Building Stable Predictive Models for Healthcare Applications: A Data-Driven Approach

by

Iman Kamkar
M. Eng.

This thesis is presented for the degree of
Doctor of Philosophy

Deakin University
July, 2016
I am the author of the thesis entitled

Building Stable Predictive Models for Healthcare Applications: A Data-Driven Approach

submitted for the degree of Doctor of Philosophy

This thesis may be made available for consultation, loan and limited copying in accordance with the Copyright Act 1968.

'I certify that I am the student named below and that the information provided in the form is correct'

Full Name: ........Iman Kamkar...........................................................................................................
(Please Print)

Signed: .................................................................

Date: ........2/12/2016..................................................................................................................
I certify the following about the thesis entitled (10 word maximum)

*Building Stable Predictive Models for Healthcare Applications: A Data-Driven Approach*

submitted for the degree of Doctor of Philosophy

a. I am the creator of all or part of the whole work(s) (including content and layout) and that where reference is made to the work of others, due acknowledgment is given.

b. The work(s) are not in any way a violation or infringement of any copyright, trademark, patent, or other rights whatsoever of any person.

c. That if the work(s) have been commissioned, sponsored or supported by any organisation, I have fulfilled all of the obligations required by such contract or agreement.

I also certify that any material in the thesis which has been accepted for a degree or diploma by any university or institution is identified in the text.

'I certify that I am the student named below and that the information provided in the form is correct'*

**Full Name:** ...........Iman Kamkar........................................................................................................

(Please Print)

**Signed:** ........................................................................................................................................

**Date:** ............2/12/2016..............................................................................................................
Contents

Abstract .................................................. xxii

Acknowledgements ........................................ xxiv

Relevant Publications ..................................... xxvi

Notation .................................................. xxviii

1 Introduction ........................................... 1

1.1 Aims and Approaches ................................... 3

1.2 Significance and Contributions ....................... 4

1.3 Outline of the Thesis ................................... 7

2 Background ............................................ 10

2.1 Electronic Medical Records .......................... 10

2.1.1 Application Areas .................................. 11

2.1.2 Research Challenges ............................... 14
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1.2.1 Incomplete Data</td>
<td>15</td>
</tr>
<tr>
<td>2.1.2.2 Sparsity and Instability</td>
<td>16</td>
</tr>
<tr>
<td>2.1.2.3 Selection Bias and Confounding Effect</td>
<td>17</td>
</tr>
<tr>
<td>2.2 Machine Learning Models</td>
<td>17</td>
</tr>
<tr>
<td>2.2.1 Predictive Modeling</td>
<td>18</td>
</tr>
<tr>
<td>2.2.1.1 Preliminaries</td>
<td>18</td>
</tr>
<tr>
<td>2.2.1.2 Statistical Learning</td>
<td>20</td>
</tr>
<tr>
<td>2.2.1.3 Linear Regression</td>
<td>24</td>
</tr>
<tr>
<td>2.2.1.4 Logistic Regression</td>
<td>26</td>
</tr>
<tr>
<td>2.2.1.5 Support Vector Machine</td>
<td>29</td>
</tr>
<tr>
<td>2.2.1.6 Naive Bayes</td>
<td>31</td>
</tr>
<tr>
<td>2.2.1.7 Decision Trees</td>
<td>32</td>
</tr>
<tr>
<td>2.2.1.8 Random Forest</td>
<td>33</td>
</tr>
<tr>
<td>2.2.2 Feature Selection</td>
<td>34</td>
</tr>
<tr>
<td>2.2.2.1 Feature Selection vs. Feature Extraction</td>
<td>34</td>
</tr>
<tr>
<td>2.2.2.2 Filter Methods</td>
<td>34</td>
</tr>
<tr>
<td>2.2.2.3 Wrapper Methods</td>
<td>37</td>
</tr>
<tr>
<td>2.2.2.4 Embedded Methods</td>
<td>37</td>
</tr>
<tr>
<td>2.2.3 Feature Selection Stability</td>
<td>39</td>
</tr>
<tr>
<td>2.2.3.1 Ensemble Approaches</td>
<td>40</td>
</tr>
</tbody>
</table>
2.2.3.2 Data Variance Reduction Approaches .......... 42
2.2.3.3 Group-Based Approaches ..................... 43
2.2.4 Evaluation Measures .............................. 45
  2.2.4.1 Stability Measurements ....................... 45
  2.2.4.2 Classification Performances .................. 50
  2.2.4.3 Combining Stability and Prediction .......... 52

3 Stable Clinical Prediction: Exploiting ICD Tree Structure 53
  3.1 Sparsity and Stability ............................ 56
  3.2 Framework Overview .............................. 57
  3.3 Baseline Methods ................................ 60
    3.3.1 Feature Selection Methods .................... 61
    3.3.2 Classification Methods ....................... 62
  3.4 Experiments ................................ 63
    3.4.1 Datasets ................................... 64
      3.4.1.1 Synthetic Datasets .................... 64
      3.4.1.2 Real-World Datasets .................. 65
    3.4.2 Evaluation Measures ......................... 66
    3.4.3 Experimental Settings ....................... 66
    3.4.4 Experimental Results ......................... 68
3.4.4.1 Stability Performance of Feature Selection Methods 68
3.4.4.2 Classification Performance .............................. 70
3.4.4.3 Comparison With Random Forest (as an Embedded Feature Selection Method) 72
3.4.4.4 Comparison with Expanded-Lasso .......................... 76
3.4.4.5 Risk Factors Obtained using Tree-Lasso ................ 77

3.5 Conclusion ...................................................... 82

4 Stable Clinical Prediction by Supervised Feature Grouping 84

4.1 Methodology .................................................... 85

4.1.1 Predictive Grouping Elastic net ............................. 85

4.1.1.1 Optimization Algorithm .................................. 87

4.1.1.2 Computational Complexity ............................... 89

4.1.1.3 Theoretical Guarantees for Convergence ................. 89

4.2 Experiments .................................................... 91

4.2.1 Datasets ...................................................... 91

4.2.1.1 Synthetic Datasets ....................................... 91

4.2.1.2 Real-World Datasets ..................................... 94

4.2.2 Baselines ..................................................... 94

4.2.3 Evaluation Measures ........................................ 95

4.2.4 Experimental Settings ....................................... 96
4.2.5 Experimental Results ........................................... 97
  4.2.5.1 Stability Performance ........................................... 98
  4.2.5.2 Predictive Performance ........................................... 99
  4.2.5.3 Execution Time ..................................................... 103
  4.2.5.4 Effect of Grouping ................................................. 104
  4.2.5.5 Risk Factors Obtained Using pg-EN ......................... 105
  4.2.5.6 Feasibility of Grouping ........................................... 106

4.3 Conclusion .......................................................... 108

5 Exploiting Feature Relationships Towards Stable Feature Selection110

5.1 Covariance Lasso ...................................................... 111
  5.1.1 The Model Formulation ............................................. 111
  5.1.2 Optimization ......................................................... 112

5.2 Tuning Hyperparameters via Kappa Selection Criterion .............. 115

5.3 Experiments .......................................................... 116
  5.3.1 Baseline algorithms ................................................. 117
  5.3.2 Tuning Parameter Selection ......................................... 118
  5.3.3 Evaluation Measures ............................................... 118
  5.3.4 Simulation Results .................................................. 119
    5.3.4.1 Estimated Covariance Matrix for Synthetic Data .... 124
6 Stable Clinical Prediction with Support Vector Machines

6.1 Covariance-SVM
  6.1.1 Proof of Convexity
  6.1.2 Algorithm for Covariance-SVM

6.2 Graph-SVM
  6.2.1 Optimization
  6.2.2 Convergence Analysis

6.3 Experiments
  6.3.1 Datasets
  6.3.2 Evaluation Measures
  6.3.3 Experimental Settings
  6.3.4 Graph Generation for graph-SVM
  6.3.5 Estimated Covariance Matrix by C-SVM
  6.3.6 Experimental Results

6.4 Conclusion

7 Stable Feature Selection in Multiple-Instance Learning with Appli-
cation to Toxicity Prediction for Cancer Patients

7.1 Methodology ........................................ 155
    7.1.1 Regularized Logistic Regression for MIL ............... 155
        7.1.1.1 Objective Function .......................... 156
    7.1.2 Stabilized $l_1$-norm MIL .......................... 157
        7.1.2.1 Optimization ............................... 157
7.2 Experiments ....................................... 161
    7.2.1 Baseline Methods ................................ 161
    7.2.2 Evaluation Measures ............................. 162
    7.2.3 Simulation Study ............................... 163
    7.2.4 Toxicity Problem ............................... 166
7.3 Conclusion ........................................ 173

8 Conclusion ........................................ 174
    8.1 Summary ........................................ 174
    8.2 Future Works .................................... 176

A Mathematical Proofs ................................ 178
    A.1 Proof of Proposition 1 ............................ 178
    A.2 Proof of Proposition 2 ............................ 179
    A.3 Proof of Proposition 3 ............................ 180
B Additional Experiments 182

B.1 Additional Experiments to Evaluate pg-EN ........................... 182

B.1.1 Grouping Results on Soil Dataset ............................ 182

B.1.2 Statistical Test Results ........................................... 183

B.1.3 Evaluating the Classification Performance without Re-balancing Data ........................................ 193

B.1.4 K-means vs Random Initialization of pg-EN ................. 193

Bibliography 196
List of Figures

2.1 Electronic medical records are repository of comprehensive information about a patient, including demographics, history of hospital visits, diagnoses, physiological measurements and interventions (Jensen et al., 2012).

2.2 The graph of a convex function. The chord (the orange line) between any two points of the graph lies above the graph.

2.3 Four well-known loss functions: 0/1-loss (black), hinge loss (blue), logistic loss (red) and square loss (green).

2.4 Graphical representation of Logit function (left) and logistic function (right).

2.5 Ensemble feature selection approach using sub-sampling of original data. First, different sub-samples of the original data are built and informative features of each sub-sample are obtained. Then the final feature set is selected using a rank aggregation method.

3.1 A sample of ICD-10 tree for diseases of musculoskeletal system and connective tissue.

3.2 A graphical representation of instability behavior of sparse methods such as Lasso when selecting features in presence of correlated feature groups. Lasso tends to select a feature randomly among a group of correlated features.
3.3 Part of the ICD-10 tree constructed for the AMI dataset used in this chapter. ........................................ 57

3.4 Overview of our proposed framework. ........................................ 60

3.5 Stability results for synthetic dataset ($\rho = 0.8$) for 10 selected features. In these plots we use a threshold $T = 0.5$ and features that are selected with a probability more than $T$ are shown in black color while others in gray color. ........................................ 68

3.6 Stability results for Cancer dataset for 10 selected features. In these plots we use a threshold $T = 0.5$ and features that are selected with a probability more than $T$ are shown in black color while others in gray color. ........................................ 69

3.7 Stability results for AMI dataset for 10 selected features. In these plots we use a threshold $T = 0.5$ and features that are selected with a probability more than $T$ are shown in black color while others are in gray color. ........................................ 71

3.8 Predictive performance of different classification methods coupled with each feature selection method for synthetic data ($\rho = 0$). .......... 72

3.9 Predictive performance of different classification methods coupled with each feature selection method for synthetic data ($\rho = 0.8$). .......... 73

3.10 Predictive performance of different classification methods coupled with each feature selection method for Cancer data. ...................... 74

3.11 Predictive performance of different classification methods coupled with each feature selection method for AMI data. ...................... 74

3.12 Predictive performance of Tree-Lasso compared to Random Forest for synthetic datasets. (a) $\rho = 0$ (b) $\rho = 0.8$ ...................... 75

3.13 Predictive performance of Tree-Lasso compared to Random Forest for real-world datasets. (a) Cancer data. (b) AMI data ...................... 76
4.1 Graphical representation of the feature grouping and weight assignment procedure in the proposed model. The yellow icons in matrix $G$ are representative of non-negative values. Features in the same group are combined together by multiplying $D$ by $G$. This gives us super-features $X$, which are representatives of the correlated features in each group. The algorithm finds the coefficients $\beta$ for super-features $X$. ......................................................... 86

4.2 Empirical convergence of pg-EN for Cancer (EMR) and AMI (EMR) datasets. ......................................................... 89

4.3 Graphical illustration of correlation matrices used for synthetic datasets. 92

4.4 Feature selection stability as measured by the Kuncheva Index for real-world datasets. Larger values indicate higher stability. ............. 97

4.5 Comparing the prediction performance of pg-EN and other baselines for Synthetic datasets. ......................................................... 100

4.6 Execution time (in seconds) of pg-EN compared to Lasso and Oscar for different number of samples on Cancer (EMR) dataset. ............. 105

4.7 Prediction and stability performance of the model with respect to variation in number of groups, shown for Cancer and AMI datasets. The recommended range for number of groups is marked by vertical lines. ......................................................... 106

4.8 Example of feature groups obtained using pg-EN for AMI data. Features in a group are consistent and related to (a) bone, (b) cardio-pulmonary, (c) psycho-emotional and (d) brain diseases. Suffixes 3M, 6M or 1Y in feature names, indicate that they occurred in previous 3 months, or 6 months or 1 year of prediction. ............. 108
5.1 Grouping properties of C-Lasso for Synthetic-I data set. (a)-(d):
There are three groups of highly correlated features, with coefficients
1, 0 and −1, illustrated in red, blue and green. (a): Empirical corre-
lation matrix of Synthetic-I; beige, blue and green indicate positive,
negative and zero correlations. (b): Density plot of estimated \( \hat{\beta} \) ob-
tained by Ridge. (c): Density plot of estimated \( \hat{\beta} \) obtained by Oscar.
(d): Density plot of estimated \( \hat{\beta} \) obtained by C-Lasso.

5.2 Grouping properties of C-Lasso for Synthetic-II data set. (a)-(d):
There are three groups of highly correlated features, with positively
and negatively correlated features in each group. (a): Empirical cor-
relation matrix of Synthetic-II. (b): Density plot of estimated \( \beta \) ob-
tained by Ridge. (c): Density plot of estimated \( \beta \) obtained by Oscar.
(d): Density plot of estimated \( \beta \) obtained by C-Lasso. Note that all
the coefficients in the blue group are zero, whereas in red and green
groups, half the coefficients are 1 and half are -1.

5.3 The pictorial representation of correlation matrix of \( X \) and the esti-
mated covariance matrix \( \Omega \) for synthetic data sets. For better repre-
sentation, we show the correlation matrix computed from \( \Omega \) matrix
by standardizing its values using equation (5.12).

5.4 The pictorial representation of estimated covariance matrix \( \Omega \) for real
datasets. For better representation, we show the correlation matrix
computed from \( \Omega \) matrix by standardizing its values using equation
(5.12).

5.5 Comparison of stability performance obtained using different selec-
tion criterion i.e. CV, BIC and Kappa for each penalized regression
method.

5.6 Comparison of predictive performance of C-Lasso with other baseline
algorithms using different selection criterion, i.e. CV, BIC and Kappa
for synthetic and real data sets.
6.1 A subgraph constructed using Jaccard similarity between EMR features in AMI dataset. The nodes indicate EMR features and edges represent interaction strength, measured using Jaccard index. 

6.2 The pictorial representation of estimated covariance matrix for real datasets. For better representation, we show the correlation matrix computed from Ω matrix by standardizing its values.

7.1 Feature construction for cancer toxicity data: prediction points are set every fortnight. Feature vectors \{i_1, i_2, \cdots, i_{14}\} form a bag. \(T_1, T_2, T_3, \ldots\) are the prediction points. The toxicity outlook is 28-days.

7.2 The number of patients in each prediction point for toxicity dataset.

7.3 The proportion of patients with toxicity in each prediction point for toxicity dataset.

7.4 Stability performance of our proposed method compared to other baselines in terms of (a) SRCC and (b) JSM for toxicity dataset at different prediction points.

7.5 Classification performance of our proposed method compared to other baselines in terms of (a) F1 score and (b) AUC for toxicity dataset at different prediction points.

7.6 Trend of changes of the average AUC and F1 score along different prediction points.

7.7 Top risk factors for toxicity selected by our proposed method.

B.1 Computed feature correlation matrix for Soil data.

B.2 Group matrix obtained for Soil dataset. (The group numbers are permuted to assist comparison with Figure B.1.)
List of Tables

2.1 Comparing stability measurements based on several criteria. In the complexity part of the table, $p$ is the number of total features, $l$ is the number of iterations and $k$ is the length of selected feature set. . 50

2.2 Confusion matrix for a binary classification problem. . . . . . . . . . 50

3.1 Statistics of the real-world datasets. . . . . . . . . . . . . . . . . 65

3.2 Comparison of Tree-Lasso with baselines in terms of different stability measures for both synthetic and real-world datasets. . . . . . . . . . 67

3.3 Comparison of Tree-Lasso with Random Forest in terms of different stability measures for both synthetic and real-world datasets. . . . . . 75

3.4 Comparison of Tree-Lasso and Lasso with Expanded-Lasso in terms of feature stability and predictive performance on synthetic data ($\rho = 0$). 77

3.5 Comparison of Tree-Lasso and Lasso with Expanded-Lasso in terms of feature stability and predictive performance on synthetic data ($\rho = 0.8$). 78

3.6 Comparison of Tree-Lasso and Lasso with expanded-Lasso in terms of feature stability and predictive performance on Cancer data. . . . . . 78

3.7 Comparison of Tree-Lasso and Lasso with expanded-Lasso in terms of feature stability and predictive performance on AMI data. . . . . . 79
3.8 Well-known risk factors for cancer reported by clinicians or other research papers, which are also obtained by Tree-Lasso with high probability. ................................ 80

3.9 Well-known risk factors of readmission after AMI reported by clinicians or other research papers, which are also obtained by Tree-Lasso with high probability. ............................ 81

3.10 An example that shows importance of stability in selecting correlated features. Features I20, I21 and I25 belong to ischaemic heart diseases. Features I50 and I51 are related to heart failure. ....................... 82

4.1 Average stability performance of pg-EN compared to other baselines in terms of SRCC and JSM, for synthetic and real datasets. The numbers in brackets show the p-values obtained by applying Wilcoxon signed-rank test to the best and the second best stability results for each dataset. ................................ 98

4.2 The p-value obtained from pairwise Wilcoxon signed-rank test of MSE applied to the Synthetic datasets (a) Synthetic_I and (b) Synthetic_II .................................101

4.3 Average classification performances of pg-EN compared to other methods for Synthetic-III dataset. The numbers in brackets show the p-values obtained by applying Wilcoxon signed-rank test to the best and the second best classification results for each dataset. ................. 102

4.4 Average classification performance of pg-EN compared to other methods for Breast Cancer dataset. The numbers in brackets show the p-values obtained by applying Wilcoxon signed-rank test to the best and the second best classification results for each dataset. ................. 102

4.5 Average classification performance of pg-EN compared to other methods for Cancer (EMR) dataset. The numbers in brackets show the p-values obtained by applying Wilcoxon signed-rank test to the best and the second best classification results for each dataset. ................. 103
4.6 Average classification performance of pg-EN compared to other methods for AMI (EMR) dataset. The numbers in brackets show the p-values obtained by applying Wilcoxon signed-rank test to the best and the second best classification results for each dataset. ............... 104

4.7 Execution time (in seconds) of pg-EN compared to some other methods for real datasets. ........................................... 105

4.8 Top selected variables for cancer obtained by pg-EN are consistent with the risk factors reported by clinicians and research articles. ... 107

4.9 Top selected variables for readmission after AMI obtained by pg-EN are consistent with those reported by clinicians and research articles. 107

5.1 Simulation results for Synthetic_III. Means and standard error over 50 iterations are reported. ..................... 122

5.2 Stability performance of C-Lasso compared to other baselines for real datasets. Means and standard error over 50 iterations are reported. .. 125

5.3 Predictive performance of C-Lasso compared to other baselines for Breast cancer dataset. Means and standard error over 50 iterations are reported. .......................... 126

5.4 Predictive performance of C-Lasso compared to other baselines for Cancer dataset. Means and standard error over 50 iterations are reported. .......................... 126

5.5 Predictive performance of C-Lasso compared to other baselines for AMI dataset. Means and standard error over 50 iterations are reported. .......................... 127

5.6 ICD-10 codes and their descriptions related to the first and the last groups shown in Figure 5.4(a) ................. 129

5.7 ICD-10 codes and their descriptions related to the first and the last groups shown in Figure 5.4(b) ................. 130
6.1 Stability performance of C-SVM and graph-SVM compared to the baselines for synthetic and real-world datasets. Means and standard error over 50 iterations are reported. .................................. 149

6.2 Classification performances of C-SVM and graph-SVM compared to the baseline algorithms for Synthetic and Breast cancer datasets. Mean and standard error over 100 iterations are reported. .......................... 150

6.3 Classification performances C-SVM and graph-SVM compared to the baselines for Cancer and AMI datasets. Mean and standard error over 100 iterations are reported. .................................. 151

7.1 Average stability performance of the proposed method compared to other baselines in terms of SRCC and JSM for synthetic dataset. The standard error are shown in parentheses. .................................. 164

7.2 Average classification performance of the proposed method compared to other methods for synthetic dataset. The numbers in brackets show the standard error over 50 iterations. .................................. 165

7.3 Features and labels used in learning toxicity predictors .................................. 166

B.1 The p-value obtained from pairwise Wilcoxon signed-rank test for (a) Positive Predictive Value (PPV) and (b) Sensitivity of different methods applied to the Synthetic-III dataset. .................................. 185

B.2 The p-value obtained from pairwise Wilcoxon signed-rank test for (a) Specificity, (b) F1 score and (c) AUC of different methods applied to the Synthetic-III dataset. .................................. 186

B.3 The p-value obtained from pairwise Wilcoxon signed-rank test for (a) Positive Predictive Value (PPV) and (b) Sensitivity of different methods applied to the Breast cancer dataset. .................................. 187
B.4 The p-value obtained from pairwise Wilcoxon signed-rank test for (a) Specificity, (b) F1 score and (c) AUC of different methods applied to the Breast cancer dataset. ........................................ 188

B.5 The p-value obtained from pairwise Wilcoxon signed-rank test for (a) Positive Predictive Value (PPV) and (b) Sensitivity of different methods applied to the Cancer (EMR) dataset. ........................ 189

B.6 The p-value obtained from pairwise Wilcoxon signed-rank test for (a) Specificity, (b) F1 score and (c) AUC of different methods applied to the Cancer (EMR) dataset. ........................ 190

B.7 The p-value obtained from pairwise Wilcoxon signed-rank test for (a) Positive Predictive Value (PPV) and (b) Sensitivity of different methods applied to the AMI (EMR) dataset. ........................ 191

B.8 The p-value obtained from pairwise Wilcoxon signed-rank test for (a) Specificity, (b) F1 score and (c) AUC of different methods applied to the AMI (EMR) dataset. ........................ 192

B.9 Average classification performance of pg-EN compared to other methods for Breast Cancer dataset without re-balancing the data. the numbers in brackets show the standard error over 100 iterations. .... 193

B.10 Average classification performance of pg-EN compared to other methods for Cancer (EMR) dataset without re-balancing the data. the numbers in brackets show the standard error over 100 iterations. .... 194

B.11 Average classification performance of pg-EN compared to other methods for AMI (EMR) dataset without re-balancing the data. the numbers in brackets show the standard error over 100 iterations. .... 195

B.12 Comparison between Random and K-means initialization of pg-EN for real datasets. ................................. 195
List of Algorithms

4.1 Algorithm for solving pg-EN optimization problem. ............... 88
5.1 Kappa selection criterion ........................................ 117
6.1 ADMM for graph-SVM ............................................. 141
Abstract

About 2500 years ago, Hippocrates started a revolution in health domain by announcing for recording evidence about patients and their symptoms. This announcement established a foundation for the evolution of modern healthcare. We are now at the edge of a revolution in data-driven healthcare, enabled by advanced information technology. Emergence of electronic medical records (EMRs) for automatic storage of patient-related data promises to change the availability of data that can be used to enhance the quality of health as well as quality and longevity of life.

Analyzing health related data enable us to transform data into predictive models. These models can be used to accurately make a diagnosis or prognosis about states of a patient, which cannot be investigated directly. The scope of this thesis lies within the realm of biomedical informatics, an interdisciplinary field at the crossroads of medicine and computer science.

In this thesis we develop techniques that can exploit EMRs to learn clinical predictive models while tackling the challenges inherent in the data. EMRs are high dimensional but the majority of its content is redundant or irrelevant, which can mislead a machine learning algorithm and negatively affect its performance. Such high dimensionality calls for sparse models. However, in practice, sparse models alone are not sufficient. We need stable models, that are, models which are robust to slight changes in data. The stability matters in clinical settings because it makes the model to be interpretable and generalizable.

We propose three different models to address the instability of sparse methods applied for prediction in clinical domain. The first model extracts the hierarchical...
structure of diagnosis codes in EMR data. This model employs Tree-Lasso to perform stable feature selection and prediction at the group level of hierarchical diagnosis codes. In many situations, features may not have such tree structure, though. For example, EMR data also consist of other types of variables such as age, sex or pathological results that do not have tree structure. Our second model, solves this problem by finding groups of correlated features in data in general form, where the groups have no intrinsic structure. In this method, feature grouping is learned within the model using supervised data and therefore is aligned with prediction goal. Nevertheless, solving the objective function of this model is a formidable challenge since it is non-convex with potentially large number of local minima. Our third proposed model utilizes a convex objective function to perform prediction and stable feature selection. This model uses new regularization formulation that encourages the similarities between features based on their relatedness. The relatedness between features is captured via a feature covariance matrix. The proposed model can simultaneously perform feature selection and capture both the positive correlation and the negative correlation between features through a convex objective function. All the models are evaluated on real applications and show better feature stability and prediction performance compared to state-of-the-art methods.

We also propose a model to predict toxicity in cancer patients that is able to perform stable feature selection. In this application, each instance is associated with daily treatment and patient-specific attributes such as chemotherapy, radiotherapy, age and diagnosis codes. The toxicity data is viewed as a set of instances, where only a subset of the instances are responsible for the outcome. We propose a multiple-instance learning framework to model this data enabling us to formulate the impact of a subset of treatments on toxicity. This model uses an sparsity-inducing norm to perform feature selection. In order to increase the stability of this model, we introduce a regularization term that encourages similarities between features based on relatedness, which is captured using a covariance matrix. We apply our model on a real application and show that its prediction performance and stability outperform baseline methods.
Acknowledgments

It is my great pleasure to give my sincere thanks to all the people who have helped me during the course of my studies in the Centre for Pattern Recognition and Data Analytics (PRaDA) for the past 3 years. Collaboration with its superb staff and wonderful students has helped to shape my thought, research and ultimately, my future.

First of all with deepest appreciation, I would like to thank my principal supervisor, Dr. Sunil Gupta, who has mentored me tirelessly. Sunil incited in me a dedication to the science, guiding me through difficult problems with emphasis on the multi-armed approach to modeling and interpretation and with reinforcement of the pillars of applied machine learning. Without his thorough supervision, the work could have never taken shape.

I would also like to thank my co-supervisor Prof. Svetha Venkatesh, for giving me a chance to do research at PRaDA and for her valuable suggestions and guidance during my PhD. Her scientific writing classes is unforgettable and helped me to improve my writing skills. Her boundless energy has always inspired me to push harder whenever I felt tired. Her positive attitude is infectious, and certainly has rubbed off on me.

I would also like to express gratitude to my associate supervisor, A/Prof. Dinh Phung for his ongoing and substantive guidance. He did an unlimited effort to prepare me and my labmates for the real-world life, not only in research issues but also in different life-walks.

My thanks also goes to my colleagues at PRaDA for mutual support and making
our workplace an engaging environment. Wonderful memories of sharing stories, movies and culture will remain with me.

I am also deeply grateful to my family. First, to my Parents, Ali and Mansoureh for their unconditional support, timely encouragement and endless patience. All those math and English courses were useful, finally, as well as the conviction that there is more in this world than material things. Thanks to my sister and my brother-in-law, Aida and Javad, who have been generous with their love and encouragement despite the long distance between us. You all provide a warm, happy and stimulating environment.

I dedicate this thesis to my lovely wife, Nooshin for supporting and (moderating) me during this journey. It was her love and belief that helped me sail through dark times of my life. She has been a ray of sunshine with her positive attitude and constant smile without which life would be dull and uninteresting. If not for her rock-solid support and patience I would not have been able to complete my degree.
Relevant Publications

Part of this thesis has been published or documented elsewhere. The details of these publications are as follows:

Chapter 3:


Chapter 4:


Chapter 5:

Chapter 6:


Chapter 7:

## Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EMRs</td>
<td>Electronic Medical Records</td>
</tr>
<tr>
<td>ICD-10</td>
<td>International Classification of Diseases (version 10)</td>
</tr>
<tr>
<td>SVM</td>
<td>Support Vector Machine</td>
</tr>
<tr>
<td>C-SVM</td>
<td>Covariance SVM</td>
</tr>
<tr>
<td>EN-SVM</td>
<td>Elastic Net SVM</td>
</tr>
<tr>
<td>Lasso</td>
<td>Least Absolute Shrinkage and Selection Operator</td>
</tr>
<tr>
<td>i.i.d</td>
<td>Independently and Identically Distributed</td>
</tr>
<tr>
<td>pg-EN</td>
<td>Predictive Grouping Elastic Net</td>
</tr>
<tr>
<td>C-Lasso</td>
<td>Covariance Lasso</td>
</tr>
<tr>
<td>Oscar</td>
<td>Octagonal Shrinkage and Clustering Algorithm for Regression</td>
</tr>
<tr>
<td>AMI</td>
<td>Acute Myocardial Infarction</td>
</tr>
<tr>
<td>JSM</td>
<td>Jaccard Similarity Measure</td>
</tr>
<tr>
<td>SRCC</td>
<td>Spearman’s Rank Correlation Coefficient</td>
</tr>
<tr>
<td>ANHD</td>
<td>Average Normal Hamming Distance</td>
</tr>
<tr>
<td>PCC</td>
<td>Pearson’s Correlation Coefficient</td>
</tr>
<tr>
<td>WCD</td>
<td>Weighted Canberra Distance</td>
</tr>
<tr>
<td>PPV</td>
<td>Positive Predictive Value</td>
</tr>
<tr>
<td>NPV</td>
<td>Negative Predictive Value</td>
</tr>
<tr>
<td>AUC</td>
<td>Area Under the Receiver Operating Characteristic Curve</td>
</tr>
<tr>
<td>ADMM</td>
<td>Alternating Direction Method of Multipliers</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

In their book, “Big data: A Revolution That Will Transform How We Live and Think”, Mayer-Schönberger and Cukier (2013) mentioned that the world is increasingly becoming “datafied” and we are drifting towards a new era of data-driven insights. Data is helping businesses in every industry and is treated as a strategic asset analyzed for making decisions and drive actionable results. Healthcare data is not different. Recent advances in information technology have changed the way health care is carried out and documented (Prokosch and Ganslandt, 2009). Nowadays, not only traditional clinical narrative but also structured data relating to all aspects of care, including diagnosis, medication and pathology results are automatically captured by databases in modern health centers. Approximately, three out of four physicians use electronic medical records (EMRs) to report the patient-related conditions (Cimino, 2013). This changeover to digital records represents an important transition on improving profits and cutting down on wasted overheads.

Electronic medical records are just data. However, what we are generally interested in is knowledge. Machine learning is a scientific field that attempts to extract such knowledge from data. This is usually done by building mathematical models to learn the underlying patterns in the data and use these patterns for decision making. Machine learning enables us to employ EMRs for usages never imagined when collected (Jensen et al., 2012). For example, EMRs enable us to access longitudinal clinical care. By applying machine learning models to this data we can predict future events or outcomes for a patient and perform personalized decision making.
Each patient related data in EMRs can be represented by a vector that contains rich information about patient, including demographics, history of hospital visits, diagnoses, physiological measurements and interventions. Even though this data falls in the category of high dimensional data, considerable amount of it is irrelevant or redundant for prediction. Therefore, to build accurate prediction models from such a high-dimensional data, feature selection is necessary. Among many feature selection methods that have been introduced in literature, sparse methods such as Lasso that use $l_1$-norm penalty to obtain parsimonious set of features have been shown to be effective (Tibshirani, 1996).

In many applications such as healthcare and bioinformatics, feature selection is an stage for knowledge discovery, where identification of strongly relevant features can uncover hitherto unrecognized connections between risk factors and occurrence or progression of diseases. In such applications, sparse models alone are not sufficient. We need stable models that are well posed to select the same set of features irrespective of slight changes in training data. In these problems, as knowledge discovery is prominent, the fact that with slight variation in dataset different sets of features will be selected, can be problematic and puzzling. Further, stability has also been linked with the generalization performance of an algorithm (Poggio et al., 2004). It is shown that the concept of stability is necessary and sufficient for an algorithm to be able to learn (Shalev-Shwartz et al., 2009). However, sparsity and stability are contradictory objectives and sparse models have a tendency to produce unstable features (Xu et al., 2012). The problem of instability aggravates in presence of highly correlated variables, which is the case in EMR data. This happens because sparse models often select one feature among a group of highly correlated features (Zou and Hastie, 2005). Therefore, instability can prevent sparse models to achieve generalizability and reproducibility from one cohort to another. In this thesis we aim to address the model instability occurred in applying sparse models to the high dimensional clinical data. To this end, we make use of information hidden in high dimensional electronic medical records to improve model stability and prediction accuracy.
1.1 Aims and Approaches

This thesis investigates into sparse models, focusing on their instability issues for knowledge discovery in healthcare applications. We aim to develop a suite of techniques to perform stable feature selection and accurate prediction for a range of clinically relevant problems such as mortality, hospital readmission and toxicity prediction. In particular, our main goals are:

- To explore the way hierarchical structure of diagnosis codes in electronic medical records can be incorporated to increase the feature stability of prediction model. We introduce a feature extraction process that generates ICD-10 tree by making use of predefined ICD-10 coding hierarchy in diagnosis codes. Our framework performs feature selection through model training with Tree-Lasso regularization. The diagnosis codes in the extracted ICD-10 tree are grouped based on disease similarity, which puts the correlated features together. Thus, feature grouping can be exploited in a tree-structure to achieve improved stability of the model.

- To build a stable feature selection and prediction model in a general context, where the structure between features is not explicitly stated. We propose a novel model, termed as Predictive Grouping Elastic net (pg-EN) that automatically finds groups of correlated features and selects informative groups instead of each individual feature. Feature grouping is inferred within the model in a supervised learning framework and therefore is aligned with prediction goal. We formulate the model as a non-convex optimization problem. To solve this problem, we propose an efficient iterative algorithm with theoretical guarantees for its convergence.

- To provide an alternative approach to perform stable feature selection and prediction in general context with a convex objective function. We propose a new model, termed Covariance Lasso (C-Lasso) that has a new regularization formulation to improve the stability of Lasso by encouraging the similarities between features based on their relatedness. The model uses a feature covariance matrix to capture the relatedness between features. The proposed model is capable of capturing both positive and negative correlation between features through a convex objective function.
1.2 Significance and Contributions

- To propose new stable feature selection method using support vector machine. It is shown that combining SVM with $l_1$-norm regularization enables the model to perform feature selection and classification simultaneously but the resulting model would be unstable. To address this instability, we propose two models to stabilize $l_1$-norm SVM, namely Covariance-SVM (C-SVM) and graph-SVM. C-SVM employs a regularization term that induces the correlated features to be simultaneously present or absent. Our second model, graph-SVM uses an undirected feature graph, in which nodes represent EMR features and edges represent relationship between the features defined using Jaccard index. It uses a convex penalty that includes a pairwise $l_\infty$ regularizer to encourage coefficients of the correlated features to be equal.

- To build a stable model to predict treatment toxicity in cancer patients. In toxicity data daily treatment and patient-specific attributes such as chemotherapy, radiotherapy, age and diagnosis codes are treated as an instance. The overall treatment data is viewed as a set of instances, where only a subset of the instances are responsible for the outcome because not all treatments are toxic. We propose a multiple-instance learning framework to model this data enabling us to formulate the impact of a subset of treatments on toxicity. Since the data is based on EMR and has dimensional features, sparsity and stability aspects of the model are desirable. Using a sparsity-inducing norm the model can select the most important risk factors involved in treatment toxicity of cancer. To increase the stability of the model, a regularization term is used that induces the correlated features to be simultaneously present or absent.

1.2 Significance and Contributions

The significance of this thesis lies in its interdisciplinary contribution that investigates the nexus between machine learning and healthcare. In other words, this thesis is organized around two central lines of work: (i) developing new algorithms and techniques that are capable to perform stable feature selection and accurate prediction and (ii) applying such frameworks to a variety of practical healthcare problems. In particular our main contributions are:
• Exploiting hierarchical structure of diagnosis codes in EMRs and incorporation of these data to Tree-Lasso to perform stable feature selection and prediction at the group level of hierarchical diagnosis codes. In this framework, we apply a feature extraction process to generate a tree out of available ICD-10 diagnosis codes in the data by making use of predefined ICD-10 coding hierarchy in diagnosis codes. Consequently, we obtain the ICD-10 tree from the data, in which feature correlations are represented in the form of a tree-structure. Due to the high correlations among parent nodes and their offspring, modeling this data using Lasso leads to feature instability. To increase the stability of the prediction model, we perform feature selection through model training with Tree-Lasso regularization, which enables us to exploit the feature correlations in the form of a tree-structure and hence, improves the stability of the model. An extensive experimental study shows stability behavior of our proposed method is significantly better than Lasso and comparable with other feature selection algorithms. Also, we have assessed the predictive performance of our feature sets using several classifiers, e.g. logistic regression, naive Bayes, SVM, decision trees and Random Forest and find that under the constraint of stable feature selection, the prediction performance of our model is consistently better than that of many feature selection algorithms, namely T-test, IG, ReliefF, and Lasso.

• Proposal of a new model aimed to achieve stable feature selection in general context. Our method aims to improve the stability of Lasso by grouping correlated features and selecting informative groups instead of each individual feature. In this method, feature grouping is learned within the model using supervised data and therefore is aligned with prediction goal. To this end, we learn a matrix $G$, where each column of $G$ represents a group such that if a feature $p$ belongs to a group $k$ then $G_{ik}$ is non-zero, otherwise it is zero. We formulate the proposed model as a non-convex constrained optimization problem combining both feature grouping and feature selection in a single step. To solve this problem, we propose an efficient iterative algorithm with theoretical guarantees for its convergence. The experimental results demonstrate that the proposed method outperforms recent state-of-the-art methods and obtains significant improvements in terms of model stability and classification performance on real-world clinical problems.
• **Proposal of a new model with convex objective function aimed to achieve stable feature selection.** This model incorporates a new regularization formulation that improves the stability of Lasso by encouraging the similarities between features based on their relatedness. The relatedness between features is captured via a feature covariance matrix. The proposed model can simultaneously perform feature selection and capture both the positive correlation and the negative correlation between features through a convex objective function. This method has the grouping effect, which means that a group of highly correlated predictors are either all selected together into the model or left out altogether. The experimental results shows that the proposed method achieves better model stability and prediction performance compared to other baselines.

• **Proposal of two stable classification models using support vector machine for clinical prediction.** We propose two convex methods to stabilize $l_1$-norm support vector machines. Our first method, makes use of a new regularization formulation that encourages the similarities between features based on their relatedness. In our formulation, the relatedness between features is captured through a feature covariance matrix. We call this model as Covariance SVM (C-SVM). We have proposed an alternating optimization algorithm to solve this objective function. In our second method, to stabilize $l_1$-norm SVM, we construct an undirected feature graph, where nodes represent EMR features and edges represent relationship between the features. We define the statistical relationship between features using Jaccard index. In this graph correlated features are connected by an edge that enables us to perform feature selection on the group level. Our proposed method uses a convex penalty that includes a pairwise $l_\infty$ regularizer to encourage coefficients of the correlated features to be equal. We refer to this method as graph-SVM. We solve the resulting objective function using alternating direction method of multipliers (ADMM). We demonstrate the effectiveness of these two methods on both synthetic and real-world datasets.

• **Proposal of a new model for prediction of treatment toxicity in cancer patients.** In this application, each instance is associated with daily treatment and patient-specific attributes such as chemotherapy, radiotherapy, age and diagnosis codes. The toxicity data is viewed as a set of instances, where only a subset of the instances are responsible for the outcome. We propose a multiple-
instance learning framework to model this data enabling us to formulate the impact of a subset of treatments on toxicity. This model uses an sparsity-inducing norm to perform feature selection. In order to increase the stability of this model, we introduce a regularization term that encourages similarities between features based on relatedness, which is captured using a covariance matrix. We have shown the efficiency of the proposed model in terms of model stability and classification performance on real-world data.

1.3 Outline of the Thesis

The rest of this thesis is organized as follows:

- In chapter 2 we provide the preliminary background related to the work in this thesis, starting with an introduction to electronic medical records (EMRs), their application areas and the research challenges in modeling this type of data. The chapter then focuses on the machine learning background related and essential for development of our key models. The material covers classification methods, feature selection algorithms, stability measures and classification metrics.

- Chapter 3 begins with presentation of our first contribution on building a stable feature selection and prediction model in clinical domain. Our framework consists of a feature extraction process to generate a tree out of available ICD-10 diagnosis codes in the data. The model then performs feature selection using Tree-Lasso regularization and the feature correlations in the form of a tree-structure to improve the stability of the model. The second half of this chapter focuses on validating the proposed framework on real-world clinical applications including prediction of mortality in cancer patients and prediction of readmission for the AMI patients.

- Chapter 4 proposes a novel framework, called predictive grouping elastic net (pg-EN) that is able to perform stable feature selection and prediction using clinical features that have no specific structure. After formulating the proposed model as a non-convex constrained optimization problem, an efficient
iterative algorithm is proposed to solve this problem. We have also proved the convergence of this algorithm theoretically. Finally, experiments are conducted for both synthetic and real datasets showing the superiority of the proposed model over other state-of-the-art feature selection and classification methods.

- Chapter 5 addresses the non-convexity problem of the pg-EN by proposing a new framework called Covariance Lasso (C-Lasso) with convex objective function. Employing a new regularization formulation for capturing similarities between features, this model improves the instability of Lasso. After presenting the problem formulation, we present an iterative solution to optimize the cost function of our proposed method. Additionally, we introduce the Kappa selection criterion, used for selecting the tuning parameter of C-Lasso, in order to further increase its feature selection stability. Finally, experiments using real datasets are conducted to validate the efficiency of the proposed model.

- Chapter 6 proposes two stable classification models for clinical prediction. In this chapter, we focus on stable feature selection with a promising classifier in clinical domain i.e. support vector machine. We propose two convex methods to stabilize $l_1$-norm support vector machines. Our first method, makes use of a new regularization formulation that encourages the similarities between features based on their relatedness. We call this model as Covariance SVM (C-SVM). In our second method, to stabilize $l_1$-norm SVM, we construct an undirected feature graph, where nodes represent EMR features and edges represent relationship between the features. We define the statistical relationship between features using Jaccard index. In this graph correlated features are connected by an edge that enables us to perform feature selection on the group level. After formulating these two models, we present an iterative solution to optimize their cost functions. Finally, we demonstrate the effectiveness of these two models on both synthetic and real-world datasets.

- Chapter 7 proposes a model to predict toxicity in cancer patients that is able to perform stable feature selection. We propose a multiple-instance learning framework to model this data enabling us to formulate the impact of a subset of treatments on toxicity. This model uses a sparsity-inducing norm to perform feature selection. To increase the stability of the model, a regularization term is used that induces the correlated features to be simultaneously present
or absent. An efficient optimization algorithm is developed to solve this problem. We apply our model on a real application and show that its prediction performance and stability outperform baseline methods.

- Chapter 8 summarizes the main content of the thesis, and discuss future lines of inquiry.
Chapter 2

Background

This chapter provides a review of related background for the work investigated in this thesis. We dedicate the first section of this thesis to introduce electronic medical records. We discuss about their application areas and also their research challenges associated with this data. In second section we discuss about related machine learning background. First, we briefly talk about statistical learning theory. We then review some predictive models such as logistic regression, support vector machines, etc. Next, we discuss about feature selection methods and the notion of feature stability. Finally, we introduce important evaluation measurements for feature stability and prediction performance.

2.1 Electronic Medical Records

It is required for doctors to document evidence about patients and their symptoms. For decades, these information were recorded on papers by hand. However, these traditional paper records have two main drawbacks. Firstly, hand writing may cause misunderstanding due to illegibility and thus may increase the rate of mistake. Secondly, paper records cause limitations in saving and searching of medical documents and also information sharing between doctors.

Recent advances in the domain of information technology has changed the way health
2.1. Electronic Medical Records

Figure 2.1: Electronic medical records are repository of comprehensive information about a patient, including demographics, history of hospital visits, diagnoses, physiological measurements and interventions (Jensen et al., 2012).

care is carried out and documented (Prokosch and Ganslandt, 2009). Nowadays, not only traditional clinical narrative but also structured data relating to all aspects of care, including diagnosis, medication, pathology results and radiological imaging data are automatically captured by databases in modern health centers (see Figure 2.1). These electronic medical records (EMRs) contain rich information about a patient, including demographics, history of hospital visits, diagnoses, physiological measurements and interventions. Even though EMRs have been primarily used for financial and billing purposes, the trend is changing towards employing these data for disease prevention, diagnosis, prognosis and treatment.

2.1.1 Application Areas

Mining EMRs can yield useful patterns hidden in the data. The findings discovered from EMRs have been applied to variety of clinical areas such as:

**Comorbidity Analysis:** Comorbidity analysis is the process of exploring relationships between co-occurring diseases. As some diseases often occur together, treating them simultaneously would be desirable. Several research studies have been performed in this domain. For example, Roque et al. (2011) by analyzing comorbidities observed that alcohol usage is associated with depression, anxiety
2.1. Electronic Medical Records

and personality disorders. Shin et al. (2010) employed the association rule mining framework to observe comorbidities associated with hypertension such as chronic renal failure and cerebral infection. An statistical framework has been used by Cao et al. (2005) to explore the association between diseases such as “myasthenia gravis” and “cushingoid facies”.

**Risk Prediction:** Risk prediction is the problem of building predictive models to estimate the probability of having a particular disease (diagnosis) or developing a specific disease (prognosis). Conducting such analysis enables us to identify individuals who are at risk of developing a specific disease or to improve a patient’s health by preventing progression of its disease.

Several statistical learning models have been proposed to assess the risk of a complication in the patient. Feldman and Chawla (2014) proposed an adaptive boost algorithm to estimate the length of stay for an infant in the ICU. Burke et al. (1995) compared 5-year predictive accuracy of different statistical models including TNM staging model, principal component analysis, classification and regression trees, logistic regression with different types of artificial neural networks (ANN) for prediction of breast cancer. Following the previous study, Burke et al. (1997) compared the predictive accuracy of the TNM staging system with that of ANN in predicting 10-year survival of patients with breast carcinoma. In this study, only the TNM variables (tumor size, number of positive regional lymph nodes, and distant metastasis) were used. Authors showed that ANN achieved prediction accuracy of 0.730 compared to TNM staging system with prediction accuracy of 0.692. Lakshmanan et al. (2013) proposed a disease progression model that continuously learns from discrete-time observations to predict disease progression in type 2 diabetes mellitus (T2DM) and cardiovascular complications.

In some cases, it is also essential that prediction models provide information about underlying risk factors, which are responsible in occurring or progression of a disease. Tran et al. (2014) proposed a framework to automatically extract features from electronic medical records of a disease in order to predict hospital readmission. By applying survival association rule mining, Simon et al. (2013) discovered that the combination of hyperlipidemia, triglycerides and brates is a risk factor for T2DM. Statistical methods are used by Harpaz et al. (2013) to identify the risk factors in-
2.1. Electronic Medical Records

Involved in pancreatitis adverse effects. Ng et al. (2012) compared three classification algorithms including naive Bayes (NB), neural networks (NN), and support vector machines (SVM) for predicting survival in cancer patients within 120 days after palliative chemotherapy. They used data of 325 patients. In developing the model, a subset of available attributes was randomly selected and the forward selection (FS) method was used to identify prognostic factors for the prediction model. Using the above-mentioned classification algorithms, prediction model was built and its performance was assessed using five-fold cross validation. All the models were compared based on the AUCs and authors showed that NNs achieved the best accuracy.

Complication Prediction: Patient’s current medical conditions may arise future complications, which are often life threatening (Hripcsak et al., 2015; Park and Ghosh, 2014). As EMRs contain patients-related data across long period of time during patient’s treatment, they can be used to predict medical complications. Predicting complications can proceed clinicians towards personalized interventions and thus improve patient care.

Different statistical and machine learning methods have been proposed to build such models. Cox proportional hazard regression has been employed by Yadav et al. (2015a) to estimate the risk of Ischemic Heart Disease (IHD) and Congestive Heart Failure (CHF) involved in T2DM. Other works proposed multiple Gaussian process techniques to model the clinical time series for predicting complications occurred in the blood data of patients after cardiac surgery (Liu and Hauskrecht, 2013; Liu et al., 2013; Liu and Hauskrecht, 2015).

Adverse Event Detection: Adverse events are unexpected medical conditions due to negligences in patient’s care, drug reactions or treatment adverse effects. Detecting such events not only can modify the quality of patient’s care but also can reduce the cost for healthcare provider.

The majority of researches in this field center around analyzing and predicting of adverse effects (toxicities) of drugs or treatments. Some researches have used statistical methods to perform toxicity prediction (Hurria et al., 2011; Kim et al., 2011). In (Hurria et al., 2011), the risk factors of chemotherapy toxicity is identified based
on the $p$-values, which are applied to a multivariate logistic regression model to compute the probability of toxicity occurrence. Kim et al. (2011) used the factors that cause radiation pneumonia in lung cancer to define the abscissa of a logistic regression function. However, these methods cannot handle the interactions between variables and hence have limited predictive power to be applied in clinical practice (El Naqa et al., 2009).

There are also a few studies that use machine learning approaches to predict toxicity (El Naqa et al., 2009; Gulliford et al., 2004; Pella et al., 2011; Li et al., 2016). In (El Naqa et al., 2009), the nonlinear relationship between variables is modeled using a SVM kernel method. An artificial neural network is used in (Gulliford et al., 2004) to predict the biological outcomes by learning the relationship between the type of treatment and its side effects. (Pella et al., 2011) predict the acute toxicity in prostate radiotherapy using traditional classification methods and large scale optimization. However, the impact of daily treatment is not considered in any of these studies. Li et al. (2016) also address the toxicity prediction in multiple prediction points. In their paper multiple instance learning has been used to capture the effect of daily treatments on toxicity outcome. A multi-task framework embedding MIL has been formulated to enhance the prediction performance at prediction points. In (Li et al., 2016), authors use a Bayesian approach to specify the generative process of toxicity outcome and employ Gibbs sampling to infer the posterior distribution of the coefficients. However, their paper uses all the features extracted from data without considering any feature selection or feature stability.

2.1.2 Research Challenges

Although EMRs have good potentials for clinical research, their nature introduces some challenges (Jensen et al., 2012). In order to analyze these data, it is required to effectively address these challenges. Some of these challenges are as follows:
2.1.2.1 Incomplete Data

Incomplete data is a frequent complication of many real-world study. We face the same challenge in mining and modeling of EMR data. Analysis of datasets with incomplete can results in biased analysis and incorrect inferences. Different reasons can lead to data incompleteness such as:

Censored Data: In clinical researches we usually follow-up patients for a certain period of time. Therefore, it is possible that our event of interest falls outside the follow-up period and so becomes unobservable. Patients with such incomplete data are called censored observations (Steyerberg, 2008). Censored observations can result in essential information loss about patients condition. For example, suppose a study is conducted to predict 5-year survival in cancer patients. If some patients have been followed for 1 year, others for 3 years, etc we lost information after the patient is censored. The censored observations can either be left censored or right censored data, where in former the event of interest occurs before the study is started and in latter it occurs after the study is finished.

Fragmentation: During its treatment a patient may visit multiple healthcare providers to seek for a better care. These health providers often do not share data with researchers conducting the study due to various reasons such as confidentiality legislation. Therefore, the patient’s trajectory would be only partially observable during the study. This is called fragmentation.

Intent to Treat: EMRs usually reflect physician’s prescriptions advice, but in most cases we are not aware whether the patient took the medication. This situation, known as Intent to Treat is another source of missing data. In some cases physicians also give some recommendations about lifestyle change. In this situation, EMRs do not even include those recommendations.
2.1.2.2 Sparsity and Instability

EMRs are high dimensional and contain diverse information about patients such as demographics, history of hospital visits, diagnoses, physiological measurements and interventions. Although comprehensive, the underlying representations of EMRs are sparse because patients generally only undergo a fraction of diagnosis-medication procedures. Often building predictive models with these types of data calls for models such as Lasso that have sparsity-inducing properties (Ye and Liu, 2012). Using sparse methods will result in parsimonious set of features. This would be computationally efficient because many learning algorithms can scale quickly with additional variables (Dunne et al., 2002). Further, with selecting predictive features the learning algorithm can easily infer the association between variables and target class, which consequently lead to higher classification accuracy.

In many applications such as healthcare and bioinformatics, feature selection is an important stage for knowledge discovery, where identification of strongly relevant features can uncover hitherto unrecognized connections between risk factors and occurrence or progression of diseases. In such applications, sparse models alone are not sufficient. We need stable models, which are well posed to select the same set of features irrespective of slight changes in training data. In these problems that knowledge discovery is prominent, the fact that with slight variation in dataset different sets of informative features will be selected, can be problematic and puzzling. Further, stability is an approach to prove the generalization performance of an algorithm (Poggio et al., 2004). It is also shown that the concept of stability is necessary and sufficient for an algorithm to be able to learn (Shalev-Shwartz et al., 2009). However, sparsity and stability are contradictory objectives and sparse models produce unstable features (Xu et al., 2012). The problem of instability even aggravates in presence of high correlated variables, which is the case in EMR data. This is because, sparse models often select one feature among a group of highly correlated features (Zou and Hastie, 2005). This may prevent obtained models to be generalizable and reproducible from one cohort to another.
2.2. Machine Learning Models

2.1.2.3 Selection Bias and Confounding Effect

Research works performed using EMRs often suffer from selection bias and confounding effects, which can contribute to underestimates or overestimates of the actual effect of an intervention, treatment or exposure (Moher et al., 1998; Schulz et al., 1995). Selection bias occurs because EMRs used for research are often obtained from a small sample of population. This data contains typical demographical attributes for the sampled cohort and thus may limit inferences that we can make about the general population. This problem frequently happens while we are dealing with developing clinical models using EMR data. Another type of selection bias known as “healthy volunteer bias” occurs when participants in research are healthier than those who remain in longitudinal study or vice verse (Starks et al., 2009).

Selection bias may lead to confounding effect (Walline, 2001). This happens when a variable (called confounding or lurking variable) is observed between a given exposure and the outcome, but does not lie on the pathway link in the chain of causation between the exposure and the outcome. For example, it was known that there is a high degree of association between usage of oral contraceptive pills and the risk of myocardial infarction. Yet, it was later proved that this association was invalid because the majority of contraceptive pills consumers in the experiments were tobacco users. Thus tobacco consumption was the confounding variable in this experiment (Yadav et al., 2015b).

Confounding effect can result in type I error, in which the outcome are falsely attributed to the exposure rather than to the confounding variable. Alternatively, confounding effect can lead to type II error, in which the study wrongly concludes there are no treatment effects.

2.2 Machine Learning Models

This section reviews several machine learning techniques used throughout this thesis. We first review the machine learning models for prediction/classification. We then introduce and review several types of feature selection methods and explain about the notion of feature stability. Finally, we survey the evaluation measures used for
for feature stability and prediction/classification performance.

2.2.1 Predictive Modeling

2.2.1.1 Preliminaries

Notations

Throughout this thesis, we use $n$ to represent the number of distinct data points or observations and $p$ to denote number of features or variables in the data. Also, we show a vector in $p$ dimension by a bold lowercase letter, such as $\mathbf{x}$ and matrices by uppercase letters. Thus, a set of $n$ points in $\mathbb{R}^p$ is an $n$ rows and $p$ column matrix denoted as $X$. The $i$th row of matrix $X$ is shown by $x_i$. In some cases, it might refer to the $i$th column of the matrix $X$. The scalars are denoted by lowercase non-bold letters. $x_{ij}$ shows the $j$th variable of the $i$th observation in matrix $X$.

Convex Optimization

In this thesis we discuss about many optimization problems that their objective functions are either convex or non-convex. We briefly define the convexity and the strict convexity of a function, which come into play in many places across the thesis. For more information about this topic we refer readers to Boyd and Vandenberghe (2004).

**Convexity:** A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if its domain $\text{dom}(f)$ is a convex set and for all $x, y \in \text{dom}(f)$ and for $\theta$ in $0 \leq \theta \leq 1$, we have:

$$f(\theta x + (1 - \theta)y) \leq \theta f(x) + (1 - \theta)f(y).$$  \hspace{1cm} (2.1)

Geometrically, the above inequality means that the line segment between $(x, f(x))$ and $(y, f(y))$, lies above the graph of $f$ (see figure 2.2). A function $f$ is strictly convex whenever the inequality in (2.1) transforms to strict inequality. A function $f$ is concave if $-f$ is convex and vice-versa. In practice it is not easy to check the convexity of a function using equation (2.1). So, we can use the first-order or
2.2. Machine Learning Models

Figure 2.2: The graph of a convex function. The chord (the orange line) between any two points of the graph lies above the graph.

second-order conditions, which are defined in the following to check the convexity of a function.

**First-Order Condition:** If $f$ is differential, it will be convex if and only if its domain is convex and

$$f(y) \geq f(x) + \nabla f(x)^T(y - x)$$

(2.2)

holds for all $x, y \in \text{dom } f$. The above inequality shows that from local information about a convex function (i.e. its value and derivative at a point) we can derive global information (i.e. a global under-estimator of the function) (Boyd and Vandenberghe, 2004).

We can also characterize the strict convexity of function using the first-order condition. $f$ is strictly convex if and only if $\text{dom}(f)$ is convex and for $x, y \in \text{dom } f$, we have

$$f(y) > f(x) + \nabla f(x)^T(y - x).$$

(2.3)

**Second-Order Condition:** Although the convexity of a function can be checked using first-order conditions, it requires a check over every possible points over $\text{dom } f$ and hence is difficult to assess. Second-order conditions can be used to assess the convexity of a function. $f$ is convex if and only if $\text{dom } (f)$ is convex and its Hessian
is positive semidefinite:
\[ \nabla^2 f(x) \succeq 0. \] (2.4)

Geometrically, the above equation can be interpreted as the requirement that the graph of the function has positive curvature at \( x \). If function \( f \in \mathbb{R} \), equation (2.4) reduces to simple condition \( f''(x) \geq 0 \), which means that the derivative is nondecreasing. Function \( f \) is strictly convex if \( \nabla^2 f(x) \succ 0 \) for all \( x \in \text{dom } f \).

### 2.2.1.2 Statistical Learning

The theory of statistical learning is inspired from the work of Vapnik and Vapnik (1998). Assuming that data points and labels are independently and identically distributed (i.i.d) from a distribution \( P(\mathcal{X}, \mathcal{Y}) \), statistical learning chooses a function \( f : \mathcal{X} \rightarrow \mathcal{Y} \), within a space of functions \( \mathcal{F} \). For example, in linear models (regression/classification), choosing \( f \) is equivalent to choosing the best hyperplane (also known as hypothesis) \( h \) among the space of hyperplanes \( \mathcal{H} \). The process of choosing the best hyperplane is learning process. But the question is what would make a good choice?

#### Loss Functions

The quality of a predictive model is assessed using a loss function \( L \) that tells us how bad is a model instead of telling how good it is. The high value of a loss function, indicates the weakness of the model in learning the data. To evaluate the effectiveness of a prediction model a loss function needs both the output of the learned model \( \hat{y} = f(x) \) as well as its true labels \( y \). In this section, we talk about four well known loss functions, namely the square loss, the 0/1-loss, the hinge loss and the logistic loss, where the first one is mainly used in regression problems and the other three are used in classification problems. Figure 2.3 shows these four loss functions.

The 0/1-loss \( L_{0-1} \) takes a value one if the prediction is wrong, and zero otherwise. It can be written as follows:
\[ L_{0-1}(yh(x)) = \frac{1 - \text{sign}(yh(x))}{2}. \] (2.5)
2.2. Machine Learning Models

The hinge loss $L_{\text{hinge}}$, is used by the support vector machine and defined as:

$$ L_{\text{hinge}}(y_h(x)) = \max(0, 1 - y_h(x)). $$  \hfill (2.6)

The hinge loss increases linearly with the distance to the classification boundary and thus is affected by mislabelled instances, which are located far from the boundary. Although hinge loss is convex, it has two limitations. Firstly, it is not strictly convex and thus it does not have a unique minimum value. Secondly, even though it is convex, its differentials are not defined for $y_h(x) = 1$.

The third loss defined for classification is Logistic loss, $L_{\text{logistic}}$, which is used for logistic regression:

$$ L_{\text{logistic}}(y_h(x)) = \log (1 + \exp(-y_h(x))). $$  \hfill (2.7)

Logistic loss is strictly convex and its derivative is defined over its whole domain.

The square loss is usually used for regression problems and is defined as:

$$ L_{\text{square}}(y, h(x)) = (y - h(x))^2. $$  \hfill (2.8)
2.2. Machine Learning Models

As the square loss is strictly convex it has a unique optimum solution.

Among all the risks discussed, only 0/1 loss can directly measure the classification error of a model, which is 1 minus the accuracy of the model. For other losses, it is difficult to directly interpret the output of the loss function. Hence, these loss functions are rarely used as performance metrics and instead they are often used as objective functions to be minimized in order to learn the model. We explain this in more details in the next section.

Risk Minimization

As discussed before, in supervised learning problems we like to learn a hypothesis \( h \in \mathcal{H} \) on the every possible observation from \( P(\mathcal{X}, \mathcal{Y}) \) distribution. We assume that we have a non-negative real-valued loss function \( L(\hat{y}, y) \), which measures the difference between the prediction \( \hat{y} \) and the true outcome \( y \). The risk associated with the hypothesis \( h(x) \) is defined as follows:

\[
R(h) = \mathbb{E}[L(h(x), y)] = \int L(h(x), y) dP(\mathcal{X}, \mathcal{Y}). \tag{2.9}
\]

The ultimate goal of a learning algorithm is to find a hypothesis \( h_0(x) \) among all possible functions \( \mathcal{H} \) that minimizes the risk \( R(h) \):

\[
h_0(x) = \arg \min_{h \in \mathcal{H}} R(h). \tag{2.10}
\]

However, since the distribution \( P(\mathcal{X}, \mathcal{Y}) \) is generally unknown, \( R(h) \) is not available. Consequently, we have to compute its approximation, called empirical risk, by averaging the loss function on the training set:

\[
R_{\text{emp}}(h) = \frac{1}{n} \sum_{i=1}^{n} L(y_i, h(x_i)). \tag{2.11}
\]

According to the empirical risk minimization principle (ERM principle), the learning algorithm should choose a hypothesis \( \hat{h} \) that minimizes the empirical risk:

\[
\hat{h}(x) = \arg \min_{h \in \mathcal{H}} R_{\text{emp}}(h). \tag{2.12}
\]
By minimizing empirical risk as above, without restricting $\mathcal{H}$ (without learning bias), the model only learns training data. In better words, the model begins to “memorizing” training data rather than “learning”. As a result, the obtained model lacks good generalization performance. This phenomenon is called overfitting. We call a model overfits the training data when the model behave perfectly on the training data but shows poor predictive performance on unseen test data. In the following we introduce more concepts and explain how to choose an appropriate learning bias.

**Vapnik-Chervonenkis Dimension**

Assume that $\mathcal{H}$ is a class of indicator functions, taking values 0 or 1. Most binary classification functions are either indicator functions or real functions approximating indicator functions. Vapnik-Chervonenkis dimension or VC dimension ($d_{VC}$) of $\mathcal{H}$ is defined as the maximum number of points $m$ that can be shattered into two classes in all $2^m$ possible ways using functions in $\mathcal{H}$.

VC dimension enables us to upper-bound the true risk (2.9) as follows:

$$R(h) \leq R_{emp}(h) + C\left(\frac{d_{VC}}{n}\right),$$

(2.13)

Where $R_{emp}(h)$ is the empirical risk, and $C$ is confidence interval. The above bound depends on $n$, the size of training data and the VC dimension, and it is no longer dependent on $P(\mathcal{X}, \mathcal{Y})$. The confidence interval is large if either the VC dimension is large or $n$ is small. It is hard to learn when the ratio between the VC dimension and the number of points is high. So, instead of the number of parameters, it is the VC dimension that matters for generalization ability of learning machine.

**Regularized Risk**

The bound in (2.13) is known to be loose. Thus minimizing its exact expression is pointless to minimize the risk (2.9). Instead it is optimized using regularization technique, where it minimizes the sum of a loss function $L$ and a regularization term $\mathcal{R}$:

$$\arg \min L + \lambda \mathcal{R}.$$  

(2.14)
In the regularization technique, the minimization of the loss function corresponds
to minimization of empirical risk and minimization of the regularization term corre-
sponds to minimization of the second term in (2.13). As the complexity of a model
is related to VC dimension, the regularization term corresponds to penalizing this
complexity. \( \lambda \) is a regularization parameter that adjusts the importance of \( L \) and \( R \).

### 2.2.1.3 Linear Regression

Linear models have been the backbone of statistics and machine learning for the past
30 years and remain one of the most important tools in this field (Hastie et al., 2005).
These models are simple and often provide a suitable description of how the inputs
affect the output. For prediction purposes they can sometimes outperform fancier
nonlinear models, especially in situations with small numbers of training cases, low
signal-to-noise ratio or sparse data (Hastie et al., 2005). Linear models are used for
regression and classification. The main difference between classification and regres-
sion problems is in the nature of their labels. In contrast to classification problems,
in which target values take categorical values, in regression problems, labels take
continuous values. As linear models can perform a direct mapping between each
coefficient \( \beta_j \) and a feature of the data, they have high degree of interpretability.
A large value of \(|\beta_j|\) means that its corresponding feature has a high weight in the
decision process, also the sign of \( \beta_j \) shows the type of contribution of each feature.
These properties enable us to grasp a good inference about the model and will be
used a lot throughout this thesis. In this section we talk about linear models used for
regression tasks, while in the next section we discuss linear models for classification.

Given \( n \) data points \( \mathbf{x}_i \), where \( \mathbf{x}_i \in \mathbb{R}^p \) and \( i \in \{1, \ldots, n\} \) we can define a linear
model \( h(\mathbf{x}) \) with unknown parameters \( \beta \in \mathbb{R}^{p+1} \) as:

\[
h(\mathbf{x}) = \beta_0 + \sum_{j=1}^{p} \beta_j x_j,
\]

(2.15)

where the term \( \beta_0 \) is the intercept (bias). For convenience we include the constant
variable 1 as the first element of each vector \( \mathbf{x}_i \) and include \( \beta_0 \) in the vector of
coefficients $\beta$. So we can rewrite the linear model as

$$h(x) = \langle x, \beta \rangle$$  \hspace{1cm} (2.16)

where $X \in \mathbb{R}^{p+1}$. In the input-output space $(x, h(x))$ represent a hyperplane.

In this model we deal with samples and their labels i.e. the pairs of $\{(x_1, y_1), \ldots, (x_n, y_n)\}$ i.e. supervised learning methods from which we aim to estimate the parameters $\beta$. To do this, the most popular estimation method is least squares, which estimates coefficients $\beta = (\beta_0, \beta_1, \ldots, \beta_p)^T$ by minimizing the residual sum of squares

$$\text{RSS}(\beta) = \sum_{i=1}^{n} (y_i - h(x_i))^2 \hspace{1cm} (2.17)$$

To minimize the above equation we define a $n \times (p + 1)$ matrix $X$, in which each row represents an input vector (with a 1 in the first position). Similarly, we define vector $y$ as a vector of outputs in the training set. We can write the above equation as

$$\text{RSS}(\beta) = (y - X\beta)^T(y - X\beta) \hspace{1cm} (2.18)$$

Assuming that matrix $X$ is full rank, we differentiate the above equation with respect to $\beta$ and set it to zero. Thus we obtain estimation of $\beta$

$$\hat{\beta} = (X^TX)^{-1}X^T y. \hspace{1cm} (2.19)$$

**Ridge Regression:**

Even though least square estimate is straight forward and can easily estimate the parameters to model the data, it often can not obtain our satisfaction because of two reasons:

- **Prediction accuracy:** the least square estimates often have large variance and low bias resulting in overfitting. We can improve the prediction accuracy of the model by shrinking or setting some coefficients to zero. Doing this reduces
the variance in price of a little bit of bias and can improve the prediction accuracy.

- Interpretation: In some applications we are interested to know among a large number of features which features exhibit the strongest effect on the output. To this end, we need to some of the small details.

One of the methods that is used to avoid linear regression to overfit the training data by shrinking the coefficients towards zero is ridge regression. To this end, it penalizes the size of regression coefficients. Specifically it minimizes a penalized sum of squares,

\[ \hat{\beta}_{\text{ridge}} = \arg\min_{\beta} \left\{ \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij} \beta_j)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\}, \]  

(2.20)

where \( \lambda \geq 0 \) is a tuning parameter and controls the amount of shrinkage. In this method the coefficients are shrunk towards zero and each other.

We can write (2.20) in matrix form as follows:

\[ \text{RSS}(\lambda) = (y - X\beta)^T (y - X\beta) + \lambda \beta^T \beta. \]  

(2.21)

Thus, the estimation of \( \beta \) will be

\[ \hat{\beta}_{\text{ridge}} = (X^T X + \lambda I)^{-1} X^T y, \]  

(2.22)

where \( I \) is the \( p \times p \) identity matrix.

In section 2.2.2 we describe feature selection approaches used with linear regression and other machine learning techniques to improve the accuracy and interpretability of the model.

### 2.2.1.4 Logistic Regression

As mentioned earlier, linear models are also used for classification. In contrast to regression problems, in which labels take continuous values, in classification prob-
Logistic regression is one such linear classifier that arises from the desire to model the posterior probabilities of different classes via linear functions in $x$, while ensuring that they sum to 1 and remain in $[0, 1]$.

Suppose we want to predict the probability that sample $i$ belongs to class 1, i.e. the probability that $y_i = 1$. If this is a probability of a binary classification problem, then we define logit function as:

$$
\text{logit}(P(y_i = 1|x_i)) = \log \frac{P(y_i = 1|x_i)}{1 - P(y_i = 1|x_i)}.
$$

The logit function is depicted in the left side of Figure 2.4. It can be imposed that a linear model $h$ is equivalent to the logit function as follows:
logit(\(P(y_i = 1|x_i)\)) = \log \frac{P(y_i = 1|x_i)}{1 - P(y_i = 1|x_i)} = h(x_i). \quad (2.24)

Hence, the probability that sample \(i\) belongs to the class 1 can be expressed as:

\[ g(h(x_i)) = P(y_i = 1|x_i) = \frac{1}{\exp(-h(x_i)) + 1}, \quad (2.25) \]

which is known as the *logistic regression*. The logistic function is defined for \(h \in [-\infty, +\infty]\), which is continuous and differentiable. The range of the function is 0 and 1, which we expect from probability. The graphical representation of logistic regression is shown in the right side of Figure 2.4. The parameters of the logistic regression have to be estimated from the data. Variety of optimization algorithms have been developed for this aim. However, most of these techniques minimize the negative log-likelihood of Equation (2.25) over all available point:

\[
\sum_{i=1}^{n} L_{\text{logistic}}(y_i h(x_i)) = \sum_{i=1}^{n} \log(1 + \exp(-y_i h(x_i))).
\]

Minimizing the negative log-likelihood leads to identifying the model that best fits the data. See(Minka, 2003) for a comprehensive review of numerical optimizers for logistic regression.

We need to distinguish between the probabilistic interpretations obtained using logistic regression and the decision process. We can decide to assign a sample \(i\) to class 1 either based on the sign of \(h\) i.e.

\[ \hat{y}_i = \text{sign}(h(x)), \]

or as follows:

\[ \hat{y}_i = \begin{cases} +1 & \text{if } g(h(x_i)) > 0.5, \\ -1 & \text{if } g(h(x_i)) < 0.5. \end{cases} \]

When \(g(h(x_i)) = 0.5\) corresponds to equal probability of associating with any class.
2.2.1.5 Support Vector Machine

The support vector machine (SVM) is a direct heritage of statistical learning theory (Boser et al., 1992), which achieved state-of-the-art performances in applications such as healthcare (Lin et al., 2013; Himes et al., 2009) and biology (Ben-Hur et al., 2008; Mukherjee, 2003). In this thesis we deal with the simplest form of SVM i.e. linear SVM, which makes decisions based on a hyperplane. Linear SVM, like other linear models, has a good model interpretability. Further, it minimizes the risk in the spirit of (2.13). In this section we first discuss about the linearly separable classes and then we investigate the case of linearly non-separable classes.

Hard-Margin SVM

SVM is a binary classification model working based on the maximal margin hyperplane, which is the separating hyperplane with maximum distance from training data points. We can compute the distance between each data point and a given separating hyperplane. The smallest of such distance is called margin:

\[ M = \min_i \frac{y_i h(x_i)}{\|\beta\|^2} \forall i \in \{1, \ldots, n\}, \]  

(2.26)

The maximal margin hyperplane is the one that has the farthest margin from the data points. This leads to the following minimax problem:

\[ \max_\beta \min_i \frac{y_i h(x_i)}{\|\beta\|^2} \forall i \in \{1, \ldots, n\}. \]  

(2.27)

This problem is ill-posed; we can find infinitely many solutions for \( \beta \). We can fix the distance of the closest points to the hyperplane equal to \( \frac{1}{\|\beta\|^2} \):

\[ \frac{y_i h(x_i)}{\|\beta\|^2} = \frac{1}{\|\beta\|^2} \forall i \in SV. \]  

(2.28)

As a result, for all data points

\[ y_i h(x_i) \geq 1 \forall i \in \{1, \ldots, n\}. \]  

(2.29)
For maximizing the margin in (2.26), we can minimize its denominator under the constraints defined in (2.29). This is called the SVM primal problem:

\[
\arg\min_{\beta} \frac{1}{2}\|\beta\|_2^2, \\
\text{subject to } y_i h(x_i) \geq 1 \forall i \in \{1, \ldots, n\}.
\] (2.30)

**SVM Dual:** We can replace the constrained SVM primal problem in (2.30) by an unconstrained equivalent form with Lagrange multipliers as

\[
L(\beta, \alpha) = \frac{1}{2}\|\beta\|_2^2 - \sum_{i=1}^{n} \alpha_i (y_i h(x_i) - 1),
\] (2.31)

where \( \alpha_i \geq 0 \) are the Lagrangian multipliers. We can now express the derivatives of \( L \) with respect to \( \beta \) and equaling them to zero. It results in the following equalities analogous to the optimality conditions of \( L \):

\[
\begin{align*}
\beta_j &= \sum_{i=1}^{n} \alpha_i y_i x_{ij} \forall j \in \{1, \ldots, p\} \\
0 &= \sum_{i=1}^{n} \alpha_i y_i.
\end{align*}
\] (2.32)

We can eliminate \( \beta \) by using the conditions in (2.31) and obtain the SVM dual problem, written in terms of \( \alpha \):

\[
L(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{k=1}^{n} \alpha_i \alpha_k y_i y_k \langle x_i, x_k \rangle,
\] (2.33)

subject to \( \alpha_i \geq 0 \forall i \in \{1, \ldots, n\} \) and \( 0 = \sum_{i=1}^{n} \alpha_i y_i \).

This problem can be minimized with respect to the \( \alpha \). As the number of variables to optimize in dual form of SVM is \( n \) (the length of \( \alpha \)) and not \( p + 1 \) (the length of \( \beta \)), it is efficient for the problems where \( n \ll p \).

**Soft-Margin SVM**

In previous section, we assumed that classes are linearly separable (non-overlapping), which is not always the case in practice. Cortes and Vapnik (1995) proposed soft-
2.2. Machine Learning Models

margin SVM to resolve this problem. In this case, the hard margin constraint are relaxed and allows some of the data points to be misclassified. These misclassified data are penalized. The slack variable \( \zeta_i \) encodes the penalization of a data point \( x_i \). The constraints of (2.29) are rewritten:

\[
y_i h(x_i) \geq 1 - \zeta_i \forall i \in \{1, \ldots, n\},
\]

where \( \zeta_i \geq 0 \). \( \zeta_i = 0 \) indicates that its corresponding points are correctly classified and are located at a distance from the hyperplane greater or equal to the margin. \( 0 < \zeta_i < 1 \) shows that its corresponding points are correctly classified but are located inside the margin. Points with \( \zeta_i = 1 \) are located on the decision boundary, i.e. the hyperplane. Points with \( \zeta_i > 1 \) are wrongly classified. Even though the objective function of the soft-margin SVM still tends to maximize the margin, it also tends to minimize the penalties of the slack variables:

\[
\arg \min_{\beta} \quad C \sum_{i=1}^{n} \zeta_i + \frac{1}{2} \|\beta\|^2,
\]

subject to

\[
\begin{align*}
y_i h(x_i) &\geq 1 - \zeta_i \quad \forall i \in \{1, \ldots, n\}, \\
\zeta_i &\geq 0 \quad \forall i \in \{1, \ldots, n\}.
\end{align*}
\]

The trade off between minimization of the slack variables and maximization of margin is controlled by \( C \). It is the inverse of a regularization constant: the larger \( C \), the lesser the model is regularized. For \( C \to \infty \), problem (2.35) becomes equivalent to the hard-margin problem. The loss that is minimized in (2.35) is the hinge loss, depicted in Figure 2.3.

2.2.1.6 Naive Bayes

Naive Bayes (NB) is a probabilistic classifier based on Bayes theorem (Stuart Russell, 2003). It assumes that given a class label, all the features are independent and posterior probability that an example \( x \in \mathbb{R}^p \) is classified to class \( c \) is obtained as

\[
Pr(C = c \mid x) \propto Pr(C = c) \prod_{j=1}^{p} Pr(x_j \mid C = c). \quad (2.36)
\]
Despite its unrealistic independence assumption, research shows that naive Bayes often works well in practice (Rish, 2001). This is because this model is very simple, (with $O(CP)$ parameters, where $C$ is the number of classes and $P$ is the number of features) and thus is not susceptible to overfitting. Furthermore, due to its independence assumption, in naive Bayes the number of parameters do not depend on the number of examples. This property helps naive Bayes to scale well for large problems.

In this classifier, the type of each feature specifies the form of class-conditional density. For example, when features are real values, we can use Gaussian distribution, when features are binary, we can use Bernoulli distribution and when features are categorical, we can employ multinoulli distribution (Murphy, 2012).

### 2.2.1.7 Decision Trees

Decision trees (DT) are well-known classification methods in the field of machine learning. They have hierarchical structure and make decisions based on sequences of simple decision steps. Popular decision tree methods include ID3, C4.5, C5, and Classification and Regression Trees (CART) (Breiman et al., 1984; Quinlan, 1986, 1993).

Decision trees recursively partition the data based on its features to construct a tree for the purpose of improving prediction accuracy. To achieve this, they use mathematical algorithms such as information gain (used in ID3, C4.5, C5), Gini index (used in CART), and Chi-squared test (used in CHAID) to specify the variable and its threshold that splits the data into two or more subgroups. The splitting of the data is repeated until the complete tree is constructed. As a typical representative of decision trees, we focus on CART here.

CART recursively perform binary partitioning of the input space and therefore produces a binary tree, where each parent node has two offspring. There is no limitation on the type of features employed by CART. They can be continuous or categorical. Based on the target variable, which can be numerical or categorical, CART can be used for both regression or classification.
2.2. Machine Learning Models

As finding the best partitioning of data is NP-complete, the learning process in
decision trees follows a greedy mechanism. To this end, the whole input space is
represented by a root node and other nodes will be added one at a time. Each
node $d$ partitions the input space into two parts based on a threshold value $T$ and
a specific feature $j \in \{1, 2, \ldots, p\}$:

- If $x_j \leq T_d$, go towards the left offspring.
- If $x_j > T_d$, go towards the right offspring.

The selecting of feature $j$ at node $d$ is based on Gini index (Murphy, 2012). Each
final node shows a prediction value as its output. This value is analogous to the
most class of samples located into the final node.

2.2.1.8 Random Forest

Random Forest (RF) is an ensemble classifier that generates multiple decision trees,
each of which is learned on a random sample of training data and aggregates their
results (Breiman, 2001).

Each tree is trained on a number of bootstrap samples of the $n$ data points of training
data, where these samples are drawn with replacement. In addition, a subset of
features is randomly selected to consider at each node of each decision tree. To
classify an example, decisions (votes) of all trees in the forest are aggregated and
the majority voting of the trees is considered as the output of the classifier.

Random forests have a bound on generalization error, which proves that increasing
the number of forests would not lead to overfitting (Breiman, 2001).

One of the issues in building clinical prediction models in their interpretability. In
this context, linear models have high degree of interpretability because they can
perform a direct mapping between each coefficient and a feature of the data. As
this property enables us to realize a good inference about the model, we will focus
on building linear models for clinical applications in this thesis.
2.2.2 Feature Selection

Feature selection methods are a set of dimensionality reduction algorithms that try to identify important features of the input space. As building predictive models for healthcare application relies on several thousand dimensions, feature selection methods are commonly used in this context. This thesis has three central objectives: (1) The feature selection method should provide features, which are sparse enough to be interpretable by the domain expert. (2) The obtained features be stable enough to the slight variations of the training data to be generalizable and reproducible from one cohort to another, (3) the selected features result in high-performance classification models with high predictive performance.

2.2.2.1 Feature Selection vs. Feature Extraction

Feature extraction methods are also another types of techniques used in dimensionality reduction. These methods, such as principal component analysis (PCA) (Jolliffe, 2002), independent component analysis (Hyvärinen et al., 2004) and auto-encoding MLP (Hinton and Salakhutdinov, 2006) reduces data dimensionality by projecting data into lower dimensional space. Although feature extraction methods aim to reduce the dimensionality, the resulting model is not interpretable because the majority of these methods are non-linear combinations of the original features.

There are a huge number of feature selection algorithms to handle the high dimensionality of the input data. These methods can be categorized into three different classes, depending on how they utilize the learning algorithm into three different classes: filters, wrappers and embedded methods.

2.2.2.2 Filter Methods

Filter methods are independent of any classifier and perform feature selection by considering the characteristic of the dataset itself. Most of the filter methods use ranking techniques and order the features based on a given criterion. For example, Fisher score (Gu et al., 2012) uses fisher criterion in order to evaluate the features.
Spectral feature selection SPEC (Zhang et al., 2009) and Laplacian score (He et al., 2005) both select features based on the analysis of the eigen system. In this thesis, mainly three filter methods are considered: the t-test, Information Gain, and ReliefF, which are described next.

Due to the fact that filter methods are independent of any classifier, they are known to be efficient, scalable and generalizable. However, their accuracy in selecting informative features is less than wrapper methods especially if the classifier is known in advance.

**T-test**

Some filter based feature selection methods work based on hypothesis testing. T-test is one such method that is based on Welch statistic. T-test calculates a ratio between the difference of two class means and the variability of the two classes. Using this ratio we can assess whether the means of two classes are statistically different from each other. In a binary classification problem, for T-test we compute the following test statistic for each feature $f_j$:

$$
t(f_j) = \frac{\bar{f}_{j0} - \bar{f}_{j1}}{\sqrt{\frac{s_{j0}^2}{N_0} + \frac{s_{j1}^2}{N_1},}}
$$

(2.37)

where $\bar{f}_{j0}$ and $\bar{f}_{j1}$ are the feature means for class 0 and class 1, $s_{j0}$ and $s_{j1}$ are the standard deviation of feature $f_j$ from class 0 and class 1 and $N_0$ and $N_1$ are size of class 0 and class 1, respectively. In this method, the value of $t(f_j)$ is used to compute the $p$-value for feature $f_j$. The $p$-value is the probability of wrongly reject the hypothesis according to which the true means of $f_j$ for each class are identical. In this method, smaller $p$-values show interesting features. A ranking can be built based on the calculated $p$-value for each feature and feature selection can be performed based on one of the two scenarios: (1) If the feature size $S$ is known in advance, we will select the $S$ features with the smallest $p$-values. (2) If a threshold is defined for $p$-value, we will select those features that have $p$-values less than the threshold.
2.2. Machine Learning Models

**Information Gain**

Information Gain (IG) (Cover, 1991) is one of the most important feature ranking methods, which measures dependency between a feature and a class label. IG of \( j \)th feature \( f_j \) and class \( y \) is calculated as

\[
IG(y|f_j) = H(y) - H(y|f_j)
\]  

where \( H(.) \) is the entropy and is a measure of the uncertainty of a random variable. In a two class classification problem, \( H(y) \) and \( H(y|f_j) \) are defined as follows:

\[
H(y) = -[P(y = 0) \log P(y = 0) + P(y = 1) \log P(y = 1)]
\]

\[
H(y|f_j) = P(y = 0|f_j) \log P(y = 0|f_j) + P(y = 1|f_j) \log P(y = 1|f_j).
\]

In this method, the IG is evaluated for each feature independently and top \( K \) features are selected as the final feature set.

**Relieff**

Relief (Kira and Rendell, 1992) is a supervised feature selection algorithm for binary classification problems. It randomly samples instances from the training data and for each sample computes the nearest instance of the same class called “near-hit” and the nearest instance of the different class called “near-miss”. The score \( S(j) \) of the \( j \)th feature is updated in each iteration of algorithm as follows:

\[
S_t(j) = S_{t-1}(j) - \frac{d(x_t - nearHit_t)}{n} + \frac{d(x_t - nearMiss_t)}{n},
\]

where \( x_t \) is the random instance at iteration \( t \), \( n \) is the number of randomly sampled examples, and \( d(.) \) is the Euclidean distance measure. Kononenko et al. (1997) proposed Relieff by using Manhattan (\( l_1 \)) norm instead of Euclidean (\( l_2 \)) norm for finding near-hit and near-miss. For selecting final feature set using Relieff, we compute \( S \) score for each feature and select top \( K \) features with best \( S \) score as the final selected features.
2.2.2.3 Wrapper Methods

Unlike filter methods, wrappers utilize a classifier to evaluate the quality of the selected features, but otherwise they use the classifier as a black box (Guyon and Elisseeff, 2003). Wrapper methods can be divided into two categories: forward and backward feature selection methods. Forward feature selection methods start by an empty set of features, and run $p$ times a given learning algorithm on each of the $p$ features. Evaluate the performance for each feature, the best feature with the optimal performance is selected. Following this, $p - 1$ classifiers are built on the remaining features and performances are evaluated again and the feature with optimal performance is added to the first one, and so on. The backward feature selection methods, start from the full set of features and in each iteration discard the feature which its elimination improves the performance.

As both forward and backward feature selection methods only consider a small subset of features, they are sub-optimal. Due to the fact that wrapper methods make use of information from the classifier, they are more accurate than the filter methods. However, they are much more computationally expensive.

2.2.2.4 Embedded Methods

Embedded methods aim to reduce the computation time taken for reclassifying different subsets which is done in wrapper methods. The main approach is to incorporate feature selection in the process of training the classifier. Generally, the objective consists of two terms that compete with each other: (1) maximizing the goodness-of-fit, and (2) minimizing the number of variables. In the following subsections we describe some embedded methods for feature selection.

**Lasso**

The Lasso is a shrinkage method that combines an $l_1$ regularization term with the square loss for regression (Tibshirani, 1996). The Lasso estimate is defined by:

$$
\beta_{\text{Lasso}} = \arg \min_{\beta} \left\{ \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij} \beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\},
$$

(2.42)
where $\lambda$ is the tuning parameter. For classification problems, Lasso is used with the logistic regression loss function, this is known as generalized Lasso (Roth, 2004) and its objective function is:

$$\arg \min_{\beta} \left\{ -\sum_{i=1}^{N} \log p(y_i = 1|x_i) + \lambda \sum_{j=1}^{p} |\beta_j| \right\}, \quad (2.43)$$

where

$$p(y_i = 1|x_i) = \frac{1}{1 + \exp(-\beta^T x_i)}.$$

The $l_1$ penalty of (generalized) Lasso makes the solutions nonlinear in the $y_i$, where it does not have closed form solution. Many solvers have been proposed to optimize this objective function. Among them Friedman et al. (2010), proposed an efficient algorithm that iteratively minimizes the cost function using coordinate descent optimization.

**Group Lasso**

Group lasso is a group analog to the Lasso that sets groups of coefficient to zero (Yuan and Lin, 2006). Group Lasso which solves the convex optimization problem

$$\arg \min_{\beta} \left\{ \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij} \beta_j)^2 + \lambda \|\beta\|_{1,2} \right\},$$

where

$$\|\beta\|_{1,2} = \sum_{i \in \mathcal{G}_1, \ldots, \mathcal{G}_k} \sqrt{\sum_{j \in \mathcal{G}_i} \beta_j^2}$$

is the regularization of the $k$ groups of features $\mathcal{G}_1, \ldots, \mathcal{G}_k$. Group Lasso is an extension of Lasso, in which acts like Lasso but in the group level i.e. all the features within a group may be selected or dropped together and there is no sparsity within a group. The tuning parameter $\lambda$ controls the amount of regularization. For large values of $\lambda$, all the coefficients will be zero.

Yu et al. (2008) proposed Dense Relevant Attribute Group Selector (DRAGS), which is another group based feature selection method even though it is not based on $l_1$ regularization. It uses a kernel density function to discover correlated features (groups). Then, the relevance of each group is defined based on the average relevance
of every feature within that group. Finally, the most relevant feature in each group is used to perform classification with any classification algorithm. The other variant of group lasso is overlapped group lasso, in which the groups of variables have overlaps and some variables can belong to more than one group (Yuan and Lin, 2006). Yet, if a variable belongs to several groups, it will be assigned different coefficient. For example, if a variable belongs to 4 different groups, then it has 4 coefficients that need to be estimated.

2.2.3 Feature Selection Stability

The stability of a feature selection algorithm is the robustness of algorithm in selecting features in different training sets drawn from a same distribution (Kuncheva, 2007; Loscalzo et al., 2009) and is an important criterion to evaluate the quality of a feature selection algorithm. Since the concept of data is fixed, the relevant features should not change across different samples of the data. In knowledge discovery applications, such as healthcare and genetic analysis, domain experts expect algorithms to select features that are always consistent even if there are new samples introduced to the data. If the algorithm is not stable, they can not be confident about the prediction made by the selected features (Gulgezen et al., 2009).

Several methods have been proposed to improve the stability of feature selection algorithm (He and Yu, 2010; Yu et al., 2008; Han and Yu, 2012; Meinshausen and Bühlmann, 2010; Abeel et al., 2010). Based on the approach that these methods use to address the instability, they can be categorized as follows:

1. Ensemble approaches
2. Data variance reduction approaches
3. Group-based approaches
2.2.3.1 Ensemble Approaches

Generally, ensemble learning methods such as bagging (Breiman, 1996) and boosting (Freund and Schapire, 1997) are widely used in statistics and machine learning. These methods combine multiple feature selection models to obtain better results than could be obtained from any single model. Ensemble feature selection methods use the following two steps in their procedure:

- They create different feature selectors.
- They aggregate the results of constituent feature selectors and generate the ensemble output.

The second step in this procedure can be modeled as a rank aggregation problem that combines multiple rankings into a consensus ranking (Boulesteix and Slawski, 2009; Dwork et al., 2001). Brown et al. (2005) show that one of the essential steps in building a successful ensemble learner is generating a set of diverse components learners. To this end, two strategies are used:

1. Ensemble feature selection methods that run a feature selection algorithm with different sub-samples. In these methods, different sub-samples are generated from original data. These sub-samples are then used by a feature selection algorithm to select the most informative features and finally a consensus output is built using a rank aggregation method. This process is shown in Figure 2.5. The methods proposed in (Abeel et al., 2010; Bach, 2008; Duda et al., 2001; Meinshausen and Bühlmann, 2010) are classified into this category.

2. Ensemble feature selection methods that use different feature selection algorithms as their component learners. These methods are different from previous methods in two ways: Firstly, they use different feature selection algorithms instead of one, and secondly, they perform local feature selection on the original data without sampling. Methods developed in (Dutkowski and Gambin, 2007; Netzer et al., 2009; Tan et al., 2009; Yang et al., 2005) fall in this category.

In forming an ensemble method, it is recommended to use feature selection algorithms that result in diverse set of features. With the high diversity, it will be more
2.2. Machine Learning Models

Figure 2.5: Ensemble feature selection approach using sub-sampling of original data. First, different sub-samples of the original data are built and informative features of each sub-sample are obtained. Then the final feature set is selected using a rank aggregation method.

likely that the methods complement each other, in contrast to feature selection algorithms that produce similar results. On the other hand, using different feature selection methods that produce similar subset of features to build an ensemble method cannot improve learning performance (Yang and Mao, 2010). Using similar motivation, an ensemble feature selection method is proposed by (Saeys et al., 2008) that significantly improved the subset stability and rank of some well-known feature selection methods, namely, Symmetrical Uncertainty, Relief, Support Vector Machine with Recursive Feature Elimination (SVM-RFE) and Random Forest.

Similarly, (Abeel et al., 2010) proposed an ensemble framework for biomarker discovery in microarray datasets. They utilized SVM-RFE to improve the feature stability. In their method, they first rank all the features using SVM. Then, they eliminate features corresponding to the least scored features. Finally, they aggregate the final feature score using a linear combination of all scores in all iterations. This ensemble method not only improves the feature selection stability but also is robust in terms of the number of selected and eliminated features in RFE step.
The improvement in the feature stability obtained by using ensemble methods is due to the following reason. There may exist different subsets of features that result in equally good prediction accuracy. As each feature selection method may select different subset of features, when there is perturbation in the data, this would lead to change in the selected features. In this situation, since a small amount of perturbation does not cause a huge change in the selected features, the new selected features are most likely placed at bottom of the list. Therefore, selecting the most relevant features using different feature selection methods leads to selecting features with the highest weight which are more robust to small amount of perturbation in the data. In fact, this is the spirit of the ensemble approach. Ensemble method reduces the risk of missing true features and also reduces the risk of choosing wrong features.

One of the limitations of ensemble approaches is the choice of feature selection methods. If the feature selection methods lead to similar features, the stability will be high. However, we will lose diversity, which is the main goal of employing ensemble methods. On the other hand, if the feature selection methods generate diverse results, the stability will be low.

### 2.2.3.2 Data Variance Reduction Approaches

Another approach to increase the stability of a feature selection method is training the models on the samples from desired regions in the space. Data variation is one of the most important reasons of instability. Hence, reducing this perturbation (variance between samples), would improve the feature stability. Han and Yu (2012) proposed a stable feature selection framework which defines the stability of feature selection from a sample variance perspective and shows that the stability of feature selection under training data variations can be improved by variance reduction techniques. Their approach assigns higher weights to preferred samples, instead of rejecting less desired samples and therefore, the algorithm could also benefit from useful information gained from samples with less weights.

This framework has two steps:

1. Transforming the original sample $x$ to a new sample $x'$ in the margin space
according to (2.45),

\[ x'_j = |x_j - x_j^M| - |x_j - x_j^H|, \quad (2.45) \]

where \( x_j^M \) is the nearest miss and \( x_j^H \) is the nearest hit. Doing this, we can capture the local feature relevance, where the larger value of \( x'_j \) shows that the \( j \)th feature has more contribution to the margin of sample \( x \). As this transformation is sensitive to the outliers and noise in the dataset, the authors suggest to use more than one nearest neighbor from each class.

\[ x'_j = \frac{n_1}{n_1 + n_2} \sum_{k=1}^{n_1} |x_j - x_{jM_k}| - \frac{n_2}{n_1 + n_2} \sum_{k=1}^{n_2} |x_j - x_{jH_k}| \]

where \( n_1 + n_2 \) equals to the total number of instances in the data excluding the given instance.

2. To assign weight to each sample \( x \) based on the average distance between \( x \) and \( x'_i \), where \( i = 1, \ldots, n - 1 \) and \( x \neq x'_i \). This weight can be used in any feature selection method that accept sample weighting.

This algorithm addresses the problem of feature selection instability from the dataset perspective, which has the most significant impact on the stability. As one of the sources of instability is the variation in the data, reducing variation is a meaningful approach to increase feature stability. However, as this method uses weighting strategy it is not strongly resistant to outliers. For example, in presence of high level of noise in the data, this weighting scheme may assign equally high weight to good and bad samples.

2.2.3.3 Group-Based Approaches

In high dimensional data, it is common to find groups of correlated features. Often these groups are consistent to the variation of training data. Therefore, group feature selection can be used as a solution to increase stability of feature selection algorithms. These groups can be identified by using prior domain knowledge (knowledge-driven) or may be learned from data (data-driven).

Knowledge-driven methods have mostly been used in the field of bioinformatics to
find genes that have coherent expression patterns in the same gene set using large protein networks. The main idea here is to find a group of genes from the same pathway, which are associated with response and then convert this group into a new super feature for subsequent feature selection. Methods discussed in (Chen and Wang, 2009; Chuang et al., 2007; Lee et al., 2008; Rapaport et al., 2007; Tai and Pan, 2007) are some examples of knowledge-driven methods that try to identify markers as gene sets instead of individual genes. These methods use different strategies for group generation and transforming groups into super features. In group generation procedure, we can use all the genes in a same pathway, or we can search for a subset of genes to obtain a better discriminating group. To convert each group to a super feature, we can use different summary statistics methods such as mean or principle component analysis.

Data-driven group formation models identify feature groups using either cluster analysis (Au et al., 2005; Hastie et al., 2001; Ma et al., 2007; Park et al., 2007) or density estimation (Loscalzo et al., 2009; Yu et al., 2008). Cluster analysis methods use clustering algorithms such as \( K \)-means to group correlated features, whereas density estimation methods tend to group correlated features using kernel density estimation.

There is another class of related works that aim to stabilize \( l_1 \)-norm methods such as Lasso. For example, Group Lasso can be used as a remedy to stabilize Lasso when feature grouping information is available. This method performs feature selection at group level (Yuan and Lin, 2006). A modification of group Lasso that operates on overlapping groups is proposed in (Jacob et al., 2009; Zhao et al., 2009). When features have an intrinsic hierarchical and tree structure, tree-Lasso can be used as a method for increasing feature selection stability (Kamkar et al., 2014; Liu and Ye, 2010).

When grouping information is not available, feature correlations may serve as an alternative. Elastic net is an example of this class, which reduces the randomness of Lasso by using a combination of \( l_1 \) and \( l_2 \) penalties (Zou and Hastie, 2005). However, the final model obtained using this combination is less sparse and has longer list of features. When features are ordered and correlated, fused Lasso is a useful method that can exploit the structure of the features (Tibshirani et al., 2005). To group features, fused Lasso enforces the successive features in a local neighborhood to be
similar. However, such an ordering on features does not exist in many applications rendering this method inapplicable. Grouping pursuit (Shen and Huang, 2010) is another method that tries to select all possible groups, but it cannot achieve sparse model. Oscar (Bondell and Reich, 2008) is another alternative that performs feature grouping and feature selection, simultaneously. By applying a combination of $l_1$ and pairwise $l_{\infty}$ norm penalties, it imposes sparsity and equal feature weight for highly correlated features. However, assigning equal weights to the features that are only partially correlated may degrade prediction performance of the model (Bühlmann et al., 2013). Furthermore, due to the property of $l_{\infty}$, if two variables are nearby they may still receive large penalty. This leads to unnecessary bias especially for large coefficients (Yang et al., 2012).

As mentioned earlier in this chapter, electronic medical records are high dimensional and contain high correlated features. Therefore, to build stable prediction models using EMRs, we focus on methods that address the instability by grouping correlated features.

### 2.2.4 Evaluation Measures

In this section we introduce some metrics, used to evaluate the stability of the feature selection methods and performance of classification methods.

#### 2.2.4.1 Stability Measurements

With the increase attention on the stability of feature selection methods, it is important to have a reliable measure to assess this stability. Several methods have been proposed to evaluate the stability based on different results of feature selection process. Based on the representation of the output of the feature selection method, these measurements can be categorized into three groups (Kalousis et al., 2007). First group, known as stability by index, considers the indices of the selected features. In this category, the selected features have no particular order or corresponding relevance weight. In the second group, known as stability by weight, degree of relevance of each feature is measured by a weight assigned to the feature.
In the third group, which is called stability by rank, the feature's order is important in evaluation of stability. In this group, each feature is assigned a rank showing its importance.

In other words, if a training set contains $p$ features denoted by the vector $f = (f_1, f_2, \ldots, f_p)$, after using a feature selection algorithm we will have:

- In case of stability by index, a subset of features: $S = (s_1, s_2, \ldots, s_p)$, $s_i \in \{0, 1\}$, where 1 shows the presence of a feature and 0 shows its absence,
- In case of stability by weight, a weighing: $w = (w_1, w_2, \ldots, w_p)$, $w \subseteq \mathbb{R}^p$
- In case of stability by rank, a ranking: $r = (r_1, r_2, \ldots, r_p)$, $1 \leq r_i \leq p$.

**Stability by Index**
As mentioned before, the selected subset of features is represented as a vector of indices; or as a binary vector $S \in \mathbb{R}^p$, where $s_i = 1$, shows that the $i$th feature is selected. Most of the measurements in this category try to evaluate the amount of overlap between selected features in order to measure the stability. Some of the stability by index measurements are as follows:

**Average Normal Hamming Distance (ANHD):** ANHD measures the amount of overlap between two subsets (Dunne et al., 2002). This measurement works with the binary representation that represents the selected feature subset $S_{ik}$, 1 and 0 indicate whether the $k$th feature was selected in the $i$th iteration or not, respectively.

$$\text{ANHD}(S_i, S_j) = \frac{1}{p} \sum_{k=1}^{p} |S_{ik} - S_{jk}|$$  \hspace{1cm} (2.46)

ANHD $\in [0, 1]$, where 0 shows the most stability and 1 shows the least stability. When $p$ is large, the ANHD will be small that shows high stability. Also, in presence of small number of features, ANHD will be small. This is because, if a feature is selected across all the folds, or if it is not selected in any of the folds it will have the same impact on the stability result. This property of ANHD may cause wrong conclusion especially when the majority of features are not selected i.e. $k \ll p$. The
other disadvantage of ANHD is that it cannot deal with the feature sets of different sizes.

**Dice Coefficient:** Dice coefficient is another similarity measure that was used in Yu *et al.* (2008) to measure the similarity between two sets. Considering $S_q$ and $S_{q'}$ as two feature vectors, Dice coefficient is defined as:

$$\text{Dice}(S_q, S_{q'}) = \frac{2|S_q \cap S_{q'}|}{|S_q| + |S_{q'}|}. \quad (2.47)$$

$\text{DICE} \in [0, 1]$, where 0 means that there is no overlap between the sets and 1 means that the two sets are identical.

**Jaccard Index:** Jaccard index also measures the amount of overlap between two sets and produces values in the same range as Dice coefficient. Jaccard index is defined as:

$$\text{Jaccard}(S_q, S_{q'}) = \frac{|S_q \cap S_{q'}|}{|S_q \cup S_{q'}|}. \quad (2.48)$$

Generally, Dice coefficient and Jaccard index behave similarly even though Dice sometimes gives slightly higher stability results with respect to intersection between the two sets. For example, if there are two subsets of selected features $S_1$ and $S_2$ with length $k = 20$, where $|S_1 \cap S_2| = 10$, which is 50% of total number of features for each set. In this case, the stability obtained using Dice is 0.5, which shows exactly the amount of overlap between two sets. However, the stability obtained using Jaccard index is 0.33. One advantage of Jaccard and Dice over ANHD is that they can deal with the sets of different cardinalities.

One issue about Dice and Jaccard measurements is that they have tendency to increase when the set of selected features approaches the total number of features $p$. In this situation, the probability of more overlap by chance increases.

**Kuncheva Index:** As mentioned before, one of the drawbacks associated with previous stability measures is that they are unable to handle the overlap occurred due to chance. To address this problem, Kuncheva (2007) proposed Kuncheva index ($I_C$)
that has correction term to avoid intersection by chance between the two subsets. Considering \( S_q \) and \( S_{q'} \) again as two feature sets, Kuncheva Index is defined as

\[
I_C(S_q, S_{q'}) = \frac{r p - k^2}{k(p - k)},
\]

(2.49)

where \(|S_q \cap S_{q'}| = r\) and \(p\) is the number of features. The Kuncheva index is bound in \([-1, 1]\), where 1 means that two feature sets are identical i.e. \( r = k \). \( I_C = -1 \) when \( k = \frac{n}{2} \) and \( r = 0 \). This metric tries to correct overlappings occurred by chance. In other words, for independently drawn features \( S_q \) and \( S_{q'} \), \( I_C(S_q, S_{q'}) \) assumes values close to zero because \( r \) is expected to be around \( \frac{k^2}{n} \).

**Stability by Weight**

In this category, degree of relevance of each feature is measured by a weight assigned to the feature. The only measurement in this category is Pearson’s Correlation Coefficient (PCC) that measures the similarity between two weights \( w \) and \( w' \) obtained from a feature selection algorithm (Kalousis et al., 2007). PCC is defined as follows:

\[
PCC(w, w') = \frac{\sum_j (w_j - \mu_w)(w'_j - \mu_{w'})}{\sqrt{\sum_j (w_j - \mu_w)^2 \sum_j (w'_j - \mu_{w'})^2}},
\]

(2.50)

where \( \mu \) is the mean. \( PCC \in [-1, 1] \) and a value of 1 means that weightings are perfectly correlated, a value of 0 means that there is no correlation between weightings and a value of -1 means they are anticorrelated. We should mention that in some feature selection algorithms such as Lasso that the weight is equal to zero for a big number of features, the PCC will be higher.

**Stability by Rank**

In this category, the feature’s order is important in evaluation of stability. The measurements in this category deal with full set of features and cannot handle feature vectors with different cardinality.

**Spearman’s Rank Correlation Coefficient**: Kalousis et al. (2007) adapted Speraman’s rank correlation coefficient (SRCC) to assess the stability of two ranked
feature sets \( r \) and \( r' \):

\[
\text{SRCC}(r, r') = 1 - 6 \sum_j \frac{(r_j - r'_j)}{p(p^2 - 1)}.
\] (2.51)

Similar to Pearson’s correlation, the possible range of values for SRCC is \([-1, 1]\), where 1 shows that two rankings are identical, 0 shows that there is no correlation between two rankings and \(-1\) shows that rankings are in reverse order.

**Canberra Distance:** Canberra distance (CD) measures the absolute difference between two ranked feature sets (Jurman et al., 2008):

\[
\text{CD}(r, r') = \sum_{i=1}^{N} \frac{|r_i - r'_i|}{r_i + r'_i}.
\] (2.52)

This measurement does not have any upper bound and its value depends on the number of features and increases with the number of features \( p \). Therefore, it is often normalized by dividing by \( p \) to restrict results between 0 and 1. Jurman et al. (2008) proposed a weighted version of CD, defined as follows:

\[
\text{WCD}^{(k+1)}(r, r') = \sum_{i=1}^{N} \frac{|\min\{r_i, k + 1\} - \min\{r'_i, k + 1\}|}{\min\{r_i, k + 1\} + \min\{r'_i, k + 1\}}
\] (2.53)

WCD is proposed because the most important features are often located in the top-\( k \) positions of the ranked feature set. Hence, there should be less variation in the lower position of the set compared to its top part.

**Stability Measurements at a Glance**

Table 2.1 compares stability measurements based on different criteria such as type, complexity, bounds, etc. In summary, the table shows that the most existing stability measurements assess the stability returned in an index format. It also shows that rank and weight stability measures cannot handle different subset sizes.
2.2. Machine Learning Models

<table>
<thead>
<tr>
<th>Stability Measure</th>
<th>Index</th>
<th>Rank</th>
<th>Weight</th>
<th>Capability</th>
<th>Different size</th>
<th>Complexity</th>
<th>Bounds</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANHD</td>
<td>⚫</td>
<td></td>
<td></td>
<td>Yes</td>
<td>$O\left(\frac{p(l-1)}{2}\right)$</td>
<td>$[1, 0]$</td>
<td></td>
</tr>
<tr>
<td>SRCC</td>
<td>⚫</td>
<td></td>
<td></td>
<td>No</td>
<td>$O\left(\frac{p(l-1)}{2}\right)$</td>
<td>$[-1, 1]$</td>
<td></td>
</tr>
<tr>
<td>PCC</td>
<td>⚫</td>
<td></td>
<td></td>
<td>No</td>
<td>$O\left(\frac{p(l-1)}{2}\right)$</td>
<td>$[0, 1]$</td>
<td></td>
</tr>
<tr>
<td>CD</td>
<td>⚫</td>
<td></td>
<td></td>
<td>No</td>
<td>$O\left(\frac{p(l-1)}{2}\right)$</td>
<td>$[0, \infty]$</td>
<td></td>
</tr>
<tr>
<td>WCD</td>
<td>⚫</td>
<td></td>
<td></td>
<td>No</td>
<td>$O\left(\frac{p(l-1)}{2}\right)$</td>
<td>$[0, \infty]$</td>
<td></td>
</tr>
<tr>
<td>Dice</td>
<td>⚫</td>
<td></td>
<td></td>
<td>Yes</td>
<td>$O\left(\frac{k(l-1)}{2}\right)$</td>
<td>$[0, 1]$</td>
<td></td>
</tr>
<tr>
<td>Jaccard</td>
<td>⚫</td>
<td></td>
<td></td>
<td>Yes</td>
<td>$O\left(\frac{k(l-1)}{2}\right)$</td>
<td>$[0, 1]$</td>
<td></td>
</tr>
<tr>
<td>Kuncheva</td>
<td>⚫</td>
<td></td>
<td></td>
<td>Yes</td>
<td>$O\left(\frac{k(l-1)}{2}\right)$</td>
<td>$[-1, 1]$</td>
<td></td>
</tr>
</tbody>
</table>

Table 2.1: Comparing stability measurements based on several criteria. In the complexity part of the table, $p$ is the number of total features, $l$ is the number of iterations and $k$ is the length of selected feature set.

<table>
<thead>
<tr>
<th>True Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total population</td>
</tr>
<tr>
<td>Predicted Value</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

Table 2.2: Confusion matrix for a binary classification problem.

2.2.4.2 Classification Performances

The results of a classification algorithm can be presented in a confusion matrix, same as the one shown in Table 2.2. The strength of a confusion matrix is that it identifies the nature of the classification errors, as well as their quantities.

Accuracy

Accuracy demonstrate the proportion of correctly classified samples and is defined based on the entries of confusion matrix as follows:

$$\text{Accuracy} = \frac{TP + TN}{\text{Total population}}.$$ (2.54)

Due to the fact that accuracy yields misleading results in presence of unbalanced datasets, it is not a reliable metric to show the real performance of a classifier. For example, if one class contains 95% of the samples and the other one contains the
remaining 5%, a naive classifier that assigns every given sample to majority class will produce accuracy of 95% even though it performs really poor on one of the two classes.

### Sensitivity and Specificity

Sensitivity or *recall* measures the proportion of positives that are correctly identified i.e.

\[
\text{Sensitivity} = \frac{TP}{TP + FN}.
\]  

(2.55)

Specificity measures the proportion of negatives that are correctly identified i.e.

\[
\text{Specificity} = \frac{TN}{FP + TN}.
\]  

(2.56)

### Positive and Negative Predictive Value

Positive predictive value (PPV) and negative predictive value (NPV) are orthogonal to sensitivity and specificity. The PPV or *precision* is defined as:

\[
\text{PPV} = \frac{TP}{TP + FP},
\]  

(2.57)

and NPV is defined as:

\[
\text{NPV} = \frac{TN}{TN + FN}.
\]  

(2.58)

### F1 Score

In some applications such as healthcare and bioinformatics, datasets are imbalanced and number of examples belonging to one class is often lower than the overall number of examples. In this situation, we are interested to mostly focus on one class (often positive class) and so we use the F1 score for this aim:

\[
\text{F1 score} = \frac{2 \times \text{PPV} \times \text{Sensitivity}}{\text{PPV} + \text{Sensitivity}}.
\]  

(2.59)

Therefore, a high value of F1 score is better and ensures us that both precision and recall are reasonably high.
Area Under the Receiver Operating Characteristic Curve (AUC)

ROC curve is obtained by plotting the sensitivity against (1-specificity) for the range of whole cutoff values. Each point in the ROC space is correspondent to a confusion matrix. A point above diagonal means that the classifier exceeds random guessing, while a point close to a top left corner indicates that the classifier has high discriminative capability. Several metrics related to ROC curve are proposed. AUC, the area under the ROC curve is the main metric among them that is used to evaluate the performance of classifiers in imbalanced datasets. It is equivalent to Wilcoxon test of ranks and the probability that a randomly chosen positive instance is rated than a negative one.

2.2.4.3 Combining Stability and Prediction

In order to evaluate the performance of a feature selection method, stability is the most appropriate method. However, in the context of regression/classification, stability alone cannot sufficiently characterize the quality of a feature selection method. For example, if a specific set of features are enforced to be selected in every iteration of the algorithm, even though the stability would be high, the model built using those features would probably have a poor classification performance. That is why the quality of a feature selection algorithm in the context of regression/classification should consider both the feature stability and prediction performance.

Similar to (Saeys et al., 2008), we can also combine the feature selection and classification metrics in a single metric. However, keeping them separate would result in better interpretability. In this thesis, feature stability and regression/classification metrics are kept separate but in order to have a good analysis of the models behavior, we always consider them together.
Chapter 3

Stable Clinical Prediction: 
Exploiting ICD Tree Structure

In the previous chapter, we reviewed the related machine learning background and the research work in healthcare for building clinical prediction models and identified open problems. Stability of a clinical prediction model is one such problem that has been relatively neglected. But stability is essential to prognosis. In addition to good predictive performance, prognostic models require to have high interpretability and stability to maintain clinical adoption. The stability is even more paramount in applications where selected features are used for explaining the observed phenomena, such as identification of risk factors in cancer survival prediction. Therefore, stable feature selection is crucial when building clinical prediction models. Generally, the stability of a feature selection algorithm is the robustness of algorithm in selecting features in different training sets which are drawn from same distribution (Kuncheva (2007); Loscalzo et al. (2009)). Since the concept of data is fixed, the relevant features should not change across different samples of the data.

Nowadays, three out of four physicians use electronic medical records (EMRs) to report the patient-related conditions (Cimino, 2013). Even though the primary usage of these data is for financial management, much promise is held by the secondary analysis of these data. For example, it is shown that mining diagnosis codes in EMRs can reveal useful patterns and detect correlations in data that provide the basis for clinical prediction (Jensen et al., 2012). EMRs are high dimensional and
Figure 3.1: A sample of ICD-10 tree for diseases of musculoskeletal system and connective tissue.

contain rich information about patients including demographic data (age, gender, postcode), procedure and diagnosis related group (DRG) codes as well as International Classification of Disease 10 (ICD-10), where some of these variables are highly correlated. These high dimensional data often require sparse predictive models (Tibshirani, 1996; Ye and Liu, 2012). However, it is shown that sparse models show instability in presence of groups of highly correlated features (Xu et al., 2012).

As mentioned in section 2.2.3.3 of chapter 2, groups of highly correlated features are consistent to the variation of training data. Therefore, group feature selection can be used as a solution to increase the stability of feature selection algorithms. ICD-10 codes used in EMR data are the same. They form overlapping but nested groups i.e. they have an intrinsic tree structure, (An example is shown in Figure 3.1 for diseases of musculoskeletal system and connective tissue). In order to address the instability in clinical prediction models, we present a new framework that performs stable clinical prediction by exploiting hierarchical structure of ICD-10 diagnosis codes in electronic medical records (EMRs). Our framework embeds Tree-Lasso algorithm to
exploit the tree structure of ICD-10 codes. Tree-Lasso has been previously proposed as a prediction model for classifying images where its pixels are considered to be the features lying on a tree (Liu and Ye, 2010), though its stability behavior is not well understood. In the structure of ICD-10 tree, the diagnosis codes are mostly grouped based on disease similarity and co-occurrences causing correlations among features. So, using Tree-Lasso enables us to exploit the feature correlations in the form of a tree-structure and hence, improves the stability of the model.

In summary, our main contributions in this chapter are:

- Introducing a new sparse framework to obtain stable feature sets for developing healthcare predictive models from ICD codes.

- An extensive experimental study that shows stability behavior of our proposed method is significantly better than Lasso and comparable with other feature selection algorithms.

- Assessing the predictive performance of models with the corresponding feature sets using several classifiers, e.g. logistic regression, naive Bayes, SVM, decision trees and Random Forest and find that under the constraint of stable feature selection, the prediction performance of our model is always better than that of many feature selection algorithms, namely T-test, IG, ReliefF, and Lasso.

- Comparing the risk-factors obtained using the proposed framework with the list of features used by clinical experts and find that our obtained risk factors are consistent with those used by domain experts.

This chapter is organized as follows. In section 3.1, we review the concept of sparsity and stability in modeling the EMR data. In section 3.2, we describe our proposed framework for stable clinical prediction using Tree-Lasso. In section 3.3, we briefly introduce some feature selection algorithms namely T-test, Information Gain (IG), ReliefF, and Lasso that are used as baselines to be compared with our framework. Also, we talk about some classification methods such as logistic regression, naive Bayes, SVM, decision trees and Random Forest, employed with the feature selection methods to evaluate their predictive performance. Section 3.4 proceeds to experiments on two synthetic and two real-world datasets showing the benefits of our
3.1 Sparsity and Stability

As mentioned in chapter 2, the underlying representation of EMRs is sparse. The primary regularizer of choice in such sparse data is Lasso because of its convexity and sparsity-inducing properties. However, in practice, sparsity alone is not enough. We also need the model to be stable i.e. robust towards the slight changes in data. It is shown that sparsity and stability are contradictory objectives, though and sparse models produce unstable features (Xu et al., 2012). The problem of instability is further aggravated in presence of high correlated variables in EMR data. This is because that sparse models tend to select one feature randomly among a group of highly correlated features (Zou and Hastie, 2005). Small changes in data result in a significant change in selected features leading to unstable models. This may prevent obtained models to be reproducible from one cohort to another. A graphical illustration of this instability is shown in Figure 3.2.

In order to stabilize sparse models, we can find groups of correlated features and perform feature selection at the group level. Often these groups are consistent to the variation of training data. Therefore, group feature selection can be used as a solution to increase stability of feature selection algorithms. One way to identify the groups of correlated features is by using prior domain knowledge. In the next
3.2 Framework Overview

In many applications, features can be naturally represented as a tree structure, e.g. ICD-10 features in healthcare data form a tree. ICD-10 is “standard diagnostic tool for epidemiology, health management and clinical purposes”\(^1\). Figure 3.3 shows a part of ICD-10 tree relevant to Acute Myocardial Infarction (AMI) dataset used in this chapter. The set of diseases shown here relates to the ischemic and pulmonary heart diseases (ICD-10 codes: I20 up to I27). According to ICD-10 hierarchy, these diseases are classified into several groups and each of these groups are further classified into several subgroups, giving rise to a tree-structure. We note that the grouping of codes is mostly based on disease similarity and co-occurrences causing correlations among features. Due to using a flat \(l_1\)-penalty on features, Lasso randomly selects only one feature from every such correlated set. Although Lasso mechanism for feature selection results in selecting less features, it causes instability in selecting important features. This drawback of Lasso is undesirable in many real-world applications such as clinical prediction.

\(^1\)http://www.who.int/classifications/icd/en/
3.2. Framework Overview

For classification and regression problems having hierarchical features, a more suitable model is the Tree-Lasso (Liu and Ye, 2010) as it can exploit the feature correlations in the form of a tree-structure. In this context, the definition of a tree is as follows. For a tree $T$ of depth $d$, all the nodes corresponding to depth $i$ are in $T_i = \{G^i_1, G^i_2, ..., G^i_{n_i}\}$, where $G^i_j$ denotes the $j$th node at depth $i$, $n_0 = 1$, $G^0_1 = \{1, 2, ..., p\}$ and $n_i \geq 1$, $i = 1, 2, ..., d$. The nodes must satisfy the following two conditions:

1. The nodes at the same depth should not have overlapping indices.
2. The index set of a child node is a subset of its parent node.

Given the above definition of feature tree, Tree-Lasso learns the classification weight vector $\beta$ by minimizing the following cost function

$$\min_{\beta} \left\{ -\sum_{i=1}^{N} \log p \left( y^{(i)} = 1 \mid x^{(i)}; \beta \right) + \lambda \phi (\beta) \right\}$$

(3.1)

where $\lambda$ is a non-negative regularization parameter. The regularization term $\phi (\beta)$ is given by

$$\phi (\beta) = \sum_{i=1}^{d} \sum_{j=1}^{n_i} w^i_j \| \beta_{G^i_j} \|$$

(3.2)

where $\beta \in \mathbb{R}^p$ is the weight for node $G^i_j$, and $\beta_{G^i_j}$ is a vector composed of the entries of $\beta$ with the indices in $G^i_j$. The other parameter in the regularization term is $w^i_j$ ($i = 0, 1, ..., d; j = 1, 2, ..., n_i$), which is a predefined weight for the node $G^i_j$. As mentioned in (Liu and Ye, 2010), this parameter can be set according to importance of feature groups. In our application, since we do not have any prior knowledge about importance of feature groups, we use $\omega_j = 1$ for all the groups.

To solve the problem efficiently, the term $\phi (\beta)$ is re-formulated through Moreau-Yosida regularization as

$$\phi_\lambda (v) = \min_{\beta} \left\{ \frac{1}{2} \| \beta - v \|^2 + \lambda \sum_{i=1}^{d} \sum_{j=1}^{n_i} w^i_j \| \beta_{G^i_j} \| \right\}$$

(3.3)

for some $\lambda > 0$. For minimizing (3.3), we traverse the tree $T$ in the reverse breadth-first order to update $u$, where we initialize $u^{d+1} = v$ and $v \in \mathbb{R}^p$. At the node $G^i_j$, $u_{G^i_j}$ is updated according to (3.4), which reduces the Euclidean norm of $u_{G^i_j}$ by at
3.2. Framework Overview

most $\lambda_j^i$.

$$
\mathbf{u}_{G_j}^i = \begin{cases} 
0 & \|\mathbf{u}_{G_j}^{i+1}\| \leq \lambda_j^i, \\
\frac{\|\mathbf{u}_{G_j}^{i+1}\| - \lambda_j^i}{\|\mathbf{u}_{G_j}^{i+1}\|} \mathbf{u}_{G_j}^{i+1} & \|\mathbf{u}_{G_j}^{i+1}\| > \lambda_j^i
\end{cases} \quad (3.4)
$$

In the above equation $G_j^i(i = 0, 1, \ldots, d, j = 1, 2, \ldots, n_i)$ are the nodes in tree $T$, and $\lambda_j^i = \lambda w_j^i$. Using this algorithm we can obtain the sparsity in group level. For example consider traversing the tree in Figure 3.3. We let $w_j^i = 1, \forall i, j$, $\lambda = 1.5$ and $\mathbf{v} = [1, 2, 1, 4, 1, 1]^T$. When we traverse the nodes of depth 2, the elements related to indices in $I_{20}$ and $I_{25}$ become zero; after proceeding traversing the tree to the nodes of depth 1, the elements related to indices in $I_{20}$ and $I_{26}$ are set to zero, however, those in $I_{25}$ remain nonzero. Finally, when we proceed to the root node, we obtain the solution as $\beta^* = [0, 0, 0, 1, 1, 0]^T$. With the analytical solution for the minimizer of (3.3), we can employ accelerated gradient descent to obtain the optimal solution for (3.2). In this context, the Morea-Yosida regularization requires to be assessed in each of its iteration (Liu and Ye, 2010).

We have built our proposed prediction framework on the patient-related data queried from EMRs databases. An schema of this framework is depicted in Figure 3.4. This data contain static information (such as gender, sex and occupation) and time-stamped features (such as emergency visits, diagnosis codes and pathological results).

In this framework, we apply a feature extraction process to generate a tree out of available ICD-10 diagnosis codes in the data by making use of predefined ICD-10 coding hierarchy in diagnosis codes. This leads to an ICD-10 tree, in which feature correlations are represented in the form of a tree-structure.

One way to build a prediction model using this data is by applying $l_1$-norm regularization that will result in a model with few nonzero weights. However, as the extracted diagnosis codes contain groups of highly correlated features, this regularization would cause instability in the model with small variations in the training data. As we mentioned earlier, the reason for such instability is that $l_1$-norm penalty tends to assign a nonzero weight to only a single feature among a group of correlated features (Yuan and Lin, 2006; Zhao and Yu, 2006). In order to increase the stability of our prediction model, we perform feature selection through model training with
3.3 Baseline Methods

We study the stability behavior of Tree-Lasso and compare it with various feature selection methods, namely T-test, Information gain, ReliefF and Lasso. Furthermore, we evaluate the predictive performance of obtained features using each feature selection method by using different types of classifiers such as logistic regression (LR), naive Bayes (NB), support vector machines (SVM), decision trees (DT) and Random Forest (RF). In section (3.3.1) we briefly introduce feature selection methods.

Tree-Lasso regularization explained earlier in this section. The diagnosis codes in the extracted ICD-10 tree are grouped based on disease similarity, which causes correlations among features. Therefore, using our proposed framework enables us to exploit the feature correlations in the form of a tree-structure and hence, improves the stability of the model.
used as baselines in this chapter and in section 3.3.2 we introduce classifiers used as baselines for evaluating predictive performance of each feature selection method.

### 3.3.1 Feature Selection Methods

**T-test**  As introduced in section 2.2.2.2 of chapter 2, T-test calculates a ratio between the difference of two class means and the variability of the two classes. Using this ratio we can assess whether the means of two classes are statistically different from each other. In this method after calculating $t$ for each feature, best features (those who have $p$-value $\leq 0.05$) are selected as final feature set.

**Information Gain**  Information Gain (IG) (Cover, 1991) measures dependency between a feature and a class label based on entropy, a measure of the uncertainty of a random variable. In this method, for each feature we evaluate IG independently and top $K$ features are selected as the final feature set. We have discussed about this method in section 2.2.2.2 of chapter 2.

**ReliefF**  As mentioned in section 2.2.2.2 of chapter 2, ReliefF (Kononenko *et al.*, 1997) is a supervised feature selection algorithm for binary classification problems. It is a simple and efficient feature weighting algorithm which considers all features together in evaluating the relevance of features. The main idea of ReliefF is to weight features according to how well their values distinguish between instances that are similar to each other.

**Lasso**  Lasso is a regularization method that is used to learn a regularized regression/classification model (Tibshirani, 1996). This method uses a $l_1$-norm penalty function, which can generate exact zero estimated coefficients and therefore can be used for feature selection. We have described this method in more details in section 2.2.2.3 of chapter 2.
3.3. Baseline Methods

3.3.2 Classification Methods

Logistic Regression  Logistic regression is the most widely used statistical learning method for outcome prediction (either diagnostic or prognostic) in clinical domain (Altman, 1990; Harrell et al., 1984; Spiegelhalter, 1986). In this method, the parameters of the model can be interpreted as changes in log odds and also the results can be interpreted in terms of probabilities (Hastie et al., 2005). See section 2.2.1.4 of chapter 2 for more details.

Naive Bayes  Naive Bayes (NB) is a probabilistic classifier based on Bayes theorem (Stuart Russell, 2003). This method chooses the most likely outcome given a set of features assuming that all the features are independent. Despite its unrealistic independence assumption, research shows that naive Bayes often works well in practice (Rish, 2001). We have discussed about this method in section 2.2.1.6 of chapter 2.

Support Vector Machines  We have described support vector machines in section 2.2.1.5 of chapter 2. Apart from linear kernels, where the structure of the model can be easily described through the coefficients that define a linear hyperplane, support vector machines use a formalism that is often unsuitable for interpretation by human experts. Therefore, in clinical domain if we are only interested in predictive accuracy, SVM can be a classifier of choice. In our experiments, we use Gaussian RBF kernels, where the kernel width $\sigma$ is of values $\{0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 2\}$ and the value of box constraint $C$ is varied between $10^{-9}$ to $10^5$ by factors of ten. The best parameters of $\sigma$ and $C$ are obtained using 5-fold cross validation.

Decision Trees  As mentioned in section 2.2.1.7 of chapter 2, decision trees (Breiman et al., 1984; Quinlan, 1986, 1993) recursively partition the data based on its features to construct a tree for the purpose of improving prediction accuracy. The recursion stops when data subsets include only instances of the same class. Based on the favorable predictive performance, obtained from preliminary runs, in this study we chose CART as our decision tree method.
Random Forest  As mentioned in section 2.2.1.8 of chapter 2, Random Forest (RF) generates multiple decision trees and aggregates their results (Breiman, 2001). Each tree is trained on a bootstrap sample of the training data. To classify an example, decisions (votes) of all trees in the forest are aggregated and the majority voting of the trees is considered as the output of the classifier. In our experiments we grow 100 trees in the forest and the number of features at each split is chosen as the square root of the number of features.

3.4 Experiments

In our experiments, we have used both synthetic and real-world datasets and demonstrate the effectiveness of the proposed Tree-Lasso by carrying out the following comparisons.

• We show that stability behavior of Tree-Lasso is better compared to several baseline feature selection algorithms namely, t-Test, IG, ReliefF and Lasso.

• We compare the predictive performance of Tree-Lasso with other baseline feature selection algorithms by using them with different classifiers namely, logistic regression, naive Bayes, SVM, decision tree and Random Forest and show that under the constraint of stable feature selection, Tree-Lasso prediction performance is constantly better than that of other baselines.

• As an extra evaluation, we compare stability and predictive performance of Tree-Lasso (with built-in logistic regression) with Random Forest, which is a well-known embedded type feature selection method in machine learning and show that Tree-Lasso achieves better results in terms of both stability and prediction.

• We show that the features obtained using Tree-Lasso for real-world datasets are consistent with the well-known risk factors used by experts in clinical domain.
3.4. Experiments

3.4.1 Datasets

3.4.1.1 Synthetic Datasets

To illustrate the stability behavior of different feature selection algorithms, we generate a synthetic data where features are grouped hierarchically in a tree structure. To keep the matter simple, we confine ourselves to shallow 2-level trees. In order to generate data, we fix the number of leaf nodes, referred to as groups. Each such group (or node) contains a set of variables such that the variables within a group are correlated to one another whilst uncorrelated with the variables from other groups. This is done by defining a correlation matrix $C$ such that its $(i, j)$-th element contains the correlation coefficient between $i$-th and $j$-th variables. Formally, the correlation matrix is defined as

$$C_{i,j} = \begin{cases} 
\rho & i, j \text{ belong to the same group} \\
0 & \text{otherwise} 
\end{cases}$$ (3.5)

In order to generate upper layers of the tree, correlation between each pair of groups from the lower layer is calculated. For each group, we find the other group having the highest correlation with it and connect them to construct the upper layer of the tree. Given the above correlation matrix, the feature vector $x^{(i)}$ is generated using a multivariate normal distribution having mean zero and covariance $C$, i.e. $x^{(i)} \sim \mathcal{N}(0, C)$, $i = 1, \ldots, N$. The true parameter vector $\beta$ is

$$\beta = (0, 0, \ldots, 0, 1, 1, \ldots 1)_{\text{50 times, 50 times}}.$$ 

Given $i$-th data vector $x^{(i)}$ and the weight vector $\beta$, the label is generated as following

$$y^{(i)} = \text{sign} \left( \beta^T x^{(i)} + \epsilon \right), \quad \epsilon \sim \mathcal{N}(0, 0.1).$$ (3.6)

For the results reported in this chapter, we simulate 100 variables, grouped into 4 leaf nodes, i.e. the first 25 variables are part of group-1, the next 25 variables are part of group-2 and so on. Using these features, we generate 200 data samples. We generate two such datasets: one with low correlation ($\rho = 0$) and the other with high correlation ($\rho = 0.8$).
3.4. Experiments

Table 3.1: Statistics of the real-world datasets.

<table>
<thead>
<tr>
<th>Name</th>
<th># Samples</th>
<th># ICD 10 Codes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cancer</td>
<td>4293</td>
<td>439</td>
</tr>
<tr>
<td>AMI</td>
<td>2941</td>
<td>528</td>
</tr>
</tbody>
</table>

3.4.1.2 Real-World Datasets

For experiments with real-world data, we used two hospital patient cohorts: Cancer and Acute Myocardial Infarction (AMI). A summary of statistics of the two datasets is provided in Table 3.1 and their details are described below:

**Cancer Dataset**  This dataset is obtained from a large regional hospital in Australia\(^2\). There are eleven different cancer types in this data recorded from patients visiting the hospital during 2010-2012. Patient data is acquired from Electronic Medical Records (EMR). The dataset consists of 4293 patients with 3867 variables including International Classification of Disease 10 (ICD-10), procedure and diagnosis related Group (DRG) codes of each patient as well as demographic data (age, gender and postcode). In this dataset, the number of patients who survived within 1 year after diagnosis of cancer is 3383 and the number of those who died within 1 year is 910. Among all the features in this dataset, disease conditions of the patients are described using 439 ICD codes. As only these features have hierarchical tree structures and can be modeled using Tree-Lasso, we incorporate them in our experiments. Using this dataset, our goal is to predict 1 year mortality of patients while ensuring the stable feature sets. We note that feature stability is crucial for clinical decision making towards cancer prognosis.

**Acute Myocardial Infarction (AMI) Dataset**  This dataset is also obtained from the same hospital in Australia. It involves patients admitted with AMI conditions and discharged later between 2007-2011. The task is to predict if a patient will be re-admitted to the hospital within 30 days after discharge. The dataset consists of 2941 patients with 2504 variables include International Classification of

\(^2\)Ethics approval obtained through university and the hospital – 12/83.
3.4. Experiments

Disease 10 (ICD-10), procedure and diagnosis-related Group (DRG) codes of each admission; details of procedures; and departments involved in the patient’s care. Other variables include demographic data and details of access to primary care facilities. In this data set the number of patients who are readmitted to the hospital within 30 days after discharge is 242 and those who are not readmitted are 2699. In this dataset disease conditions of the patients are described using 528 ICD codes, which have hierarchical tree structure and can be modeled using Tree-Lasso. We incorporate these features in our experiments.

3.4.2 Evaluation Measures

Stability Measures In order to compare the stability of Tree-Lasso with other feature selection algorithms, we use two different stability measures, Spearman’s rank correlation coefficient (SRCC), and Jaccard similarity measure (JSM). These stability measures are described in section 2.2.4.1 of chapter 2.

Classification Performance Measure To compare the prediction performance of Tree-Lasso with other feature selection methods, we use the area under the receiver operating characteristic (ROC) curve, further abbreviated as AUC (Egan, 1975). Due to its robustness across both balanced and imbalanced datasets, AUC is commonly used in clinical decision making and is becoming increasingly popular in pattern recognition community (Davis and Goadrich, 2006; Wu et al., 2010).

3.4.3 Experimental Settings

The Tree-Lasso model built in our experiments is based on ICD-10 codes that have intrinsic hierarchical structure. For example, Figure 3.3, shows part of ICD-10 codes related to ischemic and pulmonary heart diseases (ICD-10 codes: I20 up to I27). According to ICD-10 hierarchy, these diseases are classified into several groups and each of these groups are further classified into several subgroups, giving rise to a tree-structure. Consequently, at the leaf node, we have individual features that are grouped progressively in parent nodes as we move up in the ICD tree. So, feature
3.4. Experiments

Table 3.2: Comparison of Tree-Lasso with baselines in terms of different stability measures for both synthetic and real-world datasets.

<table>
<thead>
<tr>
<th></th>
<th>Synthetic data ($\rho = 0$)</th>
<th>Synthetic data ($\rho = 0.8$)</th>
<th>Cancer dataset</th>
<th>AMI dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>T-test</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRCC</td>
<td>0.2853</td>
<td>0.4878</td>
<td>0.1204</td>
<td>0.4537</td>
</tr>
<tr>
<td>JSM</td>
<td>0.5088</td>
<td>0.5656</td>
<td>0.4553</td>
<td>0.5258</td>
</tr>
<tr>
<td><strong>IG</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRCC</td>
<td>0.2528</td>
<td>0.5030</td>
<td>0.2164</td>
<td>0.5150</td>
</tr>
<tr>
<td>JSM</td>
<td>0.6621</td>
<td>0.6453</td>
<td>0.7863</td>
<td>0.6378</td>
</tr>
<tr>
<td><strong>ReliefF</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRCC</td>
<td>0.5177</td>
<td>0.5373</td>
<td>0.5362</td>
<td>0.5275</td>
</tr>
<tr>
<td>JSM</td>
<td>0.5184</td>
<td>0.5008</td>
<td>0.7576</td>
<td>0.6635</td>
</tr>
<tr>
<td><strong>Lasso</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRCC</td>
<td>0.1278</td>
<td>0.4374</td>
<td>0.2330</td>
<td>0.4738</td>
</tr>
<tr>
<td>JSM</td>
<td>0.5773</td>
<td>0.4080</td>
<td>0.5542</td>
<td>0.5028</td>
</tr>
<tr>
<td><strong>Tree-Lasso</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRCC</td>
<td><strong>0.6670</strong></td>
<td><strong>0.6282</strong></td>
<td><strong>0.7458</strong></td>
<td><strong>0.6274</strong></td>
</tr>
<tr>
<td>JSM</td>
<td><strong>0.7830</strong></td>
<td><strong>0.8777</strong></td>
<td><strong>0.8850</strong></td>
<td><strong>0.7147</strong></td>
</tr>
</tbody>
</table>

encoding is such that if a patient record contains all the offspring of some parent node, we also set the parent node to 1 along with setting all offspring nodes to 1.

To assess the variability of the experiment, we randomly divide our data into two sets: 70% of our data is considered as training set and 30% as test set. For each random split, we further split the training set into two sets: derivation set (80% of the training set) and a validation set (20% of the training set). In order to be able to select the best features, this second split is randomly repeated 100 times to generate 100 sets of derivation-validation pairs. We train all the models using each derivation set while selecting the best model parameters through model performance on the corresponding validation set. This process provides us 100 feature sets. Using the ensemble of 100 feature sets, we empirically estimate the probability of presence for each feature. Given these probability estimates, we re-train a model using derivation dataset and including only those features that occur with at least probability $p$ (a threshold that we gradually increase). Using 30% held out test set, the predictive performance of the model is evaluated using AUC while the stability of the model is computed using SRCC, and JSM.
3.4. Experiments

Figure 3.5: Stability results for synthetic dataset ($\rho = 0.8$) for 10 selected features. In these plots we use a threshold $T = 0.5$ and features that are selected with a probability more than $T$ are shown in black color while others in gray color.

3.4.4 Experimental Results

In this section, we evaluate the stability performance of Tree-Lasso and compare it with other baseline feature selection methods. We also investigate the classification performance of each algorithm with different classifiers and measure their performance in terms of AUC. In addition, we study the consistency of the features obtained using Tree-Lasso for Cancer and AMI datasets with well-known risk factors in clinical domain.

3.4.4.1 Stability Performance of Feature Selection Methods

Table 3.2 shows the stability results in terms of SRCC and JSM, for Tree-Lasso and baseline algorithms. In case of synthetic data, Tree-Lasso achieves the best
3.4. Experiments

Figure 3.6: Stability results for Cancer dataset for 10 selected features. In these plots we use a threshold $T = 0.5$ and features that are selected with a probability more than $T$ are shown in black color while others in gray color.

stability performance in terms of both SRCC and JSM. The high value of SRCC in Tree-Lasso means that for different training sets the ranks of features does not vary a lot. On the other hand, the high value of JSM means that the feature set selected does not change significantly. In terms of JSM Tree-Lasso achieves the stability of 0.7830 (when $\rho = 0$) and 0.8777 (when $\rho = 0.8$) for synthetic data which is higher compared to the other methods.

In case of Cancer dataset, Tree-Lasso again shows the best stability performance, in terms of SRCC. The SRCC results for T-test, IG and Lasso is poor, whilst that of ReliefF is somewhat average. When we turn to JSM Tree-Lasso is again the winner (0.7910) followed by IG (0.7863). The other methods achieve JSM value of 0.7576 (ReliefF), 0.5542 (Lasso) and 0.4553 (T-test).

For the AMI dataset, Tree-Lasso is once again the winner with SRCC=0.6274 and JSM=0.7147, followed by ReliefF and IG. The SRCC and JSM for T-test and Lasso
3.4. Experiments

are significantly lower.

In order to have a better understanding of the stability behavior of different feature selection methods for the datasets that contain correlated variables i.e. synthetic data ($\rho = 0.8$), Cancer data and AMI data, we show top ten features selected by various methods in different splits of data in Figures 3.5-3.7, respectively. From these figures not only we can visually compare the stability of different methods but also can infer which features are considered important by each algorithm. To better distinguish between stable features in these plots we use a threshold $T$ and features that are selected with a probability more than $T$ are shown in black color while others are in gray color. So more stable feature selection methods will have more number of black lines and less number of gray points. In our experiments, we set $T = 0.5$.

In Figure 3.5 (results for synthetic data, $\rho = 0.8$), where features within a group are highly correlated to one another, as expected Lasso shows an unstable behavior in selecting features. On the other hand, Tree-Lasso is the most stable algorithm. Moreover, it could correctly infer the true features of the model. In this dataset, ReliefF, IG and T-test are in the next stages of stability after Tree-Lasso. However, they are unable to select true features of the model.

Figures 3.6 and 3.7 show the stability behavior of each feature selection method on Cancer and AMI datasets. As it is illustrated in Figure 3.6 for Cancer dataset, again Tree-Lasso is the winner followed by IG and ReliefF. T-test and Lasso show the least stable behavior in selecting features in this dataset. The visual inspection of this figure also shows that the features selected by Tree-Lasso, IG and ReliefF are approximately similar. For AMI dataset (Figure 3.7), Tree-Lasso shows the best stability followed by ReliefF and IG. Again, Lasso achieves the least stability followed by T-test.

3.4.4.2 Classification Performance

In order to compare discrimination performance of Tree-Lasso with other baseline feature selection methods, we apply the features obtained using each feature selection algorithm to different classifiers e.g. logistic regression (LR), naive Bayes
3.4. Experiments

Figure 3.7: Stability results for AMI dataset for 10 selected features. In these plots we use a threshold $\mathcal{T} = 0.5$ and features that are selected with a probability more than $\mathcal{T}$ are shown in black color while others are in gray color.

(NB), SVM, decision trees (DT) and Random Forest (RF). As we explained in section 3.4.3, after estimating the probability of presence for each feature, we re-train the model using derivation set and include only those features that occur with at least probability $p$ (a threshold that we gradually increase). In our experiments we consider features with $p = 0.6$ to 1 with a step of 0.1. This is done to show the prediction performance under average-to-high stability constraints. We evaluate the predictive performance of each method by using 30% held out set and report it in terms of AUC.

The classification performance of various algorithms is shown in Figures 3.8-3.11. As it can be seen from these figures, irrespective of the classifier type used, AUC of Tree-Lasso is always the best and in most of the cases followed by Lasso. In terms of classifier used for each feature selection method, we can see that on average the best predictive performance is obtained using Random Forest followed by SVM and logistic regression. In addition, when we increase the stability threshold from 0.6 to
3.4. Experiments

Figure 3.8: Predictive performance of different classification methods coupled with each feature selection method for synthetic data ($\rho = 0$).

1, the AUC performance of Tree-Lasso remains stable. However, the performance of other algorithms varies a lot and that of Lasso drops suddenly. The sudden drop in performance of Lasso is due to underfitting caused by its inability to select sufficient number of stable features.

3.4.4.3 Comparison With Random Forest (as an Embedded Feature Selection Method)

Random Forest is a promising classification method that can perform feature selection and prediction, simultaneously. In order to study and compare the stability and predictive performance of Tree-Lasso with Random Forest, we grow 100 trees in the forest and the number of features at each split is chosen as the square root of the number of features. The experimental settings is identical to the settings described in section (3.4.3). Table 3.3 shows the stability results for Tree-Lasso and Random Forest. As it can be seen from this table, for all datasets (synthetic and real) the stability of Tree-Lasso is better than Random Forest, in terms of SRCC and JSM.
### 3.4. Experiments

The other comparison between Tree-Lasso and Random Forest is based on their predictive performance. To this end, obtaining the probability of presence of each feature as described in section (3.4.3), the model is trained again using derivation set by including only those features that occur with at least probability \( p \) (a threshold that we gradually increase). In our experiments we consider features with \( p = 0.6 \) to 1 with a step of 0.1. Using 30% held out test set, the predictive performance of the model is reported in terms of AUC. Figures 3.12 and 3.13 show the predictive performance of Tree-Lasso compared to Random Forest. As it can be seen from these figures, the predictive performance of Random Forest and Tree-Lasso are approximately the same when features with average stability are used (with \( p = 0.6 \) and 0.7). However, when the stability threshold is increased, the AUC of Random Forest declines steadily while that of Tree-Lasso remains stable. In synthetic data with \( \rho = 0 \), where the average correlation between groups of variables is around zero, reduction of AUC performance in Random Forest is less compared to other datasets (with correlated groups of variables). This shows that although Random Forest is a
3.4. Experiments

Figure 3.10: Predictive performance of different classification methods coupled with each feature selection method for Cancer data.

Figure 3.11: Predictive performance of different classification methods coupled with each feature selection method for AMI data.
3.4. Experiments

Table 3.3: Comparison of Tree-Lasso with Random Forest in terms of different stability measures for both synthetic and real-world datasets.

<table>
<thead>
<tr>
<th></th>
<th>Synthetic data $(\rho = 0)$</th>
<th>Synthetic data $(\rho = 0.8)$</th>
<th>Cancer dataset</th>
<th>AMI dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Forest</td>
<td>SRCC 0.4972</td>
<td>0.2635</td>
<td>0.2273</td>
<td>0.3175</td>
</tr>
<tr>
<td></td>
<td>JSM  0.6535</td>
<td>0.4024</td>
<td>0.5624</td>
<td>0.5276</td>
</tr>
<tr>
<td>Tree-Lasso</td>
<td>SRCC 0.6670</td>
<td>0.6282</td>
<td>0.7458</td>
<td>0.6274</td>
</tr>
<tr>
<td></td>
<td>JSM  0.7830</td>
<td>0.8788</td>
<td>0.8850</td>
<td>0.7147</td>
</tr>
</tbody>
</table>

Figure 3.12: Predictive performance of Tree-Lasso compared to Random Forest for synthetic datasets. (a) $\rho = 0$ (b) $\rho = 0.8$
3.4. Experiments

Figure 3.13: Predictive performance of Tree-Lasso compared to Random Forest for real-world datasets. (a) Cancer data. (b) AMI data

good classifier and shows good performance in many applications, its performance degrades in presence of correlated features. On the other hand, Tree-Lasso shows acceptable predictive performance in presence of correlated variables.

3.4.4.4 Comparison with Expanded-Lasso

One way to deal with instability of Lasso in selecting informative features can be through expanding the selected feature set by including the features that are unselected but correlated to one or more features in the selected set. We refer to this heuristic-based method as *Expanded-Lasso* and compare its feature selection stability and predictive performance with those of Lasso and Tree-Lasso. Using our synthetic and real-world datasets (same as used above), we split data into training and test sets. The model is trained on the training set and evaluated on the test set. In order to specify the tuning parameters of each method, we use 5-fold cross validation. Based on the fact that for Expanded-Lasso method we need to specify the level of correlation between selected and unselected features, in our experiments
Table 3.4: Comparison of Tree-Lasso and Lasso with Expanded-Lasso in terms of feature stability and predictive performance on synthetic data ($\rho = 0$).

<table>
<thead>
<tr>
<th>Method</th>
<th>Stability</th>
<th>Predictive performance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>JSM</td>
<td>SRCC</td>
</tr>
<tr>
<td>Expanded_Lasso</td>
<td>0.6693</td>
<td>0.2435</td>
</tr>
<tr>
<td>(Threshold=0.7)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Expanded_Lasso</td>
<td>0.6247</td>
<td>0.1976</td>
</tr>
<tr>
<td>(Threshold=0.8)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Expanded_Lasso</td>
<td>0.5925</td>
<td>0.1624</td>
</tr>
<tr>
<td>(Threshold=0.9)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lasso</td>
<td>0.5773</td>
<td>0.1278</td>
</tr>
<tr>
<td>Tree-Lasso</td>
<td>0.7830</td>
<td>0.6670</td>
</tr>
</tbody>
</table>

we use three different thresholds, i.e. 0.7, 0.8, and 0.9.

Tables 3.4 to 3.7 compare Expanded-Lasso with Tree-Lasso and Lasso in terms of feature stability and predictive performance. As seen from the tables, in terms of both JSM and SRCC Tree-Lasso is the winner, followed by Expanded-Lasso. Turning to predictive performance, again Tree-Lasso achieves the best AUC and Expanded-Lasso is the runner-up. The reason behind better performance of Tree-Lasso compared to Expanded-Lasso is because of its ability to use intrinsic hierarchical information (correlation) between ICD-10 features, whereas Expanded-Lasso needs to estimate this information. However, in problems where no information about hierarchical structure of the features is available, Expanded-Lasso can be used as a remedy to increase the stability of Lasso in selecting informative features.

### 3.4.4.5 Risk Factors Obtained using Tree-Lasso

Identifying stable features (risk factors) can assist clinical decision making towards accurate medical prognosis. In Tables 3.8 and 3.9, we show that risk factors selected using Tree-Lasso (with high probability) for both Cancer and AMI datasets are consistent with well-known risk factors used in clinical domain (Brown et al., 2013; Desai et al., 2009; Dunlay et al., 2012; Krumholz et al., 2001; Laird et al., 2013;
### Table 3.5: Comparison of Tree-Lasso and Lasso with Expanded-Lasso in terms of feature stability and predictive performance on synthetic data ($\rho = 0.8$).

<table>
<thead>
<tr>
<th>Method</th>
<th>Stability</th>
<th>Predictive performance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>JSM</td>
<td>SRCC</td>
</tr>
<tr>
<td>Expanded_Lasso (Threshold=0.7)</td>
<td>0.5753</td>
<td>0.5173</td>
</tr>
<tr>
<td>Expanded_Lasso (Threshold=0.8)</td>
<td>0.5162</td>
<td>0.5027</td>
</tr>
<tr>
<td>Expanded_Lasso (Threshold=0.9)</td>
<td>0.4587</td>
<td>0.4736</td>
</tr>
<tr>
<td>Lasso</td>
<td>0.4080</td>
<td>0.4374</td>
</tr>
<tr>
<td>Tree-Lasso</td>
<td><strong>0.8777</strong></td>
<td><strong>0.6282</strong></td>
</tr>
</tbody>
</table>

### Table 3.6: Comparison of Tree-Lasso and Lasso with expanded-Lasso in terms of feature stability and predictive performance on Cancer data.

<table>
<thead>
<tr>
<th>Method</th>
<th>Stability</th>
<th>Predictive performance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>JSM</td>
<td>SRCC</td>
</tr>
<tr>
<td>Expanded_Lasso (Threshold=0.7)</td>
<td>0.6425</td>
<td>0.3836</td>
</tr>
<tr>
<td>Expanded_Lasso (Threshold=0.8)</td>
<td>0.6027</td>
<td>0.3178</td>
</tr>
<tr>
<td>Expanded_Lasso (Threshold=0.9)</td>
<td>0.5726</td>
<td>0.2763</td>
</tr>
<tr>
<td>Lasso</td>
<td>0.5542</td>
<td>0.2330</td>
</tr>
<tr>
<td>Tree-Lasso</td>
<td><strong>0.8850</strong></td>
<td><strong>0.7458</strong></td>
</tr>
</tbody>
</table>
Table 3.7: Comparison of Tree-Lasso and Lasso with expanded-Lasso in terms of feature stability and predictive performance on AMI data.

<table>
<thead>
<tr>
<th>Method</th>
<th>Stability</th>
<th>Predictive performance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>JSM</td>
<td>SRCC</td>
</tr>
<tr>
<td>Expanded_Lasso (Threshold=0.7)</td>
<td>0.6763</td>
<td>0.5531</td>
</tr>
<tr>
<td>Expanded_Lasso (Threshold=0.8)</td>
<td>0.6228</td>
<td>0.5129</td>
</tr>
<tr>
<td>Expanded_Lasso (Threshold=0.9)</td>
<td>0.5727</td>
<td>0.4836</td>
</tr>
<tr>
<td>Lasso</td>
<td>0.5028</td>
<td>0.4738</td>
</tr>
<tr>
<td>Tree-Lasso</td>
<td>0.7147</td>
<td>0.6274</td>
</tr>
</tbody>
</table>

Pfeiffer et al., 2013; Ramchandran et al., 2013; Rana et al., 2014; Zhao and Weng, 2011). Columns 1 and 2 of the tables show the risk factors ICD-10 code and names, respectively and column 3 shows the probability of presence of the risk factor in each split of data. For example, acute respiratory failure (J96.0) in Cancer dataset with probability equal to one means that this important risk factor (based on clinical research papers) is also considered important by Tree-Lasso and is selected in every splits of the data.

We also study why it matters that selected features be stable when the prediction accuracy is good. To this end, we investigate importance of stability in two ways:

**Consistency Over Time:** In some applications such as healthcare, it is important that the obtained features to be interpretable over time. For example, we need to attribute the disease of a patient to certain risk factors consistently over time. However, in presence of correlated features, feature selection methods such as Lasso may select some features off and on, causing confusions and suspicions about the model. As an example, consider correlated features I20, I21 and I25 in AMI data that are related to ischaemic heart disease and we expect that these features are always selected together. However, as it is shown in Table 3.10, although Lasso selects I21 and I25 consistently, it selects I20 only 38% of the times. This may lead to a confusion about predictive value of I20 for AMI related hospital readmissions.
Table 3.8: Well-known risk factors for cancer reported by clinicians or other research papers, which are also obtained by Tree-Lasso with high probability.

<table>
<thead>
<tr>
<th>ICD-10 code</th>
<th>Risk factor</th>
<th>Probability of presence (Tree-Lasso)</th>
</tr>
</thead>
<tbody>
<tr>
<td>D24</td>
<td>Benign neoplasm of breast (Pfeiffer et al., 2013)</td>
<td>1.00</td>
</tr>
<tr>
<td>R11</td>
<td>Nausea and vomiting (Laird et al., 2013; Zhao and Weng, 2011)</td>
<td>1.00</td>
</tr>
<tr>
<td>R06.0</td>
<td>Dyspnea (Laird et al., 2013; Maltoni et al., 2012)</td>
<td>1.00</td>
</tr>
<tr>
<td>R63.0</td>
<td>Anorexia (Maltoni et al., 2012)</td>
<td>0.98</td>
</tr>
<tr>
<td>R53</td>
<td>Fatigue (Zhao and Weng, 2011)</td>
<td>0.95</td>
</tr>
<tr>
<td>K59.0</td>
<td>Constipation (Laird et al., 2013)</td>
<td>0.91</td>
</tr>
<tr>
<td>R19.7</td>
<td>Diarrhea (Laird et al., 2013)</td>
<td>0.90</td>
</tr>
<tr>
<td>F32</td>
<td>Depression (Laird et al., 2013; Zhao and Weng, 2011)</td>
<td>0.87</td>
</tr>
<tr>
<td>G47.0</td>
<td>Insomnia (Laird et al., 2013)</td>
<td>0.85</td>
</tr>
<tr>
<td>E11</td>
<td>Type II diabetes mellitus (Yuhara et al., 2011; Zhao and Weng, 2011)</td>
<td>0.85</td>
</tr>
</tbody>
</table>

**Choosing the Best Explanatory Features:** By using an stable feature selection method, our goal is to choose the best explanatory features. For example, in AMI dataset features I50 and I51 are both related to heart failure and so correlated. However, I50 is a basic feature used to code “heart failure” and I51 is a more specialized feature that gives details of heart failure i.e. “complications of heart disease”. Based on the features used by clinicians I50 is more important feature than I51 and selecting latter where the former is not selected would be meaningless. As it can be seen from Table 3.10, Lasso chooses I51 and ignores I50 (that is more important feature). However, this is not the case in Tree-Lasso.
Table 3.9: Well-known risk factors of readmission after AMI reported by clinicians or other research papers, which are also obtained by Tree-Lasso with high probability.

<table>
<thead>
<tr>
<th>ICD-10 code</th>
<th>Feature’s name</th>
<th>Probability of presence (Tree-Lasso)</th>
</tr>
</thead>
<tbody>
<tr>
<td>I50</td>
<td>Heart failure (Brown et al., 2013; Dunlay et al., 2012; Grady et al., 2012; Rana et al., 2014)</td>
<td>1.00</td>
</tr>
<tr>
<td>N17</td>
<td>Acute renal disorder (Dunlay et al., 2012; Grady et al., 2012; Rana et al., 2014)</td>
<td>1.00</td>
</tr>
<tr>
<td>I46</td>
<td>Cardiac arrest Krumholz et al. (2001)</td>
<td>1.00</td>
</tr>
<tr>
<td>I20</td>
<td>Angina pectoris (Brown et al., 2013; Grady et al., 2012)</td>
<td>1.00</td>
</tr>
<tr>
<td>I21</td>
<td>Acute myocardial infarction (Brown et al., 2013)</td>
<td>1.00</td>
</tr>
<tr>
<td>I25</td>
<td>Chronic ischaemic heart disease (Brown et al., 2013)</td>
<td>1.00</td>
</tr>
<tr>
<td>J18</td>
<td>Pneumonia (Dunlay et al., 2012; Grady et al., 2012)</td>
<td>0.98</td>
</tr>
<tr>
<td>E11</td>
<td>Type II diabetes mellitus (Grady et al., 2012)</td>
<td>0.95</td>
</tr>
<tr>
<td>I10</td>
<td>Hypertension (Brown et al., 2013; Desai et al., 2009; Krumholz et al., 2001)</td>
<td>0.90</td>
</tr>
<tr>
<td>E83.4</td>
<td>Cardiorespiratory failure (Dunlay et al., 2012)</td>
<td>0.87</td>
</tr>
<tr>
<td>R07</td>
<td>Pain in chest (Krumholz et al., 2001)</td>
<td>0.84</td>
</tr>
<tr>
<td>E78.0</td>
<td>Hypercholesterolemia (Desai et al., 2009; Rana et al., 2014)</td>
<td>0.80</td>
</tr>
</tbody>
</table>
3.5 Conclusion

In this chapter, we proposed a framework that performs stable clinical prediction using Tree-Lasso – a supervised learning model that is used when features are hierarchical in nature and form a tree structure. We compared its stability and prediction performance with other feature selection algorithms, T-test, Information Gain, ReliefF and Lasso. Using a synthetic and two real-world datasets (Cancer and Acute Myocardial Infarction), we showed that our proposed method is significantly more stable than Lasso and comparable to other methods e.g. Information Gain, ReliefF and T-test. We further showed that, using different types of classifiers such as logistic regression, Naive Bayes, support vector machines, decision trees and Random Forest, the classification performance of our method is comparable to Lasso and better than other methods. Our result has implications in identifying stable risk factors for many healthcare problems and therefore assists clinical decision making towards accurate medical prognosis.

The key assumption made in this work was that features have hierarchical structure, which is valid when using ICD-10 codes as features. However, there could be situations where we need to work with other types of data such as age, sex and pathology results that do not have a tree structure. In this situation it would be worth developing other methods that have ability to perform stable feature selection.
in general context. Several new techniques pursuing this objective are considered in the following chapters.
Chapter 4

Stable Clinical Prediction by Supervised Feature Grouping

In the previous chapter, we proposed a method based on Tree-Lasso for stable feature selection exploiting hierarchical structure of ICD-10 codes available for EMR data. However, in many situations, features may not have such tree structure. For example, EMR data also consist of other types of variables such as age, sex or pathological results that do not have tree structure and so Tree-Lasso cannot be applied to stabilize these type of features. For such a high dimensional data, it is common to find groups of correlated features. Often these groups are consistent to the variation of training data. Therefore, group feature selection can be used as a solution to increase stability of feature selection algorithms.

Building clinical prediction models from high dimensional data containing long list of variables, such as EMR, often require sparsity inducing feature selection. In majority of these problems, Lasso, because of its convexity, is the primary regularizer of choice (Ye and Liu, 2012). However, Lasso has been known to cause instability in selecting features, especially in clinical data (Austin and Tu, 2004). This problem is even aggravated in EMR data that contain groups of highly correlated variables (Yuan and Lin, 2006; Zhao and Yu, 2006). The Lasso’s instability in selecting features is due to its tendency to select randomly one feature from a group of correlated features. Small changes in data result in a significant change in selected features leading to unstable models.
4.1 Methodology

Addressing this gap, we propose a framework to improve the stability of Lasso by grouping correlated features and selecting informative groups instead of each individual feature. Feature grouping is learned within the model using supervised data and therefore is aligned with prediction goal. To this end, we learn a matrix $G$, where each column of $G$ represents a group such that if a feature $p$ belongs to a group $k$ then $G_{ik} = 1$, otherwise 0. Since learning a binary matrix $G$ requires integer programming and is computationally expensive, we relax $G$ to be non-negative. An added advantage of using non-negativity is that each column of $G$ now contains real-valued non-negative values, which can be interpreted as weight/importance of a feature in the group. We also impose orthogonality constraint on $G$ to ensure that a feature is part of only one group. The proposed model is formulated as a constrained optimization problem combining both feature grouping and feature selection in a single step. To solve this problem, we propose an efficient iterative algorithm with theoretical guarantees for its convergence. We demonstrate the usefulness of the model via experiments on both synthetic and real datasets. We compare our model with several other baseline models demonstrating its superiority for both feature stability and prediction performance.

This chapter is organized as follows. Section 3.3 presents our new predictive grouping technique, called predictive grouping elastic net (pg-EN) that can perform supervised feature grouping and feature selection to improve stability of the model. Further, this section presents an iterative solution to optimize the cost function of pg-EN with theoretical guarantees for its convergence. Section 4.2 describes the experimental results conducted for both synthetic and real-world datasets and compares them with the state-of-the-art methods along with further discussion and statistical analysis. Concluding remarks are given in Section 4.3.

4.1 Methodology

4.1.1 Predictive Grouping Elastic net

In this section, we propose our framework that can simultaneously group correlated variables and select the best group of variables in a supervised manner. We consider
4.1. Methodology

Features in matrix $D$

Figure 4.1: Graphical representation of the feature grouping and weight assignment procedure in the proposed model. The yellow icons in matrix $G$ are representative of non-negative values. Features in the same group are combined together by multiplying $D$ by $G$. This gives us super-features $X$, which are representatives of the correlated features in each group. The algorithm finds the coefficients $\beta$ for super-features $X$.

In presence of correlated features, our goal is to identify and group them. By doing this, not only the stability of feature selection can be improved (Jörnsten and Yu, 2003), but also the estimator’s variance can be reduced (Shen and Huang, 2010). This reduction of variance leads to better prediction performance. To this end, we learn a matrix $G \in \mathbb{R}^{p \times K}$, where each column of $G$ represents a feature grouping such that if feature $p$ is part of group $k$, then we have $G_{pk} = 1$ and 0 otherwise. By restricting $G$ to be a binary matrix, the optimization procedure of our model would...
be an integer programming problem, which is NP-hard. Therefore, for simplicity in optimization process, we relax $G$ to be non-negative. Moreover, the non-negative values in $G$ can be interpreted as weight of each feature within each group and can be used to identify the importance of each feature in its group. In Figure 4.1, the procedure of grouping features and assigning weights to each group is shown graphically. As seen, the data matrix $D$ has unknown groups of highly correlated features. The proposed method can find these correlated groups and build the grouping matrix $G$. Multiplying $D$ by $G$ obtains the “super-features” $X \in \mathbb{R}^{n \times K}$ ($K < p$), which are the representatives of the correlated features in each group. The proposed algorithm finds the coefficient vector $\beta \in \mathbb{R}^{K \times 1}$, for super-features $X$. The weight vector $w \in \mathbb{R}^{p \times 1}$ for each feature in matrix $D$ can still be obtained as $w = G \times \beta$.

Using the above feature grouping scheme, we formulate the feature selection as an optimization problem with the following objective function:

$$
\min_{\beta, G \geq 0} \quad J(\beta, G) = \| y - DG\beta \|_2^2 + \lambda \Omega(\beta) + \delta \text{tr}(I - G^T G),
$$

(4.1)

where

$$
\Omega(\beta) = (1 - \alpha) \| \beta \|_1 + \frac{\alpha}{2} \| \beta \|_2^2, \quad \alpha \in [0, 1].
$$

The first term in $J(\beta, G)$ ensures model fitting; $\Omega(\beta)$ is a regularization term that prevents overfitting, and the last term with regularization parameter $\delta \in (0, 1)$ guarantees the orthogonality of the groups i.e. it ensures that each variable belongs only to one group. We refer to this model as predictive grouping Elastic net (pg-EN).

### 4.1.1.1 Optimization Algorithm

The cost function of pg-EN in (4.1) is convex for $\beta$ and $G$ individually, but not for both. Therefore, we do not expect an optimization algorithm to find a global minimum. Thus, we minimize the cost function via an iterative algorithm that updates $G$ and $\beta$ alternatively.

A step-by-step procedure for optimization of the proposed model is provided in Algorithm 4.1. The first step in the optimization algorithm is initialization of $G$. We can either randomly assign features to the groups or this can be done using $K$-
4.1. Methodology

Algorithm 4.1 Algorithm for solving pg-EN optimization problem.

- Initialize $G$ as a solution of a $K$-means to cluster features.
  - if $d_i \in k$ then $G_{ik} = 1$
  - else $G_{ik} = 0$
  - endif

- Hold $G$ fixed and solve equation 4.1 for $\beta$. Defining $X = DG$, that is, solve
  \[
  \arg\min_{\beta} J(\beta) = \frac{1}{2} \|y - X\beta\|_2^2 + \lambda(1 - \alpha) \|\beta\|_1 + \frac{\lambda\alpha}{2} \|\beta\|_2^2. 
  \]  
  (4.2)

- Hold $\beta$ fixed and solve equation 4.1 for $G$. If we define $A = \beta\beta^T$, $B = D^TD$ and $C = D^Ty\beta$ that is, solve
  \[
  \arg\min_G \left\{ \|y - DG\|_2^2 + \delta\text{tr}(I - G^TG) \right\}
  \]  
  (4.3)

means algorithm. We have compared the effect of these two types of initialization on the final predictive performance of the model in section B.1.4 of Appendix B. As the optimization procedure is iterative, in the second step we solve the equation (4.1) with respect to $\beta$ while fixing $G$. This leads to equation (4.2), which is identical to the Elastic net cost function and is solved using coordinate descent approach (see, for example, (Friedman et al., 2007)). In the next step, we optimize equation (4.1) with respect to $G$, when $\beta$ is fixed. This results in equation (4.3). To solve this equation, we use the multiplication update rule, which is adapted from the semi-NMF model (Ding et al., 2010). Thus, $G$ is updated using

\[
G_{ik} \leftarrow G_{ik} \sqrt{\frac{(B^-GA^+)_{ik} + (B^+GA^-)_{ik} + C^+_{ik} + \delta G_{ik}}{(B^-GA^+)_{ik} + (B^+GA^-)_{ik} + C^-_{ik}}}, \]  
(4.4)

where the positive and negative parts of a matrix $T$ are defined as $T^+_{ik} = 0.5(|T_{ik}| + T_{ik})$ and $T^-_{ik} = 0.5(|T_{ik}| - T_{ik})$. The details about the update rule for $G$ (equation (4.4)), is provided in section 4.1.1.3.
4.1 Methodology

4.1.1.2 Computational Complexity

As $K < p$, the computational complexity of pg-EN for Step 1, in presence of $K$ groups, is much smaller compared to standard Lasso. Computational complexity for Step 2 is of the order $m(K^2 + Kp^2 + pK^2 + pm + pK)$, where $m$ is the number of iterations of the algorithm. The empirical convergence of Algorithm 1 is shown in Figure 4.2 for two of the real-world datasets used in the chapter. As seen from the figure, the algorithm converges usually within $50 - 60$ iterations.

4.1.1.3 Theoretical Guarantees for Convergence

In this section we prove the convergence of the cost function in equation (4.1) under the update rules of (4.2) and (4.4). The non-negative constraint of $G$, lets us to proceed along the lines of non-negative matrix factorization (Lee and Seung, 2001) and semi-nmf (Ding et al., 2010) to prove its convergence. However, the main difference here is that our model contains supervised information $(y)$.

**Theorem 1.** 1) Fixing $G$, $J(\beta,G)$ is identical to Elastic net cost function and converges using coordinate descent or proximal gradient method. 2) Fixing $\beta$, the cost function $J(\beta,G)$, decreases monotonically under the update rule of (4.4) for $G$.

**Proof.** To prove part 1, see (Tseng, 2001) that discusses about the convergence prop-
4.1. Methodology

Properties of coordinate descent for convex problems or see (Wright and Nocedal, 1999) that discusses about the proximal gradient algorithms for solving convex optimization problems. To prove part 2, which is a constrained optimization problem we show two results: a) We show that at convergence, the solution of the update rule of (4.4) satisfies the Karush-Kuhn-Tucker condition. This is presented in Proposition 1. b) We show that the cost function \( J(\beta, G) \) converges under the update rule of (4.4). This is shown in Proposition 2.

**Proposition 1.** The limiting solution of the update rule in (4.4) satisfies Karush-Kuhn-Tucker condition.

*Proof.* See section A.1 of Appendix A.

**Proposition 2.** The cost function \( J \) is non-increasing under the update rule (4.4).

*Proof.* See section A.2 of Appendix A.

**Proposition 3.** Based on the objective function \( J(H) \) defined in (A.5) with non-negative matrices, the following

\[
Z(H, H') = -\sum_{i,k} 2C_{ik}^+ H_{ik}' \left(1 + \log \frac{H_{ik}}{H_{ik}'}\right) + \sum_{i,k} 2C_{ik}^- H_{ik}^2 + H_{ik}'^2 \sum_{i,k} \left(B_{ik} A_{ik}^+ H_{ik}' + (B_{ik} A_{ik}^- H_{ik}) \right) \quad (4.5)
\]

\[
+ \sum_{i,k} \frac{(B_{ik} A_{ik}^+)_k H_{ik}^2}{H_{ik}'} - \sum_{i,j,k,l} B_{ijk} A_{kl}^+ H_{il}' \left(1 + \log \frac{H_{jk} H_{il}}{A_{kl} H_{il}'}\right) + \sum_{i,k} \frac{(B_{ik} A_{ik}^-)_k H_{ik}^2}{H_{ik}'}
\]

\[
- \sum_{i,k} \delta H_{ik}^2 \left(1 + \log \frac{H_{ik}^2}{H_{ik}^2}\right),
\]

is an auxiliary function for \( J(H) \) and it is convex. Further, its global minimum is

\[
H_{ik} = \arg \min_H Z(H, H')
\]

\[
= H_{ik}' \sqrt{\frac{(B+H'A^+)_k + (B+H'A^-)_k + C_{ik}^+ - \delta H_{ik}'}{(B+H'A^+)_k + (B+H'A^-)_k + C_{ik}^-}}.
\]

*Proof.* See section A.3 of Appendix A.
4.2 Experiments

In this section, we compare the predictive performance of pg-EN with some baseline algorithms such as Ridge regression, Lasso, Elastic net, Oscar, $K$-means+Lasso, and $K$-means+GroupLasso on both synthetic and real-world datasets. In the following subsections, we first describe the datasets used in this chapter, then we briefly introduce the baseline algorithms and evaluation measures. Following that, we talk about experimental settings used for the evaluation of different methods and finally, we discuss the experimental results.

4.2.1 Datasets

4.2.1.1 Synthetic Datasets

To illustrate the stability and predictive performance of pg-EN, we consider three controlled scenarios using synthetic data. For the first two scenarios, the data is simulated from a linear regression model $y = Dw + \epsilon$, where $\epsilon$ is a noise drawn from a normal distribution with mean 0 and standard derivation $\sigma$, i.e. $\epsilon \sim \mathcal{N}(0, \sigma^2)$. In both of these scenarios, 100 datasets are generated and each dataset consists of a training set, a validation set and a test set. Each model is fit on the training set while tuning parameters are selected using the validation set. We use the test set to evaluate the performance of each model. We use the notation ././. to show the number of instances on the training, the validation and the test set. The details of data generation are as follows:

**Synthetic-I** We simulate 100 datasets, each having 100/100/400 instances. The true parameters are:

$$w = \left(3, \ldots, 3, 0, \ldots, 0\right)_{15}$$

and $\sigma = 15$. This is done to create a scenario of sparse prediction model where the last 25 features are irrelevant. Next, we create 3 feature groups among the first 15 features as $d_{im} = Z_1 + \epsilon_{im}$, $Z_1 \sim \mathcal{N}(0, 1)$, $m = 1, \ldots, 5$.
4.2. Experiments

Figure 4.3: Graphical illustration of correlation matrices used for synthetic datasets.

\[ d_{im} = Z + \epsilon_{im}, \quad Z \sim \mathcal{N}(0, 1), \quad m = 6, \ldots, 10, \]
\[ d_{im} = Z + \epsilon_{im}, \quad Z \sim \mathcal{N}(0, 1), \quad m = 11, \ldots, 15, \]
\[ d_{im} \sim \mathcal{N}(0, 1), \quad m = 16, \ldots, 40, \]

where \( \epsilon_i \) are independent identically distributed \( \mathcal{N}(0, 0.01) \), \( i = 1, \ldots, 15 \). In this scenario, there are three equally important groups with five members within each group. The features within each group are strongly correlated. We expect these features to cause instability in feature selection process. This dataset has been used before in (Bondell and Reich, 2008; Zou and Hastie, 2005). The empirical correlation matrix of this dataset is shown in Figure 4.3a.

**Synthetic-II** We simulate 100 datasets, each having 30/30/50 instances. In this simulation, we illustrate a situation of \( p > n \), with \( p = 500 \). The instances (rows of \( D \)) are iid from a \( \mathcal{N}(0, \Sigma) \) distribution, where \( \Sigma \) is a \( p \times p \) block diagonal matrix, which is defined as follows:

\[ \Sigma_{ij} = \begin{cases} 
1 & \text{if } i = j, \\
0.8 & \text{if } i \leq 50, j \leq 50, i \neq j, \\
0.8 & \text{if } 51 \leq i \leq 100, 51 \leq j \leq 100, i \neq j, \\
0 & \text{otherwise}
\end{cases} \quad (4.7) \]
and $\epsilon_i \sim \mathcal{N}(0, 2.5^2), i = 1, \ldots n$. The true parameters are:

$$w = \begin{pmatrix} 1, \ldots, 1, -1, \ldots, -1, 0, \ldots, 0 \end{pmatrix}.$$

In this scenario, there are two groups of 50 correlated features, which are associated with response. The remaining 400 features are uncorrelated and are not associated with response. Figure 4.3b shows the empirical correlation matrix of this dataset.

**Synthetic-III (Simulation of Microarray Dataset)** Analysis of microarray datasets is extremely useful for biomarker discovery and answering diagnosis and prognosis questions. In this section, we use a simulated dataset developed in (Di Camillo et al., 2012) to compare the stability and prediction performance of pg-EN with other baseline algorithms. The effect of heterogeneity and variability of synthetic microarray data consisting of two balanced groups of 50 subjects is simulated in this dataset. To this end, each subject is simulated using a regulatory network of $p = 10000$ genes using the simulator described in (Di Camillo et al., 2009). The topology of the network is specified by a connectivity matrix $W$, where $w_{ij}$ would be non zero if gene-product $j$ directly affects the expression of gene $i$. Following this, a population of $N = 1000$ instances is simulated as follows. Subjects are modeled as regulatory networks of $p = 10000$ nodes and the first generation of population consisted of $N$ individuals with identical connectivity matrix $W$ and with $p$ dimensional vectors of expression values obtained. The subsequent generations were produced by iteration of three steps: random pairing, mutation of a randomly chosen subsequent of subjects and selection of the surviving subjects. These steps were applied only to a sub-network size $p = 900$, indicated as $W_{900}$ in the following. These three steps are discussed in more details in (Di Camillo et al., 2012). When the base population was simulated, we define two groups of 500 subjects. The pathological condition is simulated by knocking out or knocking down six target hubs, which are defined as the genes with the highest out-degree and expression value at steady state higher than 0.88. Diseased subjects had 4, 5, or 6 genes belonging to $W_{900}$ that were knocked out or down. In our studies, we partitioned the two groups of 500 healthy and 500 diseased subjects into 10 balanced non-overlapping datasets of size 50 subjects.
4.2. Experiments

4.2.1.2 Real-World Datasets

For evaluating the performance of the pg-EN on real-world datasets, we conduct our experiments on the same datasets that were utilized for demonstrating Tree-Lasso in chapter 3 and are described in section 3.4.1.2. Further, we use a breast cancer dataset, collected by Van De Vijver et al. (2002) and consists of gene expression data for 8141 genes in 295 breast cancer tumors (87 metastatic and 217 non-metastatic). In this dataset, our aim is to predict the patients’ survival based on gene expression profiles.

As these datasets are imbalanced, we balance them by using multiple replicates of each positive sample while keeping all replicates in the same fold during cross validation. We have also compared the predictive performance of pg-EN with other baselines when the real-world datasets are not re-balanced in Tables B.9-B.11 of Appendix B.

4.2.2 Baselines

To compare the performance of pg-EN with other state-of-the-art algorithms we have used the following algorithms as baseline.

**Lasso**  This is a regularization method that is used to learn a regularized regression/classification model that is sparse in feature space (Tibshirani, 1996). We have introduced this method in section 2.2.2.4 of chapter 2. The solution of the Lasso’s optimization does not have a closed form and is usually found iteratively by minimizing the cost function using pathwise coordinate optimization (Friedman et al., 2007).

**Elastic net**  As we mentioned in section 2.2.2.4 of chapter 2, this method incorporates an $l_2$-norm penalty in the Lasso’s penalty term (Zou and Hastie, 2005) and can achieve an sparse model along with a tendency for correlated variables to yield similar regression coefficients.
4.2. Experiments

**K-means + Lasso** In this baseline, we first use K-means to cluster the features and assign them to different groups based on their correlation. When we prepare matrix $G \in \mathbb{R}^{p \times K}$ from the output of K-means. Each column of $G$ represents a group such that if feature $p$ is part of group $k$, then we have $G_{pk} = 1$ and 0 otherwise. We use this matrix to merge features which are in the same group. In particular, we obtain new feature matrix $X$ from the original feature matrix $D$ as $X = DG$. Then using $X$, we apply Lasso on it to obtain coefficients $\beta$. In order to evaluate the stability of this method, we examine stability measures (SRCC, JSM, and Kuncheva index) on the coefficients of each individual feature obtained from $w = G \times \beta$.

**K-means + GroupLasso** Here, similar to the previous method, we cluster features using K-means and obtain the matrix $G$. To merge features which are in the same group, we obtain matrix $X = DG$. Following this, we apply GroupLasso on the matrix $X$ to obtain coefficients $\beta$ for each group of features. To evaluate the stability of K-means+GroupLasso, we examine stability measures on the coefficients obtained from $w = G \times \beta$. This method and K-means+Lasso are two examples that study the effect of unsupervised clustering to obtain groups of correlated features.

**Oscar** This method is a penalized method that studies supervised clustering in linear regression (Bondell and Reich, 2008). Using an octagonal constraint region, it encourages correlated features to take identical coefficients. We have discussed about this method in section 2.2.3.3 of chapter 2.

### 4.2.3 Evaluation Measures

The proposed method and the baselines are evaluated in terms of their stability in feature selection and predictive performance. The evaluation measures are described below:

**Stability Measures** To compare the stability performance of pg-EN with other baselines, we use three stability measures, Spearman’s rank correlation (SRCC),
4.2. Experiments

Jaccard similarity measure (JSM) and Kuncheva Index (Kuncheva, 2007). These stability measures are described in detail in section 2.2.4.1 of chapter 2.

**Predictive Performance Measures** To compare the predictive performance of pg-EN with other baselines for regression problems (in Synthetic-I and Synthetic-II datasets) we use Mean Squared Error (MSE). For classification problems (Synthetic-III and real-world datasets), we use five evaluation methods including Precision or Positive Predictive Value (PPV), Sensitivity, Specificity, F1 score and AUC. These classification performances are discussed in details in section 2.2.4.2 of chapter 2.

**Statistical Test** To determine whether there are significant differences between the results obtained using different algorithms on each dataset we perform statistical test. To this end, we use pairwise Wilcoxon signed-rank test (a non-parametric alternative to the paired t-test) with the significance level of 0.05 for every pair of models. We assume the null hypothesis statement as “both algorithms in the pair perform equally” and the alternative hypothesis statement as the opposite. So, if the p-value obtained from pairwise Wilcoxon signed-rank test is less than the significance level (= 0.05 in our study), we reject the null hypothesis. So, based on the population mean of the model, we can conclude which model outperforms the other.

**4.2.4 Experimental Settings**

As mentioned in section 4.2.1.1 for Synthetic-I and Synthetic-II, we simulate 100 datasets for each scenario, where each dataset consists of training set, validation set and test set. We fit the model on the training set and select parameters (tuning parameters in all the models and number of groups in pg-EN, KM+Lasso and KM+GroupLasso) using validation set. Then we evaluate its performance on the test set. The results for stability and prediction performance of each method are reported as an average over these 100 simulations.

For Synthetic-III, we partition two groups of 500 healthy and 500 diseased subjects into 10 balanced non-overlapping datasets of size 50 subjects. We do this procedure
4.2. Experiments

Figure 4.4: Feature selection stability as measured by the Kuncheva Index for real-world datasets. Larger values indicate higher stability.

10 times, so finally we will have 100 datasets of 50 subjects. We use external cross-validation loops with separate training and test phases. The final results are reported as an average over 100 datasets.

Turning to real datasets, we randomly divide data into training set and test set. All the models are trained on the training set and their performances are evaluated using the test set. Parameters of the models are selected using 5-fold cross validation on the training set. The random splitting of real datasets is done 100 times and the results (stability and predictive performances) are reported as an average over these 100 splits.

4.2.5 Experimental Results

In this section we compare stability and predictive performance of pg-EN with other baseline regression and feature selection algorithms.
Table 4.1: Average stability performance of pg-EN compared to other baselines in terms of SRCC and JSM, for synthetic and real datasets. The numbers in brackets show the p-values obtained by applying Wilcoxon signed-rank test to the best and the second best stability results for each dataset.

<table>
<thead>
<tr>
<th></th>
<th>Synthetic data</th>
<th>Real data</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Syn-I</td>
<td>Syn-II</td>
</tr>
<tr>
<td><strong>Lasso</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRCC</td>
<td>0.265</td>
<td>0.214</td>
</tr>
<tr>
<td>JSM</td>
<td>0.386</td>
<td>0.302</td>
</tr>
<tr>
<td><strong>Elastic net</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRCC</td>
<td>0.453</td>
<td>0.422</td>
</tr>
<tr>
<td>JSM</td>
<td>0.620</td>
<td><strong>0.602</strong></td>
</tr>
<tr>
<td><strong>Oscar</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRCC</td>
<td>0.482</td>
<td>0.457</td>
</tr>
<tr>
<td>JSM</td>
<td>0.602</td>
<td>0.577</td>
</tr>
<tr>
<td><strong>KM Lasso</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRCC</td>
<td>0.487</td>
<td>0.392</td>
</tr>
<tr>
<td>JSM</td>
<td>0.527</td>
<td>0.507</td>
</tr>
<tr>
<td><strong>KM GLasso</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRCC</td>
<td>0.326</td>
<td>0.387</td>
</tr>
<tr>
<td>JSM</td>
<td>0.552</td>
<td>0.519</td>
</tr>
<tr>
<td><strong>pg-EN</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRCC</td>
<td><strong>0.512</strong></td>
<td><strong>0.503</strong></td>
</tr>
<tr>
<td></td>
<td>(1.8e-31)</td>
<td>(1.0e-26)</td>
</tr>
<tr>
<td>JSM</td>
<td><strong>0.627</strong></td>
<td><strong>0.602</strong></td>
</tr>
<tr>
<td></td>
<td>(3.6e-24)</td>
<td>(0.565)</td>
</tr>
</tbody>
</table>

4.2.5.1 Stability Performance

Table 4.1 and Figure 4.4 compare the stability performance of pg-EN with other baselines in terms of SRCC, JSM and Kuncheva index. To this end, we examine these stability measures on coefficients of pg-EN obtained from \( w = G \times \beta \). The numbers in brackets in Table 4.1 show the p-values obtained by applying Wilcoxon signed-rank test to the best and the second best stability results for each dataset.

For Synthetic-I the most stable method is pg-EN with SRCC=0.512 and JSM=0.627. In terms of SRCC the second best stability belongs to KM+Lasso (0.487) and in terms of JSM it belongs to Elastic net (0.620). The small p-values obtained
4.2. Experiments

from Wilcoxon signed-rank test also confirm that there is significant difference between the stability obtained using pg-EN and the second best method. For Synthetic-II, In terms of SRCC again the most stable method is pg-EN (SRCC=0.503), which is followed by Oscar (SRCC=0.457). However, in terms of JSM, the stability performance of pg-EN is equivalent to Elastic net. In this case, the Wilcoxon test could not reject the null hypothesis. Turning to Synthetic-III (Microarray dataset) again pg-EN is the most stable method with SRCC=0.487 and JSM=0.587. In terms of SRCC, pg-EN is followed by Elastic net with SRCC=0.402 and in terms of JSM it is followed by Oscar with JSM=0.575.

In case of Breast cancer dataset, the best stability performance in terms of SRCC belongs to Oscar (0.512), which is followed by pg-EN (0.502). However, in terms of JSM again pg-EN shows the best stability (0.617), followed by Elastic net (0.583) and Oscar (0.580). In Cancer (EMR) dataset, pg-EN shows the best stability performance with SRCC=0.543 and JSM=0.622. In terms of SRCC, pg-EN is followed by Elastic net (0.443) and KM+GroupLasso (0.442), respectively. Turning to JSM, Oscar (0.530) and Elastic net (0.527) are in the next stages after pg-EN. For AMI (EMR) dataset, once again pg-EN is the winner in terms of both SRCC (0.524) and JSM (0.614), followed by Oscar with SRCC=0.467 and JSM=0.552 and Elastic net with SRCC=0.436 and JSM=0.540. As seen from the table, for all datasets Lasso has the least stability performance in terms of both SRCC and JSM.

Figure 4.4, compares the stability of pg-EN with other baselines in terms of Kuncheva index on real-world datasets. As seen from this figure, the stability performance of pg-EN consistently outperforms other methods. Also, Lasso is the least stable method among others in all datasets. These results empirically demonstrate that pg-EN can greatly stabilize Lasso.

4.2.5.2 Predictive Performance

Figure 4.5 compares the predictive performance of pg-EN with other baselines on Synthetic-I and Synthetic-II datasets in terms of Mean Squared Error (MSE). We have also reported the statistical significance of each method in Table 4.2 estimated using the pairwise Wilcoxon signed-rank test with significance level of 0.05. As seen from the box plots and the p-values in Tables 4.2(a) and (b), pg-EN results in better
4.2. Experiments

Predictive performance compared to other methods.

Table 4.3 shows the predictive performance of pg-EN compared to other methods in terms of standard classification performances, namely sensitivity, specificity, Positive Predictive Value (PPV), AUC score and F1 score for the simulation of microarray dataset (Synthetic-III). The p-values obtained from applying the Wilcoxon signed-rank test to the best and the second best classification measures are also shown in brackets in the same table. As seen from Table 4.3, pg-EN could obtain the best predictive performance among other methods. The p-values also confirm that there is significant difference between the predictive performance of pg-EN and other baselines. We have also reported the statistical significance of comparisons between different algorithm pairs in Tables B.1 and B.2 in section B.1.2 of Appendix B.

Tables 4.4, 4.5 and 4.6 show the capability and effectiveness of the pg-EN compared to the baseline algorithms in terms of standard classification performances, sensitivity, specificity, Positive Predictive Value (PPV), AUC scores and F1 score for Breast Cancer, Cancer (EMR) and AMI (EMR) datasets, respectively. The numbers in brackets in each table show the p-values obtained by applying Wilcoxon signed-rank test to the best and the second best classification results for each dataset.

For the Breast cancer dataset, Table 4.4 shows that the best classification perfor-
Table 4.2: The p-value obtained from pairwise Wilcoxon signed-rank test of MSE applied to the Synthetic datasets (a) Synthetic_I and (b) Synthetic_II

<table>
<thead>
<tr>
<th>p-value</th>
<th>Lasso</th>
<th>Elastic net</th>
<th>Oscar</th>
<th>KM+Lasso</th>
<th>KM+GLasso</th>
<th>pg-EN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ridge</td>
<td>5.17e-17</td>
<td>4.73e-18</td>
<td>5.87e-18</td>
<td>5.95e-15</td>
<td>3.64e-17</td>
<td>3.89e-18</td>
</tr>
<tr>
<td>Lasso</td>
<td>3.76e-18</td>
<td>5.59e-18</td>
<td>3.22e-05</td>
<td>0.4432</td>
<td>4.26e-18</td>
<td></td>
</tr>
<tr>
<td>Elastic net</td>
<td>5.11e-18</td>
<td>5.75e-18</td>
<td>3.75e-18</td>
<td>4.27e-18</td>
<td>4.86e-10</td>
<td></td>
</tr>
<tr>
<td>Oscar</td>
<td>5.18e-17</td>
<td>6.02e-17</td>
<td>5.18e-17</td>
<td>6.02e-17</td>
<td>3.89e-18</td>
<td></td>
</tr>
<tr>
<td>KM+Lasso</td>
<td>3.05e-04</td>
<td>4.23e-18</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM+GLasso</td>
<td>3.93e-18</td>
<td>3.93e-18</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(a)

| Ridge   | 4.23e-18 | 5.01e-18 | 3.98e-18 | 4.10e-18 | 4.05e-18 | 3.88e-18 |
| Lasso   | 3.62e-11 | 5.32e-17 | 2.74e-17 | 1.34e-12 | 4.67e-18 |
| Elastic net | 8.26e-18 | 1.63e-15 | 1.41e-17 | 3.89e-18 |
| Oscar   | 8.02e-12 | 1.49e-07 |
| KM+Lasso | 3.09e-16 | 3.09e-16 |
| KM+GLasso | 1.41e-17 | 1.41e-17 |

(b)

Performance in terms of all the classification measures belongs to pg-EN with PPV=0.325, Sensitivity=0.309, F1 score=0.429, Specificity=0.898 and AUC=0.855. This is also confirmed by the p-values obtained from pairwise Wilcoxon signed-rank test shown in brackets. The statistical significance of comparisons between different algorithm pairs are presented in Tables B.3 and B.4 in section B.1.2 of Appendix B.

Table 4.5, compares the classification performance of pg-EN with other baselines on Cancer (EMR) dataset. As seen, again pg-EN could achieve the best predictive performance among other methods with PPV=0.323, Sensitivity=0.392, Specificity=0.831 and AUC=0.728. This is also confirmed by the p-values obtained from pairwise Wilcoxon signed-rank test. However, in terms of F1 score, statistical test could not reject the null hypothesis and the performance of pg-EN and Oscar are comparable.
### Table 4.3: Average classification performances of pg-EN compared to other methods for Synthetic-III dataset. The numbers in brackets show the p-values obtained by applying Wilcoxon signed-rank test to the best and the second best classification results for each dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>PPV</th>
<th>Sensitivity</th>
<th>F1 score</th>
<th>Specificity</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ridge</td>
<td>0.785</td>
<td>0.799</td>
<td>0.792</td>
<td>0.810</td>
<td>0.759</td>
</tr>
<tr>
<td>Lasso</td>
<td>0.821</td>
<td>0.815</td>
<td>0.818</td>
<td>0.833</td>
<td>0.787</td>
</tr>
<tr>
<td>Elastic net</td>
<td>0.823</td>
<td>0.819</td>
<td>0.821</td>
<td>0.856</td>
<td>0.792</td>
</tr>
<tr>
<td>Oscar</td>
<td>0.834</td>
<td>0.83</td>
<td>0.832</td>
<td>0.865</td>
<td>0.81</td>
</tr>
<tr>
<td>KM+Lasso</td>
<td>0.815</td>
<td>0.819</td>
<td>0.817</td>
<td>0.838</td>
<td>0.792</td>
</tr>
<tr>
<td>KM+GLasso</td>
<td>0.825</td>
<td>0.825</td>
<td>0.825</td>
<td>0.858</td>
<td>0.790</td>
</tr>
<tr>
<td>pg-EN</td>
<td>0.843</td>
<td><strong>0.840</strong></td>
<td><strong>0.842</strong></td>
<td><strong>0.872</strong></td>
<td><strong>0.821</strong></td>
</tr>
</tbody>
</table>

### Table 4.4: Average classification performance of pg-EN compared to other methods for Breast Cancer dataset. The numbers in brackets show the p-values obtained by applying Wilcoxon signed-rank test to the best and the second best classification results for each dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>PPV</th>
<th>Sensitivity</th>
<th>F1 score</th>
<th>Specificity</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ridge</td>
<td>0.311</td>
<td>0.421</td>
<td>0.358</td>
<td>0.824</td>
<td>0.797</td>
</tr>
<tr>
<td>Lasso</td>
<td>0.309</td>
<td>0.420</td>
<td>0.356</td>
<td>0.823</td>
<td>0.805</td>
</tr>
<tr>
<td>Elastic net</td>
<td>0.314</td>
<td>0.424</td>
<td>0.361</td>
<td>0.827</td>
<td>0.81</td>
</tr>
<tr>
<td>Oscar</td>
<td>0.319</td>
<td>0.426</td>
<td>0.361</td>
<td>0.83</td>
<td>0.813</td>
</tr>
<tr>
<td>KM+Lasso</td>
<td>0.313</td>
<td>0.421</td>
<td>0.359</td>
<td>0.823</td>
<td>0.807</td>
</tr>
<tr>
<td>KM+GLasso</td>
<td>0.313</td>
<td>0.423</td>
<td>0.36</td>
<td>0.822</td>
<td>0.806</td>
</tr>
<tr>
<td>pg-EN</td>
<td><strong>0.325</strong></td>
<td><strong>0.437</strong></td>
<td><strong>0.373</strong></td>
<td><strong>0.837</strong></td>
<td><strong>0.822</strong></td>
</tr>
</tbody>
</table>
Table 4.5: Average classification performance of pg-EN compared to other methods for Cancer (EMR) dataset. The numbers in brackets show the p-values obtained by applying Wilcoxon signed-rank test to the best and the second best classification results for each dataset.

<table>
<thead>
<tr>
<th></th>
<th>PPV</th>
<th>Sensitivity</th>
<th>F1 score</th>
<th>Specificity</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ridge</td>
<td>0.307</td>
<td>0.371</td>
<td>0.336</td>
<td>0.805</td>
<td>0.691</td>
</tr>
<tr>
<td>Lasso</td>
<td>0.306</td>
<td>0.378</td>
<td>0.338</td>
<td>0.815</td>
<td>0.713</td>
</tr>
<tr>
<td>Elastic net</td>
<td>0.311</td>
<td>0.380</td>
<td>0.342</td>
<td>0.82</td>
<td>0.715</td>
</tr>
<tr>
<td>Oscar</td>
<td>0.315</td>
<td>0.384</td>
<td>0.346</td>
<td>0.824</td>
<td>0.721</td>
</tr>
<tr>
<td>KM+Lasso</td>
<td>0.309</td>
<td>0.379</td>
<td>0.34</td>
<td>0.822</td>
<td>0.714</td>
</tr>
<tr>
<td>KM+GLasso</td>
<td>0.309</td>
<td>0.382</td>
<td>0.342</td>
<td>0.818</td>
<td>0.715</td>
</tr>
<tr>
<td>pg-EN</td>
<td>0.323</td>
<td>0.392</td>
<td>0.354</td>
<td>0.831</td>
<td>0.728</td>
</tr>
</tbody>
</table>

4.2.5.3 Execution Time

In section 4.1.2, we discussed about the computational complexity of pg-EN. Now, we empirically compare its execution time with some of the baseline methods on real-world datasets. Table 4.7, shows that the execution time of pg-EN is bigger than Lasso and Elastic net but it is less than Oscar. Also, Figure 4.6, shows the execution time of pg-EN (using Cancer (EMR) dataset) comparing it with those of
Table 4.6: Average classification performance of pg-EN compared to other methods for AMI (EMR) dataset. The numbers in brackets show the p-values obtained by applying Wilcoxon signed-rank test to the best and the second best classification results for each dataset.

<table>
<thead>
<tr>
<th></th>
<th>PPV</th>
<th>Sensitivity</th>
<th>F1 score</th>
<th>Specificity</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ridge</td>
<td>0.296</td>
<td>0.389</td>
<td>0.336</td>
<td>0.750</td>
<td>0.602</td>
</tr>
<tr>
<td>Lasso</td>
<td>0.309</td>
<td>0.400</td>
<td>0.348</td>
<td>0.756</td>
<td>0.608</td>
</tr>
<tr>
<td>Elastic net</td>
<td>0.31</td>
<td>0.403</td>
<td>0.350</td>
<td>0.758</td>
<td>0.61</td>
</tr>
<tr>
<td>Oscar</td>
<td>0.315</td>
<td>0.406</td>
<td>0.355</td>
<td>0.763</td>
<td>0.613</td>
</tr>
<tr>
<td>KM+Lasso</td>
<td>0.307</td>
<td>0.399</td>
<td>0.347</td>
<td>0.743</td>
<td>0.607</td>
</tr>
<tr>
<td>KM+GLasso</td>
<td>0.311</td>
<td>0.405</td>
<td>0.351</td>
<td>0.76</td>
<td>0.609</td>
</tr>
<tr>
<td>pg-EN</td>
<td>0.315</td>
<td>0.419</td>
<td>0.359</td>
<td>0.773</td>
<td>0.626</td>
</tr>
</tbody>
</table>

Lasso and Oscar for increasing number of samples. As shown, again the execution time of pg-EN is bigger than Lasso and lower than Oscar. Also, it is roughly linear in the number of samples, which suggests that pg-EN scales well on large datasets.

4.2.5.4 Effect of Grouping

Figure 4.7 shows the prediction performance (in terms of classification error) and the stability performance (in terms of JSM) of pg-EN with respect to variation in number of groups for Cancer (EMR) and AMI (EMR) data. As seen from the left figure for Cancer data, when we decrease the number of feature groups in the model from around 600 to 300, both the stability and prediction performance improve. However, decreasing the number of groups further, even enforces lowly correlated features to be grouped together, which is almost acceptable for obtaining better stability, but degrades the prediction performance. Similar behavior is observed for AMI dataset. This suggest that one should not over-enforce the feature grouping. Therefore, the best number of groups in the model can be selected when there is a logical trade-off between prediction and stability performances. The recommended range for selecting the best number of groups in each dataset is shown by vertical lines in Figure 4.7.
4.2. Experiments

105

The number of samples
1000 1500 2000 2500 3000 3500 4000 4500

Execution time (sec)

0 20 40 60 80 100 120 140

Figure 4.6: Execution time (in seconds) of pg-EN compared to Lasso and Oscar for different number of samples on Cancer (EMR) dataset.

Table 4.7: Execution time (in seconds) of pg-EN compared to some other methods for real datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Lasso</th>
<th>Elastic net</th>
<th>Oscar</th>
<th>pg-EN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Breast cancer</td>
<td>76.6</td>
<td>71.4</td>
<td>170.4</td>
<td>90.9</td>
</tr>
<tr>
<td>Cancer (EMR)</td>
<td>52.9</td>
<td>54.3</td>
<td>121.9</td>
<td>87.1</td>
</tr>
<tr>
<td>AMI (EMR)</td>
<td>48.3</td>
<td>50.2</td>
<td>107.8</td>
<td>69.3</td>
</tr>
</tbody>
</table>

4.2.5.5 Risk Factors Obtained Using pg-EN

As mentioned before, identifying robust variables can assist domain experts in their decision makings towards accurate medical prognosis. In Table 4.8 and 4.9 we have reported some of the top variables obtained using pg-EN for Cancer and AMI datasets. The importance of these variables is based on the predictive weights assigned to them by the pg-EN. To select the top risk factors, we use the feature sets obtained by 100 splitting of the data and compute the mean weights of the selected variables over these 100 splits and some of the variables with highest absolute weights are reported in the Table. Also, we empirically estimate the probability of presence for each feature. In these tables, column 1 shows the variable’s name, Column 2 shows its ICD-10 code, Column 3 shows its average weight and Column 4 shows it
4.2. Experiments

Figure 4.7: Prediction and stability performance of the model with respect to variation in number of groups, shown for Cancer and AMI datasets. The recommended range for number of groups is marked by vertical lines.

probability of presence in data splits.

4.2.5.6 Feasibility of Grouping

Figure 4.8 shows some examples of feature groups estimated using pg-EN for AMI dataset. As seen, features in each group are related to a special type of disorder. The features shown in Figure 4.8(a), are all related to bone diseases, features that are listed in Figure 4.8(b), are related to cardio-pulmonary diseases. Figure 4.8(c) shows the features of psycho-emotional disorders and Figure 4.8(d) lists the features, related to brain diseases. We note that in these figures the features which are associated with suffixes 3M, 6M or 1Y, show that those features occurred in previous 3 months, or 6 months or 1 year of prediction.

We have also assessed the grouping ability of pg-EN on Soil data, which studies relation between soil characteristics and rich-cove forest diversity in the Appalachian mountains of North Carolina. Although this dataset is non-medical, its small number but highly correlated features allows for an in-depth illustration of the behavior of our proposed model. The explanation and obtained results related to this dataset is presented in Section B.1.1 of the Appendix B.
Table 4.8: Top selected variables for cancer obtained by pg-EN are consistent with the risk factors reported by clinicians and research articles.

<table>
<thead>
<tr>
<th>Risk factor</th>
<th>ICD-10 code</th>
<th>Weights</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benign neoplasm of breast</td>
<td>D24 (Pfeiffer et al., 2013)</td>
<td>0.0715 ± 0.0010</td>
<td>0.92</td>
</tr>
<tr>
<td>Type II diabetes mellitus</td>
<td>E11 (Yuhara et al., 2011; Zhao and Weng, 2011)</td>
<td>−0.0627 ± 0.0021</td>
<td>0.84</td>
</tr>
<tr>
<td>Anorexia</td>
<td>R63.0 (Maltoni et al., 2012)</td>
<td>0.0601 ± 0.0014</td>
<td>0.82</td>
</tr>
<tr>
<td>Nausea and vomiting</td>
<td>R11 (Laird et al., 2013; Zhao and Weng, 2011)</td>
<td>0.0502 ± 0.0011</td>
<td>1</td>
</tr>
<tr>
<td>Fatigue</td>
<td>R53 (Zhao and Weng, 2011)</td>
<td>0.0426 ± 0.0024</td>
<td>1</td>
</tr>
<tr>
<td>Diarrhea</td>
<td>R19.7 (Laird et al., 2013)</td>
<td>0.0402 ± 0.0016</td>
<td>0.86</td>
</tr>
</tbody>
</table>

Table 4.9: Top selected variables for readmission after AMI obtained by pg-EN are consistent with those reported by clinicians and research articles.

<table>
<thead>
<tr>
<th>Risk factor</th>
<th>ICD-10 code</th>
<th>Weights</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Angina pectoris</td>
<td>I20 (Grady et al., 2012; Krumholz et al., 2011)</td>
<td>−0.0743 ± 0.0010</td>
<td>0.98</td>
</tr>
<tr>
<td>Kidney disease</td>
<td>N17, N18 (Desai et al., 2009; Krumholz et al., 2011; Shams et al., 2014)</td>
<td>0.0616 ± 0.0006</td>
<td>0.78</td>
</tr>
<tr>
<td>Diabetes and DM complications</td>
<td>E10, E11 (Desai et al., 2009; Krumholz et al., 2011)</td>
<td>0.0545 ± 0.0002</td>
<td>0.75</td>
</tr>
<tr>
<td>COPD</td>
<td>J44 (Desai et al., 2009; Krumholz et al., 2011)</td>
<td>−0.0525 ± 0.0012</td>
<td>0.79</td>
</tr>
<tr>
<td>Hypertension</td>
<td>I10 (Brown et al., 2013; Desai et al., 2009; Krumholz et al., 2001)</td>
<td>0.0516 ± 0.0023</td>
<td>0.88</td>
</tr>
<tr>
<td>Cardiorespiratory failure</td>
<td>E83.4 (Dunlay et al., 2012)</td>
<td>0.0432 ± 0.0017</td>
<td>0.85</td>
</tr>
</tbody>
</table>
4.3 Conclusion

In this chapter, we have proposed a new model to stabilize Lasso in selecting informative features. Stability matters in applications such as Healthcare and bioinformatics, where the features carry intuitive meanings and interpreting informative features is important in decision makings. Realizing that the feature instability is caused by feature correlations, our model learns a grouping of correlated variables using data, performs feature selection at the group level in a unified framework. The model is formulated as a constrained optimization problem to jointly learn feature groups and their relevance. To solve the resulting constrained optimization problem, we have derived an efficient iterative optimization algorithm and showed that convergence of these iterative updates is mathematically guaranteed. We com-

Figure 4.8: Example of feature groups obtained using pg-EN for AMI data. Features in a group are consistent and related to (a) bone, (b) cardio-pulmonary, (c) psycho-emotional and (d) brain diseases. Suffixes 3M, 6M or 1Y in feature names, indicate that they occurred in previous 3 months, or 6 months or 1 year of prediction.
4.3. Conclusion

pere the stability and prediction performance of the proposed method with several state-of-the-art methods namely, Ridge regression, Lasso, Elastic net, Oscar, K-means+Lasso and K-means+GroupLasso using three synthetic and three real-world datasets (Breast cancer data, Cancer (EMR) data and AMI (EMR) data). We show that as the proposed model learns groups of correlated features and performs feature selection based on these groups, its feature selection stability is notably better than Lasso and comparable to other methods. In terms of prediction performance, we also demonstrate that the proposed model leads to better results than baseline algorithms. This is due to estimator’s variance reduction which itself is the result of grouping correlated features. Our results can be applied to identify stable risk factors for many problems in healthcare domain and bioinformatics and therefore can assist clinicians and domain experts in their decision makings towards more accurate medical prognosis.

The model proposed in this chapter needs to simultaneously learn the feature grouping and regression weights for each feature group. This gives rise to an optimization problem that is non-convex. Non-convex functions may have several local minima. These are the points that have least values within their local neighborhood, but do not have the lowest value in the entire set. An important question that arises, can we develop a model with convex objective function for capturing relationship between correlated variables in order to perform a stable feature selection. This is the subject of the next chapter.
Chapter 5

Exploiting Feature Relationships
Towards Stable Feature Selection

The predictive grouping elastic net (pg-EN) method proposed in the previous chapter improves the stability of $l_1$-norm based feature selection by grouping correlated features and selecting informative feature groups instead of each individual feature. In this method feature grouping is learned within the model using supervised data and hence is associated with prediction goal. However, solving its objective function is a formidable challenge since it is non-convex with potentially large number of local minima.

To overcome this problem, in this chapter, we propose another model that is able to perform prediction and stable feature selection while working with a convex objective function. Our method has a new regularization formulation that improves the stability of Lasso by encouraging the similarities between features based on their relatedness. The relatedness between features is captured via a feature covariance matrix. The proposed model can simultaneously perform feature selection and capture both the positive correlation and the negative correlation between features through a convex objective function. This method has the grouping effect, which means that a group of highly correlated predictors are either all selected together into the model or left out altogether. Another contribution we make in this chapter is the stability-driven tuning of model hyperparameters. Traditionally, model’s hyperparameters are tuned keeping the model performance. Instead, to increase the
5.1 Covariance Lasso

5.1.1 The Model Formulation

In this section, we propose a new model that addresses the instability of Lasso in selecting informative features. We consider a standard supervised learning setting with data \( \{(x_i, y_i)\}_{i=1}^n \), where \( x_i \in \mathbb{R}^p \) is the feature vector representing \( p \) features and \( y_i \) is the target value (for regression) or label (for classification). We collectively represent the data using a matrix \( X \in \mathbb{R}^{n \times p} \), where each row corresponds to the \( i \)th instance; similarly we have response vector \( y = X\beta + \epsilon \), where \( \beta \) is an unknown vector of regression coefficients and \( \epsilon \) is a random vector of uncorrelated noise terms with mean 0 and variance \( \sigma^2 \). We denote the \( j \)th column of \( X \) by \( x_j \in \mathbb{R}^n \). We
5.1. Covariance Lasso

assume that the features have been standardized to have mean 0 and $l_2$ norm of 1; in other words: $\sum_i X_{ij} = 0$, $\sum_i X_{ij}^2 = 1$. In general, we will assume that we are in high-dimensional and sparse setting where the majority of the variables are not associated with the outcome i.e. $\beta = 0$ for most $j = 1, \ldots, p$. Moreover, we assume that there are unknown groups of variables with high levels of correlations among variables. In this situation, penalized regression methods such as Lasso that use $l_1$ norm penalty on the feature weights to obtain sparse estimates, show feature instability behavior because they tend to assign a nonzero weight to only a single feature among a group of correlated features. To address this problem, we introduce a new regularization formulation that encourages similarities between features based on their relatedness. This feature relatedness is captured via a feature covariance matrix. With this solution in mind, we propose a new model, which is the solution to the following optimization problem:

$$\arg\min_{\beta, \Omega} J(\beta, \Omega) = \frac{1}{2} \| y - X\beta \|^2_2 + \lambda \| \beta \|_1 + \frac{\eta}{2} \beta^T \Omega^{-1} \beta$$

s.t. $\Omega \succeq 0$, $\text{tr}(\Omega) = 1$.

where $\Omega$ is the covariance matrix that models the relationships between features. The first constraint in (5.1) holds due to the fact that $\Omega$ is the feature covariance matrix and the second constraint restricts the complexity of the problem. The parameters $\lambda$ and $\eta$ are tuning parameters. We term the above model as Covariance Lasso (C-LASSO).

5.1.2 Optimization

Before proposing an optimization algorithm for the problem (5.1), we show that this problem is convex with respect to all variables. To do this, we first define the concepts of graph and epigraph of a function (Boyd and Vandenberghe, 2004), which will be used in the proof of convexity.

Definition 1. The graph of a function $f : \mathbb{R}^n \to \mathbb{R}$ is defined as $\{ (x, f(x) ) | x \in D(f) \}$, where $D(\cdot)$ represents the domain of a function. The graph of the function $f$ is a subset of $\mathbb{R}^{n+1}$. 
The epigraph of a function $f : \mathbb{R}^{n+1} \to \mathbb{R}$ is defined as $\text{epi} \, f = \{(x,t) | x \in D(f), f(x) \leq t\}$, which is also a subset of $\mathbb{R}^{n+1}$. The epigraph defines the link between convex functions and convex sets: A function is convex if and only if its epigraph is a convex set. Using this definition, we prove the convexity of the objective function in equation (5.1).

**Theorem 2.** Problem (5.1) is convex with respect to $\beta$ and $\Omega$.

**Proof.** It can be easily seen that the first two terms in the objective function in problem (5.1) are convex with respect to $\beta$ and $\Omega$. Also, the constraints in (5.1) are convex. The last term i.e. $\beta^T \Omega^{-1} \beta$ is called matrix fractional function (Boyd and Vandenberghe, 2004). The convexity of this term can be established via its epigraph, which is defined in Definition 1. Let us define function $f$ as: $f(\beta, \Omega) = \beta^T \Omega^{-1} \beta$. So, the epigraph of the function $f$ is:

$$
\text{epi} \, f = \{ (\beta, \Omega, t) | \Omega \succ 0, \beta^T \Omega^{-1} \beta \leq t \},
$$

using the Schur complement condition for positive semidefiniteness of a block matrix (Boyd and Vandenberghe, 2004). The convexity of this term can be established via its epigraph, which is defined in Definition 1. Let us define function $f$ as: $f(\beta, \Omega) = \beta^T \Omega^{-1} \beta$. So, the epigraph of the function $f$ is:

$$
\text{epi} \, f = \{ (\beta, \Omega, t) | \Omega \succ 0, \beta^T \Omega^{-1} \beta \leq t \},
$$

using the Schur complement condition for positive semidefiniteness of a block matrix (Boyd and Vandenberghe, 2004). The convexity of this term can be established via its epigraph, which is defined in Definition 1. Let us define function $f$ as: $f(\beta, \Omega) = \beta^T \Omega^{-1} \beta$. So, the epigraph of the function $f$ is:

$$
\text{epi} \, f = \{ (\beta, \Omega, t) | \Omega \succ 0, \beta^T \Omega^{-1} \beta \leq t \},
$$

using the Schur complement condition for positive semidefiniteness of a block matrix (Boyd and Vandenberghe, 2004). The convexity of this term can be established via its epigraph, which is defined in Definition 1. Let us define function $f$ as: $f(\beta, \Omega) = \beta^T \Omega^{-1} \beta$. So, the epigraph of the function $f$ is:

As shown above, the optimization problem in (5.1) is convex with respect to $\beta$ and $\Omega$. However, optimizing it with respect to all variables at the same time is not straightforward. To this end, we propose an iterative method to solve the problem more efficiently. In other words, we optimize the objective function with respect to $\beta$ while $\Omega$ is fixed, and then we fix $\beta$ and optimize the problem with respect to $\Omega$. This optimization procedure is defined as follows:

**Optimizing w.r.t. $\beta$ when $\Omega$ is fixed** By fixing $\Omega$, the optimization problem for finding $\beta$ becomes an unconstrained optimization problem, which can be stated as:

$$
\arg \min_{\beta} J(\beta) = \frac{1}{2} \|y - X\beta\|^2_2 + \lambda \|\beta\|_1 + \frac{\eta}{2} \beta^T \Omega^{-1} \beta.
$$

(5.3)
5.1. Covariance Lasso

This problem can be solved using the alternate direction method of multipliers (ADMM), which is an efficient optimization algorithm for solving many problems with non-smooth regularization (Boyd et al., 2011). Using ADMM, the objective function (5.3) will employ a dummy variable such that:

\[
\arg\min_{\beta} J(\beta) = \frac{1}{2} \| y - X\beta \|^2_2 + \lambda \| z \|^1_1 + \eta \beta^T \Omega^{-1} \beta \quad (5.4)
\]

subject to \( \beta = z \).

The augmented Lagrangian form of equation (5.4) becomes:

\[
\arg\min_{\beta} J(\beta) = \frac{1}{2} \| y - X\beta \|^2_2 + \lambda \| z \|^1_1 + \eta \beta^T \Omega^{-1} \beta + \mu^T (\beta - z) + \frac{\rho}{2} \| \beta - z \|^2_2, \quad (5.5)
\]

where \( \rho > 0 \) is the augmented Lagrangian parameter and \( \mu \) is the dual variable or Lagrange multiplier. For more convenience, the linear and quadratic terms in equation (5.5) can be combined together (Boyd et al., 2011). Let \( r = \beta - z \), we have

\[
\mu^T r + \left( \frac{\rho}{2} \right) \| r \|^2_2 = \frac{\rho}{2} \| r + \frac{1}{\rho} \mu \|^2_2 - \frac{1}{2\rho} \| \mu \|^2_2 \quad (5.6)
\]

\[
= \frac{\rho}{2} \| r + u \|^2_2 - \frac{\rho}{2} \| u \|^2_2,
\]

where \( u = \left( \frac{1}{\rho} \right) \mu \) is the scaled dual variable. Equations (5.5) and (5.6) are equivalent, but the scaled form of ADMM has often shorter formulas compared to the unscaled form. We use the scaled ADMM through out this chapter. ADMM algorithm consists of the iterations

\[
\beta^{k+1} := (X^T X + \eta \Omega^{-1} + \rho I)^{-1} (X^T y + \rho (z^k - u^k))
\]

\[
z^{k+1} := S_{\lambda/\rho}(\beta^{k+1} + u^k)
\]

\[
u^{k+1} := u^k + (\beta^{k+1} - z^{k+1})
\]

Optimizing w.r.t. \( \Omega \) when \( \beta \) is fixed When \( \beta \) is fixed, the optimization problem for finding \( \Omega \) becomes

\[
\min_{\Omega} \quad \beta^T \Omega^{-1} \beta \quad (5.7)
\]

subject to \( \Omega \succeq 0, \text{tr}(\Omega) = 1 \).
5.2. Tuning Hyperparameters via Kappa Selection Criterion

Let $B = \beta \beta^T$, as $\beta^T \Omega^{-1} \beta = \text{tr}(\beta^T \Omega^{-1} \beta) = \text{tr}(\Omega^{-1} \beta \beta^T)$ and $\text{tr}(\Omega) = 1$, we have

$$
\text{tr}(\Omega^{-1} B) = \text{tr}(\Omega^{-1} B) \text{tr}(\Omega)
= \text{tr}((\Omega^{-\frac{1}{2}} B \frac{1}{2})(B \frac{1}{2} \Omega^{-\frac{1}{2}})) \text{tr}(\Omega \frac{1}{2} \Omega \frac{1}{2})
\geq (\text{tr}(\Omega^{-\frac{1}{2}} B \frac{1}{2} \Omega \frac{1}{2}))^2 = (\text{tr}(B \frac{1}{2}))^2.
$$

The inequality holds because of Cauchy-Schwarz inequality for the Frobenius norm. From this inequality, we can say that $\text{tr}(\Omega^{-1} B)$ achieves its minimum value $(\text{tr}(B \frac{1}{2}))^2$ if and only if $\Omega^{-\frac{1}{2}} B \frac{1}{2} = \delta \Omega \frac{1}{2}$ for some constant $\delta$ and $\text{tr}(\Omega) = 1$. So $\Omega$ can be obtained from

$$
\Omega = \frac{(\beta \beta^T)^\frac{1}{2}}{\text{tr}((\beta \beta^T)^\frac{1}{2})}. \quad (5.8)
$$

**Computational Complexity**

The additional complexity requirements of the proposed method over Lasso is due to updating $\Omega$, which is of the order $p^2$ for each iteration. There is no extra complexities with respect to the number of instances.

5.2 Tuning Hyperparameters via Kappa Selection Criterion

Selecting an appropriate hyperparameter that balance the trade-off between model fitting and sparsity is essential for the accuracy of penalized regression models. Formally, the tuning parameters are selected based on methods such as cross validation and BIC that minimize the estimated prediction error or maximize the posterior probability. However, in feature selection problems, it is important to select hyperparameters based on feature selection stability. The main idea is that if several samples are drawn from the same distribution, similar subsets of features should be obtained by an ideal feature selection methods. In other words, the selected features should not vary much in different samples. To this end, Sun et al. (2013) proposed using the kappa coefficient to measure the similarity between two feature sets. The hyperparameter selection using kappa coefficient is formally defined as follows:
5.3 Experiments

**Feature Selection Stability** A feature selection method \( F(z^n; \lambda) \) applied on a training set \( z^n \) with a tuning parameter \( \lambda \), provides a set of selected features \( S \subset \{1, \ldots, p\} \), known as active set. Applying \( F \) on different training sets achieves different active sets. The agreement between two active sets \( S_1 \) and \( S_2 \), obtained by applying \( F \) on two training sets \( z_1^n \) and \( z_2^n \), can be measured using Cohen's kappa coefficient (Cohen, 1960),

\[
k(S_1, S_2) = \frac{Pr(a) - Pr(e)}{1 - Pr(e)},
\]

where, \( Pr(a) = (n_{11} + n_{22})/p \), is the relative agreement between \( S_1 \) and \( S_2 \) and \( Pr(e) = (n_{11} + n_{12})/p^2 + (n_{12} + n_{22})(n_{21} + n_{22})/p^2 \), is the hypothetical probability of disagreement, with \( n_{11} = |S_1 \cap S_2|, n_{12} = |S_1 \cap S_2^c|, n_{21} = |S_1^c \cap S_2|, n_{22} = |S_1^c \cap S_2^c| \), and \(|.|\) being the set cardinality. Based on (5.9), the feature selection stability is defined as below:

**Definition 2.** Feature selection stability of \( F \) is defined as

\[
s(F, \lambda, n) = E(k(F(Z_1^n; \lambda), F(Z_2^n; \lambda))),
\]

where \( Z_1^n \) and \( Z_2^n \) are two independent and identically distributed training examples and the expectation is taken with respect to them. \( F(Z_1^n; \lambda) \) and \( F(Z_2^n; \lambda) \) are two feature sets obtained by applying \( F \) to \( Z_1^n \) and \( Z_2^n \) with the same \( \lambda \).

Note that \(-1 < s(F, \lambda, n) < 1\), and larger value of \( s \) shows the more stable feature selection method. Algorithm 5.1, shows the kappa selection criterion in detail. In this chapter, we use \( \alpha_n = 0.1 \), which is the best choice based on our primary experiments.

### 5.3 Experiments

We design our experiments to show the prediction performance and stability of the proposed model on both synthetic and real-world datasets and compare it with several baseline methods that deemed to be closest to our work that currently exist, namely Ridge regression, Lasso (Tibshirani, 1996), Elastic net (EN) (Zou and Hastie,
Algorithm 5.1 Kappa selection criterion

- Partition \((x_1, \ldots, x_n)^T\) randomly into two subsets \(z^b_1 = (x^b_1, \ldots, x^b_m)^T\) and \(z_2 = (x^b_{m+1}, \ldots, x^b_{2m})^T\), where \(m = \lfloor n/2 \rfloor\).
- Apply \(F(z^b_1, \lambda)\) and \(F(z^b_2, \lambda)\) to \(z^b_1\) and \(z^b_2\), respectively and obtain active sets \(S^b_1\lambda\) and \(S^b_2\lambda\).
- Estimate the variable selection stability of \(F(\cdot; \lambda)\) in the b-th splitting by \(s^b(F, \lambda, m) = k(S^b_1\lambda, S^b_2\lambda)\).
- Compute \(s(F, \lambda, m)\) for all \(\lambda\)'s, and select \(\hat{\lambda} = \min \left\{ \lambda : \frac{s(F, \lambda, m)}{\max_{\lambda'} s(F, \lambda', m)} \geq 1 - \alpha_n \right\} \).

2005), Oscar (Bondell and Reich, 2008) and Laplacian-Lasso (Gopakumar et al., 2014). We also incorporate a new tuning parameter selection method that works based on feature selection stability to improve the stability of our proposed method and compare its stability and predictive performance with other existing tuning parameter selection criteria such as BIC and cross validation.

5.3.1 Baseline algorithms

**Lasso** This is a regularization method that is used to learn a regularized regression/classification model that is sparse in feature space (Tibshirani, 1996). We have introduced this method in section 2.2.2.4 of chapter 2.

**Elastic net** As we mentioned in section 2.2.2.4 of chapter 2, this method incorporates an \(l_2\)-norm penalty in the Lasso’s penalty term (Zou and Hastie, 2005) and can achieve an sparse model along with a tendency for correlated variables to yield similar regression coefficients.
5.3. Experiments

**Oscar** This method is a penalized method that studies supervised clustering in linear regression (Bondell and Reich, 2008). Using an octagonal constraint region, it encourages correlated features to take identical coefficients. We have discussed about this method in section 2.2.3.3 of chapter 2.

**Laplacian-Lasso** In this method, we use Laplacian to find out pairwise correlation between features (Gopakumar et al., 2014). To this end, we use the individual features and build a kernel similarity matrix over the set of features (columns of original data) as follows:

\[
A(i, j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right).
\]

Based on our primary experiments, we found that by choosing \(\sigma = 1\), we achieve the best experimental results. Using \(A\), we compute the Laplacian matrix \(L = D - A\), where \(D\) is the diagonal of matrix \(A\). So we have:

\[
\mathcal{L}_{\text{laplacian}} = \mathcal{L}_{\text{lasso}} + \eta \beta^T L \beta,
\]

where \(\eta\) is the non-negative tuning parameter.

### 5.3.2 Tuning Parameter Selection

To select the tuning parameters \(\lambda\) and \(\eta\) for synthetic data sets, we use a validation set in addition to the training and test sets. We train the model on the training set and we use the validation set to select the best tuning parameter for the final model. For real data sets, we use 5-fold cross validation to select the best tuning parameter.

### 5.3.3 Evaluation Measures

The proposed method and the baselines are evaluated in terms of their stability in feature selection and predictive performance. The evaluation measures are described below:
5.3. Experiments

Stability Measures To compare the stability performance of our proposed model with other baselines, we use two stability measures, Spearman’s rank correlation (SRCC) and Jaccard similarity measure (JSM). These stability measures are described in detail in section 2.2.4.1 of chapter 2.

Predictive Performance Measures To compare the predictive performance of C-Lasso with other baselines for regression problem (in Synthetic-III) we use Mean Squared Error (MSE). For classification problems (real-world datasets), we use five evaluation methods including Precision or Positive Predictive Value (PPV), Sensitivity, Specificity, F1 score and AUC. These classification performances are discussed in details in section 2.2.4.2 of chapter 2.

5.3.4 Simulation Results

Synthetic-I In this data set, the data is simulated from a regression model \(y = X\beta + \epsilon\), where \(\epsilon \sim \mathcal{N}(0, \sigma^2)\). The observations (rows of \(X\)), are iid from a \(\mathcal{N}(0, \Sigma)\) distribution, where \(\Sigma\) is a \(p \times p\) block diagonal matrix with three equally sized blocks. \(\Sigma\) has 1’s on the diagonal, 0.8’s within each block and 0’s elsewhere. The true parameters are \(\beta = (1, \ldots, 1, -1, \ldots, -1, 0, \ldots, 0)^T\) and \(\sigma = 1\). Figure 5.1(a) shows the correlation matrix of \(X\) for this data set. As seen from this figure, this data set results in three groups, where the features in each group are highly correlated to each other. Figures 5.1(b)-(d), show the estimated \(\beta\) obtained from performing Ridge, Oscar and C-Lasso, with tuning parameters chosen so that the resulting estimators have the same \(l_2\) norms. As seen from these figures, C-Lasso obtains the most precise and compact weight estimates.

Synthetic-II This data set is the extension of previous data set in a new context where each group consists of both positively and negatively correlated features. The true parameters are: \(\beta = (1, \ldots, 1, -1, \ldots, -1, 1, \ldots, 1, -1, \ldots, -1, 0, \ldots, 0)^T\). Figures 5.2(a)-(d), show the correlation matrix of \(X\) of this data set as well as estimated \(\beta\) obtained from performing Ridge, Oscar and C-Lasso. As seen from these figures,
5.3. Experiments

Figure 5.1: Grouping properties of C-Lasso for Synthetic-I data set. (a)-(d): There are three groups of highly correlated features, with coefficients 1, 0 and −1, illustrated in red, blue and green. (a): Empirical correlation matrix of Synthetic-I; beige, blue and green indicate positive, negative and zero correlations. (b): Density plot of estimated \( \hat{\beta} \) obtained by Ridge. (c): Density plot of estimated \( \hat{\beta} \) obtained by Oscar. (d): Density plot of estimated \( \hat{\beta} \) obtained by C-Lasso.

C-Lasso yields the best estimates that represents its ability to handle both positive and negative correlations among the features.

**Synthetic-III** In this dataset, we evaluate the performance of C-Lasso in situation of \( p > n \), with \( p = 1000 \) features. The data is simulated from a regression model \( y = X\beta + \epsilon \), in which \( \epsilon \) is a noise drawn from a normal distribution with mean 0 and standard deviation \( \sigma \), i.e. \( \epsilon \sim \mathcal{N}(0, \sigma^2) \) with \( \sigma = 2.5 \). The observations are iid from a \( \mathcal{N}(0, \Sigma) \) distribution, where \( \Sigma \) is a \( p \times p \) block diagonal matrix and its elements are as follows:

\[
\Sigma_{ij} = \begin{cases} 
1, & \text{if } i = j, \\
\rho, & \text{if } i \leq 50, j \leq 50, i \neq j, \\
-\rho, & \text{if } i \leq 50, 51 \leq j \leq 100, i \neq j, \\
\rho, & \text{if } 51 \leq i \leq 100, 51 \leq j \leq 100, i \neq j, \\
-\rho, & \text{if } 51 \leq i \leq 100, j \leq 50, i \neq j, \\
0, & \text{otherwise}
\end{cases} \tag{5.11}
\]

We have assessed the performance of C-Lasso for different values of \( \rho \), from 0 to 0.8. Moreover, \( \beta_j \sim \text{Uni}[0.9, 1.1] \) for \( 1 \leq j \leq 50 \), \( \beta_j \sim \text{Uni}[-1.1, -0.9] \) for \( 51 \leq j \leq 100 \),
5.3. Experiments

Figure 5.2: Grouping properties of C-Lasso for Synthetic-II data set. (a)-(d): There are three groups of highly correlated features, with positively and negatively correlated features in each group. (a): Empirical correlation matrix of Synthetic-II. (b): Density plot of estimated $\beta$ obtained by Ridge. (c): Density plot of estimated $\beta$ obtained by Oscar. (d): Density plot of estimated $\beta$ obtained by C-Lasso. Note that all the coefficients in the blue group are zero, whereas in red and green groups, half the coefficients are 1 and half are -1.

and $\beta_j = 0$ otherwise, where $\text{Un}[a, b]$ denotes the uniform distribution with parameters $a$ and $b$. In this data set, there are four groups of 50 correlated features that are associated with response. From these four sets of correlated features, two groups are positively correlated and the other two groups are negatively correlated. In our experiments, 50 datasets are generated, where each consists of 200 observations for training set, 200 observations for validation set and 800 observations for test set. We fit each algorithm on the training set using a range of tuning parameter values. We then select the final model to be the model that yields the smallest MSE, on the validation set. Finally, we evaluate the performance of the final model on the held out test set.

The stability and prediction performance of C-Lasso is compared with the state-of-the-art shrinkage and feature selection approaches, namely the Ridge regression, Lasso, Elastic net (EN), Oscar and Laplacian-Lasso. Table 5.1 represents the following quantities for different values of $\rho$: In order to assess the predictive performance of the model on synthetic data, we use Mean Squared Error (MSE) and to evaluate its stability we use two stability measures, Spearman’s rank correlation coefficient (SRCC) and Jaccard similarity measure (JSM). We note that as ridge regression does not perform any feature selection, we do not compute the feature stability for this method.
Table 5.1: Simulation results for Synthetic_III. Means and standard error over 50 iterations are reported.

<table>
<thead>
<tr>
<th>$\rho$</th>
<th>Method</th>
<th>MSE</th>
<th>JSM</th>
<th>SRCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>Ridge</td>
<td>115.276 (1.728)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Lasso</td>
<td>78.718 (1.872)</td>
<td>0.641 (0.012)</td>
<td>0.442 (0.024)</td>
</tr>
<tr>
<td></td>
<td>Elastic net</td>
<td>63.736 (0.762)</td>
<td>0.650 (0.009)</td>
<td>0.461 (0.016)</td>
</tr>
<tr>
<td></td>
<td>Oscar</td>
<td>62.726 (1.254)</td>
<td>0.670 (0.014)</td>
<td>0.480 (0.031)</td>
</tr>
<tr>
<td></td>
<td>Laplacian-Lasso</td>
<td><strong>62.552</strong> (1.432)</td>
<td>0.672 (0.012)</td>
<td>0.476 (0.027)</td>
</tr>
<tr>
<td></td>
<td>C-Lasso</td>
<td>62.573 (1.652)</td>
<td><strong>0.675</strong> (0.009)</td>
<td><strong>0.491</strong> (0.026)</td>
</tr>
<tr>
<td>0.1</td>
<td>Ridge</td>
<td>158.176 (0.872)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Lasso</td>
<td>67.635 (0.982)</td>
<td>0.640 (0.008)</td>
<td>0.440 (0.008)</td>
</tr>
<tr>
<td></td>
<td>Elastic net</td>
<td>58.524 (1.726)</td>
<td>0.650 (0.022)</td>
<td>0.459 (0.022)</td>
</tr>
<tr>
<td></td>
<td>Oscar</td>
<td>58.254 (2.635)</td>
<td>0.664 (0.03)</td>
<td>0.478 (0.016)</td>
</tr>
<tr>
<td></td>
<td>Laplacian-Lasso</td>
<td>58.262 (1.736)</td>
<td>0.670 (0.04)</td>
<td>0.482 (0.023)</td>
</tr>
<tr>
<td></td>
<td>C-Lasso</td>
<td><strong>58.241</strong> (1.726)</td>
<td><strong>0.675</strong> (0.021)</td>
<td><strong>0.490</strong> (0.027)</td>
</tr>
<tr>
<td>0.5</td>
<td>Ridge</td>
<td>137.928 (1.306)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Lasso</td>
<td>55.635 (1.726)</td>
<td>0.570 (0.012)</td>
<td>0.387 (0.029)</td>
</tr>
<tr>
<td></td>
<td>Elastic net</td>
<td>40.625 (2.016)</td>
<td>0.607 (0.021)</td>
<td>0.402 (0.032)</td>
</tr>
<tr>
<td></td>
<td>Oscar</td>
<td>37.726 (1.726)</td>
<td>0.622 (0.031)</td>
<td>0.409 (0.037)</td>
</tr>
<tr>
<td></td>
<td>Laplacian-Lasso</td>
<td>34.635 (1.827)</td>
<td>0.615 (0.023)</td>
<td>0.423 (0.021)</td>
</tr>
<tr>
<td></td>
<td>C-Lasso</td>
<td><strong>30.736</strong> (1.635)</td>
<td><strong>0.665</strong> (0.019)</td>
<td><strong>0.473</strong> (0.026)</td>
</tr>
<tr>
<td>0.8</td>
<td>Ridge</td>
<td>117.625 (1.635)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Lasso</td>
<td>51.625 (1.635)</td>
<td>0.514 (0.025)</td>
<td>0.327 (0.018)</td>
</tr>
<tr>
<td></td>
<td>EN</td>
<td>36.625 (2.052)</td>
<td>0.565 (0.012)</td>
<td>0.365 (0.022)</td>
</tr>
<tr>
<td></td>
<td>Oscar</td>
<td>30.736 (1.762)</td>
<td>0.571 (0.023)</td>
<td>0.366 (0.035)</td>
</tr>
<tr>
<td></td>
<td>Laplacian-Lasso</td>
<td>28.625 (2.076)</td>
<td>0.581 (0.026)</td>
<td>0.373 (0.019)</td>
</tr>
<tr>
<td></td>
<td>C-Lasso</td>
<td><strong>24.635</strong> (2.652)</td>
<td><strong>0.658</strong> (0.017)</td>
<td><strong>0.466</strong> (0.026)</td>
</tr>
</tbody>
</table>

The high value for SRCC implies that ranks of features do not vary a lot for different training sets and high value for JSM means that the selected features do not change significantly. As the table implies, in terms of prediction accuracy, due to the sparse underlying model, Elastic net always outperforms Ridge. Moreover, as there are correlations among the features, especially in larger $\rho$’s, Elastic net tends to dominate Lasso. When there is no or little correlation between variables (for $0 \leq \rho \leq 0.1$), the performance of C-Lasso is comparable to Elastic net, Oscar and Laplacian. However, when the correlation among variables in each group increases, C-Lasso outperforms these algorithms.
5.3. Experiments

Figure 5.3: The pictorial representation of correlation matrix of $X$ and the estimated covariance matrix $\Omega$ for synthetic data sets. For better representation, we show the correlation matrix computed from $\Omega$ matrix by standardizing its values using equation (5.12).
5.3. Experiments

In terms of stability performance, C-Lasso shows the best performance in terms of both JSM and SRCC. When the correlation among variables is small \((0 \leq \rho \leq 0.1)\), the stability of all methods is comparable to C-Lasso. Nevertheless, by increasing correlation between variables \((\rho \geq 0.5)\), we can see a small degradation in the stability of C-Lasso, whereas stability of other methods reduces dramatically. Although Laplacian-Lasso also tends to find the pairwise similarities between features to increase the stability of the model, it fails when there is negative correlation among features.

5.3.4.1 Estimated Covariance Matrix for Synthetic Data

In this section, we show the correlation matrix of \(X\) and the estimated covariance matrix \(\Omega\) for synthetic data sets. For the case of better representation, we show the correlation matrix computed from \(\Omega\) matrix by standardizing its values as follows:

\[
\Omega_{st}(i,j) = \frac{\Omega(i,j)}{\sqrt{\Omega(i,i)\Omega(j,j)}}.
\]

(5.12)

Figure 5.3 compares the correlation matrix for synthetic data sets and its estimated standardized covariance matrix obtained by C-Lasso. As seen, the \(\Omega\) matrix could effectively captures the relationship between the features in these data sets. Figures 5.3(b) and (c) show that C-Lasso can effectively find the positive and negative correlations among the features.

5.3.5 Application on Real-World Data

For evaluating the performance of the C-Lasso on real-world applications, we conduct our experiments on AMI and Cancer datasets, described in section 3.4.1.2 of chapter 3. Further, we use the breast cancer dataset, collected by (Van De Vijver et al., 2002) and described in section 4.2.1.2 of chapter 4.
5.3. Experiments

Table 5.2: Stability performance of C-Lasso compared to other baselines for real datasets. Means and standard error over 50 iterations are reported.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Method</th>
<th>JSM</th>
<th>SRCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Breast cancer</td>
<td>Ridge</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Lasso</td>
<td>0.387 (0.031)</td>
<td>0.217 (0.023)</td>
</tr>
<tr>
<td></td>
<td>Elastic net</td>
<td>0.523 (0.025)</td>
<td>0.452 (0.032)</td>
</tr>
<tr>
<td></td>
<td>Oscar</td>
<td>0.573 (0.027)</td>
<td><strong>0.521</strong> (0.029)</td>
</tr>
<tr>
<td></td>
<td>Laplacian-Lasso</td>
<td>0.532 (0.031)</td>
<td>0.512 (0.028)</td>
</tr>
<tr>
<td></td>
<td>C-Lasso</td>
<td><strong>0.612</strong> (0.031)</td>
<td>0.516 (0.033)</td>
</tr>
<tr>
<td>Cancer</td>
<td>Ridge</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Lasso</td>
<td>0.415 (0.036)</td>
<td>0.263 (0.031)</td>
</tr>
<tr>
<td></td>
<td>Elastic net</td>
<td>0.574 (0.025)</td>
<td>0.432 (0.027)</td>
</tr>
<tr>
<td></td>
<td>Oscar</td>
<td>0.554 (0.019)</td>
<td>0.438 (0.038)</td>
</tr>
<tr>
<td></td>
<td>Laplacian-Lasso</td>
<td>0.581 (0.027)</td>
<td>0.421 (0.027)</td>
</tr>
<tr>
<td></td>
<td>C-Lasso</td>
<td><strong>0.620</strong> (0.028)</td>
<td><strong>0.512</strong> (0.031)</td>
</tr>
<tr>
<td>AMI</td>
<td>Ridge</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Lasso</td>
<td>0.365 (0.032)</td>
<td>0.276 (0.021)</td>
</tr>
<tr>
<td></td>
<td>Elastic net</td>
<td>0.512 (0.035)</td>
<td>0.462 (0.035)</td>
</tr>
<tr>
<td></td>
<td>Oscar</td>
<td>0.483 (0.028)</td>
<td>0.423 (0.037)</td>
</tr>
<tr>
<td></td>
<td>Laplacian-Lasso</td>
<td>0.510 (0.027)</td>
<td>0.458 (0.029)</td>
</tr>
<tr>
<td></td>
<td>C-Lasso</td>
<td><strong>0.578</strong> (0.034)</td>
<td><strong>0.516</strong> (0.031)</td>
</tr>
</tbody>
</table>

**Stability Performance** Table 5.2 compares the stability performance of C-Lasso with other baselines in terms of SRCC and JSM for real-world datasets. For Breast cancer dataset, the best stability performance in terms of JSM belongs to C-Lasso (0.612), which is followed by Oscar (0.573). However, in terms of SRCC, Oscar shows the best stability (0.521), followed by C-Lasso (0.516) and Laplacian-Lasso (0.512). For Cancer dataset, C-Lasso shows the best stability performance with JSM=0.620 and SRCC=0.512. In terms of JSM, C-Lasso is followed by Laplacian-Lasso (0.581) and Elastic net (0.574), respectively. Turning to SRCC, Oscar (0.438) and Elastic net (0.432) are in the next stages after pg-Lasso. For AMI dataset, again C-Lasso is the winner in terms of both JSM (0.578) and SRCC (0.516), followed by Elastic net with JSM=0.512 and SRCC=0.462 and Laplacian-Lasso with JSM=0.510 and SRCC=0.458. As seen, in all datasets Lasso shows the least stability performance in terms of both JSM and SRCC.
Table 5.3: Predictive performance of C-Lasso compared to other baselines for Breast cancer dataset. Means and standard error over 50 iterations are reported.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Method</th>
<th>PPV</th>
<th>Sensitivity</th>
<th>F1 score</th>
<th>Specificity</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Breast cancer</td>
<td>Ridge</td>
<td>0.312</td>
<td>0.420</td>
<td>0.358</td>
<td>0.818</td>
<td>0.799</td>
</tr>
<tr>
<td></td>
<td>(0.010)</td>
<td>(0.012)</td>
<td>(0.015)</td>
<td>(0.011)</td>
<td>(0.017)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Lasso</td>
<td>0.308</td>
<td>0.423</td>
<td>0.356</td>
<td>0.820</td>
<td>0.807</td>
</tr>
<tr>
<td></td>
<td>(0.009)</td>
<td>(0.017)</td>
<td>(0.017)</td>
<td>(0.020)</td>
<td>(0.016)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Elastic net</td>
<td>0.315</td>
<td>0.428</td>
<td>0.362</td>
<td>0.825</td>
<td>0.811</td>
</tr>
<tr>
<td></td>
<td>(0.012)</td>
<td>(0.020)</td>
<td>(0.018)</td>
<td>(0.017)</td>
<td>(0.011)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Oscar</td>
<td>0.320</td>
<td>0.431</td>
<td>0.367</td>
<td>0.829</td>
<td>0.815</td>
</tr>
<tr>
<td></td>
<td>(0.017)</td>
<td>(0.012)</td>
<td>(0.012)</td>
<td>(0.015)</td>
<td>(0.016)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Laplacian-Lasso</td>
<td>0.310</td>
<td>0.428</td>
<td>0.359</td>
<td>0.826</td>
<td>0.812</td>
</tr>
<tr>
<td></td>
<td>(0.015)</td>
<td>(0.017)</td>
<td>(0.019)</td>
<td>(0.017)</td>
<td>(0.014)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>C-Lasso</td>
<td><strong>0.349</strong></td>
<td><strong>0.458</strong></td>
<td><strong>0.396</strong></td>
<td><strong>0.848</strong></td>
<td><strong>0.856</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.019)</td>
<td>(0.021)</td>
<td>(0.022)</td>
<td>(0.012)</td>
<td>(0.015)</td>
</tr>
</tbody>
</table>

Table 5.4: Predictive performance of C-Lasso compared to other baselines for Cancer dataset. Means and standard error over 50 iterations are reported.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Method</th>
<th>PPV</th>
<th>Sensitivity</th>
<th>F1 score</th>
<th>Specificity</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cancer</td>
<td>Ridge</td>
<td>0.306</td>
<td>0.374</td>
<td>0.336</td>
<td>0.810</td>
<td>0.706</td>
</tr>
<tr>
<td></td>
<td>(0.018)</td>
<td>(0.020)</td>
<td>(0.015)</td>
<td>(0.019)</td>
<td>(0.017)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Lasso</td>
<td>0.311</td>
<td>0.380</td>
<td>0.342</td>
<td>0.819</td>
<td>0.720</td>
</tr>
<tr>
<td></td>
<td>(0.017)</td>
<td>(0.020)</td>
<td>(0.019)</td>
<td>(0.016)</td>
<td>(0.015)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Elastic net</td>
<td>0.319</td>
<td>0.386</td>
<td>0.349</td>
<td>0.826</td>
<td>0.728</td>
</tr>
<tr>
<td></td>
<td>(0.015)</td>
<td>(0.016)</td>
<td>(0.014)</td>
<td>(0.011)</td>
<td>(0.012)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Oscar</td>
<td>0.328</td>
<td>0.388</td>
<td>0.355</td>
<td>0.830</td>
<td>0.738</td>
</tr>
<tr>
<td></td>
<td>(0.015)</td>
<td>(0.014)</td>
<td>(0.014)</td>
<td>(0.012)</td>
<td>(0.012)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Laplacian-Lasso</td>
<td>0.328</td>
<td>0.390</td>
<td>0.356</td>
<td>0.829</td>
<td>0.740</td>
</tr>
<tr>
<td></td>
<td>(0.016)</td>
<td>(0.015)</td>
<td>(0.015)</td>
<td>(0.014)</td>
<td>(0.015)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>C-Lasso</td>
<td><strong>0.340</strong></td>
<td><strong>0.418</strong></td>
<td><strong>0.375</strong></td>
<td><strong>0.850</strong></td>
<td><strong>0.764</strong></td>
</tr>
<tr>
<td></td>
<td>(0.020)</td>
<td>(0.015)</td>
<td>(0.016)</td>
<td>(0.020)</td>
<td>(0.018)</td>
<td></td>
</tr>
</tbody>
</table>
5.3. Experiments

Table 5.5: Predictive performance of C-Lasso compared to other baselines for AMI dataset. Means and standard error over 50 iterations are reported.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Method</th>
<th>PPV</th>
<th>Sensitivity</th>
<th>F1 score</th>
<th>Specificity</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMI</td>
<td>Ridge</td>
<td>0.298</td>
<td>0.384</td>
<td>0.335</td>
<td>0.753</td>
<td>0.607</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.014)</td>
<td>(0.017)</td>
<td>(0.016)</td>
<td>(0.015)</td>
<td>(0.016)</td>
</tr>
<tr>
<td></td>
<td>Lasso</td>
<td>0.306</td>
<td>0.405</td>
<td>0.340</td>
<td>0.750</td>
<td>0.610</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.012)</td>
<td>(0.016)</td>
<td>(0.015)</td>
<td>(0.014)</td>
<td>(0.017)</td>
</tr>
<tr>
<td></td>
<td>Elastic net</td>
<td>0.310</td>
<td>0.408</td>
<td>0.352</td>
<td>0.763</td>
<td>0.615</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.014)</td>
<td>(0.016)</td>
<td>(0.014)</td>
<td>(0.012)</td>
<td>(0.015)</td>
</tr>
<tr>
<td></td>
<td>Oscar</td>
<td>0.313</td>
<td>0.408</td>
<td>0.354</td>
<td>0.768</td>
<td>0.617</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.015)</td>
<td>(0.014)</td>
<td>(0.014)</td>
<td>(0.012)</td>
<td>(0.014)</td>
</tr>
<tr>
<td></td>
<td>Laplacian-Lasso</td>
<td>0.310</td>
<td>0.404</td>
<td>0.350</td>
<td>0.762</td>
<td>0.608</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.015)</td>
<td>(0.016)</td>
<td>(0.014)</td>
<td>(0.016)</td>
<td>(0.014)</td>
</tr>
<tr>
<td></td>
<td>C-Lasso</td>
<td>0.329</td>
<td>0.437</td>
<td>0.375</td>
<td>0.802</td>
<td>0.678</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.015)</td>
<td>(0.016)</td>
<td>(0.014)</td>
<td>(0.015)</td>
<td>(0.014)</td>
</tr>
</tbody>
</table>

**Predictive Performance**  Tables 5.3-5.5 compare the predictive performance of C-Lasso with other baselines in terms of several classification measures, namely Positive Predictive Value (PPV), sensitivity, specificity, F1 score and AUC for real-world datasets. As seen from the Table 5.3 for Breast cancer dataset, C-Lasso achieved the best predictive performance in terms of all classification measures followed by Oscar and Elastic net. For Cancer dataset, in Table 5.4, the best predictive performance in terms of all classification measures again belongs to C-Lasso and Oscar and Laplacian Lasso follow it. Turning to AMI dataset, presented in Table 5.5, again C-Lasso is the winner in terms of all classification performances, while Oscar and Elastic net follow it. Among all these three tables, Ridge regression shows the least accurate predictive performance among others.

**Estimated Covariance Matrix for Real Datasets**  Figure 5.4, shows the estimated covariance matrix for real data sets. Similar to the synthetic data sets, for the case of better representation, we show the correlation matrix computed from \( \Omega \) using the equation (5.12). From these three data sets, feature names for AMI and Cancer data sets are available and we can discuss about the groups obtained in their \( \Omega \) matrix in more details. In the \( \Omega \) matrix obtained for Cancer data set,
Figure 5.4: The pictorial representation of estimated covariance matrix $\Omega$ for real datasets. For better representation, we show the correlation matrix computed from $\Omega$ matrix by standardizing its values using equation (5.12).

the first group contains the features related to anemia, which is an important risk factor in cancer mortality prediction (Caro et al., 2001). The last group in this data set belongs to the Type 2 Diabetes Mellitus (T2DM), which is another important risk factor in cancer mortality (Coughlin et al., 2004). We have shown some of the feature’s codes and their descriptions related to these two groups in Table 5.6. In the estimated $\Omega$ matrix for AMI data set the first group of correlated features is related to some types of heart diseases, which are one of the important reasons of readmission for patients with Myocardial infarction (Eapen et al., 2013). The last group of correlated features in this data set is related to renal failure, which are another important causes of 30-day re-hospitalization after myocardial infarction (Eapen et al., 2013). The feature’s codes and their descriptions related to these two groups are shown in Table 5.7. By assigning similar weights to the correlated
Table 5.6: ICD-10 codes and their descriptions related to the first and the last groups shown in Figure 5.4(a)

<table>
<thead>
<tr>
<th>Feature Group</th>
<th>ICD-10 codes</th>
<th>Code title</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anemia</td>
<td>D50.0</td>
<td>Iron deficiency anemia</td>
</tr>
<tr>
<td></td>
<td>D59.0</td>
<td>Drug-induced autoimmune hemolytic anemia</td>
</tr>
<tr>
<td></td>
<td>D62</td>
<td>Acute posthemorrhagic anemia</td>
</tr>
<tr>
<td></td>
<td>D63.0</td>
<td>Anemia in neoplastic disease</td>
</tr>
<tr>
<td></td>
<td>D64.0</td>
<td>Hereditary sideroblastic anemia</td>
</tr>
<tr>
<td>T2DM</td>
<td>E11.22</td>
<td>T2DM with diabetic chronic kidney disease</td>
</tr>
<tr>
<td></td>
<td>E11.39</td>
<td>T2DM with ophthalmic complication</td>
</tr>
<tr>
<td></td>
<td>E11.65</td>
<td>T2DM with hyperglycaemia</td>
</tr>
<tr>
<td></td>
<td>E11.9</td>
<td>T2DM without complications</td>
</tr>
</tbody>
</table>

features our proposed method improves the feature stability.

5.3.6 Effect of Using Kappa Selection Method

This section examines the efficacy of the Kappa selection criterion on selecting tuning parameters of C-Lasso and other baseline algorithms. The performance of Kappa selection criterion is compared against two other competitors, BIC and cross validation. For Synthetic-III with $\rho = 0.8$, where the validation set is available, we use the validation set to select tuning parameters and for real datasets we use 5-fold cross validation for this aim. For both cases we use the notion CV.

In evaluating the performance of each selection criterion, we compare the stability of obtained feature sets in terms of JSM. We also compare the predictive performance through MSE (for synthetic data set) and AUC (for real data sets). The tuning parameters are selected using each selection criterion i.e. Kappa, BIC and CV. For the Kappa selection criterion, the number of random splitting $B$ is 50.

Figure 5.5, shows the stability performance in terms of JSM for each penalized regression method, in which their tuning parameters are selected using different selection criterion (i.e. CV, BIC and Kappa) for Synthetic-III with $\rho = 0.8$, and real-
Table 5.7: ICD-10 codes and their descriptions related to the first and the last groups shown in Figure 5.4(b)

<table>
<thead>
<tr>
<th>Feature Group</th>
<th>ICD-10 codes</th>
<th>Code title</th>
</tr>
</thead>
<tbody>
<tr>
<td>Heart diseases</td>
<td>I42</td>
<td>Cardiomyopathy</td>
</tr>
<tr>
<td></td>
<td>I44</td>
<td>Atrioventricular and left bundle-branch block</td>
</tr>
<tr>
<td></td>
<td>I46</td>
<td>Cardiac arrest</td>
</tr>
<tr>
<td></td>
<td>I47</td>
<td>Paroxysmal tachycardia</td>
</tr>
<tr>
<td>Renal failure</td>
<td>N17.0</td>
<td>Acute renal failure with tubular necrosis</td>
</tr>
<tr>
<td></td>
<td>N17.9</td>
<td>Acute renal failure, unspecified</td>
</tr>
<tr>
<td></td>
<td>N18.4</td>
<td>Chronic kidney disease, stage 4</td>
</tr>
<tr>
<td></td>
<td>N18.9</td>
<td>Chronic kidney disease, unspecified</td>
</tr>
</tbody>
</table>

world data sets. As seen, in all datasets feature selection stability for all penalized regression methods has been increased using Kappa selection criterion compared to BIC and CV. In more details, we can see that apart from selection criterion used, C-Lasso shows the best stability performance among others in majority of times. But when it is combined with Kappa selection criterion, its stability increases effectively. Figure 5.6 compares the predictive performance of C-Lasso with other baselines using different selection criteria i.e. CV, BIC and Kappa. As shown, Kappa selection criterion achieves accurate prediction results which are comparable with BIC and CV. By looking at the results obtained using Kappa selection criterion, we can see that for all the data sets, C-Lasso achieves better prediction accuracy compared to other methods.
5.3. Experiments

Figure 5.5: Comparison of stability performance obtained using different selection criterion i.e. CV, BIC and Kappa for each penalized regression method.

Figure 5.6: Comparison of predictive performance of C-Lasso with other baseline algorithms using different selection criterion, i.e. CV, BIC and Kappa for synthetic and real data sets.
5.4 Conclusion

In this chapter, we propose a new technique (C-Lasso) for stable feature selection in presence of highly correlated features. This method employs a new regularization formulation that encourages the similarities between features, which is captures via a feature covariance matrix. Using this formulation, our proposed method can model the relationship (both positive and negative) between features. Therefore, highly correlated features tend to be selected or rejected together. Unlike pg-EN method (see chapter 4), which has a non-convex optimization function, C-Lasso is formulated as a convex optimization problem. We have introduced an efficient alternating algorithm to solve the C-Lasso’s objective function. Moreover, we have used a new tuning parameter selection method, known as Kappa selection, to further increase the stability of C-Lasso. Our experiments show that in terms of both stability and predictive performance, C-Lasso outperforms existing state-of-the-art techniques under a range of simulated settings, and also yields better results for clinical applications, namely cancer mortality prediction, 30-day readmission prediction for AMI patients and breast cancer patients’ survival prediction based on gene expression profiles.

The loss function used in stable clinical prediction models discussed so far (in chapters 3-5), is either residual sum of squares or logit function. However, in a binary classification problem, the other widely used tool for classification is support vector machine (SVM) (Pochet and Suykens, 2006). In comparison with linear and logistic regression, SVMs are more robust to outliers because instead of considering all the points in the dataset, they only consider data points near the margin (support vectors). In the next chapter, we develop methods that perform stable feature selection using SVM.
Chapter 6

Stable Clinical Prediction with Support Vector Machines

The stable clinical prediction models proposed in previous chapters (chapter 3-5), use residual sum of squares or the logit function as their loss function. Another promising classifier in clinical domain is support vector machine (SVMs) (Pochet and Suykens, 2006). SVMs are more robust to outliers compared to linear and logistic regression. It is because unlike linear and logistic regression, which consider all the points in a dataset, SVMs only consider data points near the margin (support vectors). Further, it is proved both theoretically and empirically that “additive” algorithms, which have a similar inductive bias like SVMs, are well suited for problems with sparse entries, same as EMRs (Kivinen and Warmuth, 1995).

In this text, we focus on linear SVM because using linear kernel expresses the advantage of model interpretability, like any other linear models, which is crucial for building clinical prediction models (Steyerberg, 2008). Also, due to the large number of features in EMRs, it is shown that using non-linear kernels does not improve the performance and linear kernel should be the kernel of choice (Hsu et al., 2003). Additionally, as discussed in chapter 2, linear SVM has the extra advantage of minimizing the risk in the spirit of Equation (2.13). The feature selection problem for support vector machine, particularly the application of $l_1$ norm SVM, has been studied in previous works (Zhu et al., 2004; Zou, 2007). However, limited research has been done to address the instability of $l_1$-norm support vector machines.
To address this gap, in this chapter, we propose two convex methods to stabilize $l_1$-norm support vector machines. In general, we make two assumptions: 1) We are dealing with high dimensional but sparse setting. By sparsity we mean that the majority of the features are not predictive of the outcome. 2) Among the features, there are sets of features with high levels of correlations. In this context, $l_1$-SVM shows instability in selecting informative features because it randomly assigns a nonzero weight to a single feature among a group of correlated features and so with small changes in dataset, another feature maybe selected from the correlated group.

Our first proposed method, introduces a regularization formulation that encourages the similarities between features based on their relatedness. In our formulation, the relatedness between features is captured through a feature covariance matrix. This method can perform feature selection and capture both positive and negative correlations between features through a convex objective function. We refer to this model as Covariance SVM (C-SVM). We have proposed an alternating optimization algorithm to solve this objective function. In our second method, to stabilize $l_1$-norm SVM, we construct an undirected feature graph, where nodes represent EMR features and edges represent relationship between the features. We define the statistical relationship between features using Jaccard index. In this graph correlated features are connected by an edge that enables us to perform feature selection on the group level. Our proposed method uses a convex penalty that includes a pairwise $l_\infty$ regularizer to encourage coefficients of the correlated features to be equal. We refer to this method as graph-SVM. We solve the resulting objective function using alternating direction method of multipliers (ADMM). We demonstrate the effectiveness of these two methods on both synthetic and real-world datasets. We compare their stability and prediction performance with several baseline methods and show that our proposed models outperforms the baselines.

This chapter is organized as follows. Section 6.1 introduces our first proposed method, C-SVM, its convex objective function and the alternating optimization algorithm we used to solve it. Section 6.2, introduces our second proposed method, graph-SVM, which tries to stabilize $l_1$-norm SVM by using a graph that shows correlation between features. We also propose a convex function to model the correlation using a pairwise $l_\infty$-norm regularizer and an optimization algorithm to solve it. Section 6.3 describes our experiments on synthetic and real-world dataset showing the benefits of the proposed methods in terms of classification and stability performances.
6.1. Covariance-SVM

Compared to the state-of-the-art methods. Concluding remarks are made in Section 6.4.

6.1 Covariance-SVM

We consider a binary classification problem with training data \( \{x_i, y_i\}_{i=1}^n \), where \( x_i = (x_{i1}, \ldots, x_{ip})^T \) is the feature vector and \( y_i \in \{-1, 1\} \) is the class label. As mentioned earlier, we assume that we are in high-dimensional and sparse setting. Also, we assume that there are unknown groups of variables with high levels of correlations among variables. In this context, \( l_1 \)-SVM shows instability in selecting informative features because it randomly assigns a nonzero weight to a single feature among a group of correlated features and so with small changes in dataset, another feature maybe selected from the correlated group. To overcome this problem, we propose a new regularization formulation that encourages similarities between features based on their relations. A feature covariance matrix is used to capture the relationships between features. Our proposed model, is the solution to the following optimization problem:

\[
\arg\min_{\beta_0, \beta, \Omega} \frac{1}{n} \sum (1 - y_i (\beta_0 + x_i^T \beta))^+ + \lambda \|\beta\|_1 + \frac{\eta}{2} \beta^T \Omega^{-1} \beta
\]

s.t. \( \Omega \succeq 0, \text{tr}(\Omega) = 1 \),

where \( \beta \) is the vector of feature weights and \( \beta_0 \) is the intercept. Also, \( \Omega \) is the covariance matrix that models the relationships between features, \( \lambda \) and \( \eta \) are the tuning parameters and \( (1 - T)^+ = \max(T, 0) \) is the hinge loss. The term \( \beta^T \Omega^{-1} \beta \) ensures that feature weights follow the feature correlations, i.e. if two features are highly correlated their feature weights would become very high. We refer to the above model as Covariance SVM (C-SVM).

The objective function (6.1) is convex, which means that optimizing this function leads to a unique and global solution. In the next section, we prove the convexity of (6.1).
6.1. Proof of Convexity

In this section we show that the objective function (6.1) is convex with respect to all variables. In the proof of convexity we use the concepts of graph and epigraph of a function (Boyd and Vandenberghe, 2004) defined as follows:

**Definition 3.** The graph of a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is defined as $\{(x, f(x)) | x \in D(f)\}$, where $D(.)$ represents the domain of a function.

The epigraph of a function $f : \mathbb{R}^{n+1} \rightarrow \mathbb{R}$ is defined as $\text{epi } f = \{(x, t) | x \in D(f), f(x) \leq t\}$. Using epigraph, we can define the relation between convex functions and convex sets: A convex is convex if and only if its epigraph is a convex set.

**Theorem 3.** Problem (6.1) is convex with respect to $\beta_0, \beta$ and $\Omega$.

**Proof.** The first two terms in the objective function in problem (6.1) are convex with respect to all variables (Boyd and Vandenberghe, 2004). Moreover, the constraints in (6.1) are convex. For the proof of convexity of the last term $\beta^T \Omega^{-1} \beta$, we use the definition of epigraph. For the function $f$ defined as $f(\beta, \Omega) = \beta^T \Omega^{-1} \beta$, we use the Schur complement condition for positive semi-definiteness of a block matrix (Boyd and Vandenberghe, 2004) to define its epigraph:

$$
\text{epi } f = \{ (\beta, \Omega, t) | \Omega > 0, \beta^T \Omega^{-1} \beta \leq 0 \} \quad (6.2)
= \left\{ (\beta, \Omega, t) | \begin{bmatrix} \Omega & \beta \\ \beta^T & t \end{bmatrix} \succeq 0, \Omega > 0 \right\}.
$$

The last condition is a linear matrix inequality in $(\beta, \Omega, t)$, and therefore $\text{epi } f$ is convex. As the summation operator can preserve convexity (Boyd and Vandenberghe, 2004), the objective function (6.1) is convex.

6.1.2 Algorithm for Covariance-SVM

Although the objective function in (6.1) is convex with respect to all variables, its solution is not straightforward due to non-smooth terms. To solve this problem,
we introduce an iterative algorithm that alternatively updates $\beta$ and $\Omega$ as follows:

**Optimizing w.r.t. $\beta$ when $\Omega$ is fixed:** In this situation, the objective function can be stated as:

$$\arg\min_{\beta_0, \beta} \frac{1}{n} \left( 1 - y_i (\beta_0 + x_i^T \beta) \right)_+ + \lambda \|\beta\|_1 + \frac{\eta}{2} \beta^T \Omega^{-1} \beta.$$  \hfill (6.3)

This problem can be solved using the alternate direction method of multipliers (ADMM) (Boyd et al., 2011). Because of the nondifferentiability of the hinge loss and $l_1$ norm term in (6.3), we introduce some auxiliary variables to handle these two nondifferentiable terms. Suppose $X = (x_{ij})_{i=1,j=1}^{n,p}$ and $Y$ be a diagonal matrix, where its diagonal elements are the vector $y = (y_1, \ldots, y_n)^T$. So the problem in (6.3), can be reformulated as:

$$\arg\min_{\beta_0, \beta} \frac{1}{n} \sum_{i=1}^n (a_i)_+ + \lambda \|z\|_1 + \frac{\eta}{2} \beta^T \Omega^{-1} \beta$$  \hfill (6.4)

s.t. $a = 1 - Y (X \beta + \beta_0 1), z = \beta$.

where $a = (a_1, \ldots, a_n)$ and $1$ is a column vector of 1’s with length $n$. The augmented Lagrangian function of (6.4) is

$$L(\beta_0, \beta, a, z, u, v) = \frac{1}{n} \sum_{i=1}^n (a_i)_+ + \lambda \|z\|_1 + \frac{\eta}{2} \beta^T \Omega^{-1} \beta$$

$$+ \langle u, 1 - Y (X \beta + \beta_0 1) - a \rangle + \langle v, \beta - z \rangle,$$  \hfill (6.5)

where $u \in \mathbb{R}^n$ and $v \in \mathbb{R}^p$ are dual variables corresponding to the first and the second constraints in equation (6.4), respectively. $\langle.,.\rangle$ is the inner product in the Euclidean space and $\mu_1$ and $\mu_2$ control the convergence behavior and are usually set to 1. By solving the above equation w.r.t $u,v,(\beta_0, \beta)$, $a$ and $z$ we have:

$$\begin{align*}
(\beta_0^{k+1}, \beta^{k+1}) &= \arg\min_{\beta_0, \beta} L(\beta_0, \beta, a^k, z^k, u^k, v^k), \\
a^{k+1} &= \arg\min_a L(\beta_0^{k+1}, \beta^{k+1}, a, z^k, u^k, v^k), \\
z^{k+1} &= \arg\min_z L(\beta_0^{k+1}, \beta^{k+1}, a^{k+1}, z, u^k, v^k), \\
u^{k+1} &= u^k + \mu_1 (1 - Y (X \beta^{k+1} + \beta_0^{k+1} 1) - a^{k+1}), \\
v^{k+1} &= v^k + \mu_2 (\beta^{k+1} - z^{k+1}).
\end{align*}$$  \hfill (6.6)
The first term in (6.6) is a quadratic and differentiable objective function, so its solution can be found by solving a set of linear equations:

\[
\begin{pmatrix}
\lambda_2 \Omega^{-1} + \mu_2 I + \mu_1 X^T X & \mu_1 X^T I \\
\mu_1 X^T & \mu_1 n
\end{pmatrix}
\begin{pmatrix}
\beta^{k+1} \\
\beta_0^{k+1}
\end{pmatrix} =
\begin{pmatrix}
X^T Y u^k - \mu_1 X^T Y (a^k - 1) - v^k + \mu_2 z^k \\
1^T Y u^k - \mu_1 1^T Y (a^k - 1)
\end{pmatrix}.
\]

The second term in (6.6) can be solved by using Proposition 4.

**Proposition 4.** Let \( h_\lambda(w) = \arg \min_x \lambda x + \frac{1}{2} \|x - w\|_2^2 \). Then \( h_\lambda(w) = w \) for \( w > \lambda \), \( h_\lambda(w) = 0 \) for \( 0 \leq w \leq \lambda \) and \( h_\lambda(w) = w \) for \( w < 0 \).

So we can write the second term in (6.6) as

\[
\frac{\|u\|_2^2}{2 \mu_1} + \frac{\mu_2}{2} \|1 - Y (X \beta^{k+1} + \beta_0^{k+1} 1) - a\|_2^2 + \langle u^k, 1 - Y (X \beta^{k+1} + \beta_0^{k+1} 1) - a \rangle
\]
\[
= \frac{\mu_2}{2} \|a - (1 + \frac{u^k}{\mu_1} - Y (X \beta^{k+1} + \beta_0^{k+1} 1))\|_2^2.
\]

From above equation and Proposition 4, we can update \( a^{k+1} \) as follows:

\[
a^{k+1} = H_\lambda (1 + \frac{u^k}{\mu_1} - Y (X \beta^{k+1} + \beta_0^{k+1} 1)),
\]

where \( H_\lambda(w) = (h_\lambda(w_1), h_\lambda(w_2), \ldots, h_\lambda(w_n))^T \).

The third equation in (6.6) can be solved using soft thresholding. So we have

\[
z^{k+1} = S_\lambda \frac{u^k}{\mu_2} + \beta^{k+1},
\]

where \( S_\lambda \) is the soft threshold operator defined on vector space and \( S_\lambda(w) = (s_\lambda(w_1), \ldots, s_\lambda(w_n)) \), where \( s_\lambda(w_i) = sgn(w_i) \max \{0, |w_i| - \lambda\} \).

By combining (6.6), (6.7), (6.8) and (6.9), we obtain the ADMM algorithm for solving the objective function (6.1) with respect to \( \beta \) when \( \Omega \) is fixed.
Optimizing w.r.t $\Omega$ when $\beta$ is fixed: In this situation, the optimization problem for finding $\Omega$ becomes
\[
\min_{\Omega} \quad \beta^T \Omega^{-1} \beta \\
\text{such that} \quad \Omega \succeq 0, \text{tr}(\Omega) = 1
\]
Let $B = \beta \beta^T$, as $\beta^T \Omega^{-1} \beta = \text{tr}(\beta^T \Omega^{-1} \beta) = \text{tr}(\Omega^{-1} \beta \beta^T)$ and $\text{tr}(\Omega) = 1$, so
\[
\text{tr}(\Omega^{-1} B) = \text{tr}(\Omega^{-1} B) \text{tr}(\Omega) = \text{tr}((\Omega^{-\frac{1}{2}} B^{\frac{1}{2}})(B^{\frac{1}{2}} \Omega^{-\frac{1}{2}})) \text{tr}(\Omega^{\frac{1}{2}} \Omega^{\frac{1}{2}}) \geq (\text{tr}(\Omega^{-\frac{1}{2}} B^{\frac{1}{2}} \Omega^{\frac{1}{2}}))^2 = (\text{tr}(B^{\frac{1}{2}}))^2.
\]
The inequality holds because of Cauchy-Schwarz inequality for the Frobenius norm. From this inequality, we can say that $\text{tr}(\Omega^{-1} B)$ achieves its minimum value $(\text{tr}(B^{\frac{1}{2}}))^2$ if and only if $\Omega^{-\frac{1}{2}} B^{\frac{1}{2}} = \zeta \Omega^{\frac{1}{2}}$ for some constant $\zeta$ and $\text{tr}(\Omega) = 1$. So $\Omega$ can be obtained from
\[
\Omega = \left( \frac{\beta \beta^T}{\text{tr}(\beta \beta^T)^{\frac{1}{2}}} \right)^{\frac{1}{2}}.
\]

6.2 Graph-SVM

Graph-SVM is the second model proposed to overcome the instability of $l_1$-SVM. As mentioned earlier, $l_1$-SVM shows instability in selecting informative features in presence of correlated groups of features because it randomly assigns a nonzero weight to a single feature among a group of correlated features and so with small changes in dataset, another feature maybe selected from the correlated group. However, an unbiased model should assign similar coefficient to them. In our model, we incorporate the modified version of OSCAR (Octagonal Shrinkage and Clustering Algorithm for Regression) i.e. Graph Oscar (GOSCAR) regularizer (Yang et al., 2012) into the SVM framework. Given an undirected graph, we aim to build a classification model that uses the information of the graph structure to estimate the nonzero coefficients of the model as well as identify the feature groups. We define $(N, E)$ as undirected graph, where $N = \{1, 2, \ldots, p\}$ is a set of nodes, and $E$ is the set of edges. Node $i$ corresponds to the $i$th column of the feature matrix. If nodes $i$ and $j$ are connected by an edge in $E$, then features $d_i$ and $d_j$ tend to be grouped. We refer to our
proposed model as **graph-SVM** formulated as

\[
\min_{\beta_0, \beta} \frac{1}{n} \sum_{i=1}^{n} (1 - y_i (\beta_0 + x_i^T \beta))_+ + \lambda_1 \|\beta\|_1 + \lambda_2 \sum_{(i,j) \in E} \max\{|\beta_i|, |\beta_j|\},
\]

(6.10)

where \( \lambda_1 \) and \( \lambda_2 \) are tuning parameters. The \( l_1 \) regularizer encourages sparsity and the pairwise \( l_\infty \) regularizer encourages the coefficients to be equal (Bondell and Reich, 2008). We can write

\[
\max\{|\beta_i|, |\beta_j|\} = \frac{1}{2} (|\beta_i + \beta_j| + |\beta_i - \beta_j|).
\]

The right side of the above equation can be written as \(|u^T \beta| + |v^T \beta|\), where each \( u, v \) vector only has two non-zero entries \( u_i = u_j = \frac{1}{2}, v_i = -v_j = \frac{1}{2} \). So we can write (6.10) as

\[
\min_{\beta_0, \beta} \frac{1}{n} \sum_{i=1}^{n} (1 - y_i (\beta_0 + x_i^T \beta))_+ + \lambda_1 \|\beta\|_1 + \lambda_2 \|T \beta\|_1,
\]

(6.11)

where \( T \) is a sparse matrix constructed from the edge set \( E \).

### 6.2.1 Optimization

We use the alternating direction method of multipliers (ADMM) to solve the proposed method (Boyd *et al.*, 2011). ADMM attempts to solve a large problem by decomposing it into a small subproblems and coordinate the local solutions to find the global optimal solution. As hinge loss and \( l_1 \) norm terms in (2.13) are nondifferentiable, we introduce some auxiliary variables to handle these nondifferentiable terms. Suppose \( X = (x_{ij})_{i=1,j=1}^{n,p} \) and \( Y \) is a diagonal matrix, where its diagonal elements are the vector \( y = (y_1, \ldots, y_n)^T \). The unconstrained optimization problem in (2.13) is equivalent to the following constrained optimization problem

\[
\min_{\beta_0, \beta} \frac{1}{n} \sum_{i=1}^{n} (a_i)_+ + \lambda_1 \|p\|_1 + \lambda_2 \|q\|_1
\]

s.t. \( a = 1 - Y(X \beta + \beta_0 1) \), \( p = \beta \), \( q = T \beta \),

(6.12)
Algorithm 6.1 ADMM for graph-SVM

- Initialize $\beta, \beta_0, a, p, q, u, v, z$

- repeat
  - Update $\beta_0^{k+1}$ and $\beta^{k+1}$ by solving equation (6.14)
  - Update $a^{k+1}$ by solving equation (6.16)
  - Update $p^{k+1}$ and $q^{k+1}$ by solving equation (6.17)
  - Update $u^{k+1}, v^{k+1}$ and $z^{k+1}$ by solving equation (6.18)

- until convergence

where $a = (a_1, \ldots, a_n)$ and $1$ is the column vector of 1’s with length $n$. The Lagrangian function of the above equation can be written as:

$$
L(\beta_0, \beta, a, p, q, u, v, z) = \frac{1}{n} \sum_{i=1}^{n} (a_i)_+ + \lambda_1 \| p \|_1 + \lambda_2 \| q \|_1 + \langle u, 1 - Y(X\beta + \beta_0 1) - a \rangle + \langle v, \beta - p \rangle + \langle z, T\beta - q \rangle,
$$

where $u \in \mathbb{R}^n$ is the dual variable corresponding to $a = 1 - Y(X\beta + \beta_0 1)$, $v \in \mathbb{R}^p$ is the dual variable corresponding to $p = \beta$ and $z \in \mathbb{R}^p$ is the dual variable corresponding to $q = T\beta$. $\langle \cdot, \cdot \rangle$ denotes the standard inner product in Euclidean space. The augmented Lagrangian function of (6.12) is as follows

$$
\mathcal{L}(\beta_0, \beta, a, p, q, u, v, z) = L(\beta_0, \beta, a, p, q, u, v, z) + \frac{\rho}{2} \| 1 - Y(X\beta + \beta_0 1) - a \|^2 + \frac{\rho}{2} \| \beta - p \|^2 + \frac{\rho}{2} \| T\beta - q \|^2.
$$

The last three terms in (6.13) penalize the violation of constraints $a = 1 - Y(X\beta + \beta_0 1)$, $p = \beta$ and $q = T\beta$, respectively and $\rho > 0$ is a parameter.

**Update $\beta_0$ and $\beta$:** We update $\beta_0$ and $\beta$ by minimizing the augmented Lagrangian function in (6.13) while other variables are fixed, as in this condition the objective function is quadratic and differentiable, we can find its optimal solution by solving
6.2. Graph-SVM

a set of linear equations:

\[
\begin{bmatrix}
\rho(X^T X + I + T^T T) & \rho(X^T 1) \\
\rho(1^T X) & \rho n
\end{bmatrix}
\begin{bmatrix}
\beta \\
\beta_0
\end{bmatrix} =
\begin{bmatrix}
A \\
B
\end{bmatrix}
\] (6.14)

where \( A = X^T Y u^k - v^k - T^T z^k - \rho X^T Y (a^k - 1) + \rho (p^k) + \rho T^T q^k \) and \( B = 1^T Y u^k - \rho 1^T Y (a^k - 1) \). We use the conjugate gradient algorithm to solve the above linear system.

**Update a:** \( a^{k+1} \) can be obtained by solving

\[
\begin{align*}
\arg \min_a & \frac{1}{n} \sum_{i=1}^n (a_i) + \left\langle u^k, 1 - Y (X \beta + \beta_0 1) - a \right\rangle \\
& + \frac{\rho}{2} \| 1 - Y (X \beta + \beta_0 1) - a \|^2.
\end{align*}
\] (6.15)

We can solve (6.15), using the following proposition (Ye et al., 2011).

**Proposition 5.** If \( h_\lambda(w) = \arg \min_{x \in \mathbb{R}} \lambda x + \frac{1}{2} \| x - w \|^2 \), then we have \( h_\lambda(w) = w - \lambda \) for \( w > \lambda \), \( h_\lambda(w) = w \) for \( w < \lambda \), and \( h_\lambda(w) = 0 \) otherwise.

Also, we have

\[
\frac{\| u \|^2}{2 \rho} + \frac{\rho}{2} \| 1 - Y (X \beta^{k+1} + \beta_0^{k+1} 1) - a \|^2
\]
\[
+ \left\{ \left\langle u^k, 1 - Y (X \beta^{k+1} + \beta_0^{k+1} 1) \right\rangle - a \right\}
\]
\[
= \frac{\rho}{2} \| a - (1 + \frac{u}{\rho} - Y (X \beta^{k+1} + \beta_0^{k+1} 1)) \|^2.
\]

From the above equation and Proposition 5, we can update \( a \) as follows:

\[
a^{k+1} = H_{\frac{1}{\rho}} (1 + \frac{u^k}{\rho} - Y (X \beta^{k+1} + \beta_0^{k+1} 1)),
\] (6.16)

where \( H_{\lambda}(w) = (h_{\lambda}(w_1), h_{\lambda}(w_2), \ldots, h_{\lambda}(w_n))^T, \forall w \in \mathbb{R}^n \).

**Update p and q:** We use the soft-thresholding operator to update \( p \) and \( q \)

\[
p^{k+1} = S_{\lambda_1/\rho}(\beta^{k+1} + \frac{v^k}{\rho}), \quad q^{k+1} = S_{\lambda_2/\rho}(T \beta^{k+1} + \frac{z^k}{\rho}),
\] (6.17)
where the soft-thresholding operator is defined as $S_{\lambda}(x) = \text{sign}(x) \max(|x| - \lambda, 0)$.

Update $u$, $v$, $z$:

$$
\begin{align*}
    u^{k+1} & = u^k + \rho(1 - Y(X\beta^{k+1} + \beta_0^{k+1}1) - a^{k+1}), \\
    v^{k+1} & = z^{k+1}v^k + \rho(\beta^{k+1} - p^{k+1}), \\
    z^{k+1} & = z^k + \rho(T\beta^{k+1} - q^{k+1}).
\end{align*}
$$

A step-by-step summary of the above optimization is provided in Algorithm 6.1.

### 6.2.2 Convergence Analysis

Convergence property of the Algorithm 6.1 can be derived from the standard convergence theory of the alternating direction method of multipliers (Eckstein and Bertsekas, 1992).

**Theorem 4.** Assume that the equation (2.13) has at least one solution $(\beta_0^*, \beta^*)$ and $\lambda_1 > 0$ and $\lambda_2 > 0$. Then Algorithm 6.1 has the following property:

$$
\lim_{k \to \infty} \frac{1}{n} \sum_{i=1}^{n} (1 - y_i(\beta_0^k + x_i^T\beta^k))_+ + \lambda_1 \|\beta^k\|_1 + \lambda_2 \|T\beta^k\|_1 = \frac{1}{n} \sum_{i=1}^{n} (1 - y_i(\beta_0^* + x_i^T\beta^*))_+ + \lambda_1 \|\beta^*\|_1 + \lambda_2 \|T\beta^*\|_1.
$$

Furthermore, whenever (2.13) has a unique solution, we have: $\lim_{k \to \infty} \|(\beta_0^k, \beta^k) - (\beta_0^*, \beta^*)\| = 0$.

### 6.3 Experiments

In this section, we compare the predictive performance of C-SVM and graph-SVM with some baseline algorithms namely Lasso (Tibshirani, 1996), $l_1$-SVM (Zhu et al., 2004) and Elastic net SVM (EN-SVM) (Wang et al., 2006) on both synthetic and
real-world datasets. In the following subsections, we first describe the datasets used in this chapter, then we briefly introduce evaluation measures. Following that, we talk about experimental settings used for the evaluation of different methods and finally, we discuss the experimental results.

6.3.1 Datasets

Synthetic Dataset (Simulation of Microarray Dataset) Analysis of microarray datasets is extremely useful for biomarker discovery and answering diagnosis and prognosis questions. In this section, we use a simulated dataset developed in (Di Camillo et al., 2012) to compare the stability and prediction performance of pg-EN with other baseline algorithms. The effect of heterogeneity and variability of synthetic microarray data consisting of two balanced groups of 50 subjects is simulated in this dataset. To this end, each subject is simulated using a regulatory network of \( p = 10000 \) genes using the simulator described in (Di Camillo et al., 2009). The topology of the network is specified by a connectivity matrix \( W \), where \( w_{ij} \) would be non zero if gene-product \( j \) directly affects the expression of gene \( i \). Following this, a population of \( N = 1000 \) instances is simulated as follows. Subjects are modeled as regulatory networks of \( p = 10000 \) nodes and the first generation of population consisted of \( N \) individuals with identical connectivity matrix \( W \) and with \( p \) dimensional vectors of expression values obtained. The subsequent generations were produced by iteration of three steps: random pairing, mutation of a randomly chosen subsequent of subjects and selection of the surviving subjects. These steps were applied only to a sub-network size \( p = 900 \), indicated as \( W_{900} \) in the following. These three steps are discussed in more details in (Di Camillo et al., 2012). When the base population was simulated, we define two groups of 500 subjects. The pathological condition is simulated by knocking out or knocking down six target hubs, which are defined as the genes with the highest out-degree and expression value at steady state higher than 0.88. Diseased subjects had 4, 5, or 6 genes belonging to \( W_{900} \) that were knocked out or down. In our studies, we partitioned the two groups of 500 healthy and 500 diseased subjects into 10 balanced non-overlapping datasets of size 50 subjects.
6.3. Experiments

Real-World Dataset  For evaluating the performance of the C-SVM and graph-SVM on real-world applications, we conduct our experiments on AMI and Cancer datasets, described in section 3.4.1.2 of chapter 3. In addition, we use the breast cancer dataset, collected by Van De Vijver et al. (2002) and described in section 4.2.1.2 of chapter 4.

6.3.2 Evaluation Measures

Our two proposed models and the baselines are evaluated in terms of their stability in feature selection and predictive performance. The evaluation measures are described below:

Stability Measures  To compare the stability performance of C-SVM and graph-SVM with other baselines, we use two stability measures, Spearman’s rank correlation (SRCC) and Jaccard similarity measure (JSM). These stability measures are described in detail in section 2.2.4.1 of chapter 2.

Predictive Performance Measures  To compare the predictive performance of our proposed models with other baselines, we use five evaluation methods including Precision or Positive Predictive Value (PPV), Sensitivity, Specificity, F1 score and AUC. These classification performances are discussed in details in section 2.2.4.2 of chapter 2.

6.3.3 Experimental Settings

For Synthetic dataset, we partition two groups of 500 healthy and 500 diseased subjects into 10 balanced non-overlapping datasets of size 50 subjects. We do this procedure 10 times, so finally we will have 100 datasets of 50 subjects. We use external cross-validation loops with separate training and test phases. The final results are reported as an average over 100 datasets.

Turning to real datasets, we randomly divide data into training set and test set.
6.3. Experiments

All the models are trained on the training set and their performances are evaluated using the test set. Parameters of the models are selected using 5-fold cross validation on the training set. The random splitting of real datasets is done 100 times and the results (stability and predictive performances) are reported as an average over these 100 splits.

6.3.4 Graph Generation for graph-SVM

As we mentioned earlier, graph-SVM model uses the information of the graph structure to estimate the nonzero coefficients of the model as well as identify the feature groups. To this end, we build a graph based on the similarities between features. We use the Jaccard index that measures the percentage of agreement among feature vectors. Considering two feature vectors $F_i$ and $F_j$, it is defined as $J_{i,j} = \frac{A_{11}}{A_{01} + A_{10} + A_{11}}$, where $A_{11}$ is the number of non-zero components in $F_i$ and $F_j$, $A_{01}$ is the number of non-zero components in $F_j$ but not in $F_i$ and $A_{10}$ is the number of non-zero components in $F_i$ but not in $F_j$. Using this, we construct an undirected graph where nodes represent the features and edges represent the Jaccard score between features. A proportion of the graph constructed for AMI dataset is shown in Figure 6.1.

6.3.5 Estimated Covariance Matrix by C-SVM

We show the estimated covariance matrix for real-world datasets in Figure 6.2. For better representation, we show the correlation matrix computed from $\Omega$ matrix by standardizing its values as

$$\Omega_{st}(i,j) = \frac{\Omega(i,j)}{\sqrt{\Omega(i,j)\Omega(i,j)}}.$$ 

As feature names for AMI and Cancer datasets are available, we further discuss about some of the correlated features estimated in their $\Omega$ matrix for these two datasets. In $\Omega$ matrix of AMI, the first group are the features related to cardiac troponin and the last group are features related to discharge sodium values. Both of these features are reported as important risk factors for Mayocardial infarction (Mair et al., 1991; Eapen et al., 2013). In $\Omega$ matrix obtained for Cancer dataset, the
first group are the features related to diabetes mellitus and the last group are the features related to anemia, where both of these features are important risk factors for cancer survival prediction (Coughlin et al., 2004; Caro et al., 2001).

6.3.6 Experimental Results

**Stability Performance**  Table 6.1 shows the stability performance of Covariance-SVM and graph-SVM compared to other baselines in terms of SRCC and JSM. For synthetic dataset, the best stability performance in terms of JSM belongs to C-SVM with JSM=0.495, which is closely followed by graph-SVM with JSM=0.486. In this
dataset, graph-SVM shows the best stability performance in terms of SRCC (0.548) and C-SVM is runner up (0.540). For breast cancer dataset, the best stability performance in terms of both JSM and SRCC belongs to C-SVM with JSM=0.524 and SRCC=0.601. This is followed by graph-SVM with JSM=0.518 and SRCC=0.581. Turning to cancer dataset, graph-SVM shows the best stability in terms of JSM (0.520), which is followed by C-SVM (JSM=0.512). However, the stability of C-SVM in terms of SRCC is the best with SRCC=0.627. For AMI dataset, the best stability performance in terms of JSM belongs to C-SVM with JSM=0.531 and in terms of SRCC belongs to graph-SVM with SRCC=0.531. Table also shows that the least stability performance in terms of both JSM and SRCC belongs to Lasso and \( l_1 \)-SVM that use \( l_1 \)-norm penalty. As we mentioned earlier, such instability is due to the tendency of \( l_1 \)-norm in selecting one feature from a group of correlated features randomly. Therefore, small changes in data result in a significant change in selected features leading to unstable models.
Table 6.1: Stability performance of C-SVM and graph-SVM compared to the baselines for synthetic and real-world datasets. Means and standard error over 50 iterations are reported.

<table>
<thead>
<tr>
<th></th>
<th>Lasso</th>
<th>$l_1$-SVM</th>
<th>ENSVM</th>
<th>C-SVM</th>
<th>graph-SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synthetic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JSM</td>
<td>0.347 (0.009)</td>
<td>0.370 (0.011)</td>
<td>0.425 (0.020)</td>
<td><strong>0.495 (0.020)</strong></td>
<td>0.486 (0.020)</td>
</tr>
<tr>
<td>SRCC</td>
<td>0.204 (0.010)</td>
<td>0.231 (0.014)</td>
<td>0.489 (0.018)</td>
<td>0.540 (0.010)</td>
<td><strong>0.548 (0.014)</strong></td>
</tr>
<tr>
<td>Breast cancer</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JSM</td>
<td>0.376 (0.009)</td>
<td>0.365 (0.007)</td>
<td>0.462 (0.024)</td>
<td><strong>0.524 (0.015)</strong></td>
<td>0.518 (0.035)</td>
</tr>
<tr>
<td>SRCC</td>
<td>0.251 (0.014)</td>
<td>0.237 (0.015)</td>
<td>0.476 (0.008)</td>
<td><strong>0.601 (0.012)</strong></td>
<td>0.581 (0.009)</td>
</tr>
<tr>
<td>Cancer</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JSM</td>
<td>0.428 (0.010)</td>
<td>0.438 (0.008)</td>
<td>0.442 (0.028)</td>
<td>0.512 (0.016)</td>
<td><strong>0.520 (0.018)</strong></td>
</tr>
<tr>
<td>SRCC</td>
<td>0.258 (0.021)</td>
<td>0.261 (0.015)</td>
<td>0.523 (0.009)</td>
<td><strong>0.627 (0.015)</strong></td>
<td>0.616 (0.010)</td>
</tr>
<tr>
<td>AMI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JSM</td>
<td>0.378 (0.011)</td>
<td>0.372 (0.009)</td>
<td>0.428 (0.021)</td>
<td><strong>0.531 (0.016)</strong></td>
<td>0.519 (0.030)</td>
</tr>
<tr>
<td>SRCC</td>
<td>0.243 (0.026)</td>
<td>0.228 (0.027)</td>
<td>0.518 (0.011)</td>
<td>0.527 (0.014)</td>
<td><strong>0.531 (0.010)</strong></td>
</tr>
</tbody>
</table>

**Predictive Performance**  We have shown the predictive performance of C-SVM and graph-SVM in Tables 6.2 and 6.3 in terms of different classification measures namely, PPV, sensitivity, F1 score, specificity and AUC. As the tables imply, on average the best predictive performance belongs to our two proposed methods. More specifically, for synthetic data the best predictive performance in terms of PPV, F1 score and specificity captured by graph-SVM, while C-SVM shows better predictive performance in terms of sensitivity and AUC. For Breast cancer dataset, the best predictive performance in terms of PPV, sensitivity, F1 score and specificity belongs to C-SVM and in terms of AUC belongs to graph-SVM. Turning to cancer dataset, the predictive performance of C-SVM and graph-SVM is comparable in terms of
6.3. Experiments

Table 6.2: Classification performances of C-SVM and graph-SVM compared to the baseline algorithms for Synthetic and Breast cancer datasets. Mean and standard error over 100 iterations are reported.

<table>
<thead>
<tr>
<th></th>
<th>PPV</th>
<th>Sensitivity</th>
<th>F1 score</th>
<th>Specificity</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synthetic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lasso</td>
<td>0.815 (0.007)</td>
<td>0.817 (0.010)</td>
<td>0.816 (0.011)</td>
<td>0.829 (0.009)</td>
<td>0.779 (0.010)</td>
</tr>
<tr>
<td>$l_1$-SVM</td>
<td>0.817 (0.008)</td>
<td>0.821 (0.009)</td>
<td>0.819 (0.011)</td>
<td>0.849 (0.009)</td>
<td>0.790 (0.006)</td>
</tr>
<tr>
<td>ENSVM</td>
<td>0.825 (0.007)</td>
<td>0.820 (0.011)</td>
<td>0.822 (0.009)</td>
<td>0.850 (0.010)</td>
<td>0.795 (0.009)</td>
</tr>
<tr>
<td>C-SVM</td>
<td>0.830 (0.010)</td>
<td><strong>0.838</strong> (0.012)</td>
<td><strong>0.834</strong> (0.012)</td>
<td><strong>0.860</strong> (0.009)</td>
<td><strong>0.818</strong> (0.010)</td>
</tr>
<tr>
<td>graph-SVM</td>
<td><strong>0.837</strong> (0.006)</td>
<td>0.834 (0.008)</td>
<td><strong>0.835</strong> (0.010)</td>
<td><strong>0.867</strong> (0.009)</td>
<td>0.815 (0.007)</td>
</tr>
<tr>
<td>Breast cancer</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lasso</td>
<td>0.307 (0.007)</td>
<td>0.421 (0.009)</td>
<td>0.355 (0.008)</td>
<td>0.820 (0.011)</td>
<td>0.800 (0.003)</td>
</tr>
<tr>
<td>$l_1$-SVM</td>
<td>0.308 (0.009)</td>
<td>0.425 (0.009)</td>
<td>0.357 (0.010)</td>
<td>0.819 (0.007)</td>
<td>0.804 (0.007)</td>
</tr>
<tr>
<td>ENSVM</td>
<td>0.312 (0.010)</td>
<td>0.429 (0.007)</td>
<td>0.361 (0.008)</td>
<td>0.829 (0.009)</td>
<td>0.806 (0.001)</td>
</tr>
<tr>
<td>C-SVM</td>
<td><strong>0.335</strong> (0.008)</td>
<td><strong>0.446</strong> (0.010)</td>
<td><strong>0.382</strong> (0.011)</td>
<td><strong>0.832</strong> (0.011)</td>
<td><strong>0.819</strong> (0.009)</td>
</tr>
<tr>
<td>graph-SVM</td>
<td>0.329 (0.009)</td>
<td>0.441 (0.009)</td>
<td>0.376 (0.010)</td>
<td>0.830 (0.008)</td>
<td><strong>0.825</strong> (0.006)</td>
</tr>
</tbody>
</table>

PPV and is highest among other methods. In terms of sensitivity and F1 score, the best predictive performance belongs to C-SVM, while in terms of specificity graph-SVM has the best predictive performance. However, in this dataset the AUC of ENSVM is the best among other methods, which is closely followed by C-SVM and graph-SVM. Similarly, for AMI dataset, again ENSVM shows the best result in terms of PPV and C-SVM and graph-SVM are runner ups. In terms of other classification measures i.e. sensitivity, specificity, F1 score and AUC, C-SVM obtains the best results which is closely followed by graph-SVM.
6.4 Conclusion

In this chapter, we proposed a new model for stable clinical prediction using SVM, a promising classifier employed in clinical domain. It is shown that combining SVM with $l_1$-norm regularization enables the model to perform feature selection and classification simultaneously. However, due to the properties of $l_1$-norm penalty the resulting model would be unstable. Realizing that the model instability is caused by feature correlations, we propose two models to stabilize $l_1$-norm SVM, namely Covariance-SVM and graph-SVM. Both models focus on minimizing the variance in feature subsets and model estimation parameters. The proposed models are for-
mulated as convex optimization problems and alternating algorithms are proposed to solve them. We compare the stability and prediction performance of these two models with several state-of-the-art models namely, Lasso, $l_1$-norm SVM and Elastic net SVM using synthetic and real-world datasets. We show that as these two models perform feature selection at the group level they show better stability compared to the baselines. We also note that the grouping of correlated features performed by C-SVM and graph-SVM can reduce the estimator's variance and hence results in better predictive performance compared to the baselines.

In the next chapter, we focus on another important problem in healthcare domain i.e. prediction of toxicity in cancer patients which is a crucial problem occurred during cancer treatment. Since the underlying causes of toxicities is not well understood, oncologists cannot predict occurrence of toxicity in a patient. To address this problem, we propose a new stable predictive model to predict toxicity in cancer patients.
Chapter 7

Stable Feature Selection in Multiple-Instance Learning with Application to Toxicity Prediction for Cancer Patients

The focus in the previous chapters was on stable clinical prediction models that were built for single instance supervised learning problems, where a training data set consists of a set of instances and there is a label for each individual instance. However, in many problems, it is impossible to accurately and consistently assign a label to each individual instance. For example, to classify molecules in the context of drug design, each molecule is represented by several possible conformations. In this application even though we can experimentally evaluate the efficiency of each molecule, it is impossible to control for individual conformations (Dietterich et al., 1997). The other example is text categorization. Usually, documents related to a specific topic also contain a relevant passage. However, class labels are mostly available for the whole documents and are rarely associated on the passage level (Andrews et al., 2002). These types of applications that share the same level of label ambiguity are considered as another type of supervised classification problems called multiple instance learning (MIL). In MIL problems, class labels are associated with sets of instances, called bags, instead of individual instances. A positive bag
means that at least one example in the bag is positive. The key challenge in MIL problems is to cope with the uncertainty of not knowing which of the instances in a positive bag are the actual positive instances and which ones are not.

In this chapter, we engage in one such problem, i.e. toxicity prediction for cancer patients. A patient diagnosed with cancer undergoes several types of treatments such as chemotherapy and radiotherapy. These treatments themselves can cause severe complications and adverse effects, called toxicities. The data related to these treatments along with disease conditions and patient demography can be used to predict toxicities in future. It is worth noting that toxicity can be caused due to a specific treatment (e.g. chemotherapy) on a given day, yet treatments on other days (e.g. radiotherapy) may not cause any adverse effects. Therefore, toxicity prediction can be formulated as a MIL problem due to ambiguity in its label assignments. The toxicity data is viewed as a set of instances, where only a subset of the instances are responsible for the outcome. Formulating this problem as MIL enables us to model the impact of a subset of treatments on toxicity. Since toxicity is observed for the whole block, each bag representing the block of treatments has one label e.g. “toxicity observed” or “no toxicity”.

We use the data in the form of electronic medical records (EMRs), which contain rich information about patients and are high dimensional (Jensen et al., 2012). However, a considerable amount of EMRs is irrelevant and redundant that can mislead a machine learning algorithm and negatively affect its performance. MIL algorithms are not exception and to build an accurate MIL algorithm feature selection is essential. Few feature selection methods have been applied to multiple instance problems (Chen et al., 2006; Raykar et al., 2008; Yuan et al., 2007). Nevertheless, a relatively neglected issue is their ability to select stable features. Stability is particularly important for knowledge discovery problems, where features carry intuitive meanings and actions are taken based on these features. In case of cancer care, an stable feature selection technique would increase the confidence of oncologists about selected risk factors and enable them to confidently investigate them further to identify the underlying reasons of toxicity.

Building clinical prediction models from high dimensional data, such as EMR, often call sparse methods based on $l_1$-norm due to its sparsity-inducing property and convexity (Ye and Liu, 2012). Yet, this regularization shows instability in presence
of groups of highly correlated features (Yuan and Lin, 2006; Zhao and Yu, 2006).

To address this problem, in this chapter, we propose a new multiple-instance learning model for toxicity prediction that can perform stable feature selection to provide oncologists with reliable list of risk factors, involved in occurring toxicity during treatment process. To do so, we introduce a regularization formulation to improve instability of $l_1$-norm penalty by encouraging similarities between features according to their relatedness. Our model captures the relatedness between features using a covariance matrix. The introduced regularization term enables multiple-instance framework to not only perform feature selection but also captures the positive and negative correlation between features to improve the feature selection stability. We formulate the model using a constrained optimization problem and propose a solution based on an alternating optimization algorithm.

This chapter is organized as follows. We propose our new stable multiple-instance learning technique for toxicity prediction in cancer patients in section 7.1. Further, in this section, we present an iterative solution to optimize the cost function of our proposed method. Section 7.2 describes the experimental results conducted for both synthetic and real-world datasets and compares them with the current state-of-the-art methods along with further discussion about selected risk factors by our proposed method. Conclusions are drawn in Section 7.3.

### 7.1 Methodology

We represent an instance as a feature vector $x \in \mathbb{R}^p$. A bag containing $K$ instances is denoted by boldface $\mathbf{x} = \{x_j \in \mathbb{R}^p\}_{j=1}^K$. The label of a bag is denoted by $y = \{0, 1\}$. The training data $\mathcal{D}$ consists of $N$ bags $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^N$, where $\mathbf{x}_i = \{x_{ij} \in \mathbb{R}^p\}_{j=1}^K$ is a bag containing $K_i$ instances with the label $y_i \in \{0, 1\}$.

#### 7.1.1 Regularized Logistic Regression for MIL

In MIL framework, we have the concept of bags, where all the examples in a bag share the same label. A positive bag means that at least one example in the bag is
positive. The probability that a bag contains at least one positive instance is one
minus the probability that all of them are negative. So, the posterior probability
for a positive bag can be written as
\[
p(y = 1|x) = 1 - \prod_{j=1}^{K}[1 - \sigma(\beta^T x_j)], \tag{7.1}
\]
where, \(\sigma(z) = 1/(1 + \exp(-z))\) and the bag \(x = \{x_j\}_{j=1}^{K}\) contains \(K\) examples. The
posterior probability for a negative bag can be written as
\[
p(y = 0|x) = \prod_{j=1}^{K}[1 - \sigma(\beta^T x_j)]. \tag{7.2}
\]
This model is known as \textit{Noisy-OR} and has been used previously in (Maron and

7.1.1.1 Objective Function

Given the training data \(D\), we want to estimate parameter \(\beta\). This parameter is
typically estimated using maximum likelihood estimation as
\[
\hat{\beta} = \arg \max_{\beta} p(D \mid \beta) = \arg \max_{\beta} \left[ \log p(D \mid \beta) \right]. \tag{7.3}
\]
We define the probability that the \(i\)th bag \(x_i\) is positive as \(p_i = p(y_i = 1 \mid x_i) =
1 - \prod_{j=1}^{K_i}[1 - \sigma(\beta^T x_{ij})]\). Assuming that the training bags are independent the log-
likelihood can be written as
\[
\log p(D \mid \beta) = \sum_{i=1}^{N} y_i \log p_i + (1 - y_i) \log(1 - p_i). \tag{7.4}
\]
The maximum likelihood solution can exhibit overfitting especially for high dimen-
sional data. We can address this by using an \(l_1\) penalty, which results in both
shrinkage and automatic feature selection simultaneously and thus avoids overfit-
ting. We propose to use the following regularized objective function:
\[
\arg \min_{\beta} \sum_{i=1}^{N} C_{\text{bag}_i} + \lambda \|\beta\|_1, \tag{7.5}
\]
7.1 Methodology

where $C_{bag_i}$ denotes the cost associated with the $i$th bag prediction. More specifically, it is denoted as

$$C_{bag_i} = -\{y_i \log p_i + (1 - y_i) \log p_i\}$$  \hspace{1cm} (7.6)

7.1.2 Stabilized $l_1$-norm MIL

When there are groups of highly correlated features in data, the $l_1$ norm penalty term causes feature instability behavior. This is because that $l_1$ penalty tends to assign a nonzero weight to only a single feature among a group of correlated features. We can address this problem by encouraging similarities between features based on their relatedness, where the feature relatedness can be captured using a feature covariance matrix. With this solution in mind, we modify our formulation in (7.5) to the following minimization problem:

$$\arg \min_{\beta, \Omega} J(\beta) = \sum_{i=1}^{N} C_{bag_i} + \lambda \|\beta\|_1 + \frac{\eta}{2} \beta^T \Omega^{-1} \beta,$$

s.t. $\Omega \succeq 0, \text{tr}(\Omega) = 1$  \hspace{1cm} (7.7)

where $\Omega$ is a covariance matrix that models the relationships between features. The first constraint in (7.7) requires $\Omega$ to be a semi-definite matrix as it is covariance matrix and the second constraints restricts the complexity of the problem and makes the solution unique. The parameters $\lambda$ and $\eta$ are tuning parameters. We term the above model stabilized MIL. Since it uses covariance matrix for stabilization we refer to it as MIL-C.

7.1.2.1 Optimization

As the Noisy-OR model is not convex, its use makes the objective function (7.7) non-convex. However, the main goal of learning is a good generalization on the unseen data and indeed non-convexity does not impose any challenge in general (Yakhnenko and Honavar, 2011). Further, it is shown that non-convex problems have better performance and scalability compared to convex problems and temptation of convex approaches should not discourage researchers from investigating non-convex methods (Collobert et al., 2006).
7.1. Methodology

Optimizing (7.7) with respect to $\beta$ and $\Omega$ at the same time is not straightforward. To tackle this problem, we propose an iterative method that optimizes the objective function with respect to $\beta$ when $\Omega$ is fixed, and optimizes the objective function with respect to $\Omega$ when $\beta$ is fixed. This optimization procedure is detailed in the following:

Optimizing w.r.t $\beta$ when $\Omega$ is fixed: When the $\Omega$ is fixed, the optimization problem for finding $\beta$ becomes an unconstrained optimization problem, and can be stated as:

$$\arg \min_{\beta} J(\beta) = \sum_{i=1}^{N} C_{\text{bag}} + \lambda \| \beta \|_1 + \frac{\eta}{2} \beta^T \Omega^{-1} \beta. \quad (7.8)$$

Because of the non-differentiability of the $l_1$-norm penalty in (7.8), it is hard to solve the problem directly. We solve the above problem using the alternating direction method of multipliers (ADMM), which is developed by Gabay and Mercier (1976) and has been recently recognized as an efficient method for solving many problems with non-smooth regularization (Boyd et al., 2011). In order to derive an ADMM algorithm, we introduce an auxiliary variable to handle its non-differentiable term. We can reformulate the unconstrained problem in (7.8) into an equivalent constraint problem

$$\arg \min_{\beta} \sum_{i=1}^{N} C_{\text{bag}} + \lambda \| z \|_1 + \frac{\eta}{2} \beta^T \Omega^{-1} \beta. \quad (7.9)$$

s.t. $z = \beta$

The Lagrangian function of (7.9) is

$$L(\beta, z, v) = \sum_{i=1}^{N} C_{\text{bag}} + \lambda \| z \|_1 + \frac{\eta}{2} \beta^T \Omega^{-1} \beta + v^T (\beta - z), \quad (7.10)$$

where $v \in \mathbb{R}^p$ is a dual variable corresponding to the linear constraint $z = \beta$. The augmented Lagrangian function of (7.9) adds $\frac{\rho}{2} \| \beta - z \|_2^2$ to the Lagrangian function (7.10) to penalize the violation of linear constraint $z = \beta$. That is

$$\mathcal{L}(\beta, z, v) = L(\beta, z, v) + \frac{\rho}{2} \| \beta - z \|_2^2, \quad (7.11)$$

where $\rho > 0$ is the augmented Lagrangian parameter. For more convenience, we can combine the linear and quadratic terms in (7.11) (Boyd et al., 2011). Defining
7.1. Methodology

We have

\[ v^T r + \frac{\rho}{2} \| r \|^2 = \frac{\rho}{2} \| r + \frac{1}{\rho} v \|^2 - \frac{1}{2\rho} \| v \|^2 \]

(7.12)

\[ = \frac{\rho}{2} \| r + u \|^2 - \frac{1}{2} \rho \| u \|^2, \]

where, \( u = \left( \frac{1}{\rho} \right) v \) is the scaled dual variable. So we can write the scaled form of (7.12) as

\[ \mathcal{L}(\beta, z, u) = \sum_{i=1}^{N} -\{ y_i \log p_i + (1 - y_i) \log p_i \} + \lambda \| z \|_1 \]

(7.13)

\[ + \frac{\eta}{2} \beta^T \Omega^{-1} \beta + \frac{\rho}{2} \| \beta - z + u \|^2 + \frac{\rho}{2} \| u \|^2. \]

Although equations (7.11) and (7.13) are equivalent, the scaled form of ADMM has often shorter formulas in comparison with unscaled form. In this chapter, we use the scaled form of ADMM. The ADMM algorithm consists of the iterations

\[
\begin{cases}
\beta^{t+1} \leftarrow \arg \min_{\beta} \mathcal{L}(\beta, z^t, u^t) \\
z^{t+1} \leftarrow S_{\lambda/\rho}(\beta^{t+1} + u^t) \\
u^{t+1} \leftarrow u^k + (\beta^{t+1} - z^{t+1})
\end{cases}
\]

(7.14)

For the first equation in (7.14), it is equivalent to

\[ \arg \min_{\beta} \sum_{i=1}^{N} C_{\text{bag}_i} + \frac{\eta}{2} \beta^T \Omega^{-1} \beta + \frac{\rho}{2} \| \beta - z^t + u \|^2. \]

(7.15)

Because of non-linearity of the sigmoid in the above optimization problem, we do not have a closed form solution and we have to use gradient based methods to solve it. We optimize the above objective function using stochastic gradient descent with a varied learning rate strategy as follows:

In the \( t \)th iteration of our algorithm, we randomly choose a bag \((x_{m_t}, y_{m_t})\), where \( m_t \in \{1, 2, \ldots, n\} \) from the training set \( D \). Then our objective function in (7.15) is changed to an approximation based on the sample bag

\[ f(\beta; x_{m_t}) = C_{\text{bag}_{m_t}} + \frac{\eta}{2} \beta^T \Omega^{-1} \beta + \frac{\rho}{2} \| \beta - z + u \|^2. \]

(7.16)
Considering the gradient of the above approximation function, given by
\[ \nabla_t = -\sum_{j=1}^{K_m} x_{m_j} \left\{ \sigma(\beta^T x_{m_j})(y_{m_j} - p_{m_j}) \right\} + \eta \Omega^{-1} \beta + \rho(\beta - z + u), \] (7.17)
the weight vector can be updated using varied learning rate \( \delta_t = 1/((t + 1)\eta) \), that is \( \beta^{t+1} \leftarrow \beta^t - \delta_t \nabla_t \). In order to accelerate the rate of convergence in this part of optimization, we perform a projection operation of \( \beta \) on the \( l_2 \) ball of radius \( 1/\sqrt{\eta} \) as in (Shalev-Shwartz et al., 2011) via the following update,
\[ \beta^{t+1} \leftarrow \min\{1, 1/\|\beta^{t+1}\|\} \beta^{t+1}. \] (7.18)

**Optimizing w.r.t \( \Omega \) when \( \beta \) is fixed:** When \( \beta \) is fixed, the optimization problem w.r.t \( \Omega \) becomes
\[
\min_{\Omega} \quad \beta^T \Omega^{-1} \beta \\
\text{s.t.} \quad \Omega \succeq 0, \quad \text{tr}(\Omega) = 1
\] (7.19)
Assume \( H = \beta \beta^T \), since \( \beta^T \Omega^{-1} \beta = \text{tr}(\beta^T \Omega^{-1} \beta) = \text{tr}(\Omega^{-1} \beta \beta^T) \) and \( \text{tr}(\Omega) = 1 \), we have
\[
\text{tr}(\Omega^{-1} H) = \text{tr}(\Omega^{-1} H) \text{tr}(\Omega) = \text{tr}(\Omega^{-1/2} H^{1/2} \Omega^{-1/2}) \text{tr}(\Omega^{1/2} \Omega^{1/2}) \\
\geq (\text{tr}(\Omega^{-1/2} H^{1/2} \Omega^{1/2}))^2 = (\text{tr}(H^{1/2}))^2.
\]
The above inequality comes from the Cauchy-Schwarz inequality for the Frobenius norm. Furthermore, \( \text{tr}(\Omega^{-1} H) \) achieves its minimum value \( (\text{tr}(H^{1/2}))^2 \) if and only if \( \Omega^{-1/2} H^{1/2} = \alpha \Omega^{1/2} \) for some constant value \( \alpha \) and \( \text{tr}(\Omega) = 1 \). So the analytical solution for \( \Omega \) will be:
\[
\Omega = \frac{(\beta \beta^T)^{1/2}}{\text{tr}((\beta \beta^T)^{1/2})}. \] (7.20)
7.2 Experiments

In this section, we compare our proposed method with several state-of-the-art baseline algorithms in terms of both feature stability and predictive performances for toxicity prediction.

7.2.1 Baseline Methods

Different types of learning algorithms have been adapted to the multiple-instance learning problems. To compare MIL-C with other state-of-the-art multiple instance methods, we have used the following algorithms as baselines:

**MIL-Lasso** This model is a variant of our proposed method with $\eta = 0$. The objective function of MIL-Lasso is:

$$
\arg \min_{\beta} J(\beta) = \sum_{i=1}^{N} C_{bag_i} + \lambda \|\beta\|_1,
$$

where $C_{bag_i}$ is the cost associated with the $i$th bag prediction and is formally defined in section 7.1.1.1.

**MIL-EN** This is a variant of our proposed method, where $\Omega$ is set to be an identity matrix. In this case, the regularization term is identical to the regularization term of the Elastic net (Zou and Hastie, 2005). The objective function of MIL-EN is:

$$
\arg \min_{\beta} J(\beta) = \sum_{i=1}^{N} C_{bag_i} + \lambda \|\beta\|_1 + \eta \|\beta\|_2^2,
$$

where $C_{bag_i}$ is the cost associated with the $i$th bag prediction. This baseline is chosen to show the effectiveness of using covariance matrix $\Omega$.

**MIL-RVM** MIL-RVM is an extension of the relevance vector machine to multiple instance learning framework (Raykar et al., 2008). In this method, logistic regres-
sion is used as the base model and relying on the Bayesian automatic relevance
determination paradigm, the MIL-RVM has the ability to perform feature selection.

**MIL-Boosting** In this method, Xu and Frank (2004) proposed a variant of AdaBoost algorithm adapted for the multiple instance learning problems. In this method, we use C4.5 trees as the “weak” classifiers and 50 boosting iterations.

**MIL-SVM** This method, is a variant of SVM adapted for the multiple instance problems (Andrews et al., 2002). In our implementations we have used MIL-SVM with linear kernel and chosen the regularization parameters by 5-fold-cross-validation.

**MIL-LR** This method is a variant of logistic regression that uses soft-max function to combine posterior probabilities over the instances of a bag (Settles et al., 2008). The MIL-LR method uses $\alpha = 2.5$ for the softmax function and is optimized by minimizing squared loss via L-BFGS (Settles et al., 2008).

**$l_1$-norm LR** In order to compare the performance of the multiple-instance learning with single instance models, we used $l_1$-norm logistic regression ($l_1$-norm LR) for classification. The $l_1$-norm penalty enables this method to perform automatic feature selection. In this case, each instance takes the label of its bag and $l_1$-norm LR is trained using these instances. We also evaluate the model on test set at the instance level, by assigning bag levels to the test instances.

Of the above methods, MIL-SVM and MIL-LR do not have provision to do feature selection. Therefore, we only use these baselines for comparisons on predictive performance.

### 7.2.2 Evaluation Measures

We evaluate our proposed method and other baselines in terms of their feature selection stability and predictive performance. The evaluation measures are described
7.2. Experiments

below.

**Stability Measures**  We compare the stability performance of our proposed method with baselines using two stability measures, Spearman’s rank correlation coefficient (SRCC) and Jaccard similarity measure (JSM). We have described these stability measures in detail in section 2.2.4.1 of chapter 2.

**Predictive Performance Measures**  To compare the predictive performance of our proposed method with baselines we use five evaluation methods including Precision or Positive Predictive Value (PPV), Sensitivity, Specificity, F1 score and AUC. These classification performances are discussed in details in section 2.2.4.2 of chapter 2.

### 7.2.3 Simulation Study

To illustrate how correlation among features could lead to instability in \( l_1 \)-penalized methods, a synthetic dataset is generated that consists of groups of highly correlated features. We show that in presence of these correlated features our MIL-C method can successively stabilize the feature selection process. We simulate a multiple-instance learning problem where each \( p = 300 \)-dimensional instance is generated by one of the following Gaussian components: \( \mathcal{N}_1 = (\mu_1, \Sigma), \mathcal{N}_2 = (\mu_2, \Sigma), \mathcal{N}_3 = (\mu_3, \Sigma) \) and \( \mathcal{N}_4 = (\mu_4, \Sigma) \), where \( \mu_i, i \in \{1, \ldots, 4\} \) denote the mean of normal distribution and define as follows:

\[
\begin{align*}
\boldsymbol{\mu}_1 & = \left( +1, +1, \ldots, +1, 0, 0, \ldots, 0 \right)^T, \\
\boldsymbol{\mu}_2 & = \left( +1, \ldots, +1, -1, \ldots, -1, 0, 0, \ldots, 0 \right)^T, \\
\boldsymbol{\mu}_3 & = \left( -1, \ldots, -1, +1, \ldots, +1, 0, 0, \ldots, 0 \right)^T, \\
\boldsymbol{\mu}_4 & = \left( -1, -1, \ldots, -1, 0, 0, \ldots, 0 \right)^T.
\end{align*}
\]
Table 7.1: Average stability performance of the proposed method compared to other baselines in terms of SRCC and JSM for synthetic dataset. The standard error are shown in parentheses.

<table>
<thead>
<tr>
<th>Method</th>
<th>SRCC</th>
<th>JSM</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIL-C</td>
<td>0.5211 (0.007)</td>
<td>0.6325 (0.010)</td>
</tr>
<tr>
<td>MIL-EN</td>
<td>0.3202 (0.010)</td>
<td>0.6055 (0.008)</td>
</tr>
<tr>
<td>MIL-Lasso</td>
<td>0.2278 (0.015)</td>
<td>0.3408 (0.011)</td>
</tr>
<tr>
<td>MIL-RVM</td>
<td>0.2483 (0.011)</td>
<td>0.4072 (0.009)</td>
</tr>
<tr>
<td>MIL-Boosting</td>
<td>0.4285 (0.010)</td>
<td>0.5876 (0.005)</td>
</tr>
<tr>
<td>$l_1$-norm LR</td>
<td>0.2214 (0.007)</td>
<td>0.3297 (0.006)</td>
</tr>
</tbody>
</table>

And, $\Sigma$ is a covariance matrix, which is kept same for all the above distributions and equals to:

$$
\begin{bmatrix}
\Sigma^*_{10 \times 10} & 0_{10 \times (p-10)} \\
0_{(p-10) \times 10} & I_{(p-10) \times (p-10)}
\end{bmatrix}
$$

where in $\Sigma^*$ the diagonal elements are 1 and others are all equal to $\rho = 0.8$. With this synthetic setup, there will be a pairwise correlation of 0.8 between relevant features. Each bag comprises of at most 20 instances. A label is positive if it contains at least one instance from $N_1$. Otherwise, it is negative.

Using this model, we generate 20 positive and 20 negative bags with a total of 800 instances, which are divided as follows: 200 instances for training set, 200 instances for validation set and 400 instances for test set. The models are trained on the training set, the validation set is used to tune the parameters and the performances of the models are evaluated on the test sets. The results for stability and classification performances of each method are reported as an average over 50 different simulations trials.

**Stability Performance** Table 7.1, shows the average stability performance of MIL-C and other baselines for the synthetic data. As seen, our proposed method shows the highest stability with SRCC=0.5211 and JSM=0.6325. In terms of SRCC it is followed by MIL-Boosting (0.4285) and in terms of JSM it is followed by MIL-EN (0.6055). The table also implies that the least stability performances in terms of the
Table 7.2: Average classification performance of the proposed method compared to other methods for synthetic dataset. The numbers in brackets show the standard error over 50 iterations.

<table>
<thead>
<tr>
<th>Method</th>
<th>PPV</th>
<th>Sensitivity</th>
<th>F1 score</th>
<th>Specificity</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposed method</td>
<td>0.9091</td>
<td>0.8333</td>
<td>0.8696</td>
<td>0.9444</td>
<td>0.9464</td>
</tr>
<tr>
<td></td>
<td>(0.008)</td>
<td>(0.010)</td>
<td>(0.008)</td>
<td>(0.009)</td>
<td>(0.006)</td>
</tr>
<tr>
<td>MIL-EN</td>
<td>0.7857</td>
<td>0.9167</td>
<td>0.8462</td>
<td>0.8000</td>
<td>0.8847</td>
</tr>
<tr>
<td></td>
<td>(0.010)</td>
<td>(0.009)</td>
<td>(0.009)</td>
<td>(0.007)</td>
<td>(0.009)</td>
</tr>
<tr>
<td>MIL-Lasso</td>
<td>0.7790</td>
<td>0.8841</td>
<td>0.8282</td>
<td>0.7893</td>
<td>0.8756</td>
</tr>
<tr>
<td></td>
<td>(0.012)</td>
<td>(0.011)</td>
<td>(0.011)</td>
<td>(0.008)</td>
<td>(0.010)</td>
</tr>
<tr>
<td>MIL-RVM</td>
<td>0.7778</td>
<td><strong>0.8750</strong></td>
<td>0.8235</td>
<td>0.9375</td>
<td>0.8712</td>
</tr>
<tr>
<td></td>
<td>(0.008)</td>
<td>(0.007)</td>
<td>(0.010)</td>
<td>(0.008)</td>
<td>(0.006)</td>
</tr>
<tr>
<td>MIL-Boosting</td>
<td><strong>0.9091</strong></td>
<td>0.7143</td>
<td>0.8000</td>
<td>0.8000</td>
<td>0.8306</td>
</tr>
<tr>
<td></td>
<td>(0.007)</td>
<td>(0.009)</td>
<td>(0.008)</td>
<td>(0.010)</td>
<td>(0.009)</td>
</tr>
<tr>
<td>MIL-SVM</td>
<td>0.8182</td>
<td>0.6429</td>
<td>0.7200</td>
<td>0.8750</td>
<td>0.8098</td>
</tr>
<tr>
<td></td>
<td>(0.009)</td>
<td>(0.010)</td>
<td>(0.008)</td>
<td>(0.008)</td>
<td>(0.008)</td>
</tr>
<tr>
<td>MIL-LR</td>
<td>0.9000</td>
<td>0.5556</td>
<td>0.6871</td>
<td>0.9167</td>
<td>0.7995</td>
</tr>
<tr>
<td></td>
<td>(0.007)</td>
<td>(0.008)</td>
<td>(0.010)</td>
<td>(0.009)</td>
<td>(0.011)</td>
</tr>
<tr>
<td>l₁-norm LR</td>
<td>0.8082</td>
<td>0.6000</td>
<td>0.6887</td>
<td>0.7667</td>
<td>0.7553</td>
</tr>
<tr>
<td></td>
<td>(0.010)</td>
<td>(0.006)</td>
<td>(0.009)</td>
<td>(0.007)</td>
<td>(0.008)</td>
</tr>
</tbody>
</table>

both stability measures belong to MIL-Lasso (with SRCC=0.2278 and JSM=0.3408) and l₁-norm LR (with SRCC=0.2214 and JSM=0.3297) that both use l₁-norm regularizer. This instability is because of the intrinsic attitude of l₁-norm penalty in selecting randomly one feature among a group of highly correlated features.

**Classification Performance** Table 7.2 shows the average classification performance of our proposed method for synthetic dataset, comparing it with other baselines in terms of standard classification performances, Positive Predictive Value (PPV), sensitivity, F1 score, specificity and AUC. As seen, MIL-C could achieve the best predictive performance with F1 score=0.8696, specificity=0.9444 and AUC=0.9464. In terms of PPV, the performance of our method is comparable with MIL-Boosting and in terms of sensitivity it is runner up to MIL-RVM.
7.2. Experiments

Table 7.3: Features and labels used in learning toxicity predictors

<table>
<thead>
<tr>
<th>basic</th>
</tr>
</thead>
<tbody>
<tr>
<td>age, gender, marital_status, cancer types, tumor size, cancer stage, treatment_intent_type, metastasis_flag</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>treatment</th>
</tr>
</thead>
<tbody>
<tr>
<td>treatment duration, chemo duration, chemo intervals (mean and std.), radiation duration, the number of used drugs, the number of toxicities in past</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Diagnosis Codes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICD10 codes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>toxicity (ICD10 code)</th>
</tr>
</thead>
<tbody>
<tr>
<td>blood diseases: D60–D64,D70–D77</td>
</tr>
<tr>
<td>nervous system: G50–G59,G90–G99</td>
</tr>
<tr>
<td>digestive disorders: K00–K14</td>
</tr>
<tr>
<td>disorders of the skin and subcutaneous tissue: L55–L59</td>
</tr>
</tbody>
</table>

It is interesting to note that on this dataset, $l_1$-norm logistic regression (LR) outperforms MIL-EN and MIL-RVM in terms of PPV. It also shows better classification performance compared to MIL-LR in terms of sensitivity and F1 score. Similar observation is noted in (Ray and Craven, 2005).

7.2.4 Toxicity Problem

Dataset Description The cancer toxicity data is collected from a regional health service provider in Victoria, Australia. This dataset consists of 2001 cancer patients who have undergone both chemotherapy and radiation therapy during 2000-2015. The variables associated with each patient are explicitly described in Table 7.3. These variables consist of three parts: patient-specific basic information, treatment information and diagnosis codes in the form of International Classification of Disease 10 (ICD 10). This table also shows the details of the toxicity observed in terms of ICD codes. Chemotherapy usually causes adverse effects such as blood diseases, nervous system and digestive disorders chemotherapy, whereas radiation therapy often results in the disorders of the skin and subcutaneous tissue. We use this

---

1Ethics approval is obtained through university and the hospital (number 12/83).
2http://apps.who.int/classifications/icd10/browse/2016/en
7.2. Experiments

Figure 7.1: Feature construction for cancer toxicity data: prediction points are set every fortnight. Feature vectors \( \{i_1, i_2, \cdots, i_{14}\} \) form a bag. \( T_1, T_2, T_3, \ldots \) are the prediction points. The toxicity outlook is 28-days.

Experimental Settings  We do not use data at each prediction point independently since the number of positive samples is small at some prediction points (e.g. 9th and 10th prediction points). We merge data at all prediction points and use the
7.2. Experiments

Figure 7.2: The number of patients in each prediction point for toxicity dataset.

data to train a single model for all prediction points. We employ the idea that data at different prediction points have a shared representation (Caruana, 1997) and thus all data are merged to enhance the model performance.

To perform our experiments we randomly divide the toxicity dataset into training and test sets based on patient IDs. We use about 1300 patients included in all prediction points as our training data and the remaining patients are used as our test data. All the models are trained on the training set and their classification performances are evaluated on the test set. The tuning parameters of the models are selected using 5-fold cross validation on the training set. We have performed 50 random training/test splitting of the dataset and reported the results as an average over these 50 splits.

**Stability Performance**  Figures 7.4(a) and (b) compare the stability performance of our proposed method with other baselines in terms of SRCC and JSM at dif-
7.2. Experiments

Figure 7.3: The proportion of patients with toxicity in each prediction point for toxicity dataset.

different prediction points. As the Figures show MIL-C achieves the best stability performance in terms of both SRCC and JSM compared to other methods at all prediction points. Also, the figures imply that \( l_1 \)-norm LR model that uses \( l_1 \)-norm regularization shows the least stability in selecting features. This is because that in presence of groups of highly correlated features, \( l_1 \)-norm penalty tends to select one of them randomly.

**Classification Performance** Figures 7.5(a) and (b) compare the average classification performance of MIL-C with other techniques in terms of standard AUC and F1 score for the toxicity dataset at different prediction points. As seen, our proposed method outperforms other baselines in terms of both classification performances at all prediction points. To have detail comparison of different methods, we have also shown the trend of changes of the average AUC and F1 score along different prediction points in Figure 7.6. As seen, the AUC and F1 score of all algorithms have gradually increased between the 1–8 prediction points and then begin to decrease
7.2. Experiments

Figure 7.4: Stability performance of our proposed method compared to other base-lines in terms of (a) SRCC and (b) JSM for toxicity dataset at different prediction points.

Thereafter. This is because the positive samples reduce dramatically around the 9–10 prediction points (see Figure 7.3). Also, it is noticeable that the prediction performances of MIL algorithms that perform feature selection (i.e. MIL-C, MIL-EN, MIL-RVM and MIL-Boosting) are higher than those without feature selection (i.e. MIL-SVM and MIL-LR). Thus the methods with feature selection enable us to achieve better performance while using smaller set of features.

Risk Factors Obtained Using MIL-C  Identifying stable features can assist domain experts in their decision makings to reduce the toxicity in cancer patients. Figure 7.7 shows some of the selected features using our proposed method. As
7.2. Experiments

Figure 7.5: Classification performance of our proposed method compared to other baselines in terms of (a) F1 score and (b) AUC for toxicity dataset at different prediction points.

mentioned in experimental settings, we have performed 50 random training/test splitting of the toxicity dataset. This process provides us 50 feature sets. Using the ensemble of these feature sets, we empirically estimate the probability of presence for each feature and report the top features with the highest probability of presence as the most stable risk factors. Each bar in Figure 7.7 indicates the probability of presence of the corresponding feature. For example, our results show that one of the important risk factors for toxicity is patient’s age (Age > 65), which is consistently selected by our method. This is also verified in (Hurria and Lichtman, 2007) that because of reduced organ functionality, older patients are more likely to suffer from treatment toxicity in comparison to younger patients. Also, we have reported the
7.2. Experiments

Figure 7.6: Trend of changes of the average AUC and F1 score along different prediction points.

Figure 7.7: Top risk factors for toxicity selected by our proposed method.
average predicting weight of each feature at the top of each bar. The weights also confirms that the majority of stable risk factors have high absolute weight, which means that the proposed method could not only select the most stable features but also the most informative ones. For instance, anemia, which is one of the known risk factors for chemotherapy toxicity (Hurria and Lichtman, 2007), is also identified by our model (ICD-10 codes D50 and D51). Their weights (3.21 for D50 and 3.67 for D51) show that the model finds these risk factors as highly predictive of toxicity. Furthermore, although these two risk factors are correlated to each other ($\rho = 0.87$), MIL-C could select both of them together.

### 7.3 Conclusion

In this chapter, we studied another important problem in healthcare domain, prediction of toxicity in cancer patients. We have proposed a multiple-instance learning framework to tackle this problem. Unlike other MIL frameworks, the focus of our method is to perform stable feature selection, which is crucial in healthcare problems. In this context, the model can inform oncologists about risk factors responsible for toxicities in cancer patients and can assist them in selecting suitable treatment for patients. Realizing that the feature instability is caused by feature correlations, we introduce a regularization term to our multiple-instance learning framework that is able to model the relationship (both positive and negative) between features, so highly correlated features can be selected or rejected together. We propose an efficient alternating algorithm to solve this optimization problem. We validate the efficiency of our proposed method using a real-world dataset of cancer patients. We show that as the proposed model encourages grouping effect, where correlated features tend to be in our out of the model together, it shows better feature stability performance compared to other state-of-the-art methods, namely MIL-EN, MIL-RVM, MIL-Boosting, MIL-SVM, MIL-LR and $l_1$-norm LR. In terms of predictive performance, we also show that our proposed model leads to better classification performance compared to other methods. This is because of estimator’s variance reduction that itself is the result of the capability of the model in selecting groups of correlated features.
Chapter 8

Conclusion

8.1 Summary

This thesis has focused on developing several frameworks to enhance the stability of sparse feature selection and classification with linear models for clinical prediction, where data is high dimensional and consists of highly correlated features. This thesis realizes this aim via different approaches, summarized below.

In chapter 3, a novel framework is developed for stable feature selection by exploiting hierarchical structure of ICD-10 codes available for EMR data. In this framework, we apply a feature extraction procedure to generate a tree out of available ICD-10 diagnosis codes in the data by making use of predefined ICD-10 coding hierarchy in diagnosis codes. Consequently, we obtain the ICD-10 tree from the data, in which feature correlations are represented in the form of a tree-structure. Due to the high correlations among parent nodes and their offspring, modeling this data using Lasso leads to feature instability. To increase the stability of the prediction model, we perform feature selection through model training with Tree-Lasso regularization, which enables us to exploit the feature correlations in the form of a tree-structure and hence, improves the stability of the model. The effectiveness of this framework for building stable clinical prediction models is demonstrated on two real-world applications, prediction of cancer mortality and prediction of readmission in patients after AMI.
EMRs also consist of general features, such as age, sex or pathological results, which do not have hierarchical structure and thus the previous framework is not applicable in presence of these features. In chapter 4, we developed a new model, called predictive grouping elastic net (pg-EN) that groups correlated features in the data and selects informative groups instead of each individual feature to increase the stability of the clinical model. Feature grouping is learned within the model using supervised data and therefore is aligned with prediction goal. We formulate the proposed model as a constrained optimization problem combining both feature grouping and feature selection in a single step. To solve this problem, we propose an efficient iterative algorithm with theoretical guarantees for its convergence. We demonstrate the usefulness of the model via experiments on both synthetic and real-world datasets.

Solving objective function of pg-EN, is a formidable challenge since it is non-convex with potentially large number of local minima. To overcome this problem, we propose a new model in chapter 5, called covariance Lasso (C-Lasso) that is able to perform prediction and stable feature selection while working with a convex objective function. C-Lasso employs a new regularization formulation that improves the stability of Lasso by encouraging the similarities between features based on their relatedness. The relatedness between features is captured via a feature covariance matrix. C-Lasso can simultaneously perform feature selection and capture both the positive correlation and the negative correlation between features through a convex objective function. We introduce an efficient alternating algorithm to solve the C-Lasso’s objective function. The Efficiency of this model in terms of both feature stability and classification accuracy is shown on variety of datasets. We note the reader that even though pg-EN has non-convex objective function, it still has its own benefits. For example, as pg-EN imposes non-negativity constraint to the grouping matrix, interpreted as weight/importance of a feature in the group.

In chapter 3-5, our proposed models utilize residual sum of squares or the logit function as their loss function. In chapter 6 we employ support vector machine (SVM), which is another promising classifier in clinical domain (Pochet and Suykens, 2006). It is shown that combining SVM with \( l_1 \)-norm regularization enables the model to perform feature selection and classification simultaneously. However, due to the properties of \( l_1 \)-norm penalty the resulting model would be unstable. Realizing that the model instability is caused by feature correlations, we propose two models to stabilize \( l_1 \)-norm SVM, namely Covariance-SVM (C-SVM) and graph-SVM. C-
SVM employs a regularization formulation that encourages the similarities between features based on their relatedness. In this formulation, the relatedness between features is captured through a feature covariance matrix. Our second model, graph-SVM uses an undirected feature graph, in which nodes represent EMR features and edges represent relationship between the features defined using Jaccard index. It uses a convex penalty that includes a pairwise $l_\infty$ regularizer to encourage coefficients of the correlated features to be equal. Both C-SVM and graph-SVM are formulated as convex optimization problems and alternating algorithms are proposed to solve them. The effectiveness of these two models in terms of feature stability and classification performance is shown on real-world applications for predicting cancer mortality and readmission in patients with AMI.

In chapter 7, we studied another important problem in healthcare domain, prediction of toxicity in cancer patients. We tackle this problem by introducing a multiple-instance learning framework that is capable of selecting stable and predictive features from a long-list of features. In this context, the model can inform oncologists about risk factors responsible for toxicities in cancer patients and can assist them in selecting suitable treatment for patients. Realizing that the feature instability is caused by feature correlations, we introduce a regularization term to our multiple-instance learning framework that is able to model the relationship between features, so highly correlated features can be selected or rejected together. We formulate the model using a constrained optimization problem and propose a solution based on an alternating optimization algorithm. The efficiency of the model is shown by selecting stable risk factors and predicting toxicity in cancer patients from a regional hospital in Australia.

8.2 Future Works

There are several potential tracks for further developments, which have not been addressed in this thesis. We list them as follows:

- In this thesis, we have mainly focused on single task learning. Another promising direction can be multi-task learning, in which feature selection and prediction are performed simultaneously on several related datasets. Using multi-
task learning enables us to evaluate two types of stability: between-task and within-task stability. The between-task stability measures how similar risk factors are across multiple tasks and within-task stability measures how similar risk factors are on different sub-sampling of the data for a given task.

- In this thesis, we have confined ourselves to address the model instability, caused due to existence of highly correlated features in data by proposing new feature selection techniques. However, we can cure the instability in the phase of preprocessing the dataset, instead. It is known that the stability of the model is mostly data dependent and is affected by certain dataset characteristics. One such characteristics is existence of noise in data that decays both stability and prediction performance of the model (Klebanov and Yakovlev, 2007; Marshall, 2004). Thus, reducing noise can result in more stable and accurate algorithm.

- In this thesis, we used electronic medical records, that show the phenotype of the patients to build prediction models. In addition, genomic data is another rich source of information that shows makeup of a disease and the genotype of the patient. For example, the microarray technology SSDB95 screen a biopsy to estimate the level of activity of thousands of genes in a single experiment. Analyzing such data is promising in identifying biomarkers that may allow clinicians to make more accurate clinical prediction predictions. Another useful direction is to jointly using those different but complementary types of data to perform prediction tasks.

- This thesis dealt with the labeled data. However, in many real world problems collecting labeled data is expensive and hard and instead unlabeled examples are available. It would be interesting to extend stable feature selection and prediction models from fully to partially supervised scenarios. In building clinical models using semi-supervised data, it is essential to explore a large number of features to select a subset of features that contains the most informative and interpretable information about our prediction task.
Appendix A

Mathematical Proofs

A.1 Proof of Proposition 1

We define the Lagrangian function

$$L(G) = \text{tr}(-2\beta^T G^T D^T y + \beta^T G^T D^T D G \beta - \delta G^T G - \mu G^T),$$  \hfill (A.1)

where the Lagrangian multipliers $\mu_{ij}$ introduce nonnegative constraints, $G \geq 0$. From the zero gradient condition

$$\frac{\partial L}{\partial G} = -2D^T y \beta^T + 2D^T D G \beta \beta^T - 2\delta G - \mu = 0.$$  \hfill (A.2)

From the complementary slackness condition,

$$(-2D^T y \beta^T + 2D^T D G \beta \beta^T - 2\delta G)_{ik} G_{ik} = \mu_{ik} G_{ik} = 0.$$  \hfill (A.3)

This is the fixed point equation, and we show that the limiting solution of the update rule of (2.4) satisfies this equation. At convergence, $G^\infty = G^{(t+1)} = G^{(t)} = G$, so

$$G_{ik} = G_{ik} \sqrt{\frac{(B^+ G A^+)_{ik} + (B^+ G A^-)_{ik} + C_{ik}^+ + \delta G_{ik}}{(B^+ G A^+)_{ik} + (B^- G A^-)_{ik} + C_{ik}^-}},$$  \hfill (A.3)
where \( A = \beta \beta^T \), \( B = D^T D \), \( C = D^T y \beta^T \). As \( B = B^+ - B^- \), \( A = A^+ - A^- \), hence we have \( BGA = [B^+ - B^-]G[A^+ - A^-] \). Thus, (A.3) reduces to

\[
(-2D^T y \beta^T + 2D^T DG\beta \beta^T - 2\delta G)_{ik} G^2_{ik} = 0. \tag{A.4}
\]

Equations (A.4) and (A.2) are identical. Both the equations require that at least one of the two factors is equal to zero. The first factor in both equations is identical. For the second factor \( G_{ik} \) and \( G^2_{ik} \), if \( G_{ik} = 0 \), then \( G^2_{ik} = 0 \), and vice versa. Thus, if (A.2) holds then (A.4) also holds and vice versa. In the following propositions we prove that the iterative update algorithm converges.

### A.2 Proof of Proposition 2

We can write \( J(H) \) as

\[
J(H) = \text{tr}( -2H^T C^+ + 2H^T C^- + A^+ H^T B^+ H \\
\]

where \( A = \beta \beta^T \), \( B = D^T D \), \( C = D^T y \beta^T \), and \( H = G \). We use an auxiliary function approach similar to that used in Ding et al. (2010); Lee and Seung (2001). \( Z(H, H') \) is an auxiliary function of \( J(H) \) if it satisfies \( Z(H, H') \geq J(H) \) and \( Z(H, H) = J(H) \), for any \( H, H' \). We define the update rule

\[
H^{(t+1)} = \arg \min_H Z(H, H^{(t)}), \tag{A.6}
\]

where, \( J(H^{(t)}) = Z(H^{(t)}, H^{(t)}) \geq Z(H^{(t+1)}, H^{(t)}) \geq J(H^{(t+1)}) \). Hence, \( J(H^{(t)}) \) is non-increasing. So we should find an appropriate \( Z(H, H') \) and its global minimum. In Proposition 3, we define \( Z(H, H') \) as an auxiliary function of \( J \) that its minimum is (4.6). According to (A.6), \( H^{(t+1)} \leftarrow H \) and \( H^{(t)} \leftarrow H' \); replacing \( H = G \), we obtain (2.9).
A.3 Proof of Proposition 3

We find upper bounds for positive terms and lower bounds for negative terms. For the second term in $J(H)$, we have an upper bound

$$\text{tr}(H^TC^-) = \sum_{ik} H_{ik} C_{ik}^- \leq C_{ik}^- \frac{H_{ik}^2 + H_{ik}'^2}{2H_{ik}'},$$

using inequality $a \leq (a^2 + b^2)/2a$. The third and sixth terms in $J(H)$, are bounded by

$$\text{tr}(A^+ H^T B^+ H) \leq \sum_{ik} \frac{(B^+ H'A^+)_ik H_{ik}'^2}{H_{ik}'},$$

$$\text{tr}(A^- H^T B^- H) \leq \sum_{ik} \frac{(B^- H'A^-)_ik H_{ik}'^2}{H_{ik}},$$

using Proposition 4. Lower bounds for the remaining terms are obtained using the inequality $z \geq 1 + \log(z)$, which is true for any $z > 0$. We obtain

$$\frac{H_{ik}}{H_{ik}'} \geq 1 + \log \frac{H_{ik}}{H_{ik}'}; \quad \frac{H_{ik}H_d}{H_{ik}'H_d} \geq 1 + \log \frac{H_{ik}H_d}{H_{ik}'H_d} \quad (A.7)$$

Using (A.7), we can bound the first, fourth, fifth and last terms of $J(H)$ as follows

$$\text{tr}(H^TC^+) \geq \sum_{ik} C_{ik}^+ H_{ik}' \left(1 + \log \frac{H_{ik}}{H_{ik}'}\right),$$

$$\text{tr}(A^+ H^T B^- H) \geq \sum_{i,j,k,l} B_{ij}^- H_{jk}' A_{kl}^+ H_{il}' \left(1 + \log \frac{H_{jk}H_d}{H_{jk}'H_d}\right),$$

$$\text{tr}(A^- H^T B^+ H) \geq \sum_{i,j,k,l} B_{ij}' H_{jk} A_{kl}^- H_{il}' \left(1 + \log \frac{H_{jk}H_d}{H_{jk}'H_d}\right),$$

$$\text{tr}(H^TH) \geq \sum_{i,k,l} H_{ik}' H_{il}' (1 + \log \frac{H_{ik}H_d}{H_{ik}'H_d}).$$

We can obtain $Z(H, H')$ as in (4.5), by collecting all bounds. It is obvious that
A.3. Proof of Proposition 3

\( J(H) \leq Z(H, H') \) and \( J(H) = Z(H, H) \). To find the minimum of \( Z(H, H') \), we take

\[
\frac{\partial Z(H, H')}{\partial H_{ik}} = -2 C_{ik} H'_{ik} H_{ik} + 2 C_{ik} H_{ik} + 2 (B^+ H' A^+)_{ik} H_{ik} - 2 (B H A^-)_{ik} H'_{ik} H_{ik} + 2 (B H' A^-)_{ik} H_{ik} - 2 H'_{ik}.
\]

The Hessian of \( Z(H, H') \) is written by noting

\[
\frac{\partial^2 Z(H, H')}{\partial H_{ik} \partial H_{jl}} = \delta_{ij} \delta_{kl} Z_{ik} \]

where

\[
Z_{ik} = 2 \left[ C^+_{ik} + (B^+ H A^+)_{ik} + (B^+ H' A^-)_{ik} + \delta \right] H'_{ik} H^2_{ik} + 2 \left[ C^-_{ik} + (B^+ H' A^+)_{ik} + (B^+ H' A^-)_{ik} \right] H'_{ik}.
\]

This matrix is a diagonal matrix with positive entries. We infer that \( Z(H, H') \) is a convex function of \( H \). If we solve \( \frac{\partial Z(H, H')}{\partial H_{ik}} = 0 \) for \( H \), we recover 4.6.

For any matrices \( P \in \mathbb{R}^{n \times n} \), \( R \in \mathbb{R}^{k \times k} \), \( S \in \mathbb{R}^{n \times k} \) and \( S' \in \mathbb{R}^{n \times k} \), with symmetric \( P \) and \( R \) we have

\[
\text{tr}(S^T P S R) \leq \sum_{i=1}^{n} \sum_{j=1}^{k} \frac{(PS'R)_{ij} S^2_{ij}}{S'^2_{ij}}. \tag{A.8}
\]

If \( S_{ij} = S'_{ij} u_{ij} \), the difference between the left-hand side and the right hand side can be written as

\[
\Delta = \sum_{i,t=1}^{n} \sum_{j,q=1}^{k} P_{ij} S'_{iq} B_{qj} S'_{ij} (u_{ij}^2 - u_{ij} u_{iq}).
\]

As \( P \) and \( R \) are symmetric, this is equal to

\[
\Delta = \sum_{i,t=1}^{n} \sum_{j,q=1}^{k} P_{ij} S'_{iq} B_{qj} S'_{ij} \left( \frac{u_{ij}^2 + u_{iq}^2}{2} - u_{ij} u_{iq} \right) = \frac{1}{2} \sum_{i,t=1}^{n} \sum_{j,q=1}^{k} P_{ij} S'_{iq} B_{qj} S'_{ij} (u_{ij} - u_{iq})^2 \geq 0.
\]
Appendix B

Additional Experiments

B.1 Additional Experiments to Evaluate pg-EN

B.1.1 Grouping Results on Soil Dataset

In order to show that our model yields feasible groups, we assess our model on a small dataset called Soil data Bondell and Reich (2008). This dataset studies relation between soil characteristics and rich-cove forest diversity in the Appalachian mountains of North Carolina. Twenty 500 $m^2$ plots were surveyed. The outcome shows the number of different plant species found in the plot. 15 soil characteristics are used as variables to predict forest diversity and are shown in Figure B.1. As it can be seen from this figure, there are groups of highly correlated variables. The first seven variables that are all related to cations are highly correlated. Sodium and phosphorus are also highly correlated as well as soil pH and exchangeable acidity, which are measures of acidity. In addition, as the sum of cations can be derived through sum of all cations namely, calcium, magnesium, potassium, and sodium, the design matrix is not full rank. Figure B.2 shows the obtained grouping matrix ($G$) using pg-EN for Soil data. As it is shown in this figure, the obtained grouping matrix of the features is sparse and also there are no overlaps between groups. It also shows the ability of pg-EN in selecting and grouping correlated features. As mentioned earlier, in order to increase the stability and predictive performance of the model,
B.1. Additional Experiments to Evaluate pg-EN

unlike Lasso that treats each variable separately and randomly selects a representative, pg-EN tends to use the group of correlated features and treat them as a derived variable. As the figure shows, pg-EN could group the four selected cation variables together i.e. percent base saturation (BaseSat), sum of cations (SumCation), calcium (Ca) and magnesium (Mg). It could also group acidity-related variables i.e. pH and exchangeable acidity (ExchAc). The algorithm assigns a unique weight ($\beta$) to each group. Because highly correlated variables have the same underlying factor, supervised grouping of these variables can result in better estimation of the underlying factor of correlated variables and present more informative predictive model.

B.1.2 Statistical Test Results

As mentioned in the chapter, in order to have a better comparison between the results obtained using our proposed method and other baselines, we have performed pairwise Wilcoxon signed-rank test with significance level of 0.05 for each prediction measure. The p-values obtained from this statistical test are shown in Tables B.1 to B.8 for Synthetic-III.
Obtained grouping matrix for soil data

<table>
<thead>
<tr>
<th>Group number</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>BaseSat</td>
<td>-0.4</td>
<td>0</td>
<td>0.2</td>
<td>0.4</td>
<td>0.6</td>
</tr>
<tr>
<td>SumCation</td>
<td>-1</td>
<td>-0.8</td>
<td>-0.6</td>
<td>-0.4</td>
<td>-0.2</td>
</tr>
<tr>
<td>CECbuffer</td>
<td>-1</td>
<td>-0.8</td>
<td>-0.6</td>
<td>-0.4</td>
<td>-0.2</td>
</tr>
<tr>
<td>Ca</td>
<td>-1</td>
<td>-0.8</td>
<td>-0.6</td>
<td>-0.4</td>
<td>-0.2</td>
</tr>
<tr>
<td>Mg</td>
<td>-1</td>
<td>-0.8</td>
<td>-0.6</td>
<td>-0.4</td>
<td>-0.2</td>
</tr>
<tr>
<td>K</td>
<td>-1</td>
<td>-0.8</td>
<td>-0.6</td>
<td>-0.4</td>
<td>-0.2</td>
</tr>
<tr>
<td>Na</td>
<td>-1</td>
<td>-0.8</td>
<td>-0.6</td>
<td>-0.4</td>
<td>-0.2</td>
</tr>
<tr>
<td>P</td>
<td>-1</td>
<td>-0.8</td>
<td>-0.6</td>
<td>-0.4</td>
<td>-0.2</td>
</tr>
<tr>
<td>Cu</td>
<td>-1</td>
<td>-0.8</td>
<td>-0.6</td>
<td>-0.4</td>
<td>-0.2</td>
</tr>
<tr>
<td>Zn</td>
<td>-1</td>
<td>-0.8</td>
<td>-0.6</td>
<td>-0.4</td>
<td>-0.2</td>
</tr>
<tr>
<td>Mn</td>
<td>-1</td>
<td>-0.8</td>
<td>-0.6</td>
<td>-0.4</td>
<td>-0.2</td>
</tr>
<tr>
<td>HumicMatter</td>
<td>-1</td>
<td>-0.8</td>
<td>-0.6</td>
<td>-0.4</td>
<td>-0.2</td>
</tr>
<tr>
<td>Density</td>
<td>-1</td>
<td>-0.8</td>
<td>-0.6</td>
<td>-0.4</td>
<td>-0.2</td>
</tr>
<tr>
<td>pH</td>
<td>-1</td>
<td>-0.8</td>
<td>-0.6</td>
<td>-0.4</td>
<td>-0.2</td>
</tr>
<tr>
<td>ExchAc</td>
<td>-1</td>
<td>-0.8</td>
<td>-0.6</td>
<td>-0.4</td>
<td>-0.2</td>
</tr>
</tbody>
</table>

Figure B.2: Group matrix obtained for Soil dataset. (The group numbers are permuted to assist comparison with Figure B.1.)

(Microarray), Breast cancer, Cancer(EMR) and AMI(EMR) datasets.
### Table B.1: The p-value obtained from pairwise Wilcoxon signed-rank test for (a) Positive Predictive Value (PPV) and (b) Sensitivity of different methods applied to the Synthetic-III dataset.

(a)

<table>
<thead>
<tr>
<th>p-value</th>
<th>Lasso</th>
<th>Elastic net</th>
<th>Oscar</th>
<th>KM+Lasso</th>
<th>KM+GLasso</th>
<th>pg-EN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ridge</td>
<td>1.8e-06</td>
<td>1.7e-06</td>
<td>1.5e-06</td>
<td>2.8e-06</td>
<td>1.8e-06</td>
<td>1.2e-06</td>
</tr>
<tr>
<td>Lasso</td>
<td>0.7036</td>
<td>0.0736</td>
<td>0.0519</td>
<td>0.3820</td>
<td>1.7e-06</td>
<td></td>
</tr>
<tr>
<td>Elastic net</td>
<td>0.0039</td>
<td>0.0230</td>
<td>0.8774</td>
<td>8.4e-06</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oscar</td>
<td>2.3e-06</td>
<td>7.1e-04</td>
<td>0.0034</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM+Lasso</td>
<td>0.0057</td>
<td>2.1e-06</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM+GLasso</td>
<td>3.8e-06</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(b)

<table>
<thead>
<tr>
<th>p-value</th>
<th>Lasso</th>
<th>Elastic net</th>
<th>Oscar</th>
<th>KM+Lasso</th>
<th>KM+GLasso</th>
<th>pg-EN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ridge</td>
<td>1.2e-05</td>
<td>1.7e-06</td>
<td>1.9e-06</td>
<td>9.3e-06</td>
<td>1.7e-06</td>
<td>1.6e-06</td>
</tr>
<tr>
<td>Lasso</td>
<td>0.1064</td>
<td>0.1064</td>
<td>0.1915</td>
<td>2.2e-04</td>
<td>2.8e-06</td>
<td></td>
</tr>
<tr>
<td>Elastic net</td>
<td>4.5e-04</td>
<td>0.7189</td>
<td>0.0028</td>
<td>6.3e-06</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oscar</td>
<td>0.0017</td>
<td>0.0472</td>
<td>0.0032</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM+Lasso</td>
<td>0.0519</td>
<td>1.7e-05</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM+GLasso</td>
<td>1.4e-05</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table B.2: The p-value obtained from pairwise Wilcoxon signed-rank test for (a) Specificity, (b) F1 score and (c) AUC of different methods applied to the Synthetic-III dataset.

<table>
<thead>
<tr>
<th></th>
<th>p-value</th>
<th>Lasso</th>
<th>Elastic net</th>
<th>Oscar</th>
<th>KM+Lasso</th>
<th>KM+GLasso</th>
<th>pg-EN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ridge</td>
<td>8.4e-06</td>
<td>1.7e-06</td>
<td>1.6e-06</td>
<td>8.2e-06</td>
<td>1.7e-06</td>
<td>1.6e-06</td>
<td></td>
</tr>
<tr>
<td>Lasso</td>
<td></td>
<td>4.7e-06</td>
<td>1.9e-06</td>
<td>0.1846</td>
<td>2.6e-06</td>
<td>1.9e-06</td>
<td></td>
</tr>
<tr>
<td>Elastic net</td>
<td></td>
<td>0.0030</td>
<td>6.3e-05</td>
<td>0.3493</td>
<td>1.0e-05</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oscar</td>
<td></td>
<td>2.8e-06</td>
<td>0.0333</td>
<td>0.0047</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM+Lasso</td>
<td></td>
<td>2.3e-05</td>
<td>1.7e-05</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM+GLasso</td>
<td></td>
<td>6.3e-05</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(a)

<table>
<thead>
<tr>
<th></th>
<th>p-value</th>
<th>Lasso</th>
<th>Elastic net</th>
<th>Oscar</th>
<th>KM+Lasso</th>
<th>KM+GLasso</th>
<th>pg-EN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ridge</td>
<td>1.7e-06</td>
<td>1.7e-06</td>
<td>1.6e-06</td>
<td>1.7e-06</td>
<td>1.7e-06</td>
<td>1.5e-06</td>
<td></td>
</tr>
<tr>
<td>Lasso</td>
<td></td>
<td>0.1254</td>
<td>8.4e-06</td>
<td>0.5304</td>
<td>0.0020</td>
<td>1.6e-06</td>
<td></td>
</tr>
<tr>
<td>Elastic net</td>
<td></td>
<td>4.4e-05</td>
<td>0.0897</td>
<td>0.0300</td>
<td></td>
<td>1.9e-06</td>
<td></td>
</tr>
<tr>
<td>Oscar</td>
<td></td>
<td>6.3e-06</td>
<td>1.6e-04</td>
<td>1.8e-04</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM+Lasso</td>
<td></td>
<td>0.0039</td>
<td>2.6e-04</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM+GLasso</td>
<td></td>
<td>2.3e-06</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(b)

<table>
<thead>
<tr>
<th></th>
<th>p-value</th>
<th>Lasso</th>
<th>Elastic net</th>
<th>Oscar</th>
<th>KM+Lasso</th>
<th>KM+GLasso</th>
<th>pg-EN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ridge</td>
<td>2.6e-06</td>
<td>1.7e-06</td>
<td>1.7e-06</td>
<td>1.9e-06</td>
<td>2.1e-06</td>
<td>1.6e-06</td>
<td></td>
</tr>
<tr>
<td>Lasso</td>
<td></td>
<td>0.1306</td>
<td>5.7e-06</td>
<td>0.1650</td>
<td>0.1986</td>
<td>1.7e-06</td>
<td></td>
</tr>
<tr>
<td>Elastic net</td>
<td></td>
<td>1.89e-04</td>
<td>0.8451</td>
<td>0.5304</td>
<td>2.6e-06</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oscar</td>
<td></td>
<td>2.1e-05</td>
<td>2.5e-05</td>
<td>0.0039</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM+Lasso</td>
<td></td>
<td>0.6143</td>
<td>1.6e-06</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM+GLasso</td>
<td></td>
<td>1.7e-06</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(c)
### B.1. Additional Experiments to Evaluate pg-EN

Table B.3: The p-value obtained from pairwise Wilcoxon signed-rank test for (a) Positive Predictive Value (PPV) and (b) Sensitivity of different methods applied to the Breast cancer dataset.

<table>
<thead>
<tr>
<th></th>
<th>Lasso</th>
<th>Elastic net</th>
<th>Oscar</th>
<th>KM+Lasso</th>
<th>KM+GLasso</th>
<th>pg-EN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ridge</td>
<td>0.8936</td>
<td>0.1414</td>
<td>0.0018</td>
<td>0.2289</td>
<td>0.2287</td>
<td>2.0e-05</td>
</tr>
<tr>
<td>Lasso</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Elastic net</td>
<td>0.0719</td>
<td>3.1e-04</td>
<td>0.5999</td>
<td>0.7655</td>
<td>1.7e-04</td>
<td>0.2287</td>
</tr>
<tr>
<td>Oscar</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM+Lasso</td>
<td>0.6288</td>
<td></td>
<td>6.0e-05</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM+GLasso</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(a)

<table>
<thead>
<tr>
<th></th>
<th>Lasso</th>
<th>Elastic net</th>
<th>Oscar</th>
<th>KM+Lasso</th>
<th>KM+GLasso</th>
<th>pg-EN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ridge</td>
<td>0.9263</td>
<td>0.5038</td>
<td>0.0368</td>
<td>0.9099</td>
<td>0.5440</td>
<td>4.0e-05</td>
</tr>
<tr>
<td>Lasso</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Elastic net</td>
<td>0.3086</td>
<td>0.0937</td>
<td>0.6884</td>
<td>0.4780</td>
<td>0.0011</td>
<td></td>
</tr>
<tr>
<td>Oscar</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM+Lasso</td>
<td>0.7036</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM+GLasso</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(b)
### B.1. Additional Experiments to Evaluate pg-EN

Table B.4: The p-value obtained from pairwise Wilcoxon signed-rank test for (a) Specificity, (b) F1 score and (c) AUC of different methods applied to the Breast cancer dataset.

<table>
<thead>
<tr>
<th></th>
<th>Lasso</th>
<th>Elastic net</th>
<th>Oscar</th>
<th>KM+Lasso</th>
<th>KM+GLasso</th>
<th>pg-EN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ridge</td>
<td>0.7189</td>
<td>0.1986</td>
<td>0.0166</td>
<td>0.7036</td>
<td>0.7499</td>
<td>1.9e-04</td>
</tr>
<tr>
<td>Lasso</td>
<td>0.0787</td>
<td>0.0047</td>
<td>0.9099</td>
<td>0.9754</td>
<td>2.8e-04</td>
<td></td>
</tr>
<tr>
<td>Elastic net</td>
<td>0.2802</td>
<td>0.1359</td>
<td>0.0687</td>
<td>0.0028</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oscar</td>
<td>0.0132</td>
<td>0.0098</td>
<td>0.0270</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM+Lasso</td>
<td>0.9754</td>
<td>1.9e-04</td>
<td>0.0028</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM+GLasso</td>
<td>6.2e-04</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(a)

<table>
<thead>
<tr>
<th></th>
<th>Lasso</th>
<th>Elastic net</th>
<th>Oscar</th>
<th>KM+Lasso</th>
<th>KM+GLasso</th>
<th>pg-EN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ridge</td>
<td>0.5440</td>
<td>0.1589</td>
<td>4.9e-04</td>
<td>0.4048</td>
<td>0.1109</td>
<td>1.0e-05</td>
</tr>
<tr>
<td>Lasso</td>
<td>0.0196</td>
<td>1.0e-04</td>
<td>0.2452</td>
<td>0.1064</td>
<td>1.0e-08</td>
<td></td>
</tr>
<tr>
<td>Elastic net</td>
<td>0.1066</td>
<td>0.3494</td>
<td>0.3820</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oscar</td>
<td>0.0073</td>
<td>0.0140</td>
<td>8.3e-04</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM+Lasso</td>
<td>0.3389</td>
<td>1.0e-05</td>
<td>0.3389</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM+GLasso</td>
<td>5.0e-05</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(b)

<table>
<thead>
<tr>
<th></th>
<th>Lasso</th>
<th>Elastic net</th>
<th>Oscar</th>
<th>KM+Lasso</th>
<th>KM+GLasso</th>
<th>pg-EN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ridge</td>
<td>0.8936</td>
<td>0.1414</td>
<td>0.0018</td>
<td>0.2289</td>
<td>0.2289</td>
<td>2.0e-05</td>
</tr>
<tr>
<td>Lasso</td>
<td>0.0719</td>
<td>3.1e-04</td>
<td>0.2059</td>
<td>0.2369</td>
<td>3.0e-05</td>
<td></td>
</tr>
<tr>
<td>Elastic net</td>
<td>0.0185</td>
<td>0.5999</td>
<td>0.7655</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oscar</td>
<td>0.0185</td>
<td>0.0719</td>
<td>0.0285</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM+Lasso</td>
<td>0.6288</td>
<td>6.0e-05</td>
<td>0.6288</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM+GLasso</td>
<td>4.5e-04</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(c)
B.1. Additional Experiments to Evaluate pg-EN

<table>
<thead>
<tr>
<th>p-value</th>
<th>Lasso</th>
<th>Elastic net</th>
<th>Oscar</th>
<th>KM+Lasso</th>
<th>KM+GLasso</th>
<th>pg-EN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ridge</td>
<td>0.7499</td>
<td>0.0687</td>
<td>0.0018</td>
<td>0.2289</td>
<td>0.2287</td>
<td>1.0e-05</td>
</tr>
<tr>
<td>Lasso</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Elastic net</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oscar</td>
<td></td>
<td></td>
<td></td>
<td>0.0185</td>
<td>0.0719</td>
<td>0.0036</td>
</tr>
<tr>
<td>KM+Lasso</td>
<td></td>
<td></td>
<td></td>
<td>0.6288</td>
<td></td>
<td>3.0e-05</td>
</tr>
<tr>
<td>KM+GLasso</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2.2e-04</td>
</tr>
</tbody>
</table>

(a)

<table>
<thead>
<tr>
<th>p-value</th>
<th>Lasso</th>
<th>Elastic net</th>
<th>Oscar</th>
<th>KM+Lasso</th>
<th>KM+GLasso</th>
<th>pg-EN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ridge</td>
<td>0.0207</td>
<td>0.0148</td>
<td>3.0e-05</td>
<td>0.0087</td>
<td>4.9e-04</td>
<td>1.0e-08</td>
</tr>
<tr>
<td>Lasso</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Elastic net</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oscar</td>
<td></td>
<td></td>
<td></td>
<td>0.571</td>
<td>0.4284</td>
<td>0.0118</td>
</tr>
<tr>
<td>KM+Lasso</td>
<td></td>
<td></td>
<td></td>
<td>0.2895</td>
<td></td>
<td>1.6e-04</td>
</tr>
<tr>
<td>KM+GLasso</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.0013</td>
</tr>
</tbody>
</table>

(b)

Table B.5: The p-value obtained from pairwise Wilcoxon signed-rank test for (a) Positive Predictive Value (PPV) and (b) Sensitivity of different methods applied to the Cancer (EMR) dataset.
Table B.6: The p-value obtained from pairwise Wilcoxon signed-rank test for (a) Specificity, (b) F1 score and (c) AUC of different methods applied to the Cancer (EMR) dataset.
B.1. Additional Experiments to Evaluate pg-EN

<table>
<thead>
<tr>
<th></th>
<th>p-value</th>
<th>Lasso</th>
<th>Elastic net</th>
<th>Oscar</th>
<th>KM+Lasso</th>
<th>KM+GLasso</th>
<th>pg-EN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ridge</td>
<td>¬</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lasso</td>
<td>0.6583</td>
<td>0.0185</td>
<td>0.6435</td>
<td>0.3185</td>
<td>0.0285</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Elastic net</td>
<td>0.0719</td>
<td>0.2452</td>
<td>0.6583</td>
<td>0.0978</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oscar</td>
<td>8.9e-04</td>
<td>0.0859</td>
<td>0.7036</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM+Lasso</td>
<td>0.3820</td>
<td>0.0060</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM+GLasso</td>
<td>0.1156</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(a)

<table>
<thead>
<tr>
<th></th>
<th>p-value</th>
<th>Lasso</th>
<th>Elastic net</th>
<th>Oscar</th>
<th>KM+Lasso</th>
<th>KM+GLasso</th>
<th>pg-EN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ridge</td>
<td>1.5e-04</td>
<td>2.0e-05</td>
<td>2.0e-05</td>
<td>2.6e-04</td>
<td>1.1e-04</td>
<td>1.1e-08</td>
<td></td>
</tr>
<tr>
<td>Lasso</td>
<td>0.2536</td>
<td>0.0545</td>
<td>0.5440</td>
<td>0.0859</td>
<td>2.0e-05</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Elastic net</td>
<td>0.2369</td>
<td>0.1020</td>
<td>0.2712</td>
<td>4.0e-05</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oscar</td>
<td>0.0073</td>
<td>0.4780</td>
<td>0.0013</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM+Lasso</td>
<td>0.0333</td>
<td>1.0e-05</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM+GLasso</td>
<td>2.1e-04</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(b)

Table B.7: The p-value obtained from pairwise Wilcoxon signed-rank test for (a) Positive Predictive Value (PPV) and (b) Sensitivity of different methods applied to the AMI (EMR) dataset.
### Table B.8: The p-value obtained from pairwise Wilcoxon signed-rank test for (a) Specificity, (b) F1 score and (c) AUC of different methods applied to the AMI (EMR) dataset.

<table>
<thead>
<tr>
<th></th>
<th>Lasso</th>
<th>Elastic net</th>
<th>Oscar</th>
<th>KM+Lasso</th>
<th>KM+GLasso</th>
<th>pg-EN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ridge</td>
<td>0.0175</td>
<td>0.0104</td>
<td>1.0e-05</td>
<td>0.0243</td>
<td>1.6e-04</td>
<td>1.2e-09</td>
</tr>
<tr>
<td>Lasso</td>
<td></td>
<td>0.5857</td>
<td>0.0034</td>
<td>4.9e-04</td>
<td>0.0218</td>
<td>1.0e-05</td>
</tr>
<tr>
<td>Elastic net</td>
<td></td>
<td></td>
<td>0.0978</td>
<td>2.6e-04</td>
<td>0.3185</td>
<td>2.0e-05</td>
</tr>
<tr>
<td>Oscar</td>
<td></td>
<td></td>
<td></td>
<td>9.0e-05</td>
<td>0.2289</td>
<td>3.3e-04</td>
</tr>
<tr>
<td>KM+Lasso</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1.3e-04</td>
<td>1.0e-05</td>
</tr>
<tr>
<td>KM+GLasso</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1.0e-05</td>
</tr>
</tbody>
</table>

(a)

<table>
<thead>
<tr>
<th></th>
<th>Lasso</th>
<th>Elastic net</th>
<th>Oscar</th>
<th>KM+Lasso</th>
<th>KM+GLasso</th>
<th>pg-EN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ridge</td>
<td>1.0e-05</td>
<td>1.2e-08</td>
<td>1.0e-08</td>
<td>1.5e-05</td>
<td>1.0e-05</td>
<td>2.0e-09</td>
</tr>
<tr>
<td>Lasso</td>
<td></td>
<td>0.5038</td>
<td>0.0030</td>
<td>0.3933</td>
<td>0.0787</td>
<td>1.0e-05</td>
</tr>
<tr>
<td>Elastic net</td>
<td></td>
<td></td>
<td>0.0333</td>
<td>0.0897</td>
<td>0.4405</td>
<td>6.0e-05</td>
</tr>
<tr>
<td>Oscar</td>
<td></td>
<td></td>
<td></td>
<td>2.8e-04</td>
<td>0.0978</td>
<td>0.0333</td>
</tr>
<tr>
<td>KM+Lasso</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.0519</td>
<td>1.0e-05</td>
</tr>
<tr>
<td>KM+GLasso</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8.3e-04</td>
</tr>
</tbody>
</table>

(b)

<table>
<thead>
<tr>
<th></th>
<th>Lasso</th>
<th>Elastic net</th>
<th>Oscar</th>
<th>KM+Lasso</th>
<th>KM+GLasso</th>
<th>pg-EN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ridge</td>
<td>0.0093</td>
<td>0.0185</td>
<td>0.0020</td>
<td>0.0449</td>
<td>0.0032</td>
<td>2.3e-08</td>
</tr>
<tr>
<td>Lasso</td>
<td></td>
<td>0.4908</td>
<td>0.0118</td>
<td>0.9590</td>
<td>0.4284</td>
<td>1.5e-08</td>
</tr>
<tr>
<td>Elastic net</td>
<td></td>
<td></td>
<td>0.2134</td>
<td>0.3820</td>
<td>0.7971</td>
<td>2.0e-05</td>
</tr>
<tr>
<td>Oscar</td>
<td></td>
<td></td>
<td></td>
<td>0.0098</td>
<td>0.2210</td>
<td>1.0e-05</td>
</tr>
<tr>
<td>KM+Lasso</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.4908</td>
<td>1.2e-08</td>
</tr>
<tr>
<td>KM+GLasso</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1.0e-08</td>
</tr>
</tbody>
</table>

(c)
### Table B.9: Average classification performance of pg-EN compared to other methods for Breast Cancer dataset without re-balancing the data. The numbers in brackets show the standard error over 100 iterations.

<table>
<thead>
<tr>
<th>Method</th>
<th>PPV</th>
<th>Sensitivity</th>
<th>F1 score</th>
<th>Specificity</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ridge</td>
<td>0.254 (0.004)</td>
<td>0.365 (0.004)</td>
<td>0.299 (0.005)</td>
<td>0.763 (0.005)</td>
<td>0.698 (0.004)</td>
</tr>
<tr>
<td>Lasso</td>
<td>0.250 (0.005)</td>
<td>0.366 (0.002)</td>
<td>0.297 (0.003)</td>
<td>0.767 (0.004)</td>
<td>0.710 (0.004)</td>
</tr>
<tr>
<td>Elastic net</td>
<td>0.261 (0.004)</td>
<td>0.366 (0.004)</td>
<td>0.304 (0.005)</td>
<td>0.770 (0.003)</td>
<td>0.717 (0.004)</td>
</tr>
<tr>
<td>Oscar</td>
<td>0.268 (0.005)</td>
<td>0.370 (0.003)</td>
<td>0.310 (0.004)</td>
<td>0.775 (0.006)</td>
<td>0.721 (0.002)</td>
</tr>
<tr>
<td>KM+Lasso</td>
<td>0.259 (0.003)</td>
<td>0.362 (0.004)</td>
<td>0.298 (0.003)</td>
<td>0.768 (0.005)</td>
<td>0.712 (0.004)</td>
</tr>
<tr>
<td>KM+GLasso</td>
<td>0.260 (0.002)</td>
<td>0.365 (0.005)</td>
<td>0.303 (0.004)</td>
<td>0.768 (0.004)</td>
<td>0.714 (0.005)</td>
</tr>
<tr>
<td>pg-EN</td>
<td>0.276 (0.003)</td>
<td>0.381 (0.006)</td>
<td>0.320 (0.005)</td>
<td>0.781 (0.003)</td>
<td>0.742 (0.006)</td>
</tr>
</tbody>
</table>

**B.1.3 Evaluating the Classification Performance without Re-balancing Data**

As mentioned before, the real-world datasets used in our experiments are imbalanced and we re-balanced them by oversampling the rare class. In this section, we evaluate the predictive performance of pg-EN and other baselines when these datasets are not re-balanced. The obtained results in Tables B.9-B.11 show that, without re-balancing the datasets, the predictive performance of all methods degrades slightly. However, in all imbalanced datasets, pg-EN achieves the best predictive performance among other methods.

**B.1.4 K-means vs Random Initialization of pg-EN**

As mentioned in methodology part in section 4, the initialization of pg-EN can be done either using K-means algorithm or by random assignment. In this section, we
### Table B.10: Average classification performance of pg-EN compared to other methods for Cancer (EMR) dataset without re-balancing the data. The numbers in brackets show the standard error over 100 iterations.

<table>
<thead>
<tr>
<th>Method</th>
<th>PPV</th>
<th>Sensitivity</th>
<th>F1 score</th>
<th>Specificity</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ridge</td>
<td>0.249</td>
<td>0.298</td>
<td>0.271</td>
<td>0.728</td>
<td>0.602</td>
</tr>
<tr>
<td></td>
<td>(0.005)</td>
<td>(0.003)</td>
<td>(0.004)</td>
<td>(0.002)</td>
<td>(0.005)</td>
</tr>
<tr>
<td>Lasso</td>
<td>0.251</td>
<td>0.300</td>
<td>0.273</td>
<td>0.750</td>
<td>0.641</td>
</tr>
<tr>
<td></td>
<td>(0.004)</td>
<td>(0.005)</td>
<td>(0.005)</td>
<td>(0.006)</td>
<td>(0.004)</td>
</tr>
<tr>
<td>Elastic net</td>
<td>0.255</td>
<td>0.305</td>
<td>0.277</td>
<td>0.758</td>
<td>0.645</td>
</tr>
<tr>
<td></td>
<td>(0.005)</td>
<td>(0.003)</td>
<td>(0.003)</td>
<td>(0.004)</td>
<td>(0.005)</td>
</tr>
<tr>
<td>Oscar</td>
<td>0.260</td>
<td>0.308</td>
<td>0.281</td>
<td>0.761</td>
<td>0.650</td>
</tr>
<tr>
<td></td>
<td>(0.003)</td>
<td>(0.002)</td>
<td>(0.005)</td>
<td>(0.003)</td>
<td>(0.004)</td>
</tr>
<tr>
<td>KM+Lasso</td>
<td>0.253</td>
<td>0.302</td>
<td>0.275</td>
<td>0.758</td>
<td>0.641</td>
</tr>
<tr>
<td></td>
<td>(0.004)</td>
<td>(0.004)</td>
<td>(0.006)</td>
<td>(0.004)</td>
<td>(0.005)</td>
</tr>
<tr>
<td>KM+GLasso</td>
<td>0.254</td>
<td>0.307</td>
<td>0.277</td>
<td>0.752</td>
<td>0.643</td>
</tr>
<tr>
<td></td>
<td>(0.003)</td>
<td>(0.003)</td>
<td>(0.005)</td>
<td>(0.005)</td>
<td>(0.004)</td>
</tr>
<tr>
<td>pg-EN</td>
<td>0.271</td>
<td>0.320</td>
<td>0.293</td>
<td>0.783</td>
<td>0.681</td>
</tr>
<tr>
<td></td>
<td>(0.005)</td>
<td>(0.004)</td>
<td>(0.006)</td>
<td>(0.003)</td>
<td>(0.004)</td>
</tr>
</tbody>
</table>

Compare the effect of these two types of initialization on the final predictive performance of the model. In Table B.12 using the real datasets, we evaluate the performance of pg-EN on these data with either of two initialization methods. As seen from the table, pg-EN using K-means initialization may at times perform slightly better than pg-EN using random initialization.
### B.1. Additional Experiments to Evaluate pg-EN

#### Table B.11: Average classification performance of pg-EN compared to other methods for AMI (EMR) dataset without re-balancing the data. The numbers in brackets show the standard error over 100 iterations.

<table>
<thead>
<tr>
<th>Method</th>
<th>PPV</th>
<th>Sensitivity</th>
<th>F1 score</th>
<th>Specificity</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ridge</td>
<td>0.201</td>
<td>0.298</td>
<td>0.240</td>
<td>0.692</td>
<td>0.576</td>
</tr>
<tr>
<td></td>
<td>(0.003)</td>
<td>(0.001)</td>
<td>(0.005)</td>
<td>(0.002)</td>
<td>(0.005)</td>
</tr>
<tr>
<td>Lasso</td>
<td>0.248</td>
<td>0.345</td>
<td>0.288</td>
<td>0.702</td>
<td>0.580</td>
</tr>
<tr>
<td></td>
<td>(0.002)</td>
<td>(0.003)</td>
<td>(0.004)</td>
<td>(0.004)</td>
<td>(0.002)</td>
</tr>
<tr>
<td>Elastic net</td>
<td>0.251</td>
<td>0.349</td>
<td>0.291</td>
<td>0.708</td>
<td>0.587</td>
</tr>
<tr>
<td></td>
<td>(0.003)</td>
<td>(0.005)</td>
<td>(0.003)</td>
<td>(0.003)</td>
<td>(0.004)</td>
</tr>
<tr>
<td>Oscar</td>
<td>0.265</td>
<td>0.350</td>
<td>0.301</td>
<td>0.712</td>
<td>0.590</td>
</tr>
<tr>
<td></td>
<td>(0.004)</td>
<td>(0.003)</td>
<td>(0.005)</td>
<td>(0.002)</td>
<td>(0.002)</td>
</tr>
<tr>
<td>KM+Lasso</td>
<td>0.248</td>
<td>0.342</td>
<td>0.287</td>
<td>0.687</td>
<td>0.577</td>
</tr>
<tr>
<td></td>
<td>(0.001)</td>
<td>(0.005)</td>
<td>(0.004)</td>
<td>(0.004)</td>
<td>(0.004)</td>
</tr>
<tr>
<td>KM+GLasso</td>
<td>0.253</td>
<td>0.347</td>
<td>0.292</td>
<td>0.709</td>
<td>0.580</td>
</tr>
<tr>
<td></td>
<td>(0.002)</td>
<td>(0.004)</td>
<td>(0.005)</td>
<td>(0.002)</td>
<td>(0.006)</td>
</tr>
<tr>
<td>pg-EN</td>
<td>0.267</td>
<td>0.356</td>
<td>0.305</td>
<td>0.720</td>
<td>0.598</td>
</tr>
<tr>
<td></td>
<td>(0.004)</td>
<td>(0.003)</td>
<td>(0.005)</td>
<td>(0.004)</td>
<td>(0.004)</td>
</tr>
</tbody>
</table>

#### Table B.12: Comparison between Random and K-means initialization of pg-EN for real datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>PPV</th>
<th>Sensitivity</th>
<th>F1 score</th>
<th>Specificity</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Breast cancer</td>
<td>K-means</td>
<td>0.325</td>
<td>0.437</td>
<td>0.373</td>
<td>0.837</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.004)</td>
<td>(0.005)</td>
<td>(0.003)</td>
<td>(0.005)</td>
</tr>
<tr>
<td></td>
<td>Random</td>
<td>0.318</td>
<td>0.427</td>
<td>0.362</td>
<td>0.828</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.004)</td>
<td>(0.003)</td>
<td>(0.005)</td>
<td>(0.004)</td>
</tr>
<tr>
<td>Cancer (EMR)</td>
<td>K-means</td>
<td>0.323</td>
<td>0.392</td>
<td>0.354</td>
<td>0.831</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.005)</td>
<td>(0.002)</td>
<td>(0.004)</td>
<td>(0.002)</td>
</tr>
<tr>
<td></td>
<td>Random</td>
<td>0.316</td>
<td>0.384</td>
<td>0.342</td>
<td>0.826</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.004)</td>
<td>(0.005)</td>
<td>(0.005)</td>
<td>(0.005)</td>
</tr>
<tr>
<td>AMI (EMR)</td>
<td>K-means</td>
<td>0.315</td>
<td>0.419</td>
<td>0.359</td>
<td>0.773</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.004)</td>
<td>(0.005)</td>
<td>(0.003)</td>
<td>(0.005)</td>
</tr>
<tr>
<td></td>
<td>Random</td>
<td>0.300</td>
<td>0.407</td>
<td>0.352</td>
<td>0.767</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.003)</td>
<td>(0.004)</td>
<td>(0.003)</td>
<td>(0.005)</td>
</tr>
</tbody>
</table>
Bibliography


Shalev-Shwartz, S., Shamir, O., Srebro, N. and Sridharan, K. 2009. Learnability and stability in the general learning setting. In: *COLT*.


Every reasonable effort has been made to acknowledge the owners of copyright material. I would be pleased to hear from any copyright owner who has been omitted or incorrectly acknowledged.
This is a License Agreement between Iman Kamkar ("You") and Elsevier ("Elsevier") provided by Copyright Clearance Center ("CCC"). The license consists of your order details, the terms and conditions provided by Elsevier, and the payment terms and conditions.

All payments must be made in full to CCC. For payment instructions, please see information listed at the bottom of this form.

Supplier: Elsevier Limited
The Boulevard, Langford Lane
Kidlington, Oxford, OX5 1GB, UK

Registered Company Number: 1982084
Customer name: Iman Kamkar
Customer address: 1, Hayes court
Highton, Victoria 3216

License number: 3842820533143
License date: Apr 05, 2016

Licensed content publisher: Elsevier
Licensed content publication: Journal of Biomedical Informatics
Licensed content title: Stable feature selection for clinical prediction: Exploiting ICD tree structure using Tree-Lasso
Licensed content author: Iman Kamkar, Sunil Kumar Gupta, Dinh Phung, Svetha Venkatesh
Licensed content date: February 2015
Licensed content volume number: 53
Licensed content issue number: n/a
Number of pages: 14
Start Page: 277
End Page: 290
Type of Use: reuse in a thesis/dissertation
Portion: full article
Format: both print and electronic

Are you the author of this Elsevier article?: Yes
Will you be translating?: No
Title of your thesis/dissertation: Building stable predictive models for healthcare applications: A data-driven approach
Expected completion date: May 2016
Estimated size (number of pages): 200

Elsevier VAT number: GB 494 6272 12
This is a License Agreement between Iman Kamkar ("You") and Elsevier ("Elsevier") provided by Copyright Clearance Center ("CCC"). The license consists of your order details, the terms and conditions provided by Elsevier, and the payment terms and conditions.

**All payments must be made in full to CCC. For payment instructions, please see information listed at the bottom of this form.**

Supplier: Elsevier Limited  
The Boulevard, Langford Lane  
Kidlington, Oxford, OX5 1GB, UK

Registered Company Number: 1982084

Customer name: Iman Kamkar

Customer address: 1, Hayes court  
Highton, Victoria 3216

License number: 3842831129526

License date: Apr 05, 2016

Licensed content publisher: Elsevier

Licensed content publication: Journal of Biomedical Informatics

Licensed content title: Stabilizing l1-norm prediction models by supervised feature grouping

Licensed content author: Iman Kamkar, Sunil Kumar Gupta, Dinh Phung, Svetla Venkatesh

Licensed content date: February 2016

Licensed content volume number: 59

Licensed content issue number: n/a

Number of pages: 20

Start Page: 149

End Page: 168

Type of Use: reuse in a thesis/dissertation

Intended publisher of new work: other

Portion: full article

Format: both print and electronic

Are you the author of this Elsevier article?: Yes

Will you be translating?: No

Title of your thesis/dissertation: Building stable predictive models for healthcare applications: A data-driven approach

Expected completion date: May 2016

Estimated size (number of pages): 200
Title: Exploiting feature relationships towards stable feature selection
Conference Proceedings: Data Science and Advanced Analytics (DSAA), 2015. 36678
Author: I. Kamkar; S. K. Gupta; D. Phung; S. Venkatesh
Publisher: IEEE
Date: 19-21 Oct. 2015

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author’s approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author’s university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity’s name goes here]’s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.
This Agreement between Iman Kamkar ("You") and Springer ("Springer") consists of your license details and the terms and conditions provided by Springer and Copyright Clearance Center.

<table>
<thead>
<tr>
<th>License Number</th>
<th>3910021029798</th>
</tr>
</thead>
<tbody>
<tr>
<td>License date</td>
<td>Jul 15, 2016</td>
</tr>
<tr>
<td>Licensed Content Publisher</td>
<td>Springer</td>
</tr>
<tr>
<td>Licensed Content Publication</td>
<td>Springer eBook</td>
</tr>
<tr>
<td>Licensed Content Title</td>
<td>Stable Feature Selection with Support Vector Machines</td>
</tr>
<tr>
<td>Licensed Content Author</td>
<td>Iman Kamkar</td>
</tr>
<tr>
<td>Licensed Content Date</td>
<td>Jan 1, 2015</td>
</tr>
<tr>
<td>Type of Use</td>
<td>Thesis/Dissertation</td>
</tr>
<tr>
<td>Portion</td>
<td>Full text</td>
</tr>
<tr>
<td>Number of copies</td>
<td>1</td>
</tr>
<tr>
<td>Author of this Springer article</td>
<td>Yes and you are a contributor of the new work</td>
</tr>
</tbody>
</table>

Order reference number

Title of your thesis / dissertation
Building stable predictive models for healthcare applications: A data-driven approach

Expected completion date
May 2016

Estimated size (pages)
200

Requestor Location
Iman Kamkar
1, Hayes court

Highton, Victoria 3216
Australia
Attn: Iman Kamkar

Billing Type
Invoice

Billing Address
Iman Kamkar
1, Hayes court

Highton, Australia 3216
Attn: Iman Kamkar

Total
0.00 AUD

Terms and Conditions

Introduction
The publisher for this copyrighted material is Springer. By clicking "accept" in connection with completing this licensing transaction, you agree that the following terms and conditions apply to this transaction (along with the Billing and Payment terms and conditions established by Copyright Clearance Center, Inc. ("CCC"), at the time that you opened your Rightslink account and that are available at any time at http://myaccount.copyright.com). Limited License

With reference to your request to reuse material on which Springer controls the copyright, permission is granted for the use indicated in your enquiry under the following conditions:
- Licenses are for one-time use only with a maximum distribution equal to the number stated

https://s100.copyright.com/AppDispatchServlet