VARIABLE SELECTION AND CLASSIFICATION FOR LONGITUDINAL BINARY DATA THROUGH THREE-STEP SPARSE BOOSTING

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

DENIZ ESIN EMER

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
STATISTICS

JUNE 2022

Approval of the thesis:

VARIABLE SELECTION AND CLASSIFICATION FOR LONGITUDINAL BINARY DATA THROUGH THREE-STEP SPARSE BOOSTING

submitted by **DENIZ ESIN EMER** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Statistics Department, Middle East Technical University** by,

Prof. Dr. Halil KALIPÇILAR
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Özlem İLK DAĞ
Head of Department, **Statistics** _____

Prof. Dr. Özlem İLK DAĞ
Supervisor, **Statistics, METU** _____

Examining Committee Members:

Prof. Dr. Ceylan TALU YOZGATLIGİL
Statistics, METU _____

Prof. Dr. Özlem İLK DAĞ
Statistics, METU _____

Prof. Dr. Pınar KARAGÖZ
Computer Engineering, METU _____

Prof. Dr. Serpil AKTAŞ ALTUNAY
Statistics, Hacettepe University _____

Prof. Dr. Hilal ÖZDAĞ SEVGİLİ
Biotechnology Institute, Ankara University _____

Date: 13.06.2022

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Deniz Esin Emer

Signature :

ABSTRACT

VARIABLE SELECTION AND CLASSIFICATION FOR LONGITUDINAL BINARY DATA THROUGH THREE-STEP SPARSE BOOSTING

Emer, Deniz Esin
Ph.D., Department of Statistics
Supervisor: Prof. Dr. Özlem İLK DAĞ

June 2022, 194 pages

With the rapid evolution of technology, it is now possible to obtain the gene expression levels of thousands of genes in a single experiment. In these experiments, the sample size is relatively small but the number of covariates under consideration is extremely large, whereas only a small number of expressions may be related to the outcome of interest. Hence, the selection of causal features is much-needed along with the model estimation. In this thesis, we propose a three-step sparse boosting model for detecting the most important covariates that classify the individuals into binary groups considering the longitudinal data having spatial and temporal correlations. Following the idea of Yue, Li and Cheng (2019), in the first step, the independence of the observations is assumed and the estimation of coefficients of logistic regression is obtained by directly minimizing the binary-cross entropy loss using boosting method. Then, in the second step, the temporal correlation is considered by executing a weight matrix constructed based on the errors made in the first step. Finally, in the third step, the spatial correlation is added via a weight matrix considering the correlation structure. A Monte Carlo Simulation Study was designed and nine different temporal and spatial correlation structure scenario were run using parallel computing

methods. The proposed model decreased the number of mistakenly chosen significant covariates while establishing all the true ones as significant. As the classification performance, it got higher when the spatial and temporal correlations got higher. Also, a comparison study, including Boruta (RF), Support Vector Machine (SVM), Logistic Regression, Ridge Regression, Lasso Regression and Elastic Net algorithms, showed that i) they considered very large number of mistakenly chosen significant covariates, whereas our proposed algorithm identified at most one along with the true significant variables and, ii) Three-Step Sparse Boosting algorithm performs the best in terms of specificity and precision metrics in the simulation study. In addition, the algorithm had been applied on a real life data set of Type 1 Diabetes Prediction and Prevention (DIPP) study, using both balanced and unbalanced sets. Our algorithm identified a few numbers of genes as significant which can be beneficial regarding time and money. The comparison results showed that The Three-Step Sparse Boosting technique can be considered as performing well in terms of variable selection, estimation and classification.

Keywords: Binary longitudinal data, logistic regression, spatial and temporal correlations, sparse boosting, variable selection, classification

ÖZ

ÜÇ AŞAMALI SEYREK YÜKSELTME METODU İLE İKİLİ SONUCU OLAN UZUNLAMASINA VERİLERİN DEĞİŞKEN SEÇİMİ VE SINIFLANDIRILMASI

Emer, Deniz Esin
Doktora, İstatistik Bölümü
Tez Yöneticisi: Prof. Dr. Özlem İLK DAĞ

Haziran 2022 , 194 sayfa

Teknolojinin hızlı gelişimi ile tek bir deneyde binlerce genin gen ifade düzeylerini elde etmek artık mümkün. Bu deneylerde, numune boyutunun nispeten küçük olmasının yanında incelenen değişkenlerin sayısı son derece fazladır, buna karşın yalnızca az sayıda gen ifadesi, ilgilenilen sonuçla ilgili olabilmektedir. Bu nedenle, model tahmini ile birlikte neden ilişkisi olan değişkenlerin seçimine ihtiyaç artmıştır. Bu tezde biz, bireylerin ikili sınıflandırılması amacıyla, en önemli değişkenleri tespit etmek için sağlam ve aynı zamanda uzamsal ve zamansal korelasyonu göz önünde bulunduran, üç aşamalı seyrek bir yükseltme modeli öneriyoruz. Yue, Li and Cheng (2019) makalesindeki fikri takiben, ilk aşamada, gözlemlerin bağımsız olduğu varsayılır ve yükseltme metodu kullanılarak, ikili çapraz entropi kaybının doğrudan en aza indirilmesiyle lojistik regresyonun katsayılarının tahmini elde edilir. Daha sonra, ikinci aşamada, birinci aşamada yapılan hatalara dayalı olarak oluşturulan bir ağırlık matrisi yürütülerek zamansal korelasyon dikkate alınır. Son olarak üçüncü aşamada, korelasyon yapısı dikkate alınarak bir ağırlık matrisi aracılığıyla uzamsal korelasyon ekle-

nir. Bir Monte Carlo Simülasyon Çalışması tasarlanmış ve paralel hesaplama yöntemleri kullanılarak dokuz farklı zamansal ve uzamsal korelasyon yapısı senaryosu çalıştırılmıştır. Önerilen model, anlamlı olan bütün değişkenleri doğru bir şekilde anlamlı olarak belirlerken, yanlışlıkla anlamlı olarak belirlenen değişkenlerin sayısını azaltmıştır. Sınıflandırma performansı ise uzamsal ve zamansal korelasyon yükseldikçe yükselmektedir. Ayrıca, Boruta (RF), Ridge Regresyon, Lasso Regresyon ve Elastic Net algoritmalarını içeren bir karşılaştırma çalışması yapılmış ve şu sonuçlar elde edilmiştir, i) diğer algoritmalar çok sayıda yanlışlıkla anlamlı olarak seçilmiş değişkenleri dikkate alırken, önerilen algoritma, sadece gerçek anlamlı değişkenleri tanımlamanın yanında, en fazla bir değişkeni yanlışlıkla anlamlı olarak seçmiştir , ii) Üç Aşamalı Seyrek Yükseltme algoritması, simülasyon çalışmasında özgüllük ve kesinlik metrikleri açısından en iyi performansı göstermiştir. Ayrıca algoritma, Tip 1 Diyabet Tahmin ve Önleme (DIPP) çalışmasının gerçek yaşam veri seti üzerinde hem dengeli hem de dengesiz kümeler kullanılarak uygulanmıştır. Algoritma, birkaç önemli geni anlamlı olarak tanımlamıştır, bu sayıda genin tanımlanması da zaman ve para açısından faydalı olabilecektir. Karşılaştırma sonuçları, Üç Aşamalı Seyrek Yükseltme algoritması değişken seçimi, parametre kestirimi ve sınıflandırma açısından iyi performans gösterdiğinin kabul edilebileceğini göstermiştir.

Anahtar Kelimeler: Uzunlamasına ikili veri, lojistik regresyon, uzamsal ve zamansal korelasyonlar, seyrek yükseltme, değişken seçimi, sınıflandırma

To all my challanges

ACKNOWLEDGMENTS

First of all, I would like to express my deepest gratitude to my thesis supervisor Prof. Dr. Özlem İlk Dağ for making this journey a cheerful, informative and fruitful one. It is certainly an honor to work with her. I am grateful to her for her support, motivation and guidance throughout my Ph.D. study.

I would like to express my sincere gratitude to my thesis committee members Prof. Dr. Pınar Karagöz and Prof. Dr. Ceylan Talu Yozgatlıgil, firstly for making time to be a committee member in their really busy schedule for more than two years. Their valuable guidance, comments and suggestions have improved my thesis study significantly. Also, I would like to thank Prof. Dr. Hilal Özdağ Sevgili and Prof. Dr. Serpil Aktaş Altunay for their intellectual contributions to our study.

I am indebted to my family, Ayla, Birol, Onur, Meltem and Demir Emer, for their existence, different points of view, support and endless patience.

I would like to thank Gözde Polatkal for adding depth to my life and enriching it.

Also, I would like to thank everyone who makes me grow and evolve myself.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

AB	Absolute Bias
AIC	Akaike Information Criterion
BCE	Binary Cross-Entrophy
BiMM	Binary Mixed Model
CS	Correctly Identified as Significant
DIPP	Type 1 Diabetes Prediction and Prevention
GLM	General Linear Model
GLMM	Generalized Linear Mixed Model
MAE	Mean Absolute Error
MC	Monte Carlo
MLE	Maximum Likelihood Estimation
MRMR	Minimum Redundancy Maximum Relevance
mRNA	Messenger Ribonucleic Acid
MS	Mistakenly Identified as Significant
MSE	Mean Square Error
RF	Random Forest
SVM	Support Vector Machine

CHAPTER 1

INTRODUCTION

Because of the respectable achievement in the DNA microarray technology, it is now possible to simultaneously monitor the gene expression levels of thousands of genes in a single experiment, instead of focusing on one or two specific genes in an organism (Thomas et al. (2001), Inza et al. (2004)). This technology has been applied to identify i) differences between diseased and healthy subjects, ii) differences between different types and subtypes of diseases and, iii) marker genes. By this approach, the understanding of the disease at a molecular level can be increased and new diagnostic and prognostic methods can be developed (Bø and Jonassen (2002)). In addition, due to dynamic nature of biological systems or processes, to capture this biologically meaningful dynamic changes, researchers investigate gene expression patterns across time. With the rapid evolution of technology, time series/longitudinal microarray experiments also become widespread. Consequently, developing models for longitudinal data that predict the phenotype of a new subject carry a great importance. In these experiments, the sample size is relatively small but the number of covariates under consideration is extremely large, which imply a serious statistical challenge. In addition, although the number of gene expression evaluated is large, there may be only a small number of expressions that are associated with variations of phenotypes. Hence, the selection of causal features is much-needed along with the model estimation. Díaz-Uriarte and Alvarez de Andrés (2006) stated that biomedical researchers conduct studies based on generally one of the following objectives:

1. "To identify relevant genes for subsequent research; this involves obtaining a (probably large) set of genes that are related to the outcome of interest, and this set should include genes even if they perform similar functions and are highly

correlated.

2. To identify small sets of genes that could be used for diagnostic purposes in clinical practice; this involves obtaining the smallest possible set of genes that can still achieve good predictive performance (thus, "redundant" genes should not be selected)."

We focus in this thesis on the second item, which is variable selection under the classification framework. Variable selection methods aim to reduce the size of data while preserving the information and variance of significant variables as high as possible. These methods have been used to improve the performance of the classification model that generalizes better to unseen points and to obtain faster and cost-effective methods (related to the both storage and time), as well as more quality data. Also, underlying processes that generated the data are identified more accurately due to decreasing redundant, irrelevant and noise information. In addition, obtaining essential variables has the advantage of overfitting evading, which is a very important issue.

Variable selection carries another important advantage in the context of medical studies by improving the practicality, easing the use of models, reducing the burden and need of high volume of data collection, as well as the decision support regarding whether the expensive treatment/surgery is necessary or not. Also, elimination of irrelevant variables which are inconsequential for predicting outcomes is desired by the researchers. In addition, if there exists a significant cost associated with the collection of certain predictors, variable selection techniques present cost beneficial results.

On the other hand, including more genes in the study should provide more information and discriminating power. However, several problems may rise; i) the increase in cost and computational complexity, ii) including too many insignificant covariates in the model and, iii) decrease in the classification performance measures (Speiser, 2021)). Along with the noise reduction and increase in the performance of classification, significant genes may also carry important biological information and may be used to identify possible future research directions.

When the difference from other dimensionality reduction techniques, like those based on projection (e.g. principal component analysis) or compression (e.g. using infor-

mation theory) is considered, in contrast to them, variable selection techniques do not change the original distribution of the variables, but just select a subset of them which are considered significant (Saeys et al. (2007)).

Consider a data set where repeated measures on the same subject over time are taken and in order to predict the binary outcome, like the subject being diseased or non-diseased, a longitudinal study is designed. Longitudinal study structure imposes a correlation within a subject because observations for the same subject are dependent, as temporal and spatial correlations. Whereas a basis for model estimation is provided by the standard modeling techniques, advanced methodologies are needed to address the challenges of correlated data sets with many potential predictors. Then, the research question is raised as to determine the significant variables of the binary outcome given the measurement of many variables at each measurement time? Since the assumption that all observations are independent is violated for such data, one crucial issue in longitudinal analysis is how to take into account the correlation within subjects and make efficient and accurate inference for high dimensional data. This problem motivated us to develop a model which considers the spatial and/or temporal correlation structures of the data when classifying the subjects as positive or negative.

Support Vector Machines (Luts et al., 2012), Neural Networks (Xiong et al., 2019), Decision Tree (Abdolell (2002), Zhang and Ye (2008)) and Random Forest (Hajjem et al. (2014), Speiser et al. (2019)) algorithms have been proposed in the context of longitudinal study. Speiser (2021) stated that "The main idea behind these methods is to incorporate machine learning models into the generalized linear mixed model (GLMM) framework to account for correlation among subjects and longitudinal outcomes. Many methods employ a process similar to the Expectation-Maximization algorithm in which a machine learning model is fit, followed by a GLMM fitted with results or output from the machine learning model, and then the outcome is updated based on the fitted GLMM. This process is iteratively repeated until convergence is reached."

Therefore, our research question is defined as to determine the significant genes of the binary outcome given the measurement of thousands of probes at each measurement time. We focus on binary outcomes since many medical studies use binary class

which change over time between classes dynamically. Following, our objective can be stated as to obtain the most significant genes, in order to distinguish the diseased and non-diseased group by binary classification where data have spatial and temporal correlations.

Hence, in this thesis, we propose a three-step sparse boosting model for detecting the most important genes that classify the individuals into patient or control groups considering the longitudinal data having spatial and temporal correlations. The estimation of parameters and selection of the covariates are conducted simultaneously in order to improve the prediction performance, as well as the interpretability of the model.

Following the idea of Yue, Li and Cheng (2019), in the first step, the independence of the observations is assumed and the estimation of coefficients of logistic regression is obtained by directly minimizing the binary-cross entrophy loss using boosting method. Then, in the second step, the temporal correlation is considered by executing a weight matrix constructed based on the errors made in the first step. Finally, in the third step, the spatial correlation is added via a weight matrix considering the correlation structure.

By this approach, we overcome the challenge of obtaining the optimal tuning parameters, which is a time consuming procedure and is used broadly in Random Forest, Support Vector Machines, and regularization methods such as Lasso. Therefore, boosting is considered as an effective alternative tool for high-dimensional modeling, where base learners are formed to iteratively minimize the regarding loss function. Also, since boosting leads relatively smaller computational burden, it has much less over-fitting risk and to incorporate extra constraints it needs simple adjustments. Also, the assumption of sparsity is usually made when dealing with the high-dimensional model. Therefore, we work on sparse boosting algorithm. Boosting was proposed in the computational learning theory literature by Schapire (1990), Freund (1995), Freund and Schapire (1997). Freund and Schapire (1997) stated that, "To be easy to implement, effective, and efficient, the objective function L must be amenable to an efficient search for the best coordinate to adjust, and the amount of adjustment must also be easy to compute. Moreover, to avoid local minima, convexity and smoothness

of the function L appear to be useful qualities.".

As realizing our goal, the following contributions is made to the literature. As to our knowledge, there is no study regarding binary classification with longitudinal data through boosting algorithm. Moreover, our model can handle spatial and/or temporal correlation in different steps, while classifying the binary responses.

The remainder chapters of this thesis are organized as follows: In Chapter 2, the studies about variable selection for longitudinal data, binary classification with longitudinal data are reviewed and the base articles of this thesis are presented. Chapter 3 discusses the background information that is used in developing the model related to data and models, introduces the proposed model and assumptions, and the estimation technique of the model. Assessment of the validity of the proposed model is evaluated in Chapter 4, using different Monte Carlo simulation scenarios, as well as using a real life data set of Type 1 Diabetes Prediction and Prevention (DIPP) for both balanced and unbalanced scenarios. Comparison results between our algorithm and Boruta (Random Forest), Support Vector Machine, Logistic Regression, Ridge Regression, Lasso Regression and Elastic Net are also presented in Chapter 4 for both simulation study and real data set. Finally, discussion related to the thesis study, its main findings, implications and limitations have been made and some extension ideas of the proposed model as future work are presented.

CHAPTER 2

LITERATURE REVIEW

In this chapter, we first review the literature on variable (feature) selection studies for longitudinal data, and then continue with the studies on binary classification with longitudinal data. After reviewing some related studies, we discuss three main articles which constitute as a base to this thesis study. Our model is developed by improving and combining the methodologies that are conducted in those studies.

We review variable selection and binary classification for longitudinal data studies using known machine learning algorithms such as Naïve Bayes, Support Vector Machines, Random Forests, Neural Networks and Extreme Value Models.

2.1 Variable Selection for Longitudinal Data

Variable selection methods are first organized into three categories based on the search mechanisms in the feature subset space; filter methods, wrapper methods and embedded methods. Later, hybrid and ensemble methods are included as separate categories.

Filter methods use the intrinsic properties of data depending on four different kinds of measurement criteria, i.e. information, dependency, consistency, and distance. Also, they are independent of the classification or clustering algorithms. These methods' results are feed into the classification or clustering algorithms as inputs. Yet, they do not consider variable dependencies, which may poorly affect the performance of the algorithms. Wrapper methods, actually wrap the search algorithm around the classification or clustering model by training and evaluating a specific subset of features in the model. The wrapper methods carry high risk of overfitting compared to filter methods and they take long time to complete. In embedded methods, variable selec-

tion is utilized within the classification or clustering models. These methods contain the advantages of both filter and wrapper methods. Our proposed algorithm also falls into this category.

Beginning with the filter methods, Thomas et al. (2001) performed a regression analysis that accounts for the heterogeneity and complexity of the data to identify the diffentially expressed genes between two groups. They compared their results with t-tests or Wilcoxon rank sum statistics and stated that they found diffentially expressed genes with 1% significance at the genomic level using acute myeloid leukemia and acute lymphoblastic leukemia samples.

Dudoit, Fridlyand and Speed (2002) compared nearest neighbor classifiers, Fisher linear discriminant analysis, diagonal linear discriminant analysis, classification trees, bagging and boosting techniques to classify the tumors, based on three published cancer gene expression data. Before classification, using the ratio of genes' between-groups to within-groups sum of squares, they performed a gene selection preprocess. They used accuracy as the performance measure in comparison study, where they concluded that diagonal linear discriminant analysis and nearest neighbor classifiers perform really well compared to the others. Yet, while the former ignored the correlation between genes, the latter handled them in a black box.

Calculation of rank products from replicate experiments based on the analysis of biological reasoning was proposed by Breitling et al. (2004) to identify differentially expressed genes under two experimental conditions. Due to the noisiness of the data, this task was considered as complicated. By this methodology, also the significance level for each gene could be presented, which provided control of the false-detection rate and familywise error rate. They utilized their proposed technique on three real data set and compare it to the non-parametric t-test variant, where their approach performed more reliably and consistently in highly noisy data. Also, they stated that they established the biologically relevant expression changes by an analysis of the physiological function of the identified genes.

In their paper, Cantoni, Flemming and Ronchetti (2005) proposed a generalized version of Mallows's C_p (GC_p) by adding the rescaled weighted sum of squared error. In simulation study, they constructed two setups, using noncontaminated and con-

taminated data in a marginal longitudinal model with logistic link and compared the behavior of the z-test, the backward stepwise selection procedure and their suggested method. In z-test, they fitted the full model and selected all the variables that the Wald test gave a p-value lower than 0.05. In the backward selection, they retained the variables having a p-value lower than 0.1 based on the Wald test. As a result GC_p labeled as a good model based on the percentage of containing the true model generating the data. GC_p also worked in a more robust way in the simulation study. However, in the real data application, GC_p selected larger models.

Regarding set analysis, rather than individual analysis, Zhang et al. (2011) suggested a robust nonparametric approach to find whether a set of gene showed a dynamic pattern that differs between multiple treatments or experimental conditions to achieve reliable type I error. They used permutation tests based on two nonparametric Wald test statistics. These statistics converged to their limiting distributions. They needed the limiting distribution, since the number of genes goes to infinity and theoretically linear mixed models and generalized estimated equations were shown not to be suited for such data. They conducted simulation study and concluded that the proposed method has a greater power than other methods, like LME and GEE for various distributions and heteroscedastic correlation structures.

Bø and Jonassen (2002) proposed a method for ranking genes based on gene pairs' ability to select gene sets that generalize the differences between experiment classes, such as healthy versus diseased tissues. They considered gene pairs in their study by assigning a separation score to the pair. They compared their method with forward selection and a gene-ranking method based on evaluating each gene separately in terms of cross-validation prediction accuracy on two public datasets. They showed that evaluating genes in pairs increases the prediction accuracy, since they identified some genes "that are not obviously good discriminators alone, but discriminate well when used in pairs with other genes". Also, they stated that, gene pair selection result in a more robust way in terms of distinguishing classes.

Gevaert et at. (2006) integrated clinical data patient history, laboratory analysis, ultrasound parameters and microarray data using Bayesian networks. They evaluated decision integration, partial integration and full integration methods, and utilized them

using data on breast cancer patients to classify them into poor and a good prognosis group. They found that the decision integration where they used a weight of 0.6 for predicted probabilities of the clinical model and a weight of 0.4 for predicted probabilities of the microarray model, and partial integration performs better in terms of the area under the ROC curve. Then, again using the areas under the ROC curve, these two integration methods' classification performance on the test data were observed and it was seen that the partial integration performs the best. Also, using a Markov Blanket, they showed that Bayesian networks identified the (in)dependency relationships with the class variable by selecting the significant variables.

In their study, Ding and Peng (2005) defined a minimum redundancy — maximum relevance (MRMR) framework and applied it on gene selection, for both categorical and continuous variables. To combine relevance and redundancy, Mutual Information Difference criterion and Mutual Information Quotient criterion were used. To compare their proposed method, they used Naïve Bayes, linear discriminant analysis, logistic regression, and support vector machines on six data sets of gene expressions in terms of accuracy. They reported that "genes selected via MRMR provide a more balanced coverage of the space and capture broader characteristics of phenotypes".

As for the multiclass prediction, Yeung et al. (2003) developed the uncorrelated shrunken centroid and error-weighted, uncorrelated shrunken centroid algorithms based on shrunken centroid, where a sample was assigned to a class based on its nearest average pattern. The algorithms did not have any assumption regarding the distribution of data. They used the interdependence structure of genes to select the significant ones. They utilized their approaches on three real data set and one synthetic data set, and assessed the prediction accuracy. They also presented a comparison study with the published results. They concluded that removing highly correlated genes improves the results.

Moving on to the wrapper methods, Blanco et al. (2004) proposed a new Estimation of Distribution Algorithms based on The Univariate Marginal Distribution Algorithm wrapped with Naïve Bayes classification and evaluated the performance using the accuracy results. The proposed approach was run with two biological data sets. They showed that the accuracy level of the whole set was significantly improved.

A comparison between different filter methods and a wrapper sequential search procedure was carried out by Inza et al. (2004). In the study they used Shannon-entropy, Euclidean-distance, Kolmogorov-dependence, Kullback Leibler metrics for discrete data, and P-metric and t-score for continuous data to measure the relation between gene and the problem class. Then, four classification algorithms, namely k-nearest neighbor, Naïve Bayes, decision tree, If-Then rule based classification, were used on two well-known DNA microarray data sets. Along with different results obtained from different metrics for different classification algorithms, they concluded that if the variable selection procedure was applied then the accuracy results were better, compared to the case that the variable selection procedure was not applied.

Ruiz et al. (2006) proposed a heuristic based on the statistical significance of adding a gene from a ranked-list. Their heuristic began with rankings of genes by any evaluation criterion, selected the first ranked gene, then added genes one by one to the subset of selected genes only if such inclusion improved the classifier accuracy. They measure the improvement using incremental ranked usefulness. In the comparison study, the genes were ordered according to their individual predictive power. They used four real data sets and three representative subset evaluation measures in combination with sequential forward search engine; sequential forward, fast correlation-based filter algorithm that exploits SF search and consistency based filter algorithm that again exploited SF search for the comparison. Also, they chose three classification algorithms as Naïve Bayes, an instance-based algorithm and a decision tree algorithm, and evaluate the accuracy of the algorithms utilizing the selected genes. In conclusion, their proposed heuristic selected small subset of genes from the original set while performing on predicting as well as other methods.

Gruebner et al. (2016) assessed the spatial distribution of post disaster mental health wellness and general wellness. They showed predictors behave differently for different geographic space. They applied the spatial scan statistic and geographically weighted regression. Using a cohort study of Hurricane Ike survivors, they found spatial clusters of high likelihood wellness and spatial concentrations of low likelihood wellness.

In addition, Hsu, Finkelstein and Schoenfeld (2018) aimed to discover the groups of

genes associated with the disease process and they used the binary or failure time outcomes added to the clustering algorithm. They developed a Bayesian approach. Their model was a linear model with normally distributed cluster specific random effects for the longitudinal gene expression trajectory, and they applied an MCMC algorithm. They assumed, the genes within each cluster could be expressed as the sum of a random effect, specific to the observation, the cluster and the independent error term. They used microarray data collected from trauma patients and compared the results with and without the outcome.

As an embedded method, Díaz-Uriarte and Alvarez de Andrés (2006) proposed the random forest method as a variable selection algorithm due to some advantages of it, like noise handling, utilization for more than two classification class, usage when the number of variables is much larger than the number of observations and measurement of variable importance. They used both simulated and nine real data sets, for the comparison of "Diagonal Linear Discriminant Analysis, K nearest neighbor and Support Vector Machines with linear kernel" and they concluded that random forests often chooses very small sets of genes while having comparable performance to others.

Guyon et al. (2002) proposed Support Vector Machine methods based on Recursive Feature Elimination to select significant genes. Using two real data sets, they showed SVM takes into consideration of mutual information between genes which affects the classification performance. They compared SVM with the baseline method that makes implicit orthogonality assumptions and report better classification performance and are biologically relevant to cancer.

A method for variable selection and classification using the ROC technique was proposed by Ma and Huang (2005), where they used "a sigmoid approximation to the area under the ROC curve as the objective function for classification and the threshold gradient descent regularization method for estimation and biomarker selection". As an approximation method, they used sigmoid maximum rank correlation. They used one simulated data and two real data sets in their comparison study, where logistic model is compared. As a conclusion, their proposed technique performed well in terms of classification whereas it was not necessarily optimal for all data sets, since area under the ROC curve and classification error were two different measures of

classification performance.

Longitudinal data was also modeled using a tensor based approached by Chen, Xu and Bi (2018). To analyze longitudinal data and select variables, they proposed a tensor based quadratic interference function which took into consideration of within subject correlation. The model coefficient, a k-mode tensor, was decomposed into a summation of k tensors of the same dimension. Also, for the variable selection, they introduced a tensor latent norm as a group penalty. They used a linearized block coordinate descent algorithm for the optimization problem and used both synthetic and real data to illustrate the results of their proposed model.

Huang and Pan (2020) introduced a "smoothly clipped absolute deviation-based and least absolute shrinkage and selection operator-based penalized joint generalized estimating equation methods to simultaneously model the mean and correlations for longitudinal binary data, together with variable selection in the mean model". In the penalization approach, to calculate the tuning parameter, they used quasi-generalized cross-validation. Also, the correlation matrix was modeled through a specific parametric form. In simulation study, they evaluated model fitting and variable selection performance of the proposed Penalized joint generalized estimating equations for longitudinal binary data. They showed that their proposed method produced better variable selection consistency and parameter estimation accuracy compared to existing penalized generalized estimating equation methods.

Binary Mixed Model (BiMM) forest method was developed by Speiser (2021) which incorporated random forest into generalized linear mixed model (GLMM) framework. The author utilized the model using clustered and longitudinal data with binary outcomes. In the simulation study that the author carried on, the comparison between BiMM forest with backward elimination, BiMM forest with stepwise selection, Lasso and GLLM with backward elimination. The simulation study results suggested that BiMM forest with backward elimination had the highest number of correctly identified features, the highest accuracy score and lowest computational time compared to other methods in some scenarios and it obtained similar results in the other scenarios. In addition, the author run the model to predict mobility disability in older adults using the Health, Aging and Body Composition Study dataset and the results turned out

to be similar to other models in terms of accuracy and specificity, while BiMM forest with backward elimination having the highest sensitivity.

Klén et al. (2020) introduced a robust linear mixed binary classifier that can incorporate the data with unaligned time points. They compared their model using four simulated data with linear mixed-effects model, linear feature extraction, logit mixed-effects regression, Lasso, random forests, support vector machines and neural networks in terms of sensitivity and specificity. In all scenarios, their model had the highest sensitivity and specificity results where all methods had fairly good performances. Also, again the same algorithms were run using three different sets of real data, both their method and random forest obtained high specificity and sensitivity results.

In their study, Liu et al. (2018) combined double RBF-kernels with weighted analysis to identify relevant variables in gene expression data for Cancer classification. Filter based methods such as χ^2-statistic, maximum relevance and minimum redundancy, Relief-F, Information Gain and Fisher Score were used on four different real data sets, in the comparison part of their study. As the comparison metrics, accuracy, true positive rate and true negative rate were chosen. In conclusion, their method outperformed all other used methods in terms of classification accuracy, true positive and negative rates, and in the identification of cancer genes.

2.2 Binary Classification with Longitudinal Data

Within the context of binary classification of longitudinal data, very broad range of methods has been used including logistic regression, functional models like quadratic inference function, boosting models and deep learning based models.

Serban, Staicu and Carroll (2013) proposed a logistic regression conditioned on three latent processes describing the within and between variability and the cross-dependence of the repeated longitudinal measurements. They assumed an approximation to the logistic link function. They also compared linear and exponential approximation to logistic regression function. They assumed the correlation between two functional observation tail off as the distance between their associated points increases. The

relationship between the functional data and the latent process was stated as non-linear. They compared linear and exponential approximations to the logistic regression function in a simulation study, where they stated that "the linear approximation is computationally efficient whereas the exponential approximation applies for rare events functional data". Based on this computational efficiency, they recommended linear approximation approach for the estimation of the within covariance and spatial dependence.

Logistic prediction model which utilized the extensive research on identifying differentially expressed genes based on the false discovery rate and incorporates the sign of the coefficients as random effects was proposed by Liao and Chin (2007). To estimate the number of significant genes to be included in the logistic regression and the shrinkage parameter in penalized likelihood, they used a parametric bootstrap model based on the prediction error as the Brier Score. Two real data sets were used to measure the performance of the proposed method, where they reported very accurate results.

Vogl et al. (2015) used an elastic net regularized generalized linear model regression to predict if and when retinal disease recurrence occurs in the future. Time-to-event was predicted by using Cox proportional hazard model. They proposed a “data-driven algorithm to identify spatio-temporal predictive imaging biomarkers or signatures in longitudinal medical imaging data". They overcame the sparsity by making relevant coefficients nonzero by minimizing the regularized negative partial log-likelihood. They conducted cross-validation experiments, where they demonstrated that predictive and interpretable features were identified in the spatio-temporal signature. Also, they compared their model with elastic net in terms of mean absolute error and concluded that their model obtained lower MAE.

In addition, functional models for classification of longitudinal data are studied. Functional data analysis uses infinite dimensional data. For example, Müller (2005), extended generalized functional linear model to the case of sparse longitudinal predictors and addressed irregular noisy data. This approach included binary regression models for longitudinal data. The author utilized binomial functional regression for sparse data and using the one-leave-out prediction error criterion, the author obtain

26.54% overall misclassification rate.

Wang and Qu (2014) developed a new classifier QIFC that built on the model information derived from the quadratic inference function for longitudinal data. They defined the shortest distance to the subject based on the information between the longitudinal responses and covariates for each class, then classification was done according to that shortest distance. They stated that "for finite sample applications, this enables one to overcome the difficulty in estimating covariance matrices while still incorporating correlation into the classifier". They showed that QIFC performs better than the functional data classifier, support vector machine, logistic regression, linear discriminant analysis, Naïve Bayes and the decision tree algorithms in a simulation study. They also used two different real data sets to assess the performance of their proposed algorithm in terms of classification errors, where QIFC performs again the best.

Hyun, et al. (2016) developed a spatio-temporal Gaussian process framework which used a functional principal component model to capture a large proportion of spatio-temporal dependence structure and a partition parametric space-time covariance model to capture some local spatio-temporal correlations. They developed a three-stage estimation procedure using kriging technique. They stated that, their model could capture non-stationary and non-separable spatio-temporal dependence structure because of strong heterogeneity in longitudinal data with a small number of parameters. They used two real datasets to illustrate their results.

Lee (2019) developed a deep learning-based python package to provide the preconstructed deep learning architecture for a classification of biomarkers from multimodal data, which could work on both time series and non-time series data. In the constructed framework, there existed two phases; in the first phase, recurrent neural network with the extension of gated recurrent unit was adopted to obtain learning feature representation. In the second phase, they stated that any classification algorithm can be utilized, and they used linear regression with l_1-regularization as the classification algorithm due to easy interpretability of the coefficients. They utilized their model on simulated multimodal time series data and real data set to validate the expectation of higher accuracy by combining the linear model and the deep learning model. To

compare, they used logistic regression, random forest and support vector machine algorithms in the simulated data as well, where their method performed the best in terms of classification accuracy based on modality of the data.

2.3 The Base Articles

Now, we review the three main articles which have ideas to be improved and combined for this thesis. The first main study we considered was carried out by Yue, Li and Cheng (2019). They proposed two-step sparse boosting approach for variable selection and model-based prediction where they used boosting to provide the integration between them. They stated that, the available regularization methods for variable selection needed the optimal tuning parameters for the degree of regularization, which was a time-consuming numerical procedure. Hence, they considered boosting algorithms as an alternative. Also, in their nonparametric model, the regression coefficients varied as functions of covariates. Balanced longitudinal data was used with a continuous outcome, in high dimensional varying coefficient models. In the first step, the independence of subjects was assumed, and sparse boosting technique was applied to facilitate an estimate of the correlation structure. Again, using sparse boosting in the second step, they overcame the independence assumption, considered the within subject correlation and conducted variable selection and parameter estimation simultaneously. In this step, they used weighted least square loss function in the boosting algorithm. The advantage of their algorithm was illustrated by extensive numerical examples, and as a real example, an application of yeast cell cycle gene expression data was used.

Since our response will be binary and the boosting approach produces accurate prediction results, we reviewed the boosting algorithms for binary classification to improve the above study. Friedman, Hastie and Tibshirani (2000) and Friedman (2001) showed boosting methods can be used as stagewise estimation procedures for fitting an additive logistic regression model which minimize an exponential criterion (which to second order is equivalent to the binomial log-likelihood). Also, they proposed a boosting procedure which optimizes the Bernoulli log-likelihood, since the only justification for exponential criterion was it has a widely used population minimizer.

Menezes, Liska, Cirillo and Vivanco (2016) used the results of the previous studies, that boosting can be viewed as a method for functional estimation, and compared the logistic regression models for binary classification: one was estimated by maximum likelihood procedure, the other was estimated using the Binomial Boosting algorithm. They used coronary heart disease data to illustrate and concluded that logistic regression estimated via Binomial Boosting produces lower values of the information criteria AIC and BIC, higher sensitivity, specificity and accuracy, lower rates of false positive and false negative, making it a better fit where the response is binary.

Adewale, Dinu and Yasui (2010) proposed two new variants of boosting algorithms with correlated binary responses in order to overcome the correlation characteristic of longitudinal data. First one considered the generic functional gradient descent algorithm with a weighted L2 loss function, which was actually weighted least squares loss with the variance-covariance matrix accounting for the correlation in the data. A direct likelihood optimization boosting approach via approximation was modified as the second method. They defined likelihood-based boosting as "a stagewise optimization of the likelihood of a variety of link functions and distributions in the exponential family of distributions". For correlated data, they proposed a likelihood-based boosting via generalized linear mixed model and setting the random effect to zero. They evaluated the performances and computer efficiencies by a simulation study and illustrated the first method with the real data on acute lymphoblastic leukemia. They found that both of them performed well, but the computational efficiency of the generic functional gradient descent algorithm exceeded the other.

We can summarize these three main articles and their connection to our thesis as in the following table:

Table 2.1: The summary of four main articles

Source	**The parts we adopted**	**Our contributions**
Yue, Li and Cheng (2019)	• Stepwise sparse boosting	• Adopt it for binary response • Include spatial correlation
Menezes, Liska, Cirillo and Vivanco (2016)	• Logistic Regression estimated using boosting algorithm for binary outcome	• Modify to use in stepwise sparse boosting
Adewale, Dinu and Yasui (2010)	• The generic gradient boosting algorithm with a loss function • The likelihood optimization boosting algorithm	• Modify to use in stepwise sparse boosting

CHAPTER 3

BACKGROUND INFORMATION AND METHODOLOGY

Following the idea of Yue, Li and Cheng (2019), where they developed a two-step sparse boosting method to conduct variable selection and estimation simultaneously for varying coefficient model with longitudinal data and continuous response, we developed a *three*-step sparse boosting algorithm with *binary* response.

3.1 Background Information

In this section, background information is provided related to the longitudinal data, logistic regression and boosting techniques.

3.1.1 Related to Data

3.1.1.1 Features of Longitudinal Data with Binary Outcome

A longitudinal study is considered as an investigation where participant outcomes and possibly treatments or exposures are collected at multiple follow-up times. Longitudinal studies play a key role in epidemiology, clinical research and therapeutic evaluation. Many medical studies are longitudinal in nature because the purpose of using the same individuals is to observe any measurable change over a period of time, and binary response data commonly occur in such studies. Also, as a micro-level analysis, the same individuals can be compared over time. Therefore, it shows how changing properties of individuals fit into systemic change and enables clear recommendation for interventions to be made, since the timing of new occurrence can be correlated with recent change in subject exposure. In addition, the temporal order

of outcomes are observed. Also, within-subject changes, over time can be explicitly modeled along with the outcome's relationship with time-varying predictors. It is in-depth and comprehensive coverage of a wide range of variables, also individual specific effects and population heterogeneity can be analyzed.

Longitudinal studies have some challenges. First of all, longitudinal data is correlated. As mentioned, it consist of repeated measurements on each subject, and hence these repeated measurements are collected within subjects. Therefore, longitudinal data analysis requires special statistical techniques for valid analysis and inference which takes into account of the possibility of correlations between responses given by the same individual. If such correlation is ignored then inferences such as statistical tests or confidence intervals can be invalid. Due to these correlation, the distance between measurements on different subjects is usually expected to be greater than the distance between repeated measurements taken on the same subject. The amount of variation from subject-to-subject in the overall level of response and the magnitude of variation in the trend over time in the response can be useful in determining the types of correlated data regression models that would be appropriate. To understand the components of variation, correlation characterization is important.

The second challenge is obtaining the results in a longitudinal study taking a long time. Again because of this long time, the participation can decrease or mortality of the subjects can heighten over time, which can lead to risk of bias. If subjects that are followed to the planned end of study differ from subjects who discontinue follow-up then a naive analysis may provide summaries that are not representative of the original target population.

Thirdly, it is important to note that, although longitudinal study associate changes in exposure with changes in the outcome of interest, the direction of causality can be complicated, because there can be reciprocal influence between exposure and outcome.

Also, in many studies, this repeated measurements do not have to be at prespecified occasions. Instead, individuals may be measured at irregular intervals, with those having a history of poorer health outcomes being measured with somewhat greater frequency and regularity.

In addition, the data set can be unbalanced which means that the number of measurements are not equal for each individual subject, because some subjects can appear in the study for the first time after the first measurements are done, or some subjects can drop out before final measurements.

Another challenge can be stated for the binary outcome in a longitudinal study. When outcomes of a linear model are normally distributed, the 'marginal effects', or average differences for subpopulations defined by differing covariate values, are the same as 'subject specific effects' or expected differences for individual subjects under different covariate values. However, this not the case for binary outcome, where a nonlinear link function is needed to provide a realistic connection between a linear predictor and the mean of the observable variable (the probability of the outcome) (Neuhaus, 1992; Diggle et al., 1994).

In practice, full likelihood-based methods for fitting of marginal models for discrete longitudinal data have proven to be very challenging for the following reasons: "(i) it can be conceptually difficult to model higher-order associations in a flexible and interpretable manner that is consistent with the model for the marginal expectations, (ii) given a marginal model for the vector of repeated outcomes, the multinomial probabilities cannot, in general, be expressed in closed-form as a function of the model parameters, and (iii) the number of multinomial probabilities grows exponentially with the number of repeated measures" (Parzen et al., 2011). For example, the typical generalized linear mixed (logistic regression) model with normal random effects, which provide inferences about variability between respondents, does not provide a simple expression for the marginal odds ratios for a one unit increase in each covariate, given the other covariate values for a binary outcome. In general, a feature of random effects logistic regression models for longitudinal binary data is that the marginal functional form, when integrated over the distribution of the random effects, is no longer of logistic form. Because the marginal likelihood function cannot be evaluated in a closed form, there are computational problems in fitting a model to a binary outcome longitudinal data, along with greater complexity of parameter interpretation. These problems especially occur when there is minimal replication in occasions within individuals in longitudinal data. It is stated that, in a typical longitudinal epidemiological study, there are rarely as many as ten occasions of measurement.

3.1.1.2 Covariates on Genetics

Some examples of covariates in an analysis of longitudinal study in a clinical setting include:

- Demographic covariates such as age, sex, ethnicity,
- Renal impairment,
- Genetic makeup of drug metabolizing enzymes,
- Repetitive medication,
- Environmental factors (such as smoking),
- Specific diet,
- Age-specific penetrance function,
- The lung-function measurement,
- Gene flow, inbreeding, genetic drift,
- Personal treatment assignment,
- Well-established major genetic factors,
- Biometric traits such as weight, height, body mass index (BMI), percentage body fat (PBF, measured by bioimpedance) and waist-to-hip ratio (WHR), femoral neck bone mineral density (BMD), forearm BMD, lumbar spine BMD,
- Exposures from industrial, environmental or iatrogenic sources,
- Allelic heterogeneity: When two or more alleles of a single locus are independently associated with the same trait,
- Locus heterogeneity: When two or more DNA variations in distinct genetic loci are independently associated with the same trait,
- Phenocopy: The presence of a disease phenotype that has a non-genetic (random or environmental) basis,

- Trait heterogeneity: When a trait, or disease, has been defined with insufficient specificity such that it is actually two or more distinct underlying traits,
- Phenotypic variability: Variation in the degree, severity or age of onset of symptoms exhibited by persons who actually have the same trait or disease process,
- Gene–gene interaction: When two or more DNA variations interact either directly (DNA–DNA or DNA–mRNA interactions), to change transcription or translation levels, or indirectly by way of their protein products, to alter disease risk separate from their independent effects,
- Gene–environment interaction:When a DNA variation interacts with an environmental factor, such that their combined effect is distinct from their independent effects,
- Hematopoietic phenotypes (hemoglobin, HbA_{1c}, mean cell hemoglobin [MCH], MCH concentration, mean cell volume, number of platelets, packed cell volume, and red blood cell count),
- Immune-related phenotypes (Crohn disease, inflammatory bowel disease, ulcerative colitis, and rheumatoid arthritis),
- Metabolic phenotypes (age of menarche, fasting glucose, fasting insulin, high-density lipoprotein [HDL], HOMA-B, HOMA-IR, low-density lipoprotein [LDL], triglycerides [TG], type 2 diabetes, and total cholesterol [TC] levels),
- Neurological phenotypes (schizophrenia),
- Social phenotypes (college and educational attainment).

3.1.1.3 Handling Missing Data

In longitudinal data analysis, missing values arise whenever one or more of the sequences of measurements from subjects within the study are incomplete. This incompleteness can be due to measurements not being taken because of subject's illness, being lost or subjects dropping out of the study. The reason why the values are missing is essential in the sense that whether their being missing has any bearing on

the outcome. For example, consider a sequence of atypically low (or high) values on a particular measurement being drop out. Dropouts carry particular importance since they cannot provide any further outcome information. This can lead to biased estimate of regression parameters, which is the case that the probability of the missingness is associated with outcomes. If the missingness is not due to dropouts, then they are called intermittent missing values.

3.1.1.3.1 Approaches to Analysis with Missing Data

In the literature, there exist several statistical approaches that attempt to reduce bias due to missing data:

1. **Imputation** of missing data. Imputation means to fill missing data which require a model that links the missing data to the observed data. Last observation carried forward method is an example of imputation, which extrapolates the last observed measurement for the subject in question to the remainder of the study. This extrapolation can be carried out relative to an estimate of a time trend, rather than a constant level. This method results in conservative assessment of subjects showing improvement over the duration of the study.

2. **Modeling** of the missing data process. Dropouts can be modeled by selection models, pattern mixture models and random effects models. Selection models separate the log-likelihood into two components, one for the parameters of the measurement sub-model, the other for the parameters of the dropout sub-model. Pattern mixture model is an alternative factorization of the joint distribution used by the selection models.

   Random effect models formalize the intuitive idea that a subject's pattern of response in a study is likely to depend on many characteristics of that subject, including some which are not observable. These unobservable characteristics are included in the model as random effects.

3. **Weighting** the available data to account for missing data by the use of inverse probability or non-response weighting.

4. **Discarding** all incomplete sequences. This method has the potential to introduce bias if the missing value process and measurement process are related, since the complete cases cannot be assumed to be a random sample with respect to the distribution of the measurements. In other words, the incomplete data from the subjects who have missing measurements provide additional information about the properties of the underlying measurement process conditional on completion.

It is important to note that these methods can be applied to random missing data rather than informative missing data. Informative missing data can lead to bias which can not be corrected simply through modeling since the link between the probability of missingness to the unobserved data is not known. In this case some form of sensitivity analysis to characterize plausible estimates based on various missingness assumptions can be used.

3.1.2 Related to Models

3.1.2.1 Logistic Regression and Boosting

First of all, since our outcome is binary, we use a binary logistic regression model, which is a particular case of generalized linear models. Logistic Regression can be considered as a discriminative model, where only the conditional distribution of the binary output y given the inputs x and model parameters β is modeled. This conditional probability has a particular parametric form, which is a sigmoid function of a linear combinations of covariates. Let us suppose that the inputs are $\mathrm{x} \in \mathbb{R}^p$, then the logistic regression model for the conditional distribution over Y given x and β is:

$$p(y \mid \mathrm{x}, \beta) = Bernoulli(\sigma(\beta \cdot x)) \tag{3.1}$$

where $\sigma : \mathbb{R} \to (0, 1)$ is the sigmoid function given by $\sigma(t) = \frac{1}{1+e^{-t}}$. Since σ maps $\mathbb{R} \to (0, 1)$, hence it can be interpreted as a probability. Thus, in 3.1, Y is modeled as Bernoulli random variable with expectation $\sigma(\beta \cdot \mathrm{x})$. Hence, $\sigma(\beta^T x_i) = \mu_i$. As a result, $\sigma(\beta \cdot x)$ can be interpreted as estimating the probability that $Y = 1$.

Basically, logistic regression calculates the probability of a specific event, $p(x)$. A logistic transformation is needed to allow any value of independent variables in the model to have a corresponding outcome in the range of $(0, 1)$. This logistic transformation is called logit and applied as follows:

$$logit\left(p\left(x\right)\right) = log\left(\frac{p\left(x\right)}{1-p\left(x\right)}\right) = \beta_0 + \beta_1 x_1 + ... + \beta_p x_p \tag{3.2}$$

As long as it is greater than 1, the base of the logarithm function has a little importance, but the natural logarithm with base e is the one most often used. The ratio $p\left(x\right) / \left(1 - p\left(x\right)\right)$ is called the odds. Generally, the coefficients are estimated using maximum likelihood method. However, due to the nonlinear nature of the equations, there is no closed-form solution, and, hence, iterative approaches are used.

Then, following the idea of our base article, Yue, Li and Cheng (2019), we investigated the usage of logistic regression and boosting techniques, where Friedman, Hastie and Tibshirani (2000) developed LogitBoost based on the idea of a direct interpretation of boosting as a method for function estimation. They showed that boosting can be interpreted as a "stagewise, additive in the covariates approach". This stagewise component actually provided the variable selection methods where "(i) variables were included sequentially in a stepwise regression, (ii) the coefficients of variables already included receive no further adjustment".

Originally, boosting algorithms are considered as an ensemble methods, since the underlying idea of boosting is that it is possible to obtain a strong learner by combining weak learners. Strong and weak learner can be defined as follows. Assume that x is random sample from distribution D and $y = h^*(x)$ for some function h^*. Given that some class of functions H such that $h^* \in H$, an algorithm is said a strong learner if it can take enough observations and output $h \in H$ such that $P(|h^*(x) - h(x)| > \epsilon) < \delta$ $\forall \epsilon, \delta > 0$. On the other hand, a weak learner can output $h \in H$ for some specific constants $\epsilon_0, \delta_0 > 0$. Weak learners can be exemplified as a simple linear function, regression splines or random trees. It can be explained as, in boosting, each new model is a fit on a modified version of the original data set, in order to do that, it uses a weak learner. Here, the idea is to improve upon the predictions of the previous model, which is to include predictive power from multiple, overlapping regions of

the variable space and obtain a strong learner. The first type of boosting is adaptive boosting and it fits another model to the residuals of the previous model and improves it by doing so. It begins fitting the model by equal weights of each observation, then after evaluating the first model, the weights of difficult to classify observations are increased while the other ones are decreased. Hence, the second model built on the weighted data. The subsequent models help to classify observations that are not well classified by the previous model. Therefore, predictions of the final ensemble model are the weighted sum of the predictions made by the previous models.

AdaBoost was the firstly introduced boosting algorithm by Freund and Schapire (1997) which was used for classification problems and proven to produce very good prediction accuracy results. Later, Breiman (1999, 2001) showed that AdaBoost algorithm can be seen as a gradient descent algorithm in function space which was considered as break through observation and led to other context where boosting techniques can be used in different contexts besides classification, like regression, density estimation, survival analysis, etc.

Also, it was shown by Friedman, Hastie and Tibshirani (2000), AdaBoost implicitly minimizes exponential loss function, which is equivalent to a second-order binomial log-likelihood criterion. They considered Adaptive Boosting as a generalized additive model and then they applied the cost function of logistic regression. Hence, by this relationship, they proposed LogitBoost that directly optimizes the binomial log-likelihood. LogitBoost was stated as fitting an additive logistic regression models through a stagewise optimization of the binomial log-likelihood. They used an adaptive Newton method for LogitBoost model. Rather than minimizing error with respect to y, it minimized the weighted least squares error of the function with respect to log-likelihood error. Therefore, LogitBoost became a variant of generalization of AdaBoost to Gradient Boosting in order to handle a variety of loss functions, in this case, namely binomial log-likelihood loss. It focuses on the loss function rather than weak learners. Because, boosting is an iterative method, first we need an initialization step which consist of a starting point, a step size and the number of iterations. Letting $f(x)$ be a binary classification model, the algorithm is given as the following:

Algorithm 1 LogitBoost (two classes)

1. Start with weights $w_i = 1/N$ $i = 1, 2, \ldots, N$, $F(x) = 0$ and probability estimates $p(x_i) = 1/2$.

2. Repeat for $m = 1, 2, \ldots, M$:

   (a) Compute the working response and weights

$$z_i = \frac{y_i^* - p(x_i)}{p(x_i)(1 - p(x_i))},$$
$$w_i = p(x_i)(1 - p(x_i)).$$

   (b) Fit the function $f_m(x)$ by a weighted least-squares regression of z_i to x_i using weights w_i.

   (c) Update $F(x) \leftarrow F(x) + \frac{1}{2} f_m(x)$ and $p(x) \leftarrow (e^{F(x)})/(e^{F(x)} + e^{-F(x)})$.

3. Output the classifier $sgn[F(x)] = sing[\sum_{m=1}^{M} f_m(x)]$

Here M can be a predefined number, it is stated that usually 1000 is enough, or it can be the number where convergence occurs, or it can be the number that minimizes the cross-validated error, or it can be set by shrinkage method using predetermined small positive number called learning rate. Now, moving to Gradient Boosting, which trains many models in a gradual, additive and sequential manner. The major difference between adaptive boosting and gradient boosting is the former uses high weight data points while the latter uses gradient in the loss function. The loss function is a measure indicating how good model's coefficients are fitting the underlying data. Therefore, it allows one to optimize a user specified loss function. The loss functions can be categorized as follows; squared error, absolute error, Huber and Quantile loss functions for continuous outcome; binomial and exponential loss functions for categorical outcome, loss functions for survival models, loss functions for counts data, and customized loss functions for other types of outcome. Hence, the algorithm for gradient boosting is the following:

Algorithm 2 Gradient Boosting

1. Initialize the model with a constant value where γ is the step size:

$$F_0(x) = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, \gamma).$$

2. For $m = 1$ to M:

   - Compute the gradient of the loss function, i.e., pseudo-residuals,

$$r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)} \text{ for } i = 1, \ldots, n.$$

   - Fit a model $h_m(x)$ to pseudo-residuals calculated from the gradient of the loss function.
   - Compute step magnitude multiplier γ_m.
   - Update $F_m(x) = F_{m-1}(x) + \gamma_m h_m(x)$

Mason et al. (1999) defined a class of algorithms called AnyBoost, as "gradient descent algorithms for choosing linear combinations of elements of an inner product space so as to minimize some cost (loss) functional". The *margin* of (x, y) with respect to the classifier $\mathrm{sgn}(F(\mathrm{x}))$ was defined as $yF(\mathrm{x})$, where the voted combinations of classifiers was defined as follows:

$$F(\mathrm{x}) = \sum_{t=1}^{T} w_t f_t(\mathrm{x}) \tag{3.6}$$

Here $f_t : X \to \{\pm 1\}$ are base classifiers from some fixed class $\mathcal{F}$ and $w_t \in \mathbb{R}$ are the classifiers weights.

Given a set $\mathcal{S} = \{(\mathrm{x}_1, y_1), \cdots, (\mathrm{x}_m, y_m)\}$ of m labeled data generated according to $\mathcal{D}$, a voted combination of classifiers of the form described above so that $\mathcal{P}_{\mathcal{D}}(\mathrm{sgn}(F(\mathrm{x})) \neq y)$ is small is tried to be constructed. It is stated by Mason et al. (1999), "That is, the probability that F incorrectly classifies a random data is small. Since $\mathcal{D}$ is unknown and we are only given a training set $\mathcal{S}$, we take the approach of finding voted classifiers which minimize the sample average of some loss function of the margin. That

is, for a set $\mathcal{S}$, find F such that

$$C(F) = \frac{1}{m}\sum_{i=1}^{m} C(y_i F(\mathrm{x}_i)) \tag{3.7}$$

is minimized for some suitable loss function $C : \mathbb{R} \to \mathbb{R}$ (the notation $\mathcal{C}$ is used for loss function not to confuse with likelihood function notation).".

One way to produce a weighted combination of classifiers 3.7, which was proposed by Mason et al. (1999) was by gradient descent in function space, at each iteration a base classifier was chosen to be included in the combination so as to maximally reduce the loss function. The idea of performing gradient descent in function space in this way was due to Breiman (1999). Mason et al. (1999) viewed the base classifier $f \in \mathcal{F}$ and their combinations F as elements of an inner product space $(\mathcal{X}, \langle , \rangle)$. In this case, they stated that, $\mathcal{X}$ was a linear space of functions that contained $lin(\mathcal{F})$, the set of all linear combinations of functions in $\mathcal{F}$, and the inner product was defined by for all $F, G \in lin(\mathcal{F})$:

$$\langle F, G\rangle := \frac{1}{m}\sum_{i=1}^{m} F(\mathrm{x}_i)G(\mathrm{x}_i) \tag{3.8}$$

However, this algorithm is valid for any inner product definition.

Now, suppose we have a function $F \in \mathcal{F}$ and a new $f \in \mathcal{F}$ is sought to add to F so that the cost $C(F + \epsilon f)$ decreases, for some small value of ϵ. In other words, the direction f in function space is sought such that $C(F + \epsilon f)$ most rapidly decreases. Then, they define

$$C(F + \epsilon f) = C(F) + \epsilon \langle \nabla C(F), f\rangle \tag{3.9}$$

and the greatest reduction in loss will occur for the f which maximizes $-\langle \nabla C(F), f\rangle$.

Then, the algorithm developed by them is given as Algorithm 3.

Algorithm 3 AnyBoost

Require:

■ An inner product space $(\mathcal{X}, \langle , \rangle)$ containing functions mapping from X to some set Y.

■ A class of base classifiers $\mathcal{F} \in \mathcal{X}$.

■ A differentiable loss function $C : lin(\mathcal{F}) \to \mathbb{R}$.

■ A weak learner $\mathcal{L}(F)$ that accepts $F \in lin(\mathcal{F})$ and returns $f \in \mathcal{F}$ with a large value of $-\langle \nabla C(F), f \rangle$.

Let $F_0(\mathrm{x}) \coloneqq 0$.

for $t \coloneqq 0$ to T **do**

  Let $f_{t+1} \coloneqq \mathcal{L}(F_t)$.

  if $-\langle \nabla C(F_t), f_{t+1} \rangle \leq 0$ **then**

    return F_t.

  end if

  Choose w_{t+1}.

  Let $F_{t+1} \coloneqq F_t + w_{t+1} f_{t+1}$

end for

return F_{T+1}

The algorithm terminates when $-\langle \nabla C(F_t) f_{t+1} \leq 0 \rangle$; when a base classifier f_{t+1} no longer points in the downhill direction of the loss function $C(F)$.

In addition, they propose a normalized version of AnyBoost that only returns functions in the convex hull of the base classifier class $\mathcal{F}$, since returning any arbitrary linear combination of elements of the base classifier class has the potential to cause overfitting. The difference is the algorithm require the differentiable loss functional $C : \mathrm{co}(\mathcal{F}) \to \mathbb{R}$ and the weak learner accepts $F \in \mathrm{co}(\mathcal{F})$. In addition, the stopping criterion shifts to $-\langle \nabla C(F_t) f_{t+1} - F_t \leq 0 \rangle$.

They also showed that AdaBoost and LogitBoost (which uses logistic loss) implicitly minimize some margin loss function by gradient descent, and hence they could be considered as special cases of AnyBoost. AdaBoost involves $C(yF(\mathrm{x})) = e^{-yF(\mathrm{x})}$ as the loss function, $f : X \to [-1, 1]$ as the base classifier and the optimal step-size is found via line search at each round. LogitBoost is AnyBoost with the loss

function $C(\alpha) = \log_2(1 + e^{-2\alpha})$ and step-size chosen via single Newton-Raphson step. Therefore, it can be deducted that the major variation among many boosting algorithms is the loss function.

The Sparse Boosting used by Yue et al. (2019) was developed by Bühlmann and Yu (2006) as the SparseL_2Boost algorithm; which utilized a model selection criterion to estimate out-sample prediction error by using a measure of complexity of boosting, which is presented as Algorithm 4. Yue et al. (2019) explained sparse boosting as "iteratively pursuing gradient descending in function space using penalized empirical risk function which integrate squared loss and complexity of boosting measure". Büehlmann and Yu (2006) stated that rather than minimizing the error loss, one may reduce the out-of-sample prediction error, where "This is not exactly achievable since the out-of sample prediction error is unknown. However, it can be estimated using a model selection criterion.". To do so, a measure of complexity of boosting is needed, which can be defined via boosting operator.

Then, the boosting operator can be defined as, $\mathscr{B}_m : \mathbb{R}^n \to \mathbb{R}^n$ by $\mathscr{B}_m Y = \widehat{F}_m$ where $\widehat{F}_m$ is the boosting estimate in iteration m. Hence,

$$\mathscr{B}_m = I - (I - \nu \mathscr{H}_{\widehat{s}_m}) \cdots (I - \nu \mathscr{H}_{\widehat{s}_1}) \tag{3.10}$$

where $\widehat{s}_m$ denotes the selector in iteration m and $\mathscr{H}_{\widehat{s}}$ is the hat operator given by the matrix:

$$\mathscr{H}_{\widehat{s}} = \mathbf{X}^{(\widehat{s})} \left(\mathbf{X}^{(\widehat{s})}\right)^T, \ \mathbf{X}^{(j)} = \left(X_1^{(j)}, \ldots, X_n^{(j)}\right)^T. \tag{3.11}$$

Then, the degrees of freedom for boosting is the trace of boosting operator, $trace(\mathscr{B}_m)$. In the algorithm, it is stated that "Since the selector $\tilde{\mathcal{S}}_m$ depends not only on the current residuals, but also explicitly on all previous boosting iterations through $\tilde{\mathcal{S}}_1, \ldots, \tilde{\mathcal{S}}_{m-1}$ via the trace of $\mathscr{B}_m(\mathcal{S})$, the estimate of base procedure is not a function of the current residuals only. This implies that boosting cannot be represented as a linear combination of base procedures, each of them acting on residuals only.".

Algorithm 4 SparseL_2Boost

Step 1 (initialization). $\hat{F}_0 \equiv 0$ and set $m = 0$.

Step 2. Increase m by 1.

Search for the best selector

$$\tilde{\mathcal{S}}_m = \underset{\mathcal{S}\in\Gamma}{\arg\min}\, T(\mathbf{Y}, trace(\mathscr{B}_m(\mathcal{S}))),$$
$$\mathcal{S} = I - (I - \mathscr{H}_{\mathcal{S}})(I - \upsilon\mathscr{H}_{\tilde{\mathcal{S}}_{m-1}})\cdots(I - \upsilon\mathscr{H}_{\tilde{\mathcal{S}}_1}),$$
(for $m = 1$: $\mathcal{B}_1(\mathcal{S}) = \mathcal{H}_{\mathcal{S}}$)

Fit the residuals $U_i = Y_i - \hat{F}_{m-1}(X_i)$ $(i = 1, \cdots, n)$ with the base procedure using the selected $\tilde{\mathcal{S}}_m$ which yields a function estimate

$$\hat{f}_m(.) = \hat{g}_{\tilde{\mathcal{S}}_m;(\mathbf{X},\mathbf{U})}(.)$$

where $\hat{g}_{\mathcal{S};(\mathbf{X},\mathbf{U})}(.)$ corresponds to the hat operator $\mathcal{H}_{\mathcal{S}}$ from the base procedure.

Step 3 (update). Update,

$$\hat{F}_m(.) = \hat{F}_{m-1}(.) + \upsilon\hat{f}_m(.)$$

Step 4 (iteration). Repeat Steps 2 and 3 for a large number of iterations M.

Step 5 (stopping). Estimate the stopping iteration by

$$\hat{m} = \underset{1\le m\le M}{\arg\min}\, T(\mathbf{Y}, trace(\mathscr{B}_m)),\ \mathscr{B}_m = (I - \upsilon\mathscr{H}_{\tilde{\mathcal{S}}_m})\cdots(I - \upsilon\mathscr{H}_{\tilde{\mathcal{S}}_1})$$

The final estimate is $\hat{F}_{\hat{m}}(.)$.

The selected $\tilde{\mathcal{S}}_m$ in SparseL_2Boost minimizes a model selection criterion over all possible selectors rather than $\tilde{\mathcal{S}}_m$ that minimized the residual sum of squares.

For the initialization of boosting algorithm, Bühlmann and Hothorn (2007) suggested that a parametric form of $\hat{f}^{[0]}(.)$ can be estimated by maximum likelihood.

Stopping Criteria for boosting iterations can be established by the use of information criteria like AIC which incorporates the degrees of freedom calculated based on the trace of Hat Matrix. Pursuing this way eliminates the need of some sort of cross-validation to calculate the stopping criteria.

3.1.2.2 Loss Functions

Assume that examples (x, y) are randomly generated according to some unknown probability distribution $\mathcal{D}$ on $X \times Y$ where X is the space of measurements, $X \subseteq \mathbb{R}^n$, and Y is the space of labels where $Y \in \{-1, 1\}$.

Then, the margin by which y is classified correctly is defined as:

$$\rho_f(\mathrm{x}, y) \coloneqq yf(\mathrm{x}) \tag{3.13}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a real valued function used for classification. Then, a negative value of $\rho_f(\mathrm{x}, y)$ corresponds to an incorrect classification. Having classifiers that achieve a large margin ρ_f is desirable, since it is expected that an estimate that is reliable on the training set will also perform well on unseen data.

The purpose of loss functions to penalize incorrect classification, in our case predicted probabilities. Given X as the inputs and $Y \in \{-1, 1\}$ as the outputs, a function is sought $f : X \to \mathbb{R}$ which best maps x to y. However, it is stated that, "because of incomplete information, noise in the measurement, or probabilistic components in the underlying process, it is possible for the same x to generate different y" (Lorenzo and Tomaso, 2014). Therefore, the expected risk can be minimized, defined as:

$$I[f] = \int_{X \mathrm{x} Y} \mathcal{C}(yf(x))p(x,y)\mathrm{d}x\mathrm{d}y \tag{3.14}$$

where $\mathcal{C}(yf(x))$ is the loss function and $p(x,y)$ is the probability density function of the process that generated the data.

Selection of a loss function affects the optimal $f_{\mathcal{C}}^{\star}$ where the expected risk is minimized.

In binary classification case which is considered in this thesis, the expected risk can be calculated as:

$$\begin{aligned}
I[f] &= \int_{X x Y} \mathcal{C}(yf(x))p(x,y)\mathrm{d}x\mathrm{d}y \\
&= \int_X \int_Y \mathcal{C}(yf(x))p(y \mid x)p(x)\mathrm{d}x\mathrm{d}y \qquad \text{since } p(x,y) = p(y \mid x)p(x) \\
&= \int_X \left[\mathcal{C}\left(f(x)\right)p(1 \mid x) + \mathcal{C}\left(-f(x)\right)p(-1 \mid x)\right]p(x)\mathrm{d}x \\
&\qquad \text{since -1 and 1 are the only possible values of } y \\
&= \int_X \left[\mathcal{C}\left(f(x)\right)p(1 \mid x) + \mathcal{C}\left(-f(x)\right)\left(1 - p(1 \mid x)\right)\right]p(x)\mathrm{d}x \\
&\qquad \text{since } p(-1 \mid x) = 1 - p(1 \mid x)
\end{aligned} \tag{3.15}$$

Hence, the minimizer of $I[f]$ can be found by taking the functional derivative with respect to f and equalize it to 0:

$$\frac{\partial \mathcal{C}(f)}{\partial f}\left(p(1 \mid x)\right) + \frac{\partial \mathcal{C}(-f)}{\partial f}\left(1 - p(1 \mid x)\right) = 0 \tag{3.16}$$

Some loss functions are visualized in Figure 3.1.

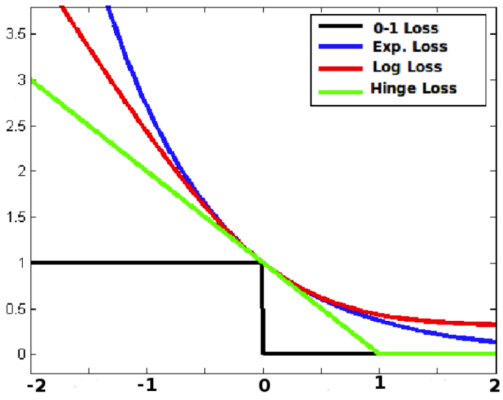


Figure 3.1: Loss Functions
(Copied from Bishop, 2006)

3.1.2.2.1 0-1 Loss

Given the binary nature of our context, a natural selection for a loss function (assuming equal cost for false positives and false negatives) would be the 0-1 loss, which takes the value of 0 if the predicted classification equals to the true class or a 1 if the predicted classification does not match the true class. Hence, the misclassification loss is:

$$\rho_{0-1}(y, f) = I_{\{\tilde{y}f \leq 0\}} \tag{3.17}$$

Using Bayes' Theorem, the optimal $f^{\star}_{0-1}$ for a binary classification problem is:

$$f^{\star}_{0-1}(x) = \begin{cases} 1 & \text{if } p(x) > 1/2 \\ -1 & \text{if } p(x) \leq 1/2 \end{cases}$$

where $p(x) = \mathbb{P}[Y = 1|X = x]$.

A loss function is said to be Bayes consistent if its optimal $f^{\star}_{\mathcal{C}}$ is such that $f^{\star}_{0/1}(x) = \text{sgn}(f^{\star}_{\mathcal{C}}(x))$ and is thus optimal under the Bayes decision rule. A Bayes consistent loss function allows us to find the Bayes optimal decision function $f^{\star}_{\mathcal{C}}$ by directly

minimizing the expected risk and without having to explicitly model the probability density functions. For convex loss $\mathcal{C}(yf(x))$, it is Bayes consistent if and only if it is differentiable at 0 and $\mathcal{C}'(0) = 0$.

However, this loss function is non-convex and non-smooth, and solving for the optimal solution is an NP-hard combinatorial optimization problem. As shown in Figure 3.1, the $0-1$ loss has two inflection points and it has infinite slope at 0.

3.1.2.2.2 Hinge Loss

Hinge Loss approximates 0-1 loss by $\min_\beta \sum_i H(\beta^T x)$ and provides an upper convex bound of the misclassification error. It is defined as $H(\beta^T x) = max(0, 1 - y \cdot f)$. Apparently H is small if we classify correctly. Hinge loss is used when $Y \in \{-1, 1\}$ and it encourages outputs to have a correct sign, assigning more error when there is a difference in the sign between actual and predicted values. Therefore, the Hinge function is:

$$\rho_{Hinge}(y, f) = [1 - \tilde{y}f]_+ \tag{3.18}$$

where $[x]_+ = xI_{\{x>0\}}$ denotes the positive part.

Its minimizer is:

$$f^\star_{Hinge}(x) = \text{sgn}(p(x) - 1/2) \tag{3.19}$$

which again corresponds to the Bayes classifier.

The Hinge function is the standard loss function in Support Vector Machines. Yet, it is noninvertable function of $p(x)$ and no direct way exist to obtain conditional probability estimates.

3.1.2.2.3 Exponential Loss

Another upper convex approximation of 0-1 loss is the exponential loss, expressed as;

$$\rho_{exp}(y, f) = exp(-yf) \tag{3.20}$$

which is convex and it is used in AdaBoost algorithm. The minimizer of the exponential loss function is:

$$f^{\star}_{exp} = \frac{1}{2} \log \frac{p(x)}{1 - p(x)} \tag{3.21}$$

where $p(x) = \mathbb{P}[Y = 1|X = x]$.

3.1.2.2.4 Least Squares Loss (L_2 Loss)

On the other hand, Least Squares Loss is expressed as:

$$\rho_{exp}(y, f) = \sum_i (y_i - f(x_i, \beta))^2 = \sum_i \left(y_i - \frac{1}{1 + e^{-x_i \beta}} \right)^2 \tag{3.22}$$

where the minimizer is:

$$f^{\star}_{L_2} = \mathbb{E}[Y|X = x] = p(x) = \mathbb{P}[Y = 1|X = x] \tag{3.23}$$

This loss function can be non-convex (due to logit transformation) and therefore, there is a possibility of finding local minima as the solution while minimizing the least squared loss. Then, gradient descent may not find the global minimum.

3.1.2.2.5 L_1 Loss

The loss function is:

$$\rho_{L_1}(y, f) = |y - f| \tag{3.24}$$

where the minimizer is the Bayes classifier:

$$f^{\star}_{L_1} = median(Y|X = x) = \begin{cases} 1 & \text{if } p(x) > 1/2 \\ -1 & \text{if } p(x) \leq 1/2 \end{cases}$$

L_1 loss has some robustness property, yet it is not differentiable at $y = f$.

3.1.2.2.6 Likelihood Loss

Next, the likelihood of observing the outputs y given the model parameters β and the inputs X as:

$$L(Y \mid X, \beta) = \prod_{i=1}^{n} \mu_i^{Y_i} \cdot \left(1 - \mu_i^{1-Y_i}\right) \tag{3.25}$$

It is more convenient to deal with the negative log-likelihood than the likelihood itself. The negative log-likelihood can be expressed as:

$$\begin{aligned} \text{Negative Log-Likelihood}(Y \mid X, \beta) &= -\sum_{i=1}^{n} \log\left(\mu_i^{Y_i}\left(1 - \mu_i^{1-Y_i}\right)\right) \\ &= -\sum_{i=1}^{n} (Y_i \log \mu_i + (1 - Y_i) \log(1 - \mu_i)) \end{aligned} \tag{3.26}$$

Then, by minimizing the negative log-likelihood, the parameters β can be estimated. The gradient of the negative log-likelihood are given below:

$$\nabla_{\beta}\text{Negative Log-Likelihood}(Y \mid X, \beta) = \sum_{i=1}^{n} X_i(\mu_i - Y_i) = X^T(\mu - Y) \tag{3.27}$$

3.1.2.2.7 Binary Cross-Entrophy Loss

In binary logistic regression, negative log-likelihood loss function leads to the usual loss function, which is the Binary Cross-Entropy Loss or Log-Loss:

$$BCE = -\frac{1}{n}\sum_{i=1}^{N} y_i \log(\widehat{y}_i) + (1-y_i)\log(1-\widehat{y}_i) \tag{3.28}$$

Cross-entropy calculates a score that summarizes the average difference between actual and predicted probability distributions for predicting $Y = 1$. This method is equivalent to using Newton's method only with the expected second derivative of log-likelihood, instead of its actual value. Hence, this loss function is equivalent to Iteratively Reweighted Least Squares Loss as stated in Charnes et al. (1976), "The method of iterative weighted least squares can be used to estimate the parameters in a nonlinear regression model. If the dependent variables are observations from a member of the regular exponential family, then under mild conditions it is shown that the IWLS estimates are identical to those obtained using the maximum likelihood principle". Also, the cross entropy loss is stated as "closely related to the Kullback–Leibler divergence between the empirical distribution and the predicted distribution".

Then, the logistic loss function is defined as:

$$\mathcal{C}(yf(x)) = \frac{1}{\log(2)}\log(1+e^{-yf(x)}) \tag{3.29}$$

The logistic loss is convex and grows linearly for negative values which make it less sensitive to outliers. The logistic loss is used in the LogitBoost algorithm.

The minimizer of $I[f]$ for the logistic loss function is:

$$f^{\star}_{Logistic} = \log\frac{p(1 \mid x)}{1-p(1 \mid x)} \tag{3.30}$$

In fact, the logistic loss and binary cross entropy loss are also the same up to a multiplicative constant $\frac{1}{\log(2)}$.

3.2 Proposed Model

The data has the following form:

$$[\{t_{ij}, X_i(t_{ij}), Y_i(t_{ij})\}, i = 1, \ldots, n, j = 1, \ldots, T_i] \qquad \text{for } n \text{ subjects}$$

where:

T_i denotes the total number of repeated measures for subject i,

$X_i(t_{ij}) = \{X_{i1}(t_{ij}), \ldots, X_{ip}(t_{ij})\}'$ denotes the vector of p covariates,

$Y_i(t_{ij})$ denotes the binary response variable measured at time t_{ij} for subject i,

i is the index for different individuals, where $i = 1, \ldots, n$,

j is the index for measurements of subject i taken at time $t_{ij} \in T_i$, where $j = 1, \ldots, T_i$.

For simplicity let us drop the subscript of ij from t_{ij}. Let $Y_i(t) \in \{0, 1\}$, and take the values one and zero with probabilities $p_i(t)$ and $1 - p_i(t)$, respectively. Since the probabilities $p_i(t)$ depend on a vector of observed covariates $X_i(t)$, we can let $p_i(t)$ be a linear function of covariates. Then, the closed form of our proposed model is:

$$\text{logit}\left(p_i\left(t\right)\right) = \log\left(\frac{p_i(t)}{1 - p_i(t)}\right) = \beta_0 + \sum_{d=1}^{p} X_{i,d}(t)^T \beta_d, \quad i = 1, \ldots, n \qquad (3.31)$$

where $\beta_0, \ldots, \beta_d$ are regression coefficients.

As mentioned, for the longitudinal data that will be used, sample size n is relatively small but the number of covariates p under consideration is extremely large. Since these repeated measurements are collected within subjects, all of the measurements will have the same between-subject errors and can be correlated within a subject. There is no assumption of balanced longitudinal study, i.e., t_{ij} does not need to be equivalent to t_{kj} for $i! = k$ and n_i does not need to be equivalent to a constant for all i. Although balanced data set was assumed by Yue et al. (2019), our proposed model runs on unbalanced data set, as well.

Next, the estimation procedure is explained as follows. In the first step, as Yue et al. (2019), the independence of observations is assumed, i.e., the within subject correlation is ignored and boosting algorithm is applied by minimizing an empirical risk

function, BCE, which introduce the initial deviance residuals. As model selection criterion AIC will be used. After obtaining the initial deviance residuals, the $T \times T$ weight matrix is derived based on them where T is the union of times that measurements are taken for both diseased and non-diseased individuals. Therefore, if there exist measurements for only one class, then it is calculated only using them, otherwise it will be calculated using both. In the second step, utilizing the weight matrix, the estimated temporal correlation structure within repeated measurements will be added to the empirical risk function.

Then, using the weight matrix, the coefficients are estimated again using boosting approach with the related empirical risk function. In the third step, spatial correlation is added by estimating the correlation structure within the data. After obtaining the coefficient estimates in the second step, the $n*T \times n*T$ covariance matrix $\mathrm{Cov}(Y_i) \equiv \Sigma$ is derived using the probability estimates calculated in the second step and used as a weight matrix in the third step for spatial correlation. As model selection criterion AIC will be used similar to the second step.

One reason we separately add the temporal correlation from the spatial correlation is that the weight matrix will be too large and computationally incomprehensible even for very small number of observations. The other reason is that, if the researcher wants to include the temporal correlation but ignore the spatial correlation or robustness property, s/he can stop after step two or assign 1 to weights on step 3.

Also, the reason why we use empirical risk function instead of loss function is that, we want the loss to be small for many observations in the data set. In other words, loss function measures how good our model is on a particular observation. Minimization of the loss function is:

$$f^* = \arg\min_f \frac{1}{n} \sum_{i=1}^{n} L\left(X_i, Y_i, f(X_i)\right) \tag{3.32}$$

Thus, when we let $n \to \infty$:

$$\frac{1}{n} \sum_{i=1}^{n} L\left(X_i, Y_i, f(X_i)\right) \xrightarrow{n\to\infty} \mathrm{E}_p\left[L\left(X_i, Y_i, f(X_i)\right)\right] \tag{3.33}$$

where p is the true distribution over the inputs and hence $\mathrm{E}_p\left[L\left(X_i, Y_i, f(X_i)\right)\right]$ is the true risk function. Since, we do not know the true distribution, then the true risk function can be approximated by the empirical risk function, $\mathrm{E}_p\left[L\left(X_i, Y_i, f(X_i)\right)\right] \approx \widehat{\mathrm{E}}\left[L\left(X_i, Y_i, f(X_i)\right)\right]$. Yet, for unbounded loss function, the assumption of the empirical risk function being a reasonable estimate of the true risk can be treated carefully.

Since there are not enough time measurements for smooth curve estimation, we do not consider varying coefficients, and continue with time-independent coefficients.

3.3 Three-step Sparse Boosting Techniques

Our base procedure is component wise binary cross-entrophy which finds the best set of covariates by reducing the AIC the most in every iteration. This selected variable is not necessarily different from the one selected in the previous iteration. Also, only the selected variable's coefficient is updated and a logistic model fit, including variable selection, is obtained in each iteration.

Our method involves the Hessian Matrix as hat matrix based on Newton's method instead of a step length for gradient, where B_m is an approximation hat matrix. Hat matrix is the matrix that maps the response variable to their fitted values. Degrees of freedom is used for information criteria in AIC as the penalty term. Our algorithm is presented below.

Algorithm 5 Three-Step Sparse Boosting Algorithm

Step I: Use sparse boosting to estimate covariance matrix

a. Initialization. Let $k_1 = 0$ and $\beta_0^{[k_1]} = 0, \ldots, \beta_p^{[k_1]} = 0$.

b. Increase k_1 by 1. Calculate $\widehat{s}_{k_1} = \arg\min_{0 \le d \le p-1} \left(\mathcal{C}_d(yF(x))^{[k_1]}\right)$ where $\mathcal{C}(yF(x))$ and is the chosen loss function or model selection criterion.

c. Update. $\beta_{\widehat{s}_{k_1}}^{[k_1]} = \beta_{\widehat{s}_{k_1}}^{[k_1-1]}$ for $d \neq \widehat{s}_{k_1}$ and $\beta_{\widehat{s}_{k_1}}^{[k_1]} = \beta_{\widehat{s}_{k_1}}^{[k_1-1]} + \nu\widehat{\lambda}_{\widehat{s}_{k_1}}^{[k_1]}$, where ν is the step-size parameter and $\widehat{\lambda}_d^{[k_1]}$ is the model which is fit by gradient descent method using the loss function or the model selection criterion.

d. Iteration. Repeat step (b)-(c) for some large iteration number K_1.

e. Stopping. The optimal iteration number can be taken as $\widehat{K_1} = \arg\min_{1 \le k_1 \le K_1} \left(\mathcal{C}_{\widehat{s}_{k_1}}(yF(x))^{[k_1]}\right)$.

Thus, $\beta^{[\widehat{K_1}]} = \left(\left(\beta_0^{[\widehat{K_1}]}\right)^T, \ldots, \left(\beta_p^{[\widehat{K_1}]}\right)^T\right)^T$ is the first step estimator for β from sparse boosting ignoring the within-subject correlation. $\mathrm{Cov}(Y_i) \equiv \Sigma$ can be estimated by the empirical estimator:

$$\widehat{\Sigma} = \frac{1}{n}\sum_{i=1}^{n} \widehat{\varepsilon}_i \widehat{\varepsilon}_i^T$$

where $\widehat{\varepsilon_i} = Y_i - \mathbf{X}_i \beta^{[\widehat{K_1}]}, i = 1, \ldots, n$.

Step II: Use sparse boosting again by incorporating covariance matrix estimator

a. Initialization. Let $k_2 = 0$ and $\beta^{\star[k_2]} = \beta^{[\widehat{K_1}]}$.

b. Increase k_2 by 1. Calculate $\widehat{s}_{k_2} = \arg\min_{0 \le d \le p-1} \left(\mathcal{C}_d^{\star}(yF(x))^{[k_2]}\right)$.

c. Update. $\beta_{\widehat{s}_{k_2}}^{\star[k_2]} = \beta_{\widehat{s}_{k_2}}^{\star[k_2-1]}$ for $d \neq \widehat{s}_{k_2}$ and $\beta_{\widehat{s}_{k_2}}^{\star[k_2]} = \beta_{\widehat{s}_{k_2}}^{\star[k_2-1]} + \nu \widehat{\lambda}_{\widehat{s}_{k_2}}^{\star[k_2]}$.

d. Iteration. Repeat step (b)-(c) for some large iteration number K_2.

e. Stopping. The optimal iteration number can be taken as $\widehat{K_2} = \arg\min_{1 \le k_2 \le K_2} \left(\mathcal{C}_{\widehat{s}_{k_2}}^{\star}(yF(x))^{[k_2]}\right)$.

Therefore, $\beta^{\star[\widehat{K_2}]} = \left(\left(\beta_0^{\star[\widehat{K_2}]}\right)^T, \ldots, \left(\beta_p^{\star[\widehat{K_2}]}\right)^T\right)^T$ is the second estimator for $\beta^{\star}$ by the three-step sparse boosting. Then update $\mathrm{Cov}(Y_i) \equiv \Sigma$ using the estimated probabilities.

Step III: Use sparse boosting again by incorporating covariance matrix estimator

a. Initialization. Let $k_3 = 0$ and $\beta^{\star[k_3]} = \beta^{[\widehat{K_2}]}$.

b. Increase k_3 by 1. Calculate $\widehat{s}_{k_3} = \arg\min_{0 \le d \le p-1} \left(\mathcal{C}_d^{\star}(yF(x))^{[k_3]}\right)$.

c. Update. $\beta_{\widehat{s}_{k_3}}^{\star[k_3]} = \beta_{\widehat{s}_{k_3}}^{\star[k_3-1]}$ for $d \neq \widehat{s}_{k_3}$ and $\beta_{\widehat{s}_{k_3}}^{\star[k_3]} = \beta_{\widehat{s}_{k_3}}^{\star[k_3-1]} + \nu \widehat{\lambda}_{\widehat{s}_{k_3}}^{\star[k_3]}$.

d. Iteration. Repeat step (b)-(c) for some large iteration number k_3.

e. Stopping. The optimal iteration number can be taken as $\widehat{k_3} = \arg\min_{1 \le k_3 \le K_3} \left(\mathcal{C}_{\widehat{s}_{k_3}}^{\star}(yF(x))^{[k_3]}\right)$.

The final estimate for Y is $\widehat{Y} = \mathbf{X}\beta^{\star[\widehat{K_3}]}$.

The selection of step size υ was stated to have a minor importance by Bühlmann and Yu (2006). A small step size factor can be seen as a shrinkage of the base procedure by the factor ν. This point was stated as "implying low variance but potentially large estimation bias" in Bühlmann and Hothorn (2007). Smaller υ may produce more

accurate estimation, yet it is also more computationally intensive. Yue et al. (2019) used $\upsilon = 0.1$ since it is commonly used in literature (Friedman, 2001), thus we also use the same step-size.

CHAPTER 4

RESULTS

Our main motivation and consideration is selecting significant variables while developing three-step sparse boosting. Yet, since it is a classification algorithm, results and performance measures are presented for both variable selection performance and classification point of view.

4.1 Simulation Study

We have conducted simulation studies to assess the performance of the proposed three-step sparse boosting algorithm. In the simulation study setup is constructed as follows:

- One of the covariates, $X_{ij,1}$, is generated using Binomial Distribution using 0.5 as success probability to represent binary covariate for example sex.
- Other covariates $X_{ij} = (X_{ij,2}, \ldots, X_{ij,p})$, $i = 2, \ldots, n$, $j = 1, \ldots, m$ are generated from multivariate normal distribution with mean 0.
- To specify the covariance matrix of the variables, Σ is calculated using the Kronecker product of the temporal correlation matrix and the spatial correlation matrix.
- We consider $n = 150$ for subject size and $m = 4$ for time points.
- As suggested by Cheng et al. (2014), Yue and Li (2017) and Xia et al. (2016), we keep $\lfloor \frac{n}{\log n} \rfloor$ variables as $p = 29$ as the total covariate number.

- Coefficients are defined ad $\beta_0 = 1.5$ (for the intercept), $\beta_1 = 1.6$ (for the binary covariate), and $\beta_2 = 1.8$, $\beta_3 = 1.3$, $\beta_4 = 1.7$, $\beta_5 = 1.4$ (for the normally distributed covariates).
- The maximum number of boosting iterations are limited to 500 for all the three steps.
- The number of Monte Carlo simulation is 100.

Since the main aim of our algorithm is to handle temporally and spatially correlated data, 9 different scenarios are developed using different values of ρ_{temp} and ρ_{spat} as temporal and spatial correlations, respectively. Each scenario takes around 160 hours to be completed for 100 run Monte Carlo Simulation by applying parallel computing methods in the R code using a computer with Intel(R) Core (TM) i7-7600U CPU @ 2.80 GHz, 16,0 GB RAM. R codes of our algorithm is presented in Appendix A.

As the results and performance measures of Monte Carlo Simulations Study of our variable selection algorithm, the following metrics are presented for each different correlation structure scenarios.

- The median of significant covariates that have been identified correctly (CS),
- The median of significant covariates that have been identified mistakenly (MS),
- Bias for the coefficient estimators (BIAS) which is computed as $\widehat{\beta}_l$-β_l,
- Absolute Bias (AB) which is calculated as $|\widehat{\beta}_l$-$\beta_l|$ to define "the distance between an estimator and its target parameter",
- Mean Squared Error of an estimate (MSE) which is defined as the expectation of its squared deviation from the truth and calculated as $\sum_1^p[(\widehat{\beta}_l$-$\beta_l)^2]$,
- Loss function outcomes as AICs for each iteration in each step,
- Confusion matrix at the end of each step and related statistics with accuracy information.

In the following subsections, first variable selection results are summarized with performance measures. Then, for each different correlation structure, SHAP values and

the results of loss functions are presented separately. After that, the classification performances of our algorithm are discussed. Finally, the comparison between our algorithm and Boruta (Random Forest), Support Vector Machine (SVM), Logistic Regression, Ridge Regression, Lasso Regression and Elastic Net algorithms are made considering both variable selection and classification performances.

After completing these studies, we also conducted a simulation study including only Step 1 and Step 3 consecutively, in order to have an idea of the contribution of Step 2. These results are presented in Appendix D.

4.1.1 Variable Selection Results

CS, MS, BIAS, AB and MSE's are presented in a unified manner in order to make the comparison easier. Other detailed metrics are presented under the related correlation structure sections. In Table 4.1 the performance of variable selection is summarized with correctly identified significant variables (CS), falsely identified significant variable (MS).

Table 4.1: Variable selection results in simulation for different correlation structures

Case			**p=29**					
			CS			MS		
	ρ_{temp}	ρ_{spat}	Step I	Step II	Step III	Step I	Step II	Step III
I	0.8	0.8	6	2	6	2	0	1
II	0.8	0.6	6	3	6	3	0	2
III	0.8	0.2	6	6	6	3	0	1
IV	0.6	0.8	6	2	6	2	0	1
V	0.5	0.5	6	3	6	3	0	1
VI	0.4	0.2	6	6	6	3	0	2
VII	0.2	0.8	6	2	6	2	0	1
VIII	0.2	0.4	6	5	6	3	0	1
IX	0.2	0.2	6	6	6	3	0	1

When we analyze the results of the above nine scenarios, the followings can be observed.

- After the first step which assumes the independence of covariates, the number of correctly identified significant covariates is the highest, yet the number of falsely identified significant covariates is high, too.
- At the end of the second step, although the falsely identified significant covariates become zero, correctly identified ones decrease as well.
- At the end of Step 3, the algorithm reaches the highest possible number of true significant covariates, and mistakenly identified ones are lower then the results of Step 1.
- As a summary, the algorithm reduces the mistakenly chosen significant covariates while establishing the true ones as significant.

In Tables 4.2- 4.7, the bias, absolute bias and MSE for each of the estimated coefficients are presented.

Table 4.2: Bias, Absolute Bias and MSE for $\widehat{\beta_0}$

Case			**p=29**								
			BIAS($\widehat{\beta_0}$)			AB($\widehat{\beta_0}$)			MSE($\widehat{\beta_0}$)		
	ρ_{temp}	ρ_{spat}	Step I	Step II	Step III	Step I	Step II	Step III	Step I	Step II	Step III
I	0.8	0.8	-0.006	1.408	1.380	0.207	1.408	1.380	0.065	2.075	2.018
II	0.8	0.6	0.016	1.251	1.425	0.203	1.254	1.425	0.066	1.791	2.183
III	0.8	0.2	0.058	0.563	1.745	0.183	0.611	1.745	0.052	0.731	3.192
IV	0.6	0.8	-0.016	1.441	1.364	0.220	1.441	1.364	0.073	2.137	1.958
V	0.5	0.5	0.036	1.234	1.405	0.184	1.254	1.405	0.053	1.781	2.089
VI	0.4	0.2	0.096	0.716	1.699	0.185	0.724	1.699	0.055	0.853	3.045
VII	0.2	0.8	0.060	1.452	1.415	0.204	1.452	1.415	0.070	2.155	2.083
VIII	0.2	0.4	0.111	0.969	1.629	0.202	0.972	1.629	0.064	1.240	2.796
IX	0.2	0.2	0.075	0.599	1.737	0.181	0.633	1.737	0.053	0.713	3.165

Table 4.3: Bias, Absolute Bias and MSE for $\widehat{\beta}_1$

Case			**p=29**								
			BIAS($\widehat{\beta}_1$)			AB($\widehat{\beta}_1$)			MSE($\widehat{\beta}_1$)		
	ρ_{temp}	ρ_{spat}	Step I	Step II	Step III	Step I	Step II	Step III	Step I	Step II	Step III
I	0.8	0.8	0.080	1.493	0.119	0.312	1.496	0.406	0.149	2.336	0.234
II	0.8	0.6	0.117	1.285	0.084	0.318	1.286	0.411	0.175	1.890	0.274
III	0.8	0.2	0.190	0.571	0.291	0.327	0.626	0.426	0.155	0.595	0.260
IV	0.6	0.8	0.091	1.497	0.182	0.326	1.499	0.374	0.178	2.356	0.246
V	0.5	0.5	0.028	1.280	0.021	0.258	1.281	0.370	0.111	1.881	0.207
VI	0.4	0.2	0.078	0.585	0.183	0.256	0.628	0.343	0.106	0.591	0.191
VII	0.2	0.8	0.009	1.479	0.074	0.275	1.479	0.369	0.120	2.304	0.243
VIII	0.2	0.4	0.083	0.798	0.097	0.231	0.804	0.320	0.089	0.912	0.150
IX	0.2	0.2	0.201	0.627	0.321	0.289	0.654	0.399	0.126	0.624	0.223

Table 4.4: Bias, Absolute Bias and MSE for $\widehat{\beta}_2$

Case			**p=29**								
			BIAS($\widehat{\beta}_2$)			AB($\widehat{\beta}_2$)			MSE($\widehat{\beta}_2$)		
	ρ_{temp}	ρ_{spat}	Step I	Step II	Step III	Step I	Step II	Step III	Step I	Step II	Step III
I	0.8	0.8	-0.018	1.490	-0.100	0.260	1.490	0.340	0.114	2.397	0.204
II	0.8	0.6	0.053	1.383	-0.024	0.207	1.384	0.280	0.066	2.178	0.125
III	0.8	0.2	0.120	0.632	0.239	0.171	0.638	0.364	0.045	0.682	0.177
IV	0.6	0.8	-0.037	1.529	-0.071	0.257	1.530	0.381	0.114	2.519	0.262
V	0.5	0.5	0.026	1.302	-0.077	0.180	1.306	0.282	0.048	1.985	0.127
VI	0.4	0.2	0.131	0.776	0.210	0.168	0.784	0.340	0.043	0.848	0.156
VII	0.2	0.8	-0.020	1.500	-0.114	0.250	1.500	0.363	0.104	2.407	0.246
VIII	0.2	0.4	0.095	0.954	0.061	0.163	0.954	0.282	0.040	1.118	0.108
IX	0.2	0.2	0.142	0.654	0.259	0.175	0.656	0.355	0.046	0.641	0.170

Table 4.5: Bias, Absolute Bias and MSE for $\widehat{\beta}_3$

Case			**p=29**								
			BIAS($\widehat{\beta}_3$)			AB($\widehat{\beta}_3$)			MSE($\widehat{\beta}_3$)		
	ρ_{temp}	ρ_{spat}	Step I	Step II	Step III	Step I	Step II	Step III	Step I	Step II	Step III
I	0.8	0.8	0.059	0.663	0.046	0.362	0.730	0.479	0.191	0.719	0.314
II	0.8	0.6	0.078	0.768	0.032	0.252	0.782	0.343	0.099	0.744	0.177
III	0.8	0.2	0.094	0.471	0.192	0.153	0.491	0.262	0.033	0.416	0.095
IV	0.6	0.8	0.141	0.709	0.138	0.350	0.761	0.424	0.186	0.715	0.263
V	0.5	0.5	0.039	0.847	-0.055	0.202	0.847	0.305	0.065	0.852	0.141
VI	0.4	0.2	0.081	0.545	0.135	0.139	0.547	0.254	0.029	0.456	0.092
VII	0.2	0.8	0.131	0.657	0.107	0.368	0.700	0.450	0.204	0.648	0.326
VIII	0.2	0.4	0.081	0.633	0.064	0.175	0.638	0.255	0.049	0.544	0.096
IX	0.2	0.2	0.088	0.452	0.174	0.138	0.454	0.262	0.027	0.332	0.095

Table 4.6: Bias, Absolute Bias and MSE for $\widehat{\beta}_4$

Case			**p=29**								
			BIAS($\widehat{\beta}_4$)			AB($\widehat{\beta}_4$)			MSE($\widehat{\beta}_4$)		
	ρ_{temp}	ρ_{spat}	Step I	Step II	Step III	Step I	Step II	Step III	Step I	Step II	Step III
I	0.8	0.8	0.059	0.974	-0.045	0.422	0.987	0.538	0.258	1.203	0.445
II	0.8	0.6	0.037	0.922	-0.066	0.228	0.927	0.371	0.077	1.056	0.197
III	0.8	0.2	0.127	0.556	0.245	0.185	0.581	0.357	0.050	0.522	0.178
IV	0.6	0.8	-0.050	0.866	-0.128	0.395	0.887	0.549	0.246	0.989	0.490
V	0.5	0.5	-0.018	0.932	-0.161	0.193	0.942	0.332	0.055	1.078	0.166
VI	0.4	0.2	0.096	0.656	0.180	0.151	0.662	0.329	0.036	0.628	0.148
VII	0.2	0.8	-0.017	0.879	-0.194	0.356	0.887	0.550	0.200	0.968	0.495
VIII	0.2	0.4	0.102	0.757	0.094	0.183	0.757	0.297	0.052	0.721	0.124
IX	0.2	0.2	0.104	0.588	0.213	0.148	0.588	0.345	0.035	0.505	0.155

Table 4.7: Bias, Absolute Bias and MSE for $\widehat{\beta_5}$

Case			**p=29**								
			BIAS($\widehat{\beta_5}$)			AB($\widehat{\beta_5}$)			MSE($\widehat{\beta_5}$)		
	ρ_{temp}	ρ_{spat}	Step I	Step II	Step III	Step I	Step II	Step III	Step I	Step II	Step III
I	0.8	0.8	0.012	1.204	-0.020	0.302	1.204	0.360	0.147	1.580	0.211
II	0.8	0.6	0.036	1.114	0.005	0.201	1.114	0.268	0.071	1.410	0.114
III	0.8	0.2	0.119	0.542	0.219	0.158	0.553	0.294	0.035	0.510	0.119
IV	0.6	0.8	0.087	1.258	0.093	0.314	1.266	0.381	0.158	1.681	0.235
V	0.5	0.5	0.058	1.098	-0.016	0.172	1.099	0.237	0.043	1.378	0.080
VI	0.4	0.2	0.076	0.631	0.114	0.148	0.636	0.287	0.033	0.570	0.115
VII	0.2	0.8	0.089	1.218	0.094	0.296	1.220	0.384	0.141	1.594	0.240
VIII	0.2	0.4	0.069	0.824	0.030	0.166	0.828	0.247	0.039	0.861	0.091
IX	0.2	0.2	0.114	0.542	0.189	0.174	0.550	0.301	0.041	0.455	0.123

While the algorithm reduced the number of mistakenly chosen significant covariates, the BIAS's, AB's and MSE's of coefficients get higher in Step 2 compared to Step 1. In Step 3, these metrics are reduced, yet remain higher than those in Step 1. That is because the algorithm is sparse, and it tends to decrease the coefficients in each step as aimed. In addition, at the end of Step 3, the number of mistakenly chosen significant covariates decreases while establishing the all the true ones as significant.

The detailed results for each different correlation structure are presented separately. Since similar results are obtained, the case of $\rho_{spat} = 0.8, \rho_{temp} = 0.8$ is presented in the main body of the Report, and the remaining cases are discussed in Appendix B.

The case of $\boldsymbol{\rho_{spat} = 0.8, \rho_{temp} = 0.8}$

As presented in Table 4.1, the results in this scenario is as follows:

Table 4.8: Variable selection results

Case			**p=29**								
			CS			MS			Size		
	ρ_{temp}	ρ_{spat}	Step I	Step II	Step III	Step I	Step II	Step III	Step I	Step II	Step III
I	0.8	0.8	6	2	6	2	0	1	8	2	7

In order to understand and explain the complexity of the model results and analyze the relationship between genes and the outcome, SHAP values have been calculated and presented. SHAP values are actually reframing Shapley Value Approach developed in game theory that try to "divide a prize so that everyone gets a fair share based on their different skill sets in a group of n players" (Shapley 1953). It was stated that, "the outcome of each possible combination (or coalition) of players should be considered to determine the importance of a single player". Štrumbelj and Kononenko (2010) and Štrumbelj and Kononenko (2014) firstly used Shapley Values in machine learning algorithms.

Hence, SHAP values answer the question of what is the influence of each input features to get that output, in other words the feature importance in a model is investigated by them. In addition, the impact of features are measured by taking into account the interaction with other features considering each possible combination of all features, since the order in which a feature enters can affect its predictions. Therefore, they are defined as the weighted average of marginal contribution of a feature value across all possible combinations. To calculate SHAP, all possible combinations of feature values have to be evaluated with and without the j-th feature. Consequently, "summing the SHAP values of each feature of a given observation yields the difference between the prediction of the model and the null model". Since, each predicted outcome is different based on the observations, SHAP values which are basically marginal contribution of the covariates to the outcome, are also different for each subject for each time measurement.

The contribution of final covariates that are identified as significant is illustrated using SHAP values as explained above. Figure 4.1 and Figure 4.2 represent SHAP values for the 1^{st} observation, whose observed response is 0, and for the 600^{th} observation (i.e. 150^{th} subject at time 4), for whom the observed response is 1, respectively. These graphs contain feature importance as the variables are ranked in descending order and the horizontal location indicates whether the effect of that feature is associated with a higher or lower prediction. The numbers presented in the graphs show the contributions of each individual covariates to that particular output. For example, 3^{rd} covariate has the most effect on the output of 0 and there is a negative relationship between this covariate and the output of zero. On the other hand, 6^{th} covariate has a

positive relationship with the output of 0. These contributions are vary between each time observation for each subject.

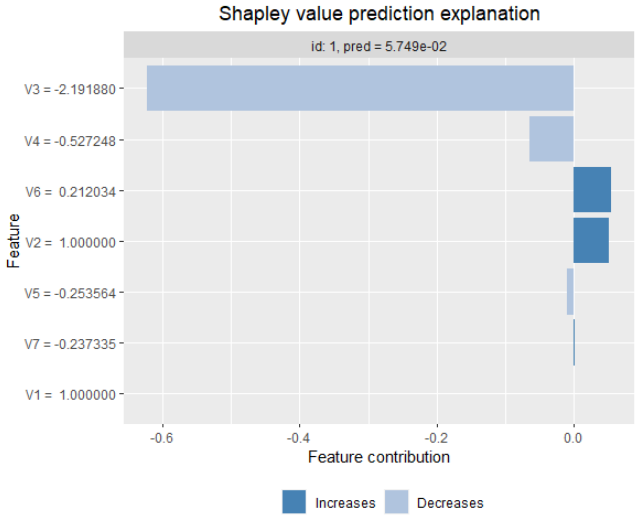


Figure 4.1: Contribution of Each Covariate for Class 0

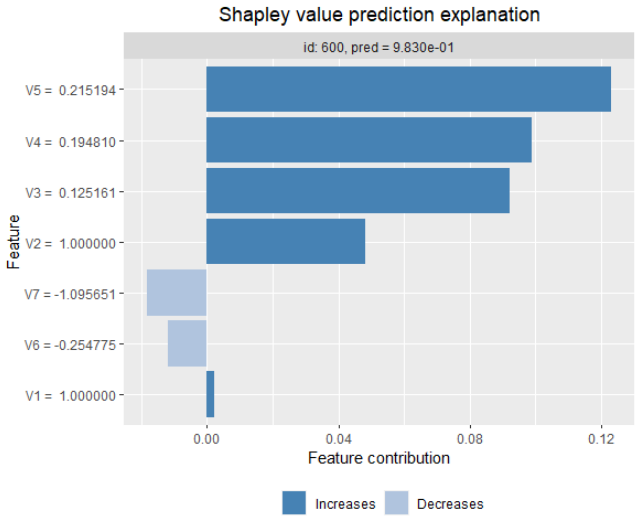


Figure 4.2: Contribution of Each Covariate for Class 1

The loss function results at the end of each iteration and in each step is presented as AIC values in Figure 4.3. It can be seen that AICs after each iteration and step gets smaller.

Also, when we analyze the results obtained using 500 as the maximum number of

boosting iterations, it is observed that, after 100^{th} iteration in each step, decrease in the loss function is minimal. Therefore, fixing the maximum number of boosting iterations to 100 would be enough to obtain reliable results.

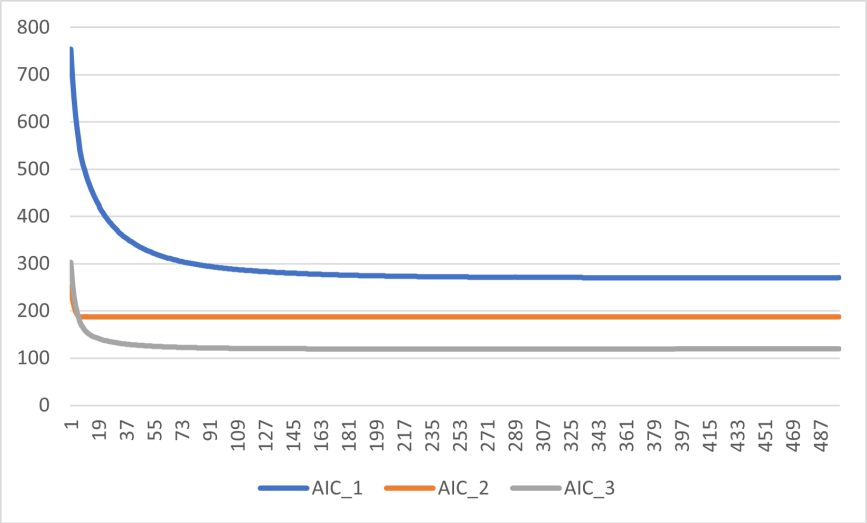


Figure 4.3: AIC's in Each Step and Iteration

4.1.2 Classification Results

In addition to the variable selection performance of our algorithm, we also investigated the classification performance of the algorithm. The following performance measures were considered based the confusion matrix identified as Figure 4.4, where TP stands for True Positive, FP stands for False Positive, FN stands for False Negative, TN stands for True Negative.

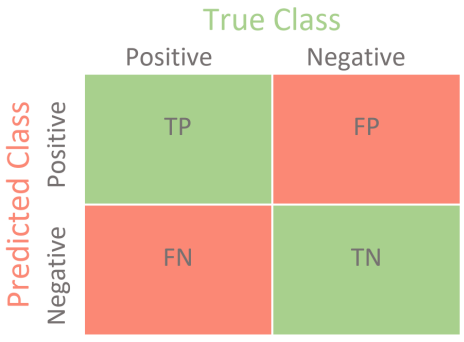


Figure 4.4: Confusion Matrix

- *Sensitivity* is the measurement to determine how much the prediction captures the actual positive class, where the positive class was defined as 1 in our study. Sensitivity is calculated as $\frac{TP}{TP+Fn}$.
- *Specificity* measures the ability to detect the negative class correctly, where the positive class was defined as 0 in our study. Specificity is calculated as $\frac{TN}{TN+FP}$.
- *Precision* is the measurement to determine how much of the positive prediction is actually positive case. Precision is calculated as $\frac{TP}{TP+FP}$.
- *F1 Score* is the weighted average score of sensitivity and precision.
- *Accuracy* is the measurement of how many cases the model is correctly identified. Accuracy is calculated as $\frac{TP+TN}{TP+TN+FP+FN}$.
- *The kappa statistic* defined as "a metric that compares an Observed Accuracy with an Expected Accuracy (random chance)". The higher value is expected as better for this metric.

In Tables 4.9 and 4.10 the performance measures calculated at the end of each steps are presented for $m = 4$ times measurements of $n = 150$ different subjects. The confusion matrices for all different scenarios are presented in Appendix C

Table 4.9: Classification metrics in simulation for different correlation structures

Metric	ρ_{temp}	ρ_{spat}	Step I	Step II	Step III
Sensitivity	0.8	0.8	0.922	0.754	0.813
	0.8	0.6	0.939	0.734	0.805
	0.8	0.2	0.942	0.901	0.783
	0.6	0.8	0.941	0.771	0.860
	0.5	0.5	0.912	0.897	0.749
	0.4	0.2	0.942	0.951	0.743
	0.2	0.8	0.945	0.762	0.822
	0.2	0.4	0.940	0.878	0.818
	0.2	0.2	0.919	0.916	0.667
Specificity	0.8	0.8	0.850	0.969	0.973
	0.8	0.6	0.783	0.920	0.931
	0.8	0.2	0.760	0.808	0.928
	0.6	0.8	0.870	0.959	0.948
	0.5	0.5	0.812	0.831	0.972
	0.4	0.2	0.706	0.686	0.967
	0.2	0.8	0.871	0.972	0.968
	0.2	0.4	0.780	0.852	0.923
	0.2	0.2	0.789	0.789	0.964
Precision	0.8	0.8	0.910	0.976	0.981
	0.8	0.6	0.913	0.957	0.966
	0.8	0.2	0.911	0.924	0.966
	0.6	0.8	0.939	0.975	0.972
	0.5	0.5	0.898	0.906	0.980
	0.4	0.2	0.903	0.899	0.985
	0.2	0.8	0.928	0.980	0.978
	0.2	0.4	0.908	0.931	0.961
	0.2	0.2	0.901	0.901	0.975

Table 4.10: Classification metrics in simulation for different correlation structures - cont

Metric	ρ_{temp}	ρ_{spat}	Step I	Step II	Step III
F1	0.8	0.8	0.916	0.851	0.889
	0.8	0.6	0.926	0.831	0.878
	0.8	0.2	0.926	0.912	0.865
	0.6	0.8	0.940	0.861	0.913
	0.5	0.5	0.905	0.901	0.849
	0.4	0.2	0.922	0.924	0.847
	0.2	0.8	0.937	0.858	0.894
	0.2	0.4	0.924	0.904	0.884
	0.2	0.2	0.910	0.908	0.792
Accuracy	0.8	0.8	0.895	0.835	0.873
	0.8	0.6	0.893	0.788	0.842
	0.8	0.2	0.892	0.875	0.823
	0.6	0.8	0.918	0.832	0.888
	0.5	0.5	0.877	0.873	0.828
	0.4	0.2	0.882	0.883	0.800
	0.2	0.8	0.918	0.838	0.875
	0.2	0.4	0.892	0.870	0.850
	0.2	0.2	0.877	0.875	0.763
Kappa	0.8	0.8	0.775	0.673	0.745
	0.8	0.6	0.737	0.561	0.658
	0.8	0.2	0.723	0.695	0.618
	0.6	0.8	0.813	0.654	0.760
	0.5	0.5	0.729	0.725	0.658
	0.4	0.2	0.675	0.675	0.574
	0.2	0.8	0.822	0.677	0.745
	0.2	0.4	0.738	0.703	0.676
	0.2	0.2	0.715	0.712	0.539

The following arguments can be made according to the above tables.

- Specificity and precision metrics obtain their highest value in Step 3 for 7 cases out of 9. For the 2 cases, it outperforms Step 1, and only around 0.5% lower than Step 2.
- For sensitivity, Step 3 results are on the average 16% lower than Step 1. The lowest difference occurs for the case $\rho_{spat} = 0.6, \rho_{temp} = 0.8$. The highest difference between Step 1 and Step 3 is realized in the case $\rho_{spat} = 0.2, \rho_{temp} = 0.2$ as expected, since Step 1 assumes temporal and spatial independence.
- When F1 Score is considered, Step 3 results are 6% lower than Step 1 results on the average. The lowest difference between Step 1 and Step 3 occurs for the case $\rho_{spat} = 0.8, \rho_{temp} = 0.8$ and $\rho_{spat} = 0.6, \rho_{temp} = 0.8$. The highest difference is realized in the case of $\rho_{spat} = 0.2, \rho_{temp} = 0.2$.
- In Step 3, accuracy results are again 6% lower than Step 1 results on the average. Again, the lowest difference between Step 1 and Step 3 occurs for the case of $\rho_{spat} = 0.8, \rho_{temp} = 0.8$ and the highest difference is realized in the case of $\rho_{spat} = 0.2, \rho_{temp} = 0.2$.
- For kappa metric, Step 3 results are again lower around 11% than Step 1, on the average. Again, the lowest difference between Step 1 and Step 3 occurs for the case of $\rho_{spat} = 0.8, \rho_{temp} = 0.8$ and the highest difference is realized in the case of $\rho_{spat} = 0.2, \rho_{temp} = 0.2$.
- In addition, it can be seen that the performance is getting higher when the spatial and temporal correlation get higher.
- It seems that, when the spatial correlation is high, the algorithm classification performances increase.

4.1.3 Comparison Results

The comparison between our algorithm and other algorithms that are used as variable selection in the Literature are presented in this subsection considering both variable

selection and classification performances. These algorithms are Boruta (Random Forest), Support Vector Machine (SVM), Logistic Regression, Ridge Regression, Lasso Regression and Elastic Net. Random Forest, SVM and Logistic Regression algorithms are well known and well used algorithms in the Literature. For the parameter tuning step, the number of decision trees used in Boruta algorithm are tuned using grid search. Although SVM are not generally used for variable classification, following Guyon et al. (2002), SVM is utilized based on Recursive Feature Elimination to select significant covariates. In addition, Ridge Regression, Lasso Regression and Elastic Net are chosen in the comparison study since they utilize a data driven penalty similar to AIC which is used as model selection criterion in our methodology.

4.1.3.1 Variable Selection Performance

The simulated data contains a total of 29 covariates and first 6 of them are significant. The number of covariates correctly selected as significant (CS), the number of covariates mistakenly selected as significant (MS), the number of overlapping covariates selected by the corresponding method and Three-Step Sparse Boosting as significant (Overlap) are given as some performance metrics for variable selection.

In Table 4.11, the variable selection results of the mentioned algorithms presented as an example for the case of $\rho_{spat} = 0.8, \rho_{temp} = 0.8$. Then, CS is calculated as the number of "X"s that corresponds to C1, C2, C3, C4, C5, C6 and MS is the number of remaining covariates selected as significant for each algorithm individually. Overlap is calculated as the mutual intersection of Three-Step Sparse Boosting algorithm and the related algorithm regarding the identification of significant covarites. Overlap metric does not consider whether the significant covariate identification is correct or not.

Table 4.11: Significant Covariates Identified by Algorithms - An example

Methods	Three-Step Sparse Boosting	Boruta (RF)	SVM	Logistic Regression	Ridge Regression	Lasso Regression	Elastic Net
C1	X						
C2	X	X	X	X	X	X	X
C3	X	X	X	X	X	X	X
C4	X	X	X	X	X	X	X
C5	X	X	X	X	X	X	X
C6	X	X	X	X	X	X	X
C7		X	X		X		X
C8		X	X		X	X	X
C9		X	X		X		X
C10		X	X		X		X
C11		X	X		X		X
C12		X	X		X		
C13		X	X		X		
C14		X	X		X	X	X
C15		X		X	X		X
C16		X	X		X		X
C17			X		X	X	X
C18			X		X		
C19			X		X		
C20					X		
C21			X		X		
C22			X		X	X	X
C23					X		
C24					X		
C25					X		X
C26					X		X
C27					X		
C28		X		X	X	X	X
C29				X	X	X	X

As mentioned above, Boruta (Random Forest), SVM, Logistic Regression, Ridge Regression, Lasso Regression and Elastic Net algorithms are compared with our Three-Step Sparse Boosting algorithm and some results are presented in Tables 4.12-4.14. The indicated figures for Three-Step Sparse Boosting algorithm are the final identified significant covariates as a result of Step 3 at the end of Monte Carlo Simulations.

Table 4.12: Comparison with Other Algorithms - Variable Selection

ρ_{temp}	ρ_{spat}	**Method**	**CS**	**MS**	**Overlap**
0.8	0.8	Three-Step Sparse Boosting	6	0	-
		Boruta (RF)	5	11	5
		SVM	5	14	5
		Logistic Regression	5	4	5
		Ridge Regression	5	23	5
		Lasso Regression	5	6	5
		Elastic Net	5	14	5
0.8	0.6	Three-Step Sparse Boosting	6	0	-
		Boruta (RF)	5	6	5
		SVM	5	14	5
		Logistic Regression	5	0	5
		Ridge Regression	5	23	5
		Lasso Regression	5	5	5
		Elastic Net	5	0	5
0.8	0.2	Three-Step Sparse Boosting	6	0	-
		Boruta (RF)	5	2	5
		SVM	5	1	5
		Logistic Regression	5	1	5
		Ridge Regression	5	23	5
		Lasso Regression	5	5	5
		Elastic Net	5	23	5
0.6	0.8	Three-Step Sparse Boosting	6	0	-
		Boruta (RF)	5	9	5
		SVM	5	21	5
		Logistic Regression	5	0	5
		Ridge Regression	5	23	5
		Lasso Regression	5	6	5
		Elastic Net	5	1	5

Table 4.13: Comparison with Other Algorithms - Variable Selection - cont

ρ_{temp}	ρ_{spat}	**Method**	**CS**	**MS**	**Overlap**
0.5	0.5	Three-Step Sparse Boosting	6	0	-
		Boruta (RF)	5	3	5
		SVM	5	7	5
		Logistic Regression	5	2	5
		Ridge Regression	5	23	5
		Lasso Regression	5	9	5
		Elastic Net	5	11	5
0.4	0.2	Three-Step Sparse Boosting	6	0	-
		Boruta (RF)	5	1	5
		SVM	5	2	5
		Logistic Regression	5	3	5
		Ridge Regression	5	23	5
		Lasso Regression	5	8	5
		Elastic Net	5	17	5
0.2	0.8	Three-Step Sparse Boosting	6	1	-
		Boruta (RF)	5	8	5
		SVM	5	18	5
		Logistic Regression	5	0	5
		Ridge Regression	5	23	5
		Lasso Regression	5	10	5
		Elastic Net	5	2	5
0.2	0.4	Three-Step Sparse Boosting	6	1	-
		Boruta (RF)	5	2	5
		SVM	5	1	5
		Logistic Regression	5	2	5
		Ridge Regression	5	23	5
		Lasso Regression	5	8	5
		Elastic Net	5	6	5

Table 4.14: Comparison with Other Algorithms - Variable Selection - cont

ρ_{temp}	ρ_{spat}	**Method**	**CS**	**MS**	**Overlap**
0.2	0.2	Three-Step Sparse Boosting	6	1	-
		Boruta (RF)	5	2	5
		SVM	5	4	5
		Logistic Regression	5	1	6
		Ridge Regression	5	23	6
		Lasso Regression	5	7	6
		Elastic Net	5	4	6

The following observations are done according to the results:

- Three-Step Sparse Boosting algorithm is the only algorithm that identifies all the significant covariates as significant for all 9 cases.
- Also, the intercept is identified as significant by only Three-Step Sparse Boosting algorithm.
- As the temporal and spatial correlations get lower, Three-Step Sparse Boosting algorithm identifies one covarite as mistakenly significant.
- The maximum number of mistakenly chosen covariates as significant is one in Three-Step Sparse Boosting algorithm, whereas this number can be large for other algorithms.
- Ridge Regression finds all 28 covariates as significant in all different correlation structures, except the intercept, whereas only 6 of them are significant.
- Generally, algorithms, except our algorithm, deduct very large number of mistakenly chosen significant covariates.
- Three-Step Sparse Boosting method produces the most sparse model.
- Also, algorithms', except ours, performance of identifying significant covariates decreases when the correlation of covariates get higher.

As a summary, Boruta (Random Forest), SVM, Logistic Regression, Ridge Regression, Lasso Regression and Elastic Net algorithms miss identifying intercept as significant in all scenarios, and generally, they consider very large number of mistakenly chosen significant covariates.

4.1.3.2 Classification Performance

The classification performance measures are calculated and presented in Tables 4.15 and 4.16.

Table 4.15: Comparison with Other Algorithms - Classification

ρ_{temp}	ρ_{spat}	**Method**	**Sensitivity**	**Specificity**	**Precision**	**F1 Score**	**Accuracy**	**Kappa**
0.8	0.8	Three-Step Sparse Boosting	0.836	**0.976**	**0.985**	0.904	0.885	0.763
		Boruta (RF)	0.905	0.869	0.933	0.919	0.893	0.763
		SVM	0.853	0.931	0.872	0.860	0.900	0.787
		Logistic Regression	0.941	0.895	0.943	**0.942**	**0.925**	**0.835**
		Ridge Regression	**0.956**	0.833	0.900	0.927	0.908	0.804
		Lasso Regression	0.947	0.845	0.910	0.928	0.908	0.802
		Elastic Net	0.920	0.904	0.951	0.936	0.915	0.811
0.8	0.6	Three-Step Sparse Boosting	0.824	**0.959**	**0.977**	0.894	0.868	0.724
		Boruta (RF)	0.889	0.852	0.936	0.912	0.878	0.716
		SVM	0.818	0.938	0.876	0.840	0.898	0.766
		Logistic Regression	0.917	0.853	0.931	0.924	0.897	0.764
		Ridge Regression	0.943	0.81	0.898	0.920	0.895	0.768
		Lasso Regression	**0.946**	0.830	0.911	**0.928**	**0.905**	**0.789**
		Elastic Net	0.889	0.898	0.958	0.922	0.892	0.744
0.8	0.2	Three-Step Sparse Boosting	0.707	**0.966**	**0.981**	0.821	0.782	0.559
		Boruta (RF)	0.839	0.795	0.941	0.887	0.830	0.547
		SVM	0.741	0.920	0.792	0.763	0.868	0.672
		Logistic Regression	0.905	0.825	0.934	0.919	0.883	0.710
		Ridge Regression	**0.937**	0.791	0.909	**0.923**	**0.892**	**0.743**
		Lasso Regression	0.927	0.762	0.894	0.910	0.875	0.704
		Elastic Net	0.897	0.826	0.937	0.916	0.878	0.695
0.6	0.8	Three-Step Sparse Boosting	0.851	**0.975**	**0.981**	0.912	0.900	0.798
		Boruta (RF)	0.913	0.880	0.923	0.918	0.900	0.790
		SVM	0.886	0.928	0.893	0.888	0.912	0.815
		Logistic Regression	0.936	0.895	0.931	0.934	**0.920**	**0.833**
		Ridge Regression	**0.967**	0.846	0.887	0.925	0.913	0.823
		Lasso Regression	0.953	0.850	0.893	0.922	0.908	0.811
		Elastic Net	0.918	0.924	0.953	**0.935**	**0.920**	0.831
0.5	0.5	Three-Step Sparse Boosting	0.833	**0.945**	**0.972**	0.897	0.867	0.711
		Boruta (RF)	0.865	0.844	0.948	0.904	0.860	0.646
		SVM	0.830	0.933	0.847	0.836	0.902	0.766
		Logistic Regression	0.925	0.871	0.948	0.936	0.910	0.783
		Ridge Regression	**0.943**	0.810	0.912	0.927	0.900	0.768
		Lasso Regression	**0.943**	0.806	0.909	0.926	0.898	0.764
		Elastic Net	0.924	0.897	0.959	**0.942**	**0.917**	**0.797**

Table 4.16: Comparison with Other Algorithms - Classification - cont

ρ_{temp}	ρ_{spat}	**Method**	**Sensitivity**	**Specificity**	**Precision**	**F1 Score**	**Accuracy**	**Kappa**
0.4	0.2	Three-Step Sparse Boosting	0.739	**0.967**	**0.981**	0.843	0.807	0.604
		Boruta (RF)	0.829	0.842	0.957	0.889	0.832	0.551
		SVM	0.805	0.933	0.843	0.819	0.895	0.745
		Logistic Regression	0.919	0.862	0.945	**0.932**	0.903	0.765
		Ridge Regression	0.935	0.835	0.929	**0.932**	**0.905**	**0.774**
		Lasso Regression	**0.940**	0.824	0.922	0.931	0.903	0.772
		Elastic Net	0.917	0.846	0.938	0.927	0.897	0.749
0.2	0.8	Three-Step Sparse Boosting	0.839	**0.957**	**0.973**	0.901	0.880	0.751
		Boruta (RF)	0.909	0.887	0.944	0.909	0.902	0.780
		SVM	0.871	0.941	0.894	0.880	0.917	0.816
		Logistic Regression	0.936	0.894	0.944	**0.940**	**0.922**	**0.827**
		Ridge Regression	**0.961**	0.827	0.895	0.927	0.908	0.804
		Lasso Regression	0.950	0.857	0.918	0.934	0.915	0.816
		Elastic Net	0.915	0.931	0.967	**0.940**	0.920	0.820
0.2	0.4	Three-Step Sparse Boosting	0.827	**0.945**	**0.972**	0.894	0.863	0.706
		Boruta (RF)	0.855	0.841	0.947	0.899	0.852	0.624
		SVM	0.804	0.935	0.847	0.821	0.895	0.747
		Logistic Regression	0.927	0.854	0.938	0.932	**0.905**	**0.774**
		Ridge Regression	0.949	0.795	0.899	0.924	0.897	0.764
		Lasso Regression	**0.95**	0.803	0.904	0.926	0.900	0.771
		Elastic Net	0.900	0.914	0.969	**0.933**	0.903	0.760
0.2	0.2	Three-Step Sparse Boosting	0.771	**0.951**	**0.977**	0.862	0.820	0.614
		Boruta (RF)	0.846	0.860	0.968	0.903	0.848	0.564
		SVM	0.779	0.940	0.835	0.802	0.897	0.733
		Logistic Regression	0.928	0.829	0.938	0.933	0.902	0.749
		Ridge Regression	0.939	0.774	0.909	0.923	0.890	0.729
		Lasso Regression	**0.941**	0.789	0.915	0.928	0.897	0.745
		Elastic Net	0.919	0.869	0.957	**0.937**	**0.907**	**0.756**

Sensitivity, specificity, precision and F1 Scores are also visualized in Figures 4.5-4.13 for different temporal and spatial correlation structures.

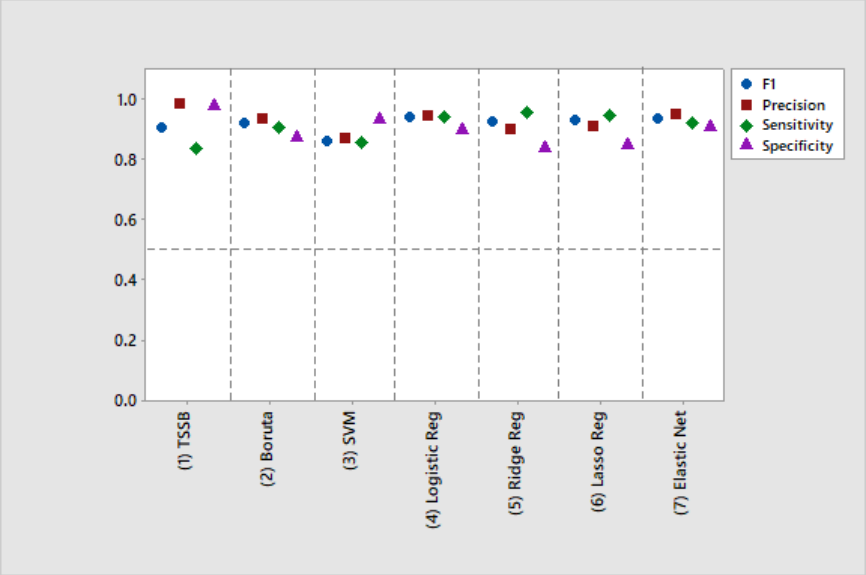


Figure 4.5: Model classification performance in simulated data - $\rho_{spat} = 0.8, \rho_{temp} = 0.8$

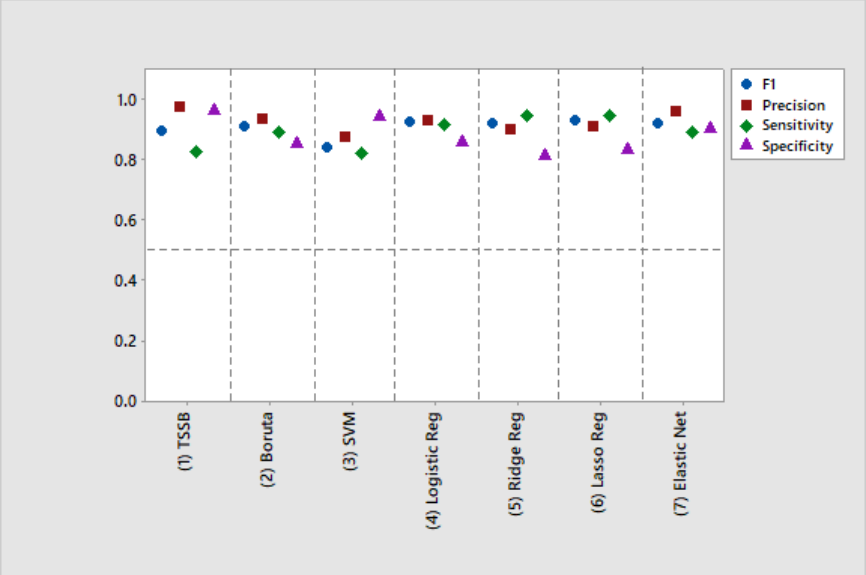


Figure 4.6: Model classification performance in simulated data - $\rho_{spat} = 0.8, \rho_{temp} = 0.6$

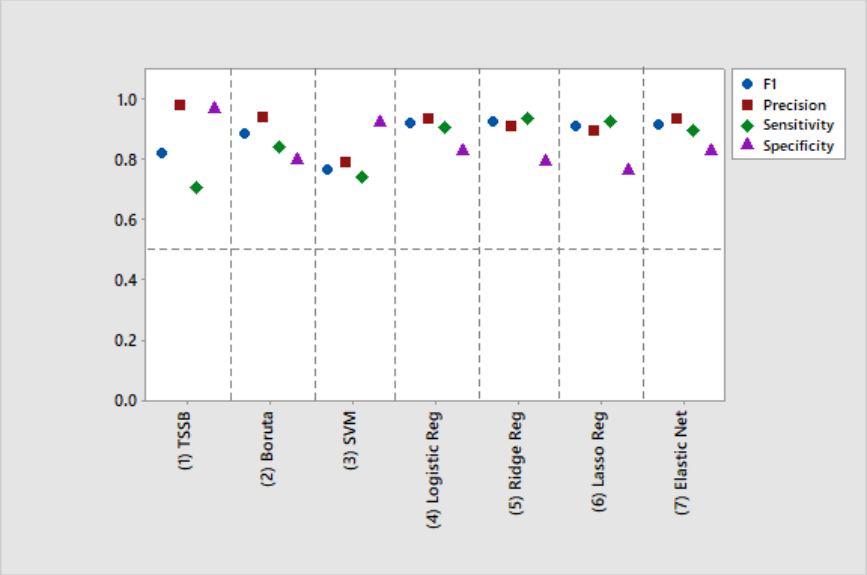


Figure 4.7: Model classification performance in simulated data - $\rho_{spat} = 0.8, \rho_{temp} = 0.2$

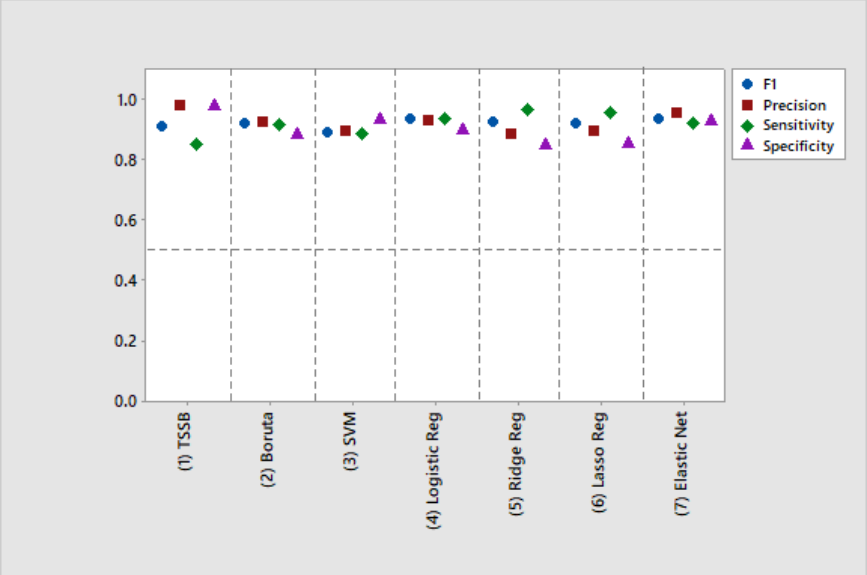


Figure 4.8: Model classification performance in simulated data - $\rho_{spat} = 0.6, \rho_{temp} = 0.8$

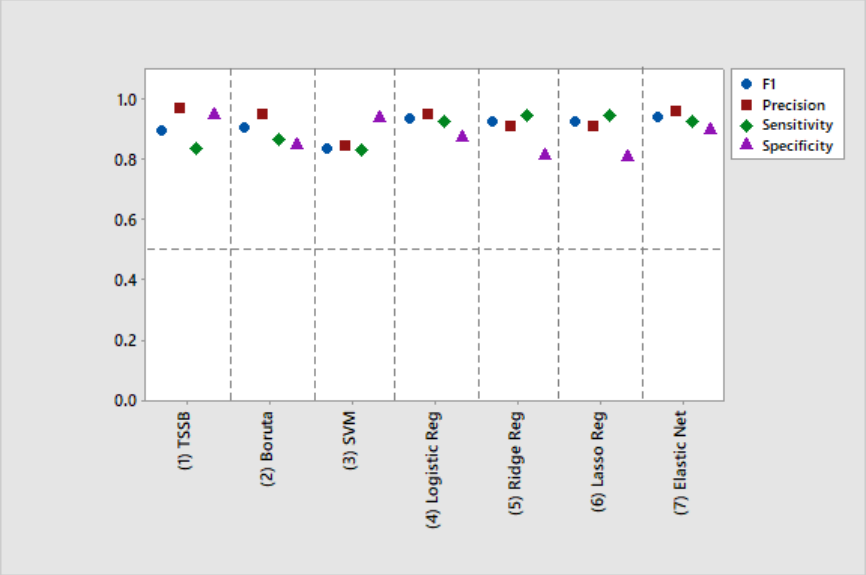


Figure 4.9: Model classification performance in simulated data - $\rho_{spat} = 0.5, \rho_{temp} = 0.5$

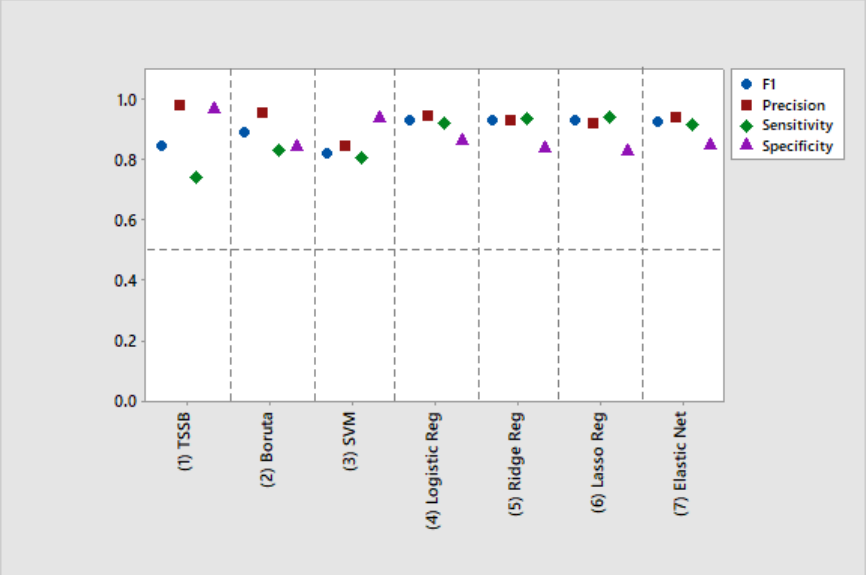


Figure 4.10: Model classification performance in simulated data - $\rho_{spat} = 0.4, \rho_{temp} = 0.2$

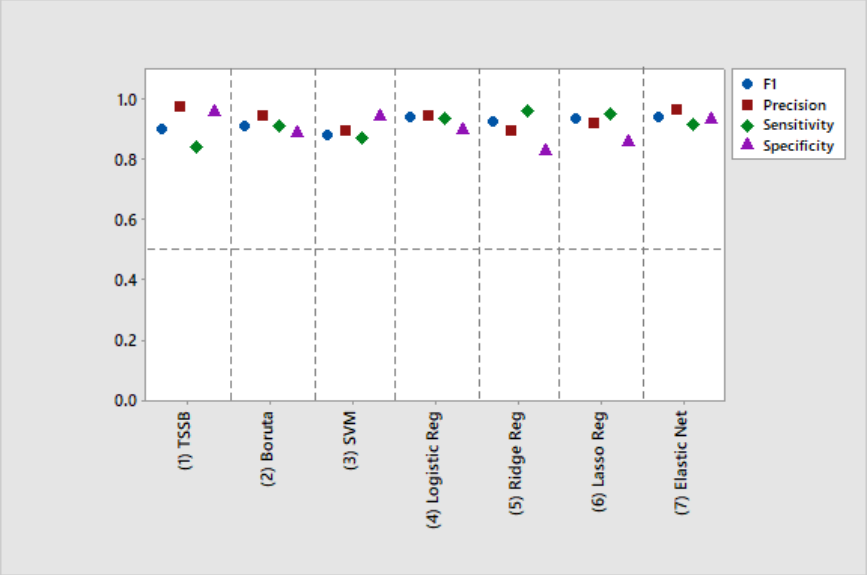


Figure 4.11: Model classification performance in simulated data - $\rho_{spat} = 0.2, \rho_{temp} = 0.8$

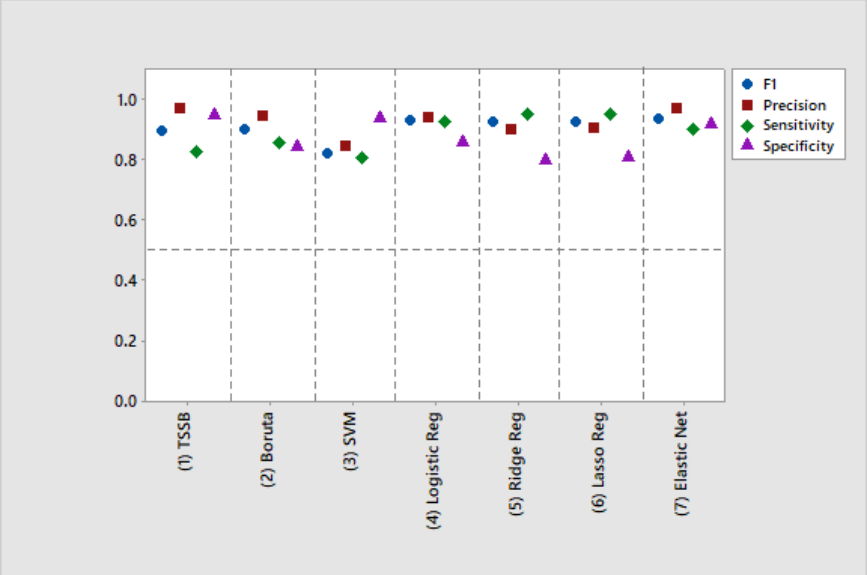


Figure 4.12: Model classification performance in simulated data - $\rho_{spat} = 0.2, \rho_{temp} = 0.4$

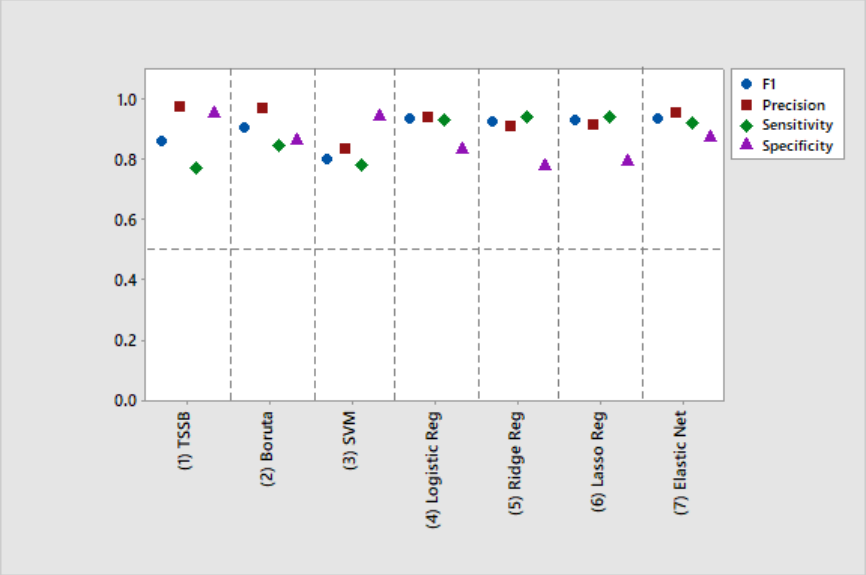


Figure 4.13: Model classification performance in simulated data - $\rho_{spat} = 0.2, \rho_{temp} = 0.2$

According to the above tables and figures:

- Specificity and precision have attained their highest value among all algorithms by Three-Step Sparse Boosting algorithm for all different correlation structures.
- For the other metrics, their highest value attained by different algorithms in each different correlation structure cases.
- Sensitivity metric for Three-Step Sparse Boosting algorithm decreases when the temporal and spatial correlation decrease, while this is not the case for the other algorithms.
- Accuracy measures can be considered as very close to each other for all algorithms in all cases.

Therefore, from classification point of view, the performance of Three-Step Sparse Boosting algorithm can be considered as very good and obtains comparative results with Boruta (RF), SVM, Logistic Regression, Ridge Regression, Lasso Regression and Elastic Net while identifying all the true significant covariates and at most one covariate as mistakenly significant.

4.2 Real Data Analysis

As the real data set, the Finnish Type 1 Diabetes Prediction and Prevention (DIPP) study is used, which can be downloaded from The National Center for Biotechnology Information web page (`https://www.ncbi.nlm.nih.gov/`) using Gene Expression Omnibus identifier GSE 30211. This study involved preprocessed mRNA expression levels of 49,386 probes corresponding to different genes that were measured over time on Affymetrix Human Genome U219 microarray from Homo Sapiens. Each probe was z-scored in the original data. The mRNA data consists of two separate measurement sets: samples before or at the time of the appearances of autoantibodies (seroconversion) and samples starting after seroconversion, in other words, seroconverted children and progressors. In addition, "a persistently autoantibody-negative control child was matched with each case, based on the date and place of birth, sex, and HLA-DQB1 genotype", which means that each seroconverted children and progressors matched with a different control subject.

In this study, the case and controls were labeled as diagnosed ($y = 1$) or not diagnosed ($y = 0$) with Type 1 diabetes.

4.2.1 Common Practices in Preprocessing of Microarray Data

In the preprocess stage, some data cleansing processes are implemented. For instance, it is common practice to take log_2 of the measurements in probe data. After this transformation, genes which have less than 3 as a measurement for all individuals can be eliminated, since these genes do not provide any useful information. Also, genes which have low variability over all individuals, both diseased and non-diseased individuals, can be ruled out. Usage of z-scores of the measurements in probe data in the algorithms can also be implemented, defined as $z = (x - m)/s$ where m is the mean, and s is the standard deviation calculated using all the time points in the matched control time series. In addition, the exclusion of probes with median expression lower than the median of median expressions is another common practice of preprocessing step of microarray data.

4.2.2 Balanced Data Set

At first, following Klén et al (2020) to obtain balanced data set, the measurements of seroconverted children were used including the first three follow-up samples. Probes with median expression lower than the median of median expressions (5.47) were excluded and the remaining 24,693 ones were run in the algorithm following Klén et al. (2020). Our goal is to determine the key probes that are statistically important for progression to Type 1 Diabetes.

We analyze two different data sets under balanced data set part, one covers all children's measurements for the first 3 follow-up times. The other case follows Klén et al. (2020) as only including the seroconverted children's first 3 follow-up measurements in order to make comparison.

Whole Data Set

Hence, firstly, as the inputs, 24,693 expressions of 56 children for 3 follow-up times were used as the balanced data set. Since the computing time is over one month for 24,693 probes, we use the independence screening approach to screen out the unimportant covariates as an initial step. This process followed the ranking method suggested in Klén et al. (2020) as a ranking procedure of the probes in two ways: 1) a ranking based on Wilcoxon's rank sum test P-value between cases and controls for all samples, and 2) a ranking based on Wilcoxon's rank sum test P-value between cases and controls for subjectwise median values. We tested the hypothesis of measurement of the subjects differs significant by time using nonparametric factorial ANOVA based on align-and-ranking for between x within subjects designs, and found that there is no statistically significant difference between measurements for different times. Hence, the two rankings could be combined by taking the average rank and as suggested by Cheng et al. (2014), Yue and Li (2017) and Xia et al. (2016), we keep the top $\lfloor \frac{n}{\log n} \rfloor$ variables to be used in the algorithm and the rest of the covariates are screened out. After the screening process 15 probes were kept. The maximum number of boosting iterations in three-step sparse boosting were all set to 500. The completion time was 13 minutes 23 seconds using a computer with Intel(R) Core (TM) i7-7600U CPU @ 2.80 GHz, 16,0 GB RAM. The number of significant probes selected by the end of

each step of our algorithm is presented in Table 4.17:

Table 4.17: Number of Significant Probes in Each Step - All Subjects

Step 1	**Step 2**	**Step 3**
2	2	2

The probes that are found significant are the same at each step and also presented in Table 4.18.

Table 4.18: Significant Probes in Each Step - All Subjects

Step 1	**Step 2**	**Step 3**
11729976_a_at	11729976_a_at	11729976_a_at
11762237_at	11762237_at	11762237_at

The association between the final 2 probes to genes can be established for only one probe using the mapping provided in the original data set:

Table 4.19: Genes Associated with the Significant Probes - All Subjects

Probes	**Genes**
11729976_a_at	PTPRN2
11762237_at	-

These genes' contributions to the final classification is illustrated using SHAP values in Figure 4.14 and Figure 4.15 for the 1^{st} observation, whose observed response is 0, and for the 165^{th} observation, for whom the observed response is 1, respectively. These graphs contain feature importance as the variables are ranked in descending order and the horizontal location indicates that whether the effect of that feature is associated with a higher or lower prediction. It can be seen that, for the not diagnosed class, both genes effects are negative, and the effect of PTPRN2 gene is higher. For the diagnosed class, PTPRN2 has negative but lower influence on the outcome,

whereas the other gene has a positive relationship with the outcome.

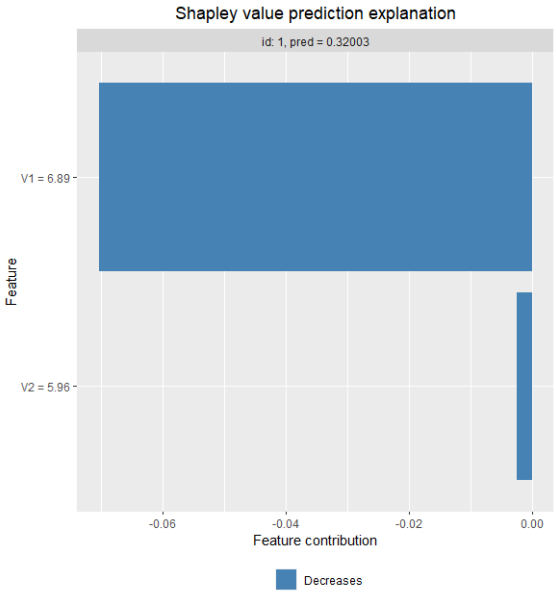


Figure 4.14: Contribution of Each Gene for Not Diagnosed Class - All Subjects

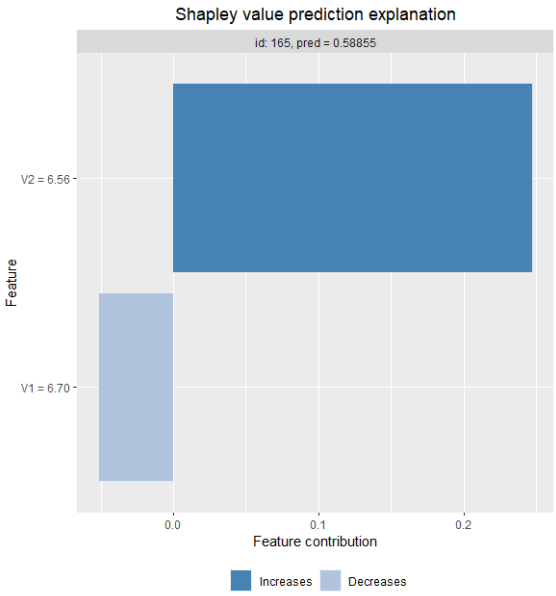


Figure 4.15: Contribution of Each Gene for Diagnosed Class - All Subjects

The loss function results at the end of each iteration and in each step are presented as AIC values in Figure 4.16. The loss in Step 2 is higher than the other two steps. AICs decrease by the iterations for early iterations, however stay almost constant after

approximately 100 iterations for all different data sets.

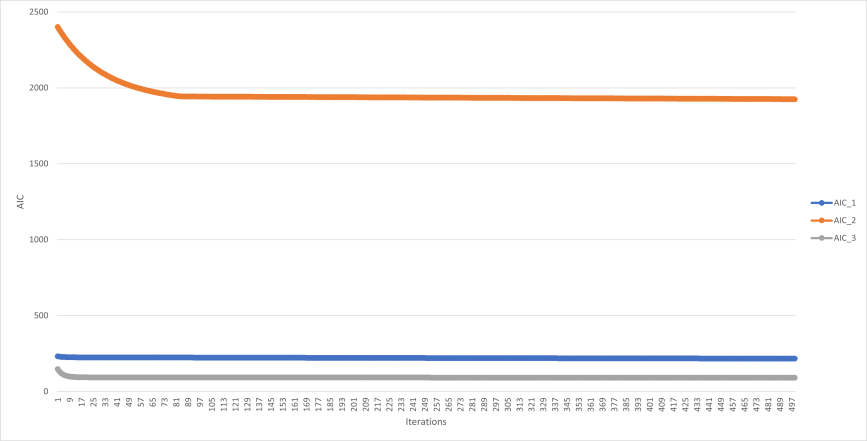


Figure 4.16: AIC's in Each Step and Iteration for the Whole Data Set

Now, when we investigate these genes that are presented in 4.19 about their relationship with Type 1 diabetes, the following information can be obtained. PTPRN2 gene is well established connection with Type 1 diabetes in the literature, obtaining 11.21[1] relevance score that is calculated based on matching documents in the literature. The relationship was studied in 40 articles and it was stated by Lee (2019) that "protein tyrosine phosphatase, receptor type N2 (PTPRN2) encodes a major islet autoantigen in type-1 diabetes". Unfortunately, since there exist no the mapping of the other significant probe, 11762237_at , to any gene, we cannot gather information regarding type 1 diabetes association.

In addition, performance metrics of classification results of our algorithm were calculated based on the following confusion matrices obtained at the end of each step for 56 children with 3 follow-up times.

[1] https://www.genecards.org/Search/Keyword?queryString=diabetes

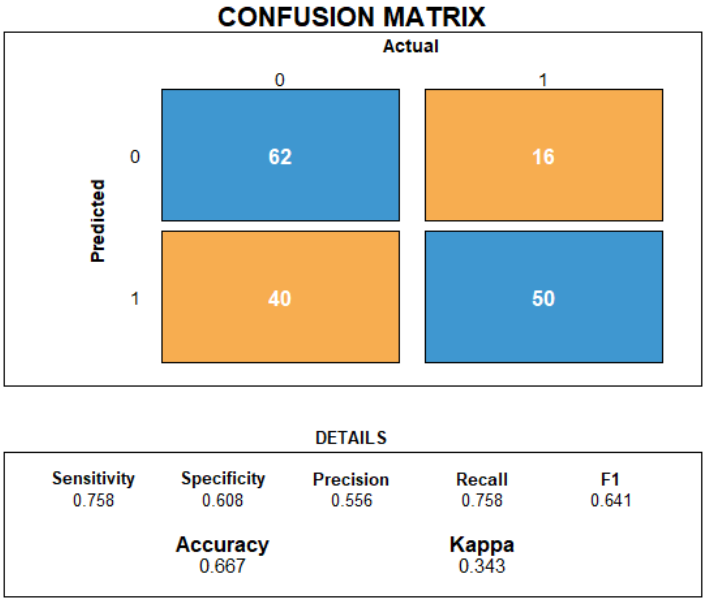


Figure 4.17: Confusion Matrix for The First Step - All Subjects

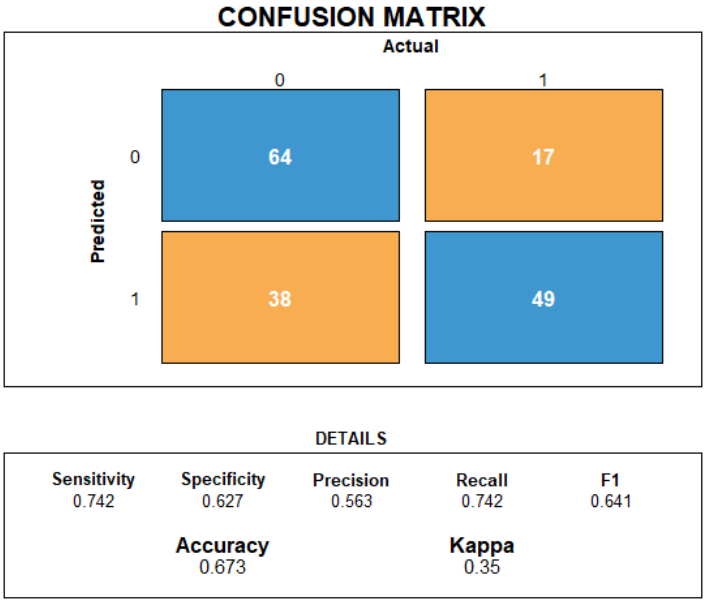


Figure 4.18: Confusion Matrix for The Second Step - All Subjects

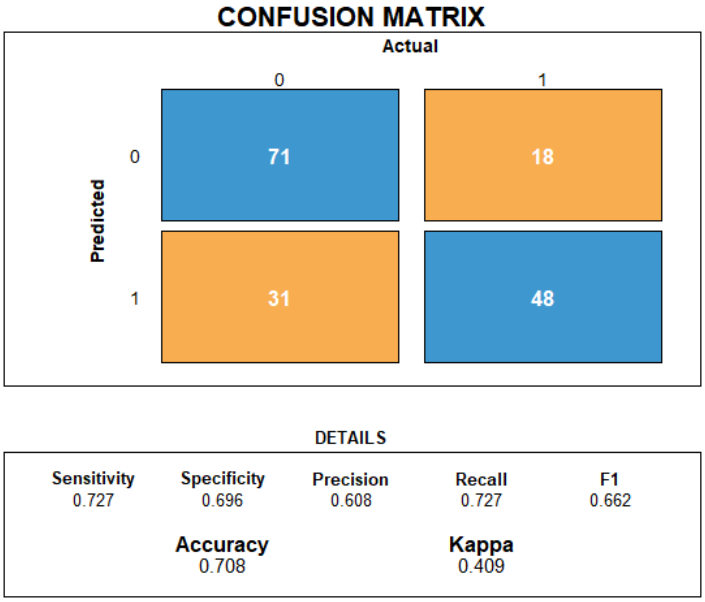


Figure 4.19: Confusion Matrix for The Third Step - All Subjects

From the classification results, the following arguments can be made:

- Only sensitivity measure decreases around 4% from Step 1 (76%) to Step 3 (73%).
- On the other hand, specificity, precision, F1 Score, accuracy and kappa increase in each step. The final results are 70% for specificity, 61% for precision, 66% for F1 Score, 71% for accuracy.

Comparison

In order to compare our algorithm with other ones that are used as variable selection algorithms in Literature, Boruta (Random Forest), SVM, Logistic Regression, Ridge Regression, Lasso Regression and Elastic Net algorithms are considered using the prescreened 15 probes (which eventually mapped to 14 different genes). For the parameter tuning step, the number of decision trees used in Boruta algorithm are tuned using grid search, again. The results are presented in Table 4.20, namely; number of genes selected (# of Significant Genes), the number of overlapping selected genes and Three-Step Sparse Boosting (Overlap) and F1 Score (F1) as some performance

metrics. Since cross validation is not applicable due to the low number of observations, out of sample performance measures cannot be calculated.

Table 4.20: Comparison with Other Algorithms - All Subjects

Method	**# of Significant Genes**	**Overlap**	**F1**
Three-Step Sparse Boosting	2	-	66%
Boruta (RF)	10	2	66%
SVM	5	1	81%
Logistic Regression	5	0	72%
Ridge Regression	14	2	44%
Lasso Regression	10	1	55%
Elastic Net	12	2	65%

The following figure illustrates the significant genes that are identified by the 7 considered algorithms and their overlapping structure. Because the 15 probes in our data set mapped to 14 different genes, the graph contains 14 rows. The red bars at the left side of each row lengthen with the more algorithm identifies the genes as significant.

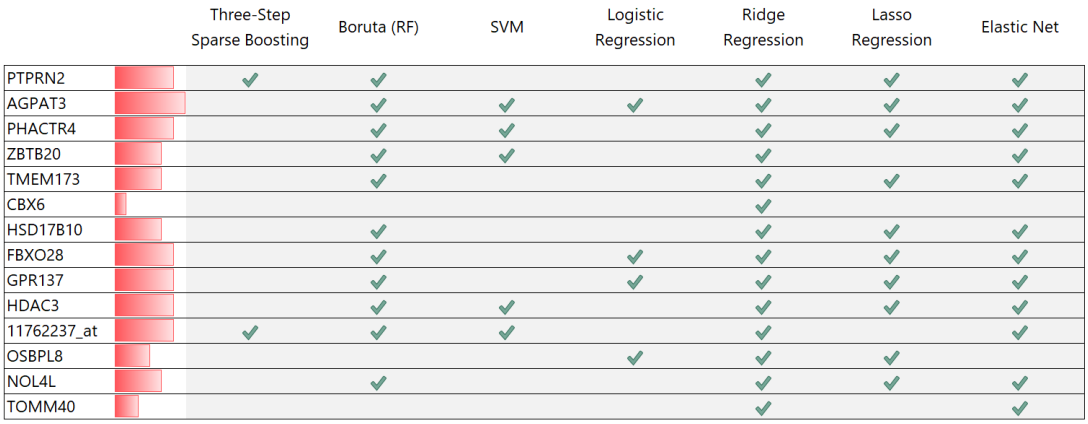


Figure 4.20: Significant Genes Identified - All Subjects

The following observations are done according to these results:

- SVM has the largest F1 score as 81%. However, it misses PTPRN2, which is found to have the highest relevance score with diabetes in the literature among

these 15 genes.

- Logistic Regression reaches 72% F1 Score by identifying 5 genes as significant, while Three-Step Sparse Boosting algorithm considers only 2 genes as significant and obtains 66% F1 Score. This might be beneficial for genetists, since Real-time PCR tests, which are used for post analysis, are expensive, and genetists usually prefer to justify a small number of significant genes.
- Ridge Regressions finds again all of the 14 genes as significant and has the lowest F1 Score.
- Lasso Regression identifies 10 genes as significant and only one overlaps with Three-Step Sparse Boosting algorithm.
- Elastic Net identifies 12 of 14 genes as significant and obtain 65% as F1 Score.
- AGPAT3 and HDAC3 genes have around 1 relevance score, yet Three-Step Sparse Boosting algorithm misses to identify, while the others do not.
- On the other hand, Three-Step Sparse Boosting algorithm do not identify PHACTR4 as significant which is found as so by all the other algorithms except logistic regression. Yet, these is no established relationship between this gene and diabetes in the literature.
- The similar argument can be made for GPR137 gene, as well, which is found significant by all algorithms except SVM and ours and does not constitute any known relationship with diabetes in the literature.

In summary, the three-step sparse boosting procedure can be considered as performing well in terms of variable selection and classification as eliminating the irrelevant genes better than other algorithms.

As an addition study, we run our algorithm, utilizing the firstly ranked 20 and 25 probes, instead of 15 probes. The results turned out to be the same for 20 and 25 probes data sets, and 2 probes were identified as significant. One of them also found as significant in 15 probes case. The results are presented as follows:

Table 4.21: Genes Associated with the Significant Probes - All Subjects - 20&25 Probes Case

Probes 15 Probes Case	**Genes 15 Probes Case**	**Probes 20&25 Probes Case**	**Genes 20&25 Probes Case**
11729976_a_at	PTPRN2	11719293_s_at	ITGB1
11762237_at	-	11762237_at	-

When we investigate the differently identified significant gene, ITGB1, T1D relationship is stated in the literature having 2.67 relevance score. For example, Lu et al. (2019) found that "The following two genes, ITGB1 and ITGB2 also have been reported to have association with T1D in gene-based pathway analysis and NOD/ltJ mice experiment, respectively".

Seroconverted Subjects

As a second data set, only seroconverted children are included, again following Klén et al. (2020). This data set involves 3 follow-up measurement of 20 children. For this setup, two different approaches were utilized. First, the independence screening approach is not used. This data set includes 24,693 gene expressions of 20 children for 3 follow-up times. Secondly, similar to the whole data set setup, the independence screening approach was used and top 15 probes were selected to be run in the algorithm.

In the first approach, all 24,693 gene expressions are included in the algorithm. The maximum number of boosting iterations in three-step sparse boosting were all set to 500, again. The completion time was around 35 days, using a computer with Intel(R) Core (TM) i7-7600U CPU @ 2.80 GHz, 16,0 GB RAM.

The number of significant probes selected by the end of each step of our algorithm is presented Table 4.22:

Table 4.22: Number of Significant Probes in Each Step - Seroconverted - No Prescreening

Step 1	Step 2	Step 3
6	6	2

The probes that are found significant are also presented in Table 4.23.

Table 4.23: Significant Probes in Each Step - Seroconverted - No Prescreening

Step 1	Step 2	Step 3
11722146_s_at	11722146_s_at	11722146_s_at
11736196_a_at	11736196_a_at	11736196_a_at
11726363_x_at	11726363_x_at	-
11726913_a_at	11726913_a_at	-
11733454_a_at	11733454_a_at	-
11719341_s_at	11722926_a_at	-

The association between the final 2 probes to genes using the mapping provided in the original data set is the following:

Table 4.24: Genes Associated with the Significant Probes

Probes	Genes
11722146_s_at	PAIP2B
11736196_a_at	DCX

Now, when we investigate these genes about their relationships with diabetes, DCX has 1.06 relevance score. In addition, PAIP1 which is related to PAIP2B, according to Derry et al. (2006), was stated as "mRNA and protein expression of PAIP1, a miRNA-340 target gene, was found down-regulated in GDM women, accordingly." by Stirm et al. (2018).

Secondly, the independence screening approach is used and top 15 probes are selected to be run in the algorithm. The maximum number of boosting iterations in three-step sparse boosting were all set to 500, again. The completion time was 11 minutes 23 seconds, using a computer with Intel(R) Core (TM) i7-7600U CPU @ 2.80 GHz, 16,0 GB RAM. The number of significant probes selected by the end of each step of our algorithm is presented in Table 4.25:

Table 4.25: Number of Significant Probes in Each Step - Seroconverted

Step 1	**Step 2**	**Step 3**
2	2	2

The probes that are found significant again are the same at each step and also presented in Table 4.26.

Table 4.26: Significant Probes in Each Step - Seroconverted

Step 1	**Step 2**	**Step 3**
11725093_a_at	11725093_a_at	11725093_a_at
11737802_a_at	11737802_a_at	11737802_a_at

The association between the final 2 probes to genes can be established for only one probe using the mapping provided in the original data set:

Table 4.27: Genes Associated with the Significant Probes - Seroconverted

Probes	**Genes**
11725093_a_at	RFX5
11737802_a_at	MAD1L1

Again,the contributions of these genes to the final classification is illustrated using SHAP values in Figure 4.21 and Figure 4.22 for the 1^{st} observation, whose observed response is 0, and for the 57^{th} observation, for whom the observed response is 1,

respectively. It can be seen that, for the not diagnosed class, RFX5 gene has positive influence on the outcome whereas MAD1L1 gene has a negative relationship and it has relatively less contribution. On the other hand, for the diagnosed class, the vice-versa situation applies and RFX5 gene has a positive impact and less contribution to the output, while MAD1L1 gene has positive effect and relatively more influence on the output.

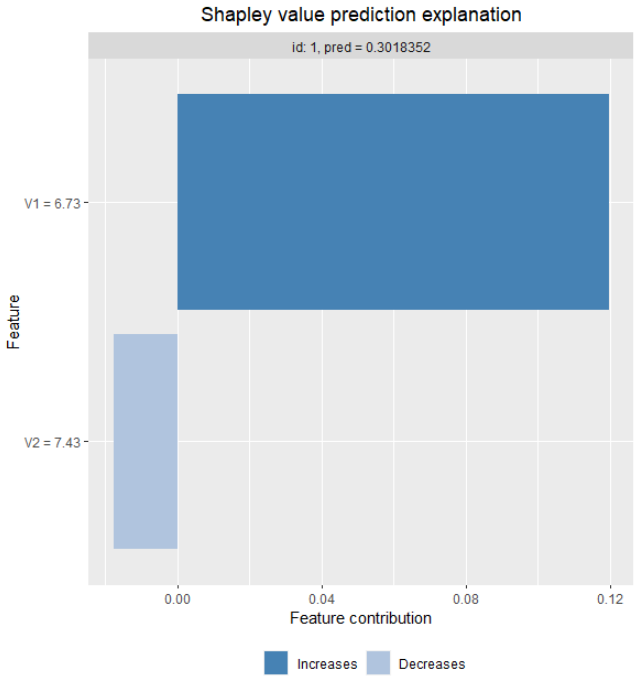


Figure 4.21: Contribution of Each Gene for Not Diagnosed Class - Seroconverted

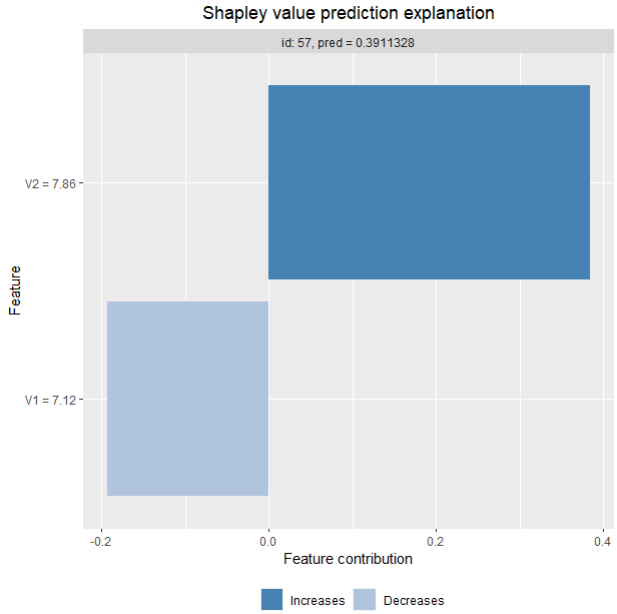


Figure 4.22: Contribution of Each Gene for Diagnosed Class - Seroconverted

The loss function results at the end of each iteration and in each step are presented as AIC values in Figure 4.23. The loss in Step 2 is higher than the other two steps, again.

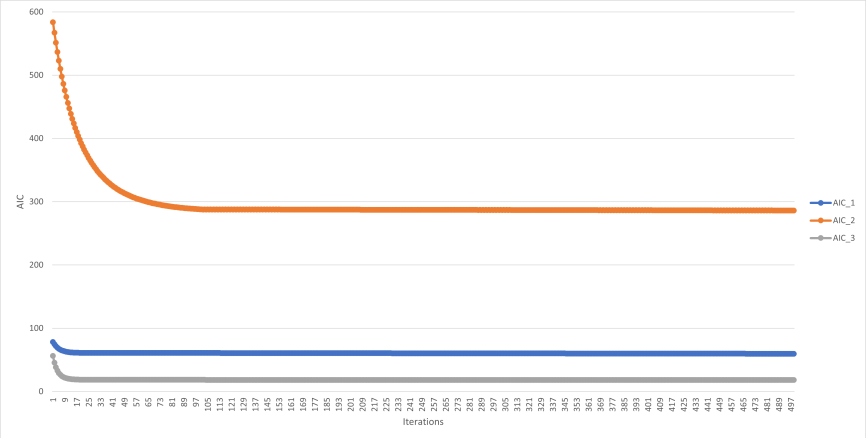


Figure 4.23: AIC's in Each Step and Iteration for Seroconverted Case

Again, when we investigate these genes that are presented in 4.27 with their relationships with Type 1 diabetes, for RFX5, it was stated that "Five regulatory factor X (RFX) transcription factors (TFs)–RFX1-5–have been previously characterized in the

human genome, which have been demonstrated to be critical for development and are associated with an expanding list of serious human disease conditions including major histocompatibility (MHC) class II deficiency and ciliaophathies." by Aftab et al. (2008). Yet, there exist no known connection between MAD1L1 gene with diabetes.

In addition, performance metrics of classification results of our algorithm were calculated based on the following confusion matrices obtained at the end of each step for 20 children with 3 follow-up times.

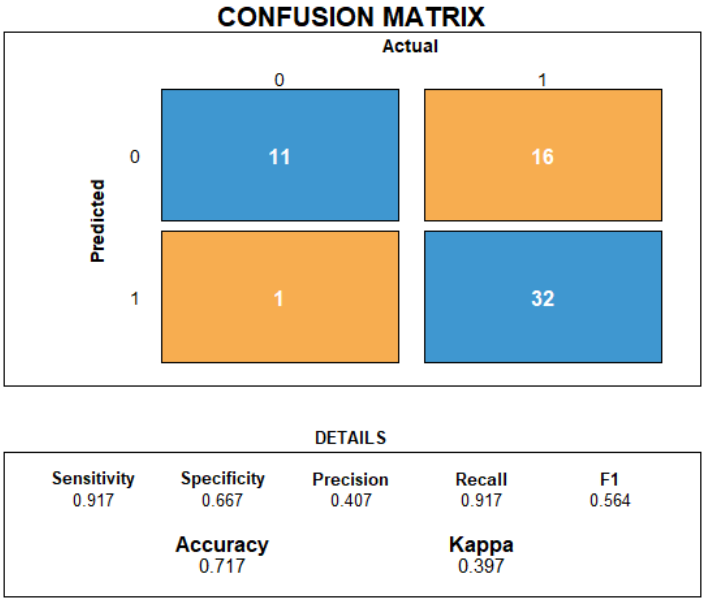


Figure 4.24: Confusion Matrix for The First Step - Seroconverted

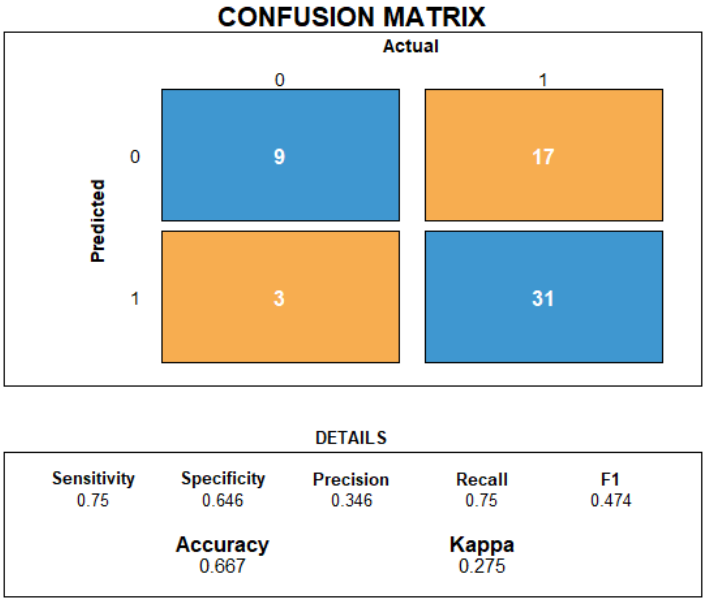


Figure 4.25: Confusion Matrix for The Second Step - Seroconverted

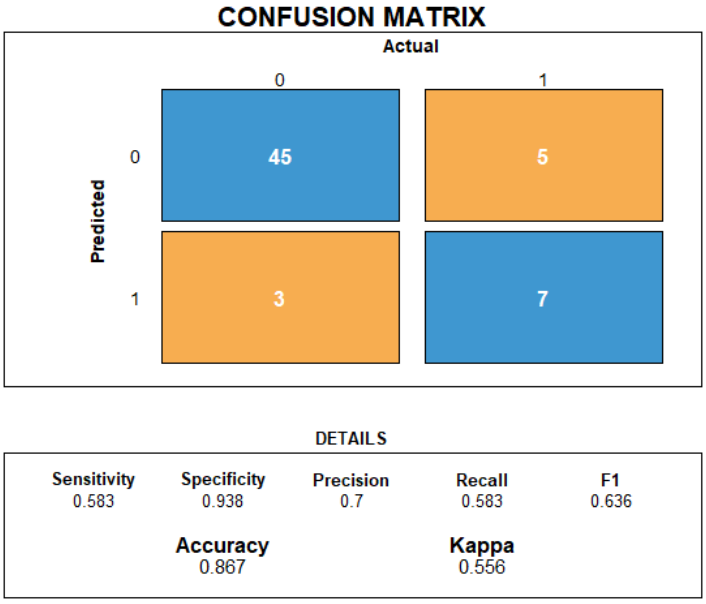


Figure 4.26: Confusion Matrix for The Third Step - Seroconverted

From the classification results, the following arguments can be made:

- Only sensitivity measure decreases around 36% from Step 1 (92%) to Step 3

(58%).

- On the other hand, specificity, precision, F1 Score and accuracy increases from Step 1 to Step 3. The final results are 94% for specificity, 70% for precision, 64% for F1 Score, 87% for accuracy.

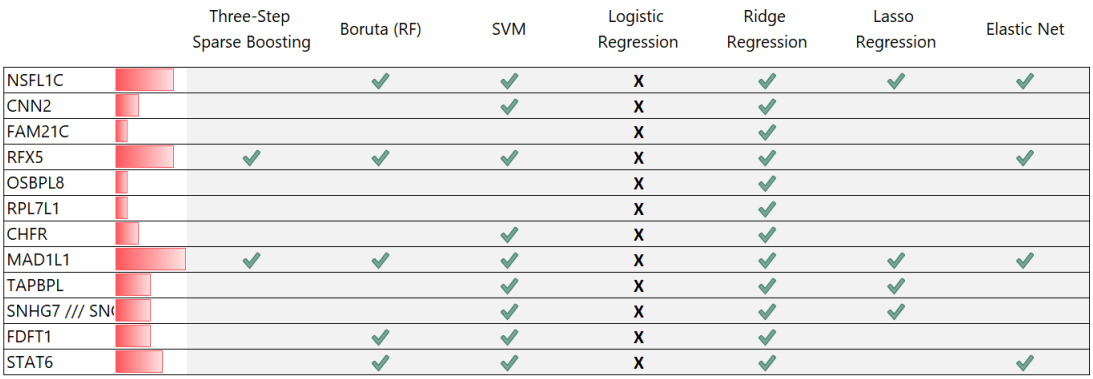
Comparison

Again, using Boruta (Random Forest), SVM, Logistic Regression, Ridge Regression, Lasso Regression and Elastic Net algorithms some results are presented in Table 4.28, namely; number of genes selected (# of Significant Genes), the number of overlapping selected genes and three-step sparse boosting (Overlap) and F1 Score (F1) as some performance metrics. In this data set, the prescreened 15 probes mapped to 12 different genes.

Table 4.28: Comparison with Other Algorithms - Seroconverted

Method	**# of Significant Genes**	**Overlap**	**F1**
Three-Step Sparse Boosting	2	-	64%
Boruta (RF)	5	2	38%
SVM	9	2	93%
Logistic Regression	Does not converge		-
Ridge Regression	12	2	29%
Lasso Regression	4	1	63%
Elastic Net	4	1	59%

The following figure illustrates the significant genes that are identified by the comparison algorithms and their overlapping structure. The red bars, again, lengthen with the more algorithm identifies the genes as significant.



	Three-Step Sparse Boosting	Boruta (RF)	SVM	Logistic Regression	Ridge Regression	Lasso Regression	Elastic Net
NSFL1C		✓	✓	X	✓	✓	✓
CNN2			✓	X	✓		
FAM21C				X	✓		
RFX5	✓	✓	✓	X	✓		✓
OSBPL8				X	✓		
RPL7L1				X	✓		
CHFR			✓	X	✓		
MAD1L1	✓	✓	✓	X	✓	✓	✓
TAPBPL			✓	X	✓	✓	
SNHG7 /// SN(			✓	X	✓	✓	
FDFT1		✓	✓	X	✓		
STAT6		✓	✓	X	✓		✓

Figure 4.27: Significant Genes Identified - Seroconverted

The following observations are done according to the results:

- SVM has the largest F1 score as 81% by identifying 9 genes as significant, while Three-Step Sparse Boosting algorithm follows it by 64% F1 Score and considers only 2 genes as significant.
- Boruta identifies 5 genes as significant and obtain very low F1 Score.
- Logistic Regression does not converge.
- Ridge Regressions finds again all of the 12 genes as significant and has the lowest F1 Score.
- Lasso Regression identifies 4 genes as significant and only one overlaps with Three-Step Sparse Boosting algorithm.
- Elastic Net identifies 4 of 12 genes as significant and obtain 59% as F1 Score.
- STAT6 gene is found significant by 4 of the algorithms and it has around 1 relevance score.
- On the other hand, similar to the above data set, NSFL1C gene is found significant by all the algorithms except Three-Step Sparse Boosting and there is no established relationship of it with diabetes in the literature.

In summary, the three-step sparse boosting procedure can be considered as performing well in terms of variable selection and classification.

Ensemble Approach

After analyzing the classification results of real data set, it is observed that in Step 1, sensitivity results are very good, on the other hand, specificity results are quite high in Step 3. Hence, an ensemble approach at the end of the algorithm to improve the classification performance of balanced real data set is utilized. Based on this approach, a final logistic regression model was run, considering the disease probabilities that are calculated in Step 1 and Step 3, where thresholds for logistic regression models were determined by ROC analysis. 5 different models were tried as indicated below and the one that improved the result the most is presented in Table 4.29, using the output as diagnosed/not diagnosed.

1. Inputs: Estimated probabilities in Step 1 and Step 3
2. Inputs: Estimated probabilities in Step 1 and Step 3, and their interaction
3. Inputs: Estimated probabilities in Step 1 and Step 3, and their squares
4. Inputs: Estimated probabilities in Step 1 and Step 3, their squares and all combinations of interactions
5. Inputs: (Estimated probabilities in Step 1-Estimated probabilities in Step 3)

- For Balanced Data Set - Whole Data

At the end of 3^{rd} ensemble approach utilized for whole balanced data set, specificity, precision and accuracy measures increased while others decreased.

- For Balanced Data Set - Seroconverted

When only the seroconverted children was considered, it seems that, this approach works better and all the metrics increase; sensitivity increases around 17%, specificity increases around 4% and attaining 98%, precision increses around 20%, F1 increses around 18%, accuracy increases around 6% and attaining 93% and kappa increases around 22%. These results were obtained again using the 3^{rd} ensemble approach described above.

The results are presented below.

Table 4.29: Ensemble Approach Results

Data Set	Method	Sensitivity	Specificity	Precision	F1 Score	Accuracy	Kappa
Whole Data	Step 3	0.727	0.696	0.608	0.662	0.708	0.409
	Ensemble - Model 3	0.636	0.765	0.636	0.636	0.714	0.401
Seroconverted	Step 3	0.583	0.938	0.700	0.636	0.867	0.556
	Ensemble - Model 3	0.750	0.979	0.900	0.818	0.933	0.778

4.2.3 Unbalanced Data Set

After obtaining the balanced data set results, the algorithm was implemented using the whole follow-up samples that makes the data unbalanced. These follow-ups vary between 3 to 12 times. Again, as the first step in preprocessing, probes with median expression lower than the median of median expressions (5.47) are excluded and the remaining 24,693 ones are kept. Also, similar to the balanced data study, using the independence screening approach to screen out the unimportant covariates described in the above section, we keep the top ranked 15 probes to be used in the algorithm and the rest of the covariates are screened out. Again, the maximum number of boosting iterations in three-step sparse boosting is 500.

Whole Data Set with No Imputation

As a first case, no imputation of the missing follow-up measurements is applied. The algorithm just ignores NA, and uses all the information that can be gathered from whole data set. The completion time was 1 hour 17 minutes 9 seconds, using a computer with Intel(R) Core (TM) i7-7600U CPU @ 2.80 GHz, 16,0 GB RAM. The number of significant probes selected by the end of each step of our algorithm is presented in Table 4.30:

Table 4.30: Number of Significant Probes in Each Step - Whole Set, No Imputation

Step 1	Step 2	Step 3
3	5	3

The probes that are found significant are also presented in Table 4.31.

Table 4.31: Significant Probes in Each Step - Whole Set, No Imputation

Step 1	**Step 2**	**Step 3**
11729976_a_at	11726625_at	11726625_at
11755359_x_at	11722692_x_at	11717515_a_at
11723273_at	11729976_a_at	11723273_at
-	11754416_s_at	-
-	11741228_a_at	-

Again, the association between the final 3 probes to genes using the mapping provided in the original data set is the following:

Table 4.32: Genes Associated with the Significant Probes - Whole Set, No Imputation

Probes	**Genes**
11726625_at	CCNG2
11717515_a_at	HSD17B10
11723273_at	FEM1C

These genes' contributions to the final classification is illustrated using SHAP values in Figure 4.28 and Figure 4.29 for the 1^{st} observation, whose observed response is 0, and for the 347^{th} observation, for whom the observed response is 1, respectively. It can be seen that, for the not diagnosed class, CCNG2 gene has the most influence and a negative relationship with the output, while HSD17B10 gene has a positive relationship and least effect on the output. For the diagnosed class, CCNG2 gene seems the most influential and has a negative relationship with the output. The least effective gene for this case is again FEM1C gene, yet this time has a negative relationship.

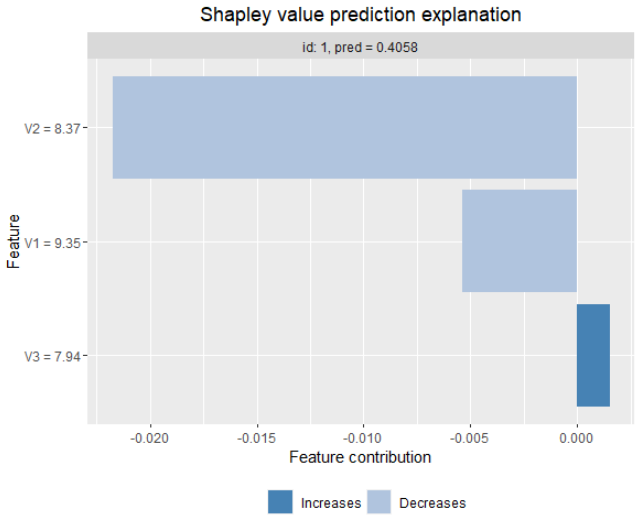


Figure 4.28: Contribution of Each Gene for Not Diagnosed Class - Whole Set, No Imputation

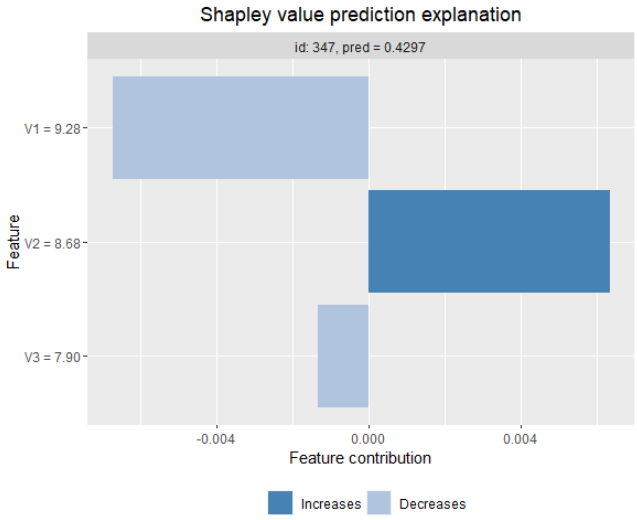


Figure 4.29: Contribution of Each Gene for Diagnosed Class - Whole Set, No Imputation

The loss function results at the end of each iteration and in each step are presented as AIC values in Figure 4.30. It can be seen that AICs after each iteration and step gets smaller. For Step 2 and Step 3, AICs get very close to 0.

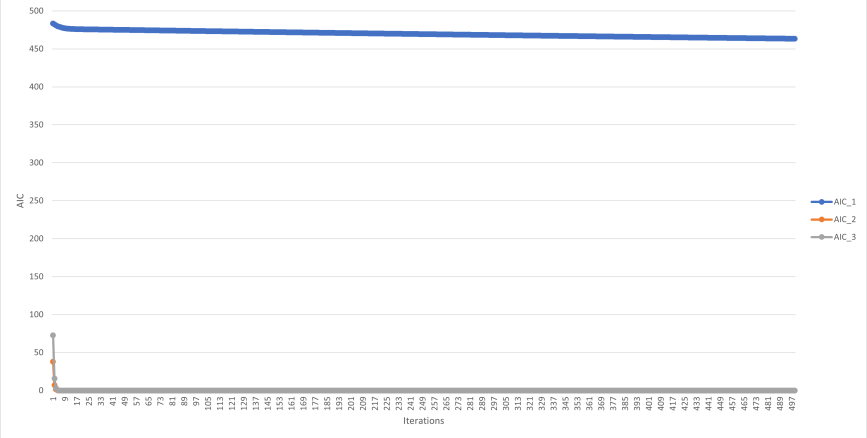


Figure 4.30: AIC's in Each Step and Iteration for the Whole Data Set with No Imputation

When we investigate these genes that are presented in 4.32 with their relationships with Type 1 diabetes, Zhao et al. (2020) found that CCNG2 gene "can protect against renal injury and fibrosis associated with diabetic nephropathy". Secondly, HSD17B10 is associated with Type 2 diabetes in the literature by Tarnopolsky et al. (2007) and Lim at al. (2011). Lastly, FEM1A and FEM1B are considered as associated with Type 2 diabetes in the literature (Bailey, 2010 and McGeachie, 2009).

For the classification performance, the following confusion matrix and related statistics are calculated at the end of each step.

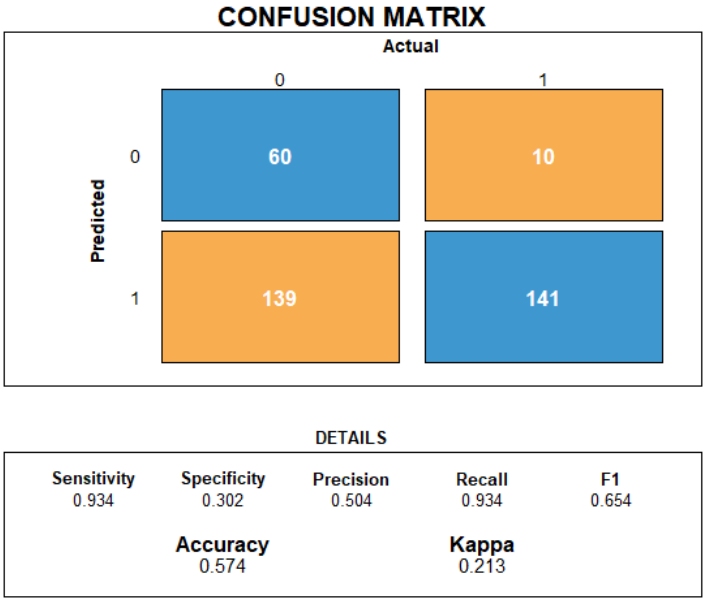


Figure 4.31: Confusion Matrix for The First Step - Whole Set, No Imputation

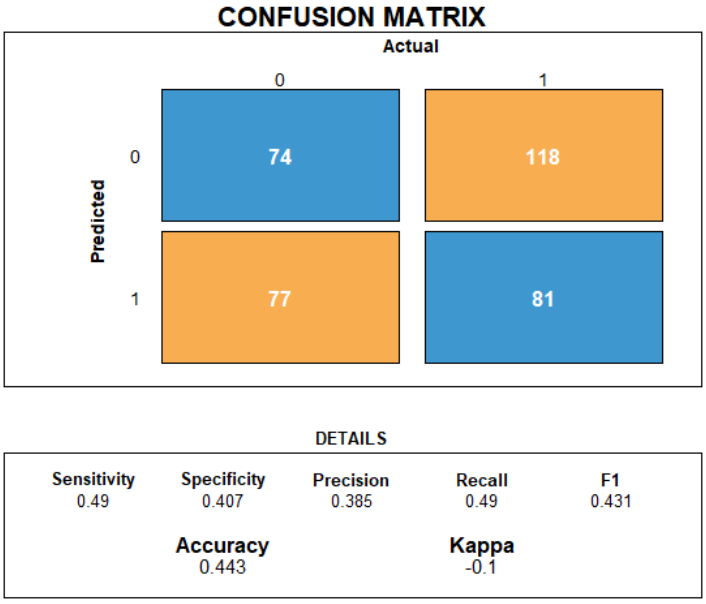


Figure 4.32: Confusion Matrix for The Second Step - Whole Set, No Imputation

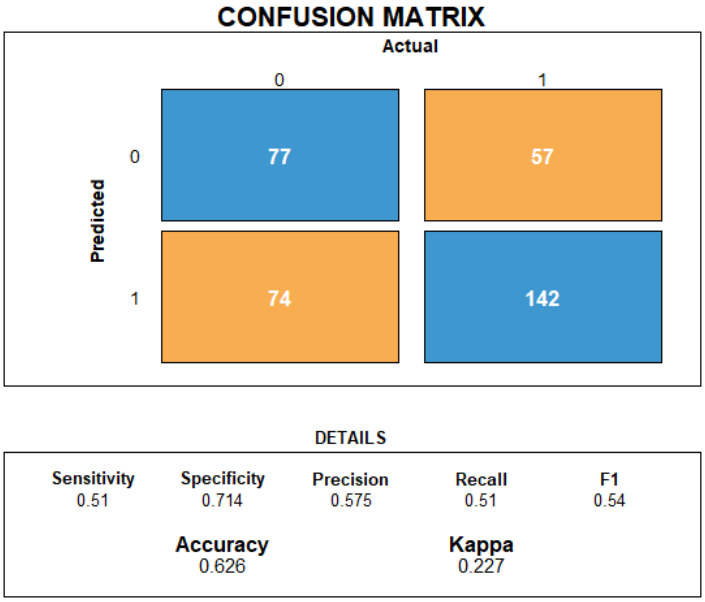


Figure 4.33: Confusion Matrix for The Third Step - Whole Set, No Imputation

From the results, the following arguments can be made:

- Similar to the balanced data set results, sensitivity measure decreases around 42% from Step 1 (93%) to Step 3 (51%).
- Yet, F1 score also decreases around 9% from Step 1 (65%) to Step 3 (54%).
- On the other hand, specificity, precision, and accuracy increases from Step 1 to Step 3. The final results are 71% for specificity, 58% for precision, 63% for accuracy.

Whole Data Set with Imputation with 0

Secondly, in order to compare results, all the missing follow-up measurements are filled by 0 to represent no measurement, and when all covariates and output is filled with 0 no output or metric are affected by that imputation. Hence, the algorithm again uses all the information that can be gathered from whole data set and the comparison between no imputation can be done. The completion time was 1 hour 35 minutes 36 seconds, using a computer with Intel(R) Core (TM) i7-7600U CPU @ 2.80 GHz,

16,0 GB RAM. The number of significant probes selected by the end of each step of our algorithm is presented in Table 4.33:

Table 4.33: Number of Significant Probes in Each Step - Whole Set, Imputation with 0

Step 1	**Step 2**	**Step 3**
3	2	3

The probes that are found significant are also presented in Table 4.34.

Table 4.34: Significant Probes in Each Step - Whole Set, Imputation with 0

Step 1	**Step 2**	**Step 3**
11717515_a_at	11717515_a_at	11717515_a_at
11738898_a_at	11738898_a_at	11738898_a_at
11736954_a_at	-	11736954_a_at

Again, the association between the final 3 probes to genes using the mapping provided in the original data set is the following:

Table 4.35: Genes Associated with the Significant Probes - Whole Set, Imputation with 0

Probes	**Genes**
11717515_a_at	HSD17B10
11738898_a_at	CLEC4C
11736954_a_at	CLEC4C

It seems that the algorithm identifies HSD17B10 gene as significant in both no imputation case and all filled by 0 case.

Again, using SHAP values, the significant genes' contributions to the final classification is illustrated in Figure 4.34 and Figure 4.35 for the 1^{st} observation, whose ob-

served response is 0, and or the 648^{th} observation, for whom the observed response is 1, respectively. It can be seen that, for the not diagnosed class, both genes effects are positive, whereas for the diagnosed class, there exists a negative relationship. Also, for both cases, HSD17B10 gene has more influence on the outcome than CLEC4C gene.

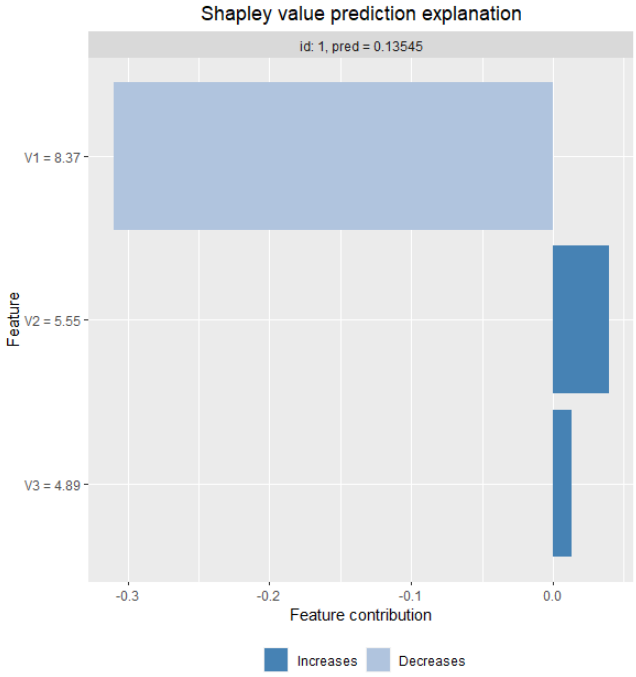


Figure 4.34: Contribution of Each Gene for Not Diagnosed Class - Whole Set, Imputation with 0

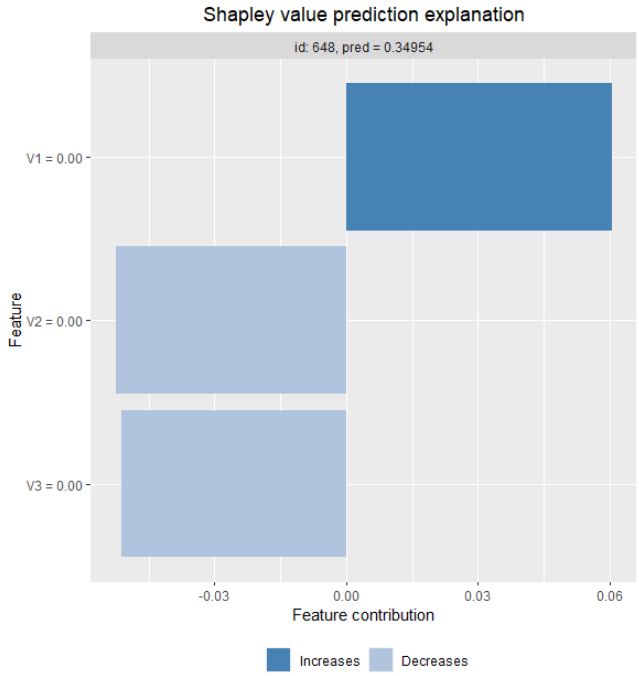


Figure 4.35: Contribution of Each Gene for Diagnosed Class - Whole Set, Imputation with 0

The loss function results at the end of each iteration and in each step are presented as AIC values in Figure 4.36. It can be seen that after each iteration AICs get smaller, even though the second step losses are too higher than the first and the third steps', which seem very close to each other in the graph.

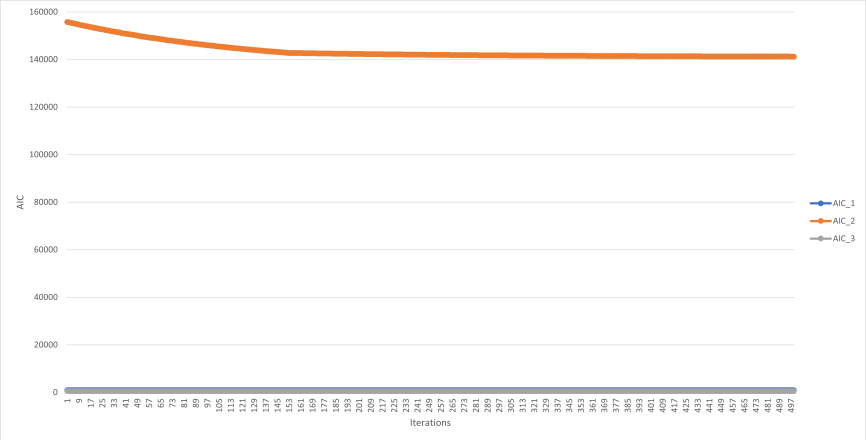


Figure 4.36: AIC's in Each Step and Iteration for the Whole Data Set with Imputation with 0

Again, when we investigate these genes that are presented in 4.35 with their relationships with Type 1 diabetes, as stated in the above case, HSD17B10 is associated with Type 2 diabetes in the literature by Tarnopolsky (2007) and Lim (2011). For CLEC4C, Riboldi et al. (2011) stated that "We found that CLEC4C recognizes complex type sugars with terminal galactose.". Also, CLEC4A and CLEC4M have known association with diabetes in the literature (Grimwood et al, 2004 and Renier, 2008).

For the classification performance, the following confusion matrix and related statistics are calculated at the end of each step.

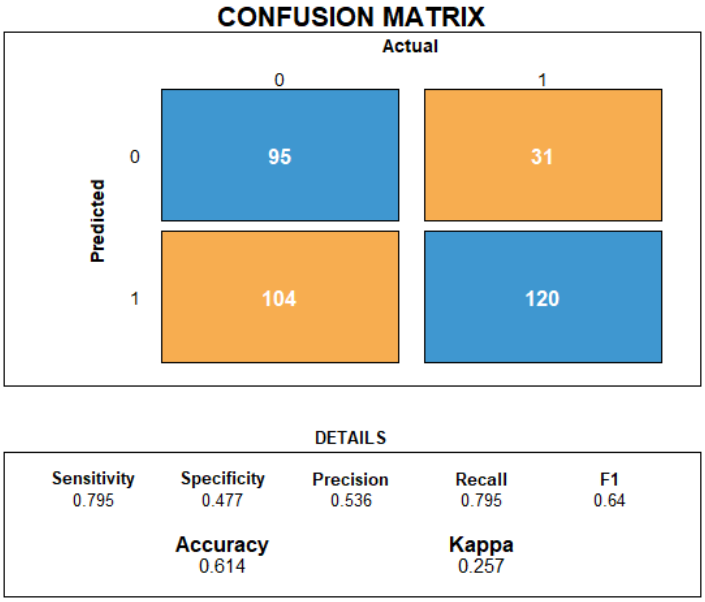


Figure 4.37: Confusion Matrix for The First Step - Whole Set, Imputation with 0

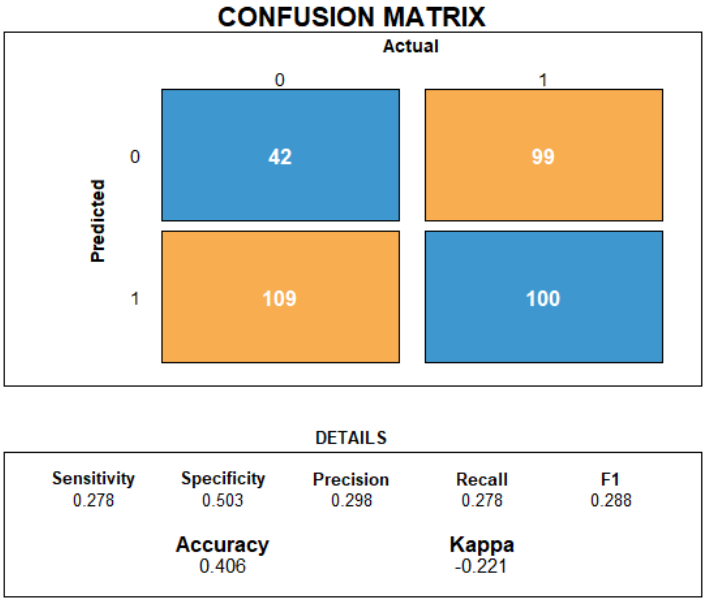


Figure 4.38: Confusion Matrix for The Second Step - Whole Set, Imputation with 0

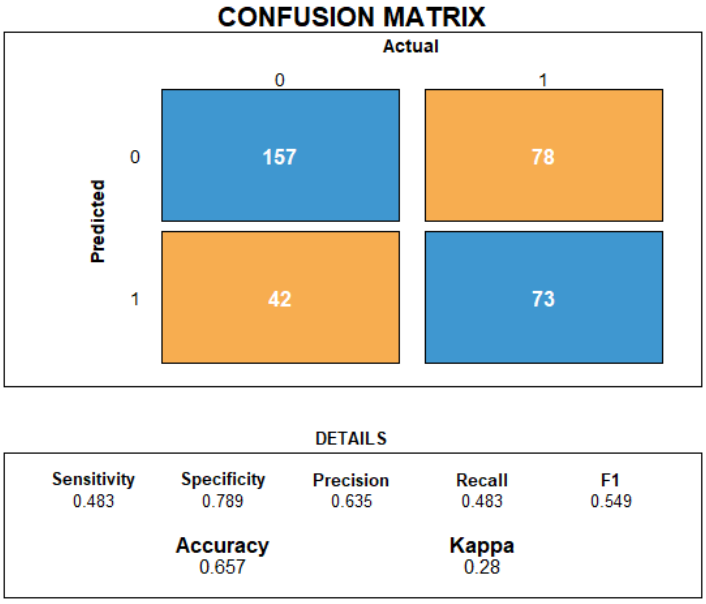


Figure 4.39: Confusion Matrix for The Third Step - Whole Set, Imputation with 0

From the results, the following arguments can be made:

- Sensitivity measure decreases around 32% from Step 1 (80%) to Step 3 (48%).

- F1 score also decreases around 9% from Step 1 (64%) to Step 3 (55%), similar to the no imputation case.
- On the other hand, specificity, precision, F1 Score and accuracy increases from Step 1 to Step 3. The final results are 79% for specificity, 64% for precision, 66% for accuracy.
- These results are higher than no imputation case.

When the comparison between no imputation and 0 filling case is made from the variable selection point of view, there is no significant difference between them, since both identifies two genes as significant which is beneficial regarding time and money. On the other hand, classification performance of missing value filling by 0 case is higher than no imputation case.

CHAPTER 5

DISCUSSION AND CONCLUSION

In longitudinal studies, many variables are measured. Often, these variables incorporate both temporal and spatial correlations, especially in the field of biomedical research. Classification using this type data is challenging due to high dimensionality and relatively small number of observations. In addition, whereas the number of potential covariates are large, only a small number of them hold a strong correlation with the outcome. For example, for a binary classification of cancer subtype problem, it was found that 50 genes are sufficient (Golub et al. (1999)). Therefore, selecting the significant variables and excluding the irrelevant, redundant and noisy ones, increase the prediction performance and enhance model parsimony while decreasing the computational burden. Also, when the variable selection algorithm is applied for DNA microarray experiments, the biological relationship with the target diseases of the selected genes is more easily identified and additional scientific understanding of the problem can be obtained (Ding and Peng (2005)).

Therefore, in this thesis, we propose a three-step sparse boosting model for detecting the most important genes that classify the individuals into diseased or non-diseased groups considering the longitudinal data having spatial and temporal correlations. The algorithm was constructed as the variable selection and coefficient estimation are conducted simultaneously. Also, no parameter tuning is necessary in the algorithm which is advantageous for computational time and complexity.

In the first step, assuming the independence in the data, logistic regression model is constructed using boosting method which estimates the coefficients and selects significant variables at the same time. Then, in the second step, using the deviance residuals obtained at the end of the first step, a weight matrix is calculated and by this

matrix, temporal correlation structure is included. In the third step, spatial correlation is added via a weight matrix calculated based on the correlation between covariates.

In order to measure the performance of the algorithm, firstly, a Monte Carlo Simulation Study was designed and nine different temporal and spatial correlation structure scenario were run using parallel computing methods. Simulation study showed that, due to sparsity of the algorithm, the minimum BIAS's, AB's and MSE's are obtained at the end of the first step, yet the number of mistakenly chosen significant covariates were the lowest at the end of Step 3. In addition, at the end of Step 3, the number of mistakenly chosen significant covariates decreases while establishing all the true ones as significant. As the classification performance, it got higher when the spatial and temporal correlation got higher and when the spatial correlation was high, the algorithm classification performances increased.

In addition to the simulation, a comparison study was made which includes Boruta (RF), Support Vector Machine (SVM), Logistic Regression, Ridge Regression, Lasso Regression and Elastic Net algorithms due to penalized approaches they incorporate similar to the our proposed model. From variable selection perspective, the results showed that, they considered very large number of mistakenly chosen significant covariates, whereas our proposed algorithm identified only the true significant variables. In addition, from classification point of view, Three-Step Sparse Boosting algorithm performs the best in terms of specificity and precision metrics.

Secondly, using the Finnish Type 1 Diabetes Prediction and Prevention (DIPP) study as the real data set, our proposed model was run on both balanced and unbalanced sets. Our algorithm identified a few number of genes as significant which can be beneficial regarding time and money. Also, Boruta (RF), SVM, Logistic Regression, Ridge Regression, Lasso Regression and Elastic Net algorithms were utilized on this data set. In addition, an ensemble method has been developed based on the probabilities calculated in Step 1 and Step 3 considering 5 different combination of them and their higher orders. It has been observed that this approach, in which probabilities of Step 1, Step 3 and their squares were considered, increased the classification performance metrics. Therefore, we recommend using this approach at the end of the algorithm for balanced data set to obtain better classification results. Also, since our

algorithm produced very well results with very few time points measurement, namely 3, converting unbalanced data set to balanced data set is recommended, which provides benefits in terms of time, variable selection and classification performance. Yet, when usage of unbalanced data set is absolutely necessary, our algorithm works well with unbalanced data set, also. We recommend utilizing unbalanced data set with 0 imputation for missing values. The Three-Step Sparse Boosting technique can be considered as performing well in terms of variable selection and classification.

As a conclusion, the numerical studies indicate that the significant covariates can be accurately selected by our proposed model and the real data analysis also demonstrates its useful results based on microarray data.

For the future extension, making the algorithm more robust to outliers could be studied, for example using MM-estimator as proposed by Rantaiainen et al. (2015). Also, random effect would be added to the model to represent the combined effect of all omitted subject-specific covariates that causes some subjects to be more prone to the disease than others. If the observation time points are many, varying coefficient structure could be incorporated, which capture the dynamical impacts of the covariates on the response variable.

REFERENCES

[1] Abdolell, M., LeBlanc, M., Stephens, D.A., and Harrison, R. (2002). Binary partitioning for continuous longitudinal data: categorizing a prognostic variable. *Statistics in Medicine*, 21 22, 3395-409. https://doi.org/10.1002/sim.1266

[2] Adewale, A. J., Dinu, I., Yasui, Y. (2010). Boosting for correlated binary classification. *Journal of Computational and Graphical Statistics* 19(1), 140-153. https://doi.org/10.1198/jcgs.2009.07118

[3] Aftab, S., Semenec, L., Chu, J. S., and Chen, N. (2008). Identification and characterization of novel human tissue-specific RFX transcription factors. *BMC Evolutionary Biology*, 8, 226. https://doi.org/10.1186/1471-2148-8-226

[4] Bailey, S. D., Xie, C., Do, R., Montpetit, A., Diaz, R., Mohan, V., Keavney, B., Yusuf, S., Gerstein, H. C., Engert, J. C., Anand, S., and DREAM investigators (2010). Variation at the NFATC2 locus increases the risk of thiazolidinedione-induced edema in the Diabetes Reduction Assessment with ramipril and rosiglitazone Medication (DREAM) study. Diabetes Care, 33(10), 2250–2253. https://doi.org/10.2337/dc10-0452

[5] Bartlett, P., and Traskin, M. (2007). AdaBoost is Consistent. *Journal of Machine Learning Research*, 8, 2347-2368. https://doi.org/10.7551/mitpress/7503.003.0018

[6] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. New York: Springer.

[7] Blanco, R., Larranaga, P., Inza, I. and Sierra, B. (2004). Gene selection for cancer classification using wrapper approaches. *International Journal of Pattern Recognition and Artificial Intelligence*, 18(08), 1373-1390. https://doi.org/10.1142/S0218001404003800

[8] Boulesteix, A.-L., Janitza, S., Kruppa, J., and König, I.R. (2012). Overview

of random forest methodology and practical guidance with emphasis on computational biology and bioinformatics. *WIREs Data Mining and Knowledge Discovery*, 2, 493-507. https://doi.org/10.1002/widm.1072

[9] Bø, T., and Jonassen, I. (2002). New feature subset selection procedures for classification of expression profiles. *Genome biology*, 3(4), RESEARCH0017. https://doi.org/10.1186/gb-2002-3-4-research0017

[10] Breiman, L. (1999). Prediction games and arcing algorithms. *Neural Computation*, 11, 1493–1517. https://doi.org/10.1162/089976699300016106

[11] Breiman, L. (2001). Random forests. *Machine Learning*, 45, 5–32. https://doi.org/10.1023/A:1010933404324

[12] Breiman, L. (1998). Arcing Classifiers. *The Annals of Statistics*, 26(3), 801–824. https://doi.org/10.1214/aos/1024691079

[13] Breitling, R., Armengaud, P., Amtmann, A., and Herzyk, P. (2004). Rank products: a simple, yet powerful, new method to detect differentially regulated genes in replicated microarray experiments. *FEBS letters*, 573(1-3), 83–92. https://doi.org/10.1016/j.febslet.2004.07.055

[14] Buja, A., Stuetzle, W. and Shen, Y. (2005). Loss Functions for Binary Class Probability Estimation and Classification: Structure and Applications. Technical report, University Washington. Available at `http://www.stat.washington.edu/wxs/Learning-papers/paper-proper-scoring.pdf`

[15] Burnham, K. P., and Anderson, D. R. (2004). Multimodel Inference: Understanding AIC and BIC in Model Selection. *Sociological Methods & Research*, 33(2), 261–304. https://doi.org/10.1177/0049124104268644

[16] Bühlmann, P. (2006). Boosting for high-dimensional linear models. *The Annals of Statistics*, 34:2, 559-583. https://doi.org/10.1214/009053606000000092

[17] Bühlmann, P. and Hothorn, T. (2007). Boosting Algorithms: Regularization, Prediction and Model Fitting. *Statistical Science*, 22(4), 477–505. https://doi.org/10.1214/07-STS242

[18] Bühlmann, P. and Yu, B. (2003). Boosting with the L2 loss. *Journal of the American Statistical Association*, 98:462, 324-339. https://doi.org/10.1198/016214503000125

[19] Bühlmann, P. and Yu, B. (2006). Sparse boosting. *Journal of Machine Learning Research*, 7, 1001–1024.

[20] Cantoni, E., Flemming, J., and Ronchetti, E. (2005). Variable Selection for Marginal Longitudinal Generalized Linear Models. *Biometrics*, 61, 507-14. https://doi.org/10.1111/j.1541-0420.2005.00331.x

[21] Cantoni, E., Field, C., Mills Flemming, J., and Ronchetti, E. (2007). Longitudinal variable selection by cross-validation in the case of many covariates. *Statistics in Medicine*, 26(4), 919–930. https://doi.org/10.1002/sim.2572

[22] Carlin, B. P. and Louis, T. A. (2000) *Bayes and Empirical Bayes Method for Data Analysis*. New York, NY: Chapman and Hall/CRC.

[23] Cano, G., Rodríguez, J., Garcia-Garcia, A., Pérez-Sánchez, H., Benediktsson, J., Thapa, A., and Barr, A. (2017). Automatic Selection of Molecular Descriptors using Random Forest: Application to Drug Discovery. *Expert Systems with Applications*, 72, 151–159. https://doi.org/10.1016/j.eswa.2016.12.008

[24] Charnes, A., Frome, E. and Yu, Po. (1976). The equivalence of generalized least squares and maximum likelihood estimates in the exponential family. *Journal of The American Statistical Association*, 71, 169-171. https://doi.org/10.1080/01621459.1976.10481508

[25] Chen, K., Xu, T., Bi, J. (2018). Latent sparse modeling of longitudinal multidimensional data. In AAAI Conference on Artificial Intelligence.

[26] Cheng, M.-Y., Honda, T., Li, J., Peng, H. (2014). Nonparametric independence screening and structure identification for ultra-high dimensional longitudinal data.*The Annals of Statistics*, 42 (5), 1819–1849. https://doi.org/10.1214/14-AOS1236

Ming-Yen Cheng, Toshio Honda, Jialiang Li, Heng Peng

[27] Cutler, D.R., Edwards, T.C., Jr., Beard, K.H., Cutler, A., Hess, K.T., Gibson, J. and Lawler, J.J. (2007), Random forest for classification in ecology. *Ecology*, 88, 2783-2792. https://doi.org/10.1890/07-0539.1

[28] Dauwan, M., van der Zande, J. J., van Dellen, E., Sommer, I. E., Scheltens, P., Lemstra, A. W., and Stam, C. J. (2016). Random forest to differentiate dementia with Lewy bodies from Alzheimer's disease. *Alzheimer's & dementia (Amsterdam, Netherlands)*, 4, 99–106. https://doi.org/10.1016/j.dadm.2016.07.003

[29] De'ath, G. (2002), Multivariate regression trees: a new technique for modeling species-environment relationships. *Ecology*, 83, 1105-1117. https://doi.org/10.1890/0012-9658(2002)083[1105:MRTANT]2.0.CO;2

[30] Dempster, A.P., Laird, N.M., and Rubin, D.B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, B, 39, 1–38. https://doi.org/10.1111/j.2517-6161.1977.tb01600.x

[31] Derry M. C., Yanagiya A., Martineau Y., and Sonenberg N. (2006). Regulation of poly(A)-binding protein through PABPinteracting proteins. *Cold Spring Harb Symp Quant Biol*, 71, 537–543. https://doi.org/10.1101/sqb.2006.71.061

[32] Díaz-Uriarte, R., and Alvarez de Andrés, S. (2006). Gene selection and classification of microarray data using random forest. *BMC Bioinformatics*, 7, 3. https://doi.org/10.1186/1471-2105-7-3

[33] Diggle, P.J. (1989). Testing for random dropouts in repeated measurement data. *Biometrics*, 45, 1255–58. https://doi.org/2531777

[34] Diggle, P.J. and Kenward, M.G. (1994). Informative dropout in longitudinal data analysis (with discussion). *Applied Statistics*, 43, 49–73. https://doi.org/2986113

[35] Dine, A., Larocque, D., amd Bellavance, F. (2009). Multivariate trees for mixed outcomes. *Computational Statistics & Data Analysis*, 53, 3795-3804. https://doi.org/10.1016/j.csda.2009.04.003

[36] Ding, C., and Peng, H. (2005). Minimum redundancy feature selection from microarray gene expression data. *Journal of Bioinformatics and Computational Biology*, 3(2), 185–205. https://doi.org/10.1142/s0219720005001004

[37] Dudoit, S., Fridlyand J. and Speed, T. P. (2002). Comparison of discriminant methods for the classification of tumors using gene expression data, *Journal of the American Statistical Association*, 97:457, 77-87. https://doi.org/10.1198/016214502753479248

[38] Dziak, J. J., and Li, R. (2007). An overview on variable selection for longitudinal data. In *Quantitative Medical Data Analysis Using Mathematical Tools and Statistical Techniques* (pp. 3-24). World Scientific Publishing Co.. https://doi.org/10.1142/9789812772121_0001

[39] Eo, S. -H., and Cho H. (2014). Tree-structured mixed-effects regression modeling for longitudinal data. *Journal of Computational and Graphical Statistics*, 23,(3), 740-760. https://doi.org/10.1080/10618600.2013.794732

[40] Falba, T., Jofre-Bonet, M., Busch, S., Duchovny, N., and Sindelar, J. (2004). Reduction of quantity smoked predicts future cessation among older smokers. *Addiction (Abingdon, England)*, 99(1), 93–102. https://doi.org/10.1111/j.1360-0443.2004.00574.x

[41] Fokkema, M., Smits, N., Zeileis, A., Hothorn, T., and Kelderman, H. (2018). Detecting treatment-subgroup interactions in clustered data with generalized linear mixed-effects model trees. *Behavior Research Methods*, 50(5), 2016–2034. https://doi.org/10.3758/s13428-017-0971-x

[42] Freund, Y. (1995). Boosting a weak learning algorithm by majority. *Information and Computation*, 121(2), 256-285. https://doi.org/10.1006/inco.1995.1136

[43] Freund, Y., and Schapire, R.E. (1996). Experiments with a new boosting algorithm. In Proceedings of the *Thirteenth International Conference on Machine Learning*, 96, 148-156.

[44] Freund, Y., and Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1), 119-139. https://doi.org/10.1006/jcss.1997.1504

[45] Friedman, J. H. (2001). Greedy function approximation: a gradi-

ent boosting machine. *The Annals of Statistics*, 29(5), 1189-1232. https://doi.org/10.1214/aos/1013203451

[46] Friedman, J. H., Hastie, T., Tibshirani, R. (2000). Additive logistic regression: a statistical view of boosting. *The Annals of Statistics*, 28(2), 337-407. https://doi.org/10.1214/aos/1016218223

[47] Fu, W., and Simonoff, J. (2015). Unbiased regression trees for longitudinal and clustered data. *Computational Statistics & Data Analysis*, 88, 53-74. https://doi.org/10.1016/j.csda.2015.02.004.

[48] Garam, L., Byungkon, K., Kwangsik, N., Kyung-Ah, S., and Dokyoon, K. (2019). MildInt: Deep learning-based multimodal longitudinal data integration framework. *Frontiers in Genetics*, 10, Article 617. https://doi.org/10.3389/fgene.2019.00617

[49] Gevaert, O., De Smet, F., Timmerman, D., Moreau, Y., and De Moor, B. (2006). Predicting the prognosis of breast cancer by integrating clinical and microarray data with Bayesian networks. *Bioinformatics*, 22(14), e184–e190. https://doi.org/10.1093/bioinformatics/btl230

[50] Goldstein, B. A., Polley, E. C., and Briggs, F. B. (2011). Random forests for genetic association studies. *Statistical applications in genetics and molecular biology*, 10(1), 32. https://doi.org/10.2202/1544-6115.1691

[51] Golub, T. R., Slonim, D. K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J. P., Coller, H., Loh, M. L., Downing, J. R., Caligiuri, M. A., Bloomfield, C. D., and Lander, E. S. (1999). Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science (New York, N.Y.)*, 286(5439), 531–537. https://doi.org/10.1126/science.286.5439.531

[52] Greenshtein, E., and Ritov, Y. (2004). Persistence in high-dimensional linear predictor selection and the virtue of overparametrization. *Bernoulli*, 10(6), 971-988. https://doi.org/10.3150/bj/1106314846

[53] Grimwood, J., Gordon, L. A., Olsen, A., Terry, A., Schmutz, J., Lamerdin, J., Hellsten, U., Goodstein, D., Couronne, O., Tran-Gyamfi, M., Aerts, A., Altherr, M., Ashworth, L., Bajorek, E., Black, S., Branscomb, E., Caenepeel,

S., Carrano, A., Caoile, C., Chan, Y. M., ... Lucas, S. M. (2004). The DNA sequence and biology of human chromosome 19. *Nature*, 428(6982), 529–535. https://doi.org/10.1038/nature02399

[54] Gruebner, O., Lowe, S. R., Tracy, M., Cerdá, M., Joshi, S., Norris, F. H., Galea, S. (2016). The geography of mental health and general wellness in Galveston Bay after Hurricane Ike: A spatial epidemiologic study with longitudinal data. *Disaster Med Public Health Prep* 10(2), 261-273. https://doi.org/10.1017/dmp.2015.172

[55] Guyon, I., Weston, J., Barnhill, S., and Vapnik, V. (2002). Gene Selection for Cancer Classification using Support Vector Machines. *Machine Learning*, 46, 389–422. https://doi.org/10.1023/A:1012487302797

[56] Hajjem, A., Bellavance, F., and Larocque, D. (2011). Mixed effects regression trees for clustered data. *Statistics & Probability Letters*, 81, 451-459. https://doi.org/10.1016/j.spl.2010.12.003

[57] Hajjem, A., Bellavance F., and Larocque, D. (2014). Mixed-effects random forest for clustered data. *Journal of Statistical Computation and Simulation*, 84(6), 1313-1328. https://doi.org/10.1080/00949655.2012.741599

[58] Hajjem, A., Larocque, D., and Bellavance, F. (2017). Generalized mixed effects regression trees. *Statistics & Probability Letters*, 126, 114-118. https://doi.org/10.1016/j.spl.2017.02.033

[59] Hansen, M., and Yu, B. (2001). Model selection and the principle of minimum description length. *Journal of the American Statistical Association*, 96:454, 746-774. https://doi.org/10.1198/016214501753168398

[60] Hansen, M., and Yu, B. (2003). Minimum description length model selection criteria for generalized linear models. Lecture Notes-Monograph Series, 40, 145-163.

[61] Hastie, T., Tibshirani, R., and Freidman, J. (2001). *The Elements of Statistical Learning; Data Mining, Inference and Prediction*. Springer, New York.

[62] Hsu, J. J., Finkelstein, D. M., Schoenfeld, D. A. (2018). Informatively clustering longitudinal microarrays using binary or survival outcome data. *Communications in Statistics: Case Studies, Data Analysis and Applications*, 4(1), 18-27. https://doi.org/10.1080/23737484.2018.1455542

[63] Huang, L., Jin, Y., Gao, Y., Thung, K. H., Shen, D., and Alzheimer's Disease Neuroimaging Initiative (2016). Longitudinal clinical score prediction in Alzheimer's disease with soft-split sparse regression based random forest. *Neurobiology of aging*, 46, 180–191. https://doi.org/10.1016/j.neurobiolaging.2016.07.005

[64] Huang, Y., and Pan, J. (2021). Penalized joint generalized estimating equations for longitudinal binary data. *Biometrical Journal*, 64, 57– 73. https://doi.org/10.1002/bimj.202000336

[65] Hurvich, C.M., Simonoff, J.S., and Tsai, C. L. (1998). Smoothing parameter selection in nonparametric regression using an improved Akaike information criterion. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 60, 271-293. https://doi.org/10.1111/1467-9868.00125

[66] Hyun, J. W., Li, Y., Huang, C., Styner, M., Lin, W., Zhu, H. (2016). STGP: Spatio-temporal Gaussian process models for longitudinal neuroimaging data. *Neuroimage* 134, 550-562. https://doi.org/10.1016/j.neuroimage.2016.04.023

[67] Inza, I., Larrañaga, P., Blanco, R., and Cerrolaza, A. J. (2004). Filter versus wrapper gene selection approaches in DNA microarray domains. *Artificial intelligence in medicine*, 31(2), 91–103. https://doi.org/10.1016/j.artmed.2004.01.007

[68] Jiang, W. (2004). Process consistency for Adaboost. *The Annals of Statistics*, 32(1), 13–29. https://doi.org/10.1214/aos/1079120128

[69] Kanamori T., Takenouchi T., Eguchi S., Murata N. (2004). The most robust loss function for boosting. *Neural Information Processing Lecture Notes in Computer Science*, 3316. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-30499-9_76

[70] Kang, J., Shen, X., Yang, L., Feng, S., Li, D., and Yuan, H. (2020). Screening of potential biomarkers in the occurrence and development of type 1 diabetes mellitus based on transcriptome analysis. *Endokrynologia Polska*, 71(1), 58–65. https://doi.org/10.5603/EP.a2019.0060

[71] Klén, R., Karhunen, M. and Elo, L.L. (2020). Likelihood contrasts: a machine learning algorithm for binary classification of longitudinal data. *Sci Rep*, 10, 1016. https://doi.org/10.1038/s41598-020-57924-9

[72] Laird N. M. (1988). Missing data in longitudinal studies. *Statistics in Medicine*, 7, 305-315. https://doi.org/10.1002/sim.4780070131

[73] Larivière, B., and Van den Poel, D. (2005). Predicting customer retention and profitability by using random forests and regression forests techniques. *Expert Systems with Applications*, 29, 472-484. https://doi.org/10.1016/j.eswa.2005.04.043

[74] Lee, S. (2005). On generalized multivariate decision tree by using GEE. *Computational Statistics & Data Analysis*, 49. 1105-1119. https://doi.org/10.1016/j.csda.2004.07.003

[75] Lee, S. (2019). The association of genetically controlled CpG methylation (cg158269415) of protein tyrosine phosphatase, receptor type N2 (PTPRN2) with childhood obesity. *Sci Rep* 9, 4855. https://doi.org/10.1038/s41598-019-40486-w

[76] Li, H., Shu, D., Zhang, Y., and Yi, G. Y. (2018). Simultaneous variable selection and estimation for multivariate multilevel longitudinal data with both continuous and binary responses. *Computational Statistics & Data Analysis*, 118, 126-137. https://doi.org/10.1016/j.csda.2017.09.004

[77] Lim, Y. A., Grimm, A., Giese, M., Mensah-Nyagan, A. G., Villafranca, J. E., Ittner, L. M., Eckert, A., and Götz, J. (2011). Inhibition of the mitochondrial enzyme ABAD restores the amyloid-β-mediated deregulation of estradiol. *PloS One*, 6(12), e28887. https://doi.org/10.1371/journal.pone.0028887

[78] Liu, S., Xu, C., Zhang, Y., Liu, J., Yu, B., Liu, X., and Dehmer, M. (2018). Feature selection of gene expression data for Cancer clas-

sification using double RBF-kernels. *BMC Bioinformatics*, 19(1), 396. https://doi.org/10.1186/s12859-018-2400-2

[79] Liao, J.G., and Chin K.V. (2007). Logistic regression for disease classification using microarray data: model selection in a large p and small n case. *Bioinformatics*, 23(15), 1945–1951. https://doi.org/10.1093/bioinformatics/btm287

[80] Little, R. J. A. (1982). Models for nonresponse in sample surveys. *Journal of the American Statistical Association*, 77, 237-250. https://doi.org/2287227

[81] Lu, J. M., Chen, Y. C., Ao, Z. X., Shen, J., Zeng, C. P., Lin, X., Peng, L. P., Zhou, R., Wang, X. F., Peng, C., Xiao, H. M., Zhang, K., and Deng, H. W. (2019). System network analysis of genomics and transcriptomics data identified type 1 diabetes-associated pathway and genes. *Genes and Immunity*, 20(6), 500–508. https://doi.org/10.1038/s41435-018-0045-9

[82] Luts, J., Molenberghs, G., Verbeke, G., Huffel, S., and Suykens, J. (2012). A mixed effects least squares support vector machine model for classification of longitudinal data. *Computational Statistics & Data Analysis*. 56. 611-628. https://doi.org/10.1016/j.csda.2011.09.008

[83] Ma, S., and Huang, J. (2005). Regularized ROC method for disease classification and biomarker selection with microarray data. *Bioinformatics*, 21(24), 4356–4362. https://doi.org/10.1093/bioinformatics/bti724

[84] Mason, L., Baxter, J., Bartlett, P. L. and Frean, M. (2000). Boosting algorithms as gradient descent. *Advances in Neural Information Processing Systems*, 12, 512-518.

[85] Mason, L., Baxter, J., Bartlett, P. L. and Frean, M. (1999). Functional gradient techniques for combining hypotheses. *Advances in Large Margin Classifiers*, 221-246, MIT Press.

[86] Mease, D. (2004). Cost-weighted boosting with jittering and over / under-sampling : JOUS-Boost. *Journal of Machine Learning Research*, 8, 409–439.

[87] McGeachie, M., Ramoni, R. L., Mychaleckyj, J. C., Furie, K. L., Dreyfuss, J. M., Liu, Y., Herrington, D., Guo, X., Lima, J. A., Post, W., Rotter, J. I., Rich,

S., Sale, M., and Ramoni, M. F. (2009). Integrative predictive model of coronary artery calcification in atherosclerosis. *Circulation*, 120(24), 2448–2454. https://doi.org/10.1161/CIRCULATIONAHA.109.865501

[88] Menezes, F. S., Liska, G. R., Cirillo, M. A., Vivanco, M. J. F. (2016). Data classification with binary response through the Boosting algorithm and logistic regression. *Expert System With Applications 69*, 62-73. https://doi.org/10.1016/j.eswa.2016.08.014

[89] Müller, H. G. (2005). Functional modelling and classification of longitudinal data. *Scandinavian Journal of Statistics*, 32(2), 223-240. https://doi.org/10.1111/j.1467-9469.2005.00429.x

[90] Neuhaus, J.M., Hauck, W.W., and Kalbfleisch, J.D. (1992). The effects of mixture distribution misspecification when fitting mixed-effects logistic models. *Biometrika*, 79, 755–62. https://doi.org/2337231

[91] Ngufor, C., Van Houten, H., Caffo, B. S., Shah, N. D., and McCoy, R. G. (2019). Mixed effect machine learning: A framework for predicting longitudinal change in hemoglobin A1c. *Journal of Biomedical Informatics*, 89, 56–67. https://doi.org/10.1016/j.jbi.2018.09.001

[92] Parzen, M., Ghosh, S., Lipsitz, S., Sinha, D., Fitzmaurice, G. M., Mallick, B. K., and Ibrahim, J. G. (2011). A generalized linear mixed model for longitudinal binary data with a marginal logit link function. *The Annals of Applied Statistics*, 5(1), 449–467. https://doi.org/10.1214/10-AOAS390

[93] Pomsuwan, T., and Freitas, A. A. (2017). Feature Selection for the Classification of Longitudinal Human Ageing Data. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)* (739-746). https://doi.org/10.1109/ICDMW.2017.102.

[94] Rantaiainen, M., Lindgren, C. M., Holmes, C. C. (2015). Robust linear models for cis-eQTL analysis. *PLoS ONE*, 10(5), 1-16. https://doi.org/10.1371/journal.pone.0127882

[95] Redondo, M. J., Fain, P. R., and Eisenbarth, G. S. (2001). Genetics

of type 1A diabetes. *Recent progress in hormone research*, 56, 69–89. https://doi.org/10.1210/rp.56.1.69

[96] Renier G. (2008). Lectin-like oxidized low-density lipoprotein receptor-1 (LOX-1), a relevant target for diabetic vasculopathy?. *Cardiovascular & Hematological Disorders Drug Targets*, 8(3), 203–211. https://doi.org/10.2174/187152908785849107

[97] Riboldi, E., Daniele, R., Parola, C., Inforzato, A., Arnold, P. L., Bosisio, D., Fremont, D. H., Bastone, A., Colonna, M., and Sozzani, S. (2011). Human C-type lectin domain family 4, member C (CLEC4C/BDCA-2/CD303) is a receptor for asialo-galactosyl-oligosaccharides. *The Journal of Biological Chemistry*, 286(41), 35329–35333. https://doi.org/10.1074/jbc.C111.290494

[98] Ridout, M. (1991). Testing for random dropouts in repeated measurement data. *Biometrics*, 47, 1617–21. https://doi.org/2532413

[99] Rosasco, Lorenzo; Poggio, Tomaso (2014). A Regularization Tour of Machine Learning, MIT-9.520 Lectures Notes, Manuscript.

[100] Rubin, D. B. (1976). Inference and missing data. *Biometrika*, 63, 581-592. https://doi.org/2335739

[101] Ruiz, R., Riquelme, J., and Aguilar-Ruiz, J. (2006). Incremental wrapper-based gene selection from microarray data for cancer classification. *Pattern Recognition*, 39, 2383-2392. https://doi.org/10.1016/j.patcog.2005.11.001

[102] Saeys, Y., Inza, I., Larrañaga, P. (2007). A review of feature selection techniques in bioinformatics. *Bioinformatics*, 23(19), 2507–2517. https://doi.org/10.1093/bioinformatics/btm344

[103] Segal, M. R. (1992). Tree-Structured methods for longitudinal data. *Journal of the American Statistical Association*, 87(418), 407–418. https://doi.org/10.2307/2290271

[104] Sela, R.J., and Simonoff, J.S. (2012). RE-EM trees: a data mining approach for longitudinal and clustered data. *Machine Learning*, 86, 169–207. https://doi.org/10.1007/s10994-011-5258-3

[105] Serban, N., Staicu, A. M., Carroll, R. J. (2013). Multilevel cross-dependent binary longitudinal data. *Biometrics*, 69, 903-913. https://doi.org/10.1111/biom.12083

[106] Schapire, R. E. (1990). The strength of weak learnability. *Machine Learning*, 5, 197-227.

[107] Shapley, L. S. (1953). A Value for n-Person Games. In Harold W. Kuhn and Albert W. Tucker (Eds.), Contributions to the Theory of Games II, 307–17. Princeton: Princeton University Press.

[108] Siroky, D. S. (2009). Navigating random forests and related advances in algorithmic modeling. *Statistics Survey*, 3, 147-163. https://doi.org/10.1214/07-SS033

[109] Stirm, L., Huypens, P., Sass, S., Batra, R., Fritsche, L., Brucker, S., Abele, H., Hennige, A. M., Theis, F., Beckers, J., Hrabě de Angelis, M., Fritsche, A., Häring, H. U., and Staiger, H. (2018). Maternal whole blood cell miRNA-340 is elevated in gestational diabetes and inversely regulated by glucose and insulin. *Scientific Reports*, 8(1), 1366. https://doi.org/10.1038/s41598-018-19200-9

[110] Speiser, J. L., Durkalski, V. L., and Lee, W. M. (2015). Random forest classification of etiologies for an orphan disease. *Statistics in Medicine*, 34(5), 887–899. https://doi.org/10.1002/sim.6351

[111] Speiser, J. L., Wolf, B. J., Chung, D., Karvellas, C. J., Koch, D. G., and Durkalski, V. L. (2019). BiMM forest: A random forest method for modeling clustered and longitudinal binary outcomes. *Chemometrics and Intelligent Laboratory Systems : An International Journal Sponsored by the Chemometrics Society*, 185, 122–134. https://doi.org/10.1016/j.chemolab.2019.01.002

[112] Speiser, J. L., Wolf, B. J., Chung, D., Karvellas, C. J., Koch, D. G., and Durkalski, V. L. (2020). BiMM tree: A decision tree method for modeling clustered and longitudinal binary outcomes. *Communications in statistics: Simulation and computation*, 49(4), 1004–1023. https://doi.org/10.1080/03610918.2018.1490429

[113] Speiser J. L. (2021). A random forest method with feature selection for developing medical prediction models with clustered and longitudinal data. *Journal of biomedical informatics*, 117, 103763. https://doi.org/10.1016/j.jbi.2021.103763

[114] Štrumbelj, E. and Igor K. (2010). An Efficient Explanation of Individual Classifications Using Game Theory. *Journal of Machine Learning Research*, 11 (March), 1–18. https://doi.org/10.1145/1756006.1756007

[115] Štrumbelj, E. and Igor K. (2014). Explaining prediction models and individual predictions with feature contributions. *Knowledge and Information Systems*, 41 (3), 647–65. https://doi.org/10.1007/s10115-013-0679-x

[116] Svetnik V., Liaw A., Tong C., Wang T. (2004). pplication of Breiman's random forest to modeling structure-activity relationships of pharmaceutical molecules. In: Roli F., Kittler J., Windeatt T. (eds) Multiple Classifier Systems. Lecture Notes in Computer Science, vol 3077, 334–343. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-25966-4_33

[117] Tang, R., Sinnwell, J. P., Li, J., Rider, D. N., de Andrade, M., and Biernacka, J. M. (2009). Identification of genes and haplotypes that predict rheumatoid arthritis using random forests. *BMC proceedings*, 3 Suppl 7(Suppl 7), S68. https://doi.org/10.1186/1753-6561-3-s7-s68

[118] Tarnopolsky, M. A., Rennie, C. D., Robertshaw, H. A., Fedak-Tarnopolsky, S. N., Devries, M. C., and Hamadeh, M. J. (2007). Influence of endurance exercise training and sex on intramyocellular lipid and mitochondrial ultrastructure, substrate use, and mitochondrial enzyme activity. *American Journal of Physiology. Regulatory, Integrative and Comparative Physiology*, 292(3), R1271–R1278. https://doi.org/10.1152/ajpregu.00472.2006

[119] Tian, S., and Wang, C. (2019). Feature Selection for Longitudinal Data by Using Sign Averages to Summarize Gene Expression Values over Time. *BioMed Research International*, 2019, 1724898. https://doi.org/10.1155/2019/1724898

[120] Thomas, J. G., Olson, J. M., Tapscott, S. J., and Zhao, L. P. (2001). An efficient and robust statistical modeling approach to discover differentially expressed

genes using genomic expression profiles. *Genome research*, 11(7), 1227–1236. https://doi.org/10.1101/gr.165101

[121] Touw, W. G., Bayjanov, J. R., Overmars, L., Backus, L., Boekhorst, J., Wels, M., and van Hijum, S. A. (2013). Data mining in the Life Sciences with Random Forest: a walk in the park or lost in the jungle?. *Briefings in Bioinformatics*, 14(3), 315–326. https://doi.org/10.1093/bib/bbs034

[122] Törn, C., Liu, X., Hagopian, W. et al. (2016). Complement gene variants in relation to autoantibodies to beta cell specific antigens and type 1 diabetes in the TEDDY Study. *Sci Rep* 6, 27887. https://doi.org/10.1038/srep27887

[123] Tutz, G., and Groll, A. (2013). Likelihood-based boosting in binary and ordinal random effects models. *Journal of Computational and Graphical Statistics*, 22(2), 356-378. https://doi.org/10.1080/10618600.2012.694769

[124] Vogl, W. D., Waldstein, S. M., Gerendas, B. S., Simader, C., Glodan, A. M., Podkowinski, D. et al. (2015). Spatio-temporal signatures to predict retinal disease recurrence. *Inf Process Med Imaging* 24, 152–163. https://doi.org/10.1007/978-3-319-19992-4_12

[125] Wang, Y., Tetko, I. V., Hall, M. A., Frank, E., Facius, A., Mayer, K. F., and Mewes, H. W. (2005). Gene selection from microarray data for cancer classification-a machine learning approach. *Computational Biology and Chemistry*, 29(1), 37–46. https://doi.org/10.1016/j.compbiolchem.2004.11.001

[126] Wang, X., and Qu, A. (2014). Efficient classification for longitudinal data. *Computational Statistics & Data Analysis*, 78, 119-134. https://doi.org/10.1016/j.csda.2014.04.008

[127] Wray, L. A., Herzog, A. R., Willis, R. J., and Wallace, R. B. (1998). The impact of education and heart attack on smoking cessation among middle-aged adults. *Journal of Health and Social Behavior*, 39(4), 271–294.

[128] Xia, X., Jiang, B., Li, J., Zhang, W. (2016). Low-dimensional confounder adjustment and high-dimensional penalized estimation for survival analysis. *Lifetime data analysis*, 22(4), 547–569. https://doi.org/10.1007/s10985-015-9350-z

[129] Xiong, Y., Kim H. J., and Singh, V. (2019). Mixed effects neural networks (MeNets) with applications to gaze estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 7735-7744, https://doi.org/10.1109/CVPR.2019.00793

[130] Uh, H. W., Mertens, B. J., Jan van der Wijk, H., Putter, H., van Houwelingen, H. C., and Houwing-Duistermaat, J. J. (2007). Model selection based on logistic regression in a highly correlated candidate gene region. *BMC proceedings, 1 Suppl 1*(Suppl 1), S114. https://doi.org/10.1186/1753-6561-1-S1-S114

[131] Yeung, K.Y. and Bumgarner, R.E. (2003). Multiclass classification of microarray data with repeated measurements: application to cancer. *Genome Biology*, 4, R83. https://doi.org/10.1186/gb-2003-4-12-r83

[132] Yue, M., Li, J. (2017). Improvement screening for ultra-high dimensional data with censored survival outcomes and varying coefficients. *The International Journal of Biostatistics*, 13(1), 20170024. https://doi.org/10.1515/ijb-2017-0024

[133] Yue, M., Li, J., Cheng, M. (2019). Two-step sparse boosting for high-dimensional longitudinal data with varying coefficients. *Computational Statistics and Data Analysis*, 131, 222-234. https://doi.org/10.1016/j.csda.2018.10.002

[134] Zhang, H., and Ye, Y. (2008). A tree-based method for modeling a multivariate ordinal response. *Statistics and Its Interface*, 1(1), 169–178. https://doi.org/10.4310/sii.2008.v1.n1.a14

[135] Zhang, K., Wang, H., Bathke, A. C., Harrar, S. W., Piepho, H., Deng, Y. (2011). Gene set analysis for longitudinal gene expression data. *Bioinformatics*, 12, 273-284. https://doi.org/10.1186/1471-2105-12-273.

[136] Zhao, C., Gao, J., Li, S., Liu, Q., Hou, X., Xing, X., Wang, D., Sun, M., Wang, S., and Luo, Y. (2020). Cyclin G2 regulates canonical Wnt signalling via interaction with Dapper1 to attenuate tubulointerstitial fibrosis in diabetic nephropathy. *Journal of Cellular and Molecular Medicine*, 24(5), 2749–2760. https://doi.org/10.1111/jcmm.14946

[137] Zhou, Q., Hong, W., Luo, L., and Yang, F. (2010). Gene Selection Using Random Forest and Proximity Differences Criterion on DNA Microarray Data. *Journal of Convergence Information Technology*, 5, 161-170. https://doi.org/10.4156/jcit.vol5.issue6.17

APPENDIX A

THREE-STEP SPARSE BOOSTING R CODE

```
# Global Variable Definition ----------------------------
----------------------
set.seed(42)
p = 29 #dimension
n = 150 #individuals
m = 4 #measurements

v = 0.1 #step-size

iteration1 = 100
iteration2 = 100
iteration3 = 100

reps = 100

library(foreach)
library(doParallel)
cl <- makeCluster(detectCores())
registerDoParallel(cl)


coverage <- function(b, se, true, level = 0.95, df = Inf)
{ # Estimate,
```

```
# standard error,
# true parameter,
# confidence level,
# and df
qtile <- level + (1 - level)/2 # Compute the proper quantile
lower.bound <- b - qt(qtile, df = df)*se # Lower bound
upper.bound <- b + qt(qtile, df = df)*se # Upper bound
# Is the true parameter in the confidence interval? (yes = 1)
true.in.ci <- ifelse(true >= lower.bound & true <= upper.
bound, 1, 0)
cp <- mean(true.in.ci) # The coverage probability
mc.lower.bound <- cp - 1.96*sqrt((cp*(1-cp))/length(b))
mc.upper.bound <- cp + 1.96*sqrt((cp*(1-cp))/length(b))
return(list(coverage.probability = cp, #Return results
true.in.ci = true.in.ci,
ci = cbind(lower.bound, upper.bound),
mc.eb = c(mc.lower.bound, mc.upper.bound)))
}

foreach(j=1:reps) %dopar% {

# Simulation Data Generation ----------------------------
------------------

rho_temp = 0.8
rho_spat = 0.8

times_temp <- 1:m
sigma_temp <- 1

times_spat <- 1:(p-2)
sigma_spat <- 1
```

```
H_temp <- abs(outer(times_temp, times_temp, "-"))
Temp <- sigma_temp * rho_temp^H_temp
Temp[cbind(1:nrow(Temp), 1:nrow(Temp))] <-
Temp[cbind(1:nrow(Temp), 1:nrow(Temp))] * sigma_temp

H_spat <- abs(outer(times_spat, times_spat, "-"))
Spat <- sigma_spat * rho_spat^H_spat
Spat[cbind(1:nrow(Spat), 1:nrow(Spat))] <-
Spat[cbind(1:nrow(Spat), 1:nrow(Spat))] * sigma_spat

Sigma = Temp %x% Spat

library(MASS)
x_normals <- mvrnorm(n, rep(0, (p-2)*m), Sigma)
x_binary <- as.data.frame(rbinom(n, 1, 0.5))

x_temporary <- matrix(0,n,(p-2))
x_temporary2 <- list()
for (i in 1:n) {
for (j in 1:m) {
x_temporary <- c(1,x_binary[i,],x_normals[i,((j-1)*(p-2)+
1):(j*(p-2))])
x_temporary2 <- append(x_temporary2, list(x_temporary))
}
}

x <- do.call(rbind,x_temporary2)

#true betas
t_beta0 <- 1.5
t_beta1 <- 1.6
t_beta2 <- 1.8
t_beta3 <- 1.3
```

```
t_beta4 <- 1.7
t_beta5 <- 1.4

true_betas <- cbind(t_beta0, t_beta1, t_beta2, t_beta3, t_
beta4, t_beta5)

num_sign_cov <- length(true_betas)

probtrue <- rep(0,(n*m))
probtrue <- 1/(1+exp(-(t_beta0*x[,1]+t_beta1*x[,2]+t_beta2*x[,
3]+t_beta3*x[,4]+t_beta4*x[,5]+t_beta5*x[,6])))
w <- probtrue*(1-probtrue)

y <- rbinom(n*m, 1, 1/(1+exp(-(t_beta0*x[,1]+t_beta1*x[,2]+
t_beta2*x[,3]+t_beta3*x[,4]+t_beta4*x[,5]+t_beta5*x[,6]))))


# First Step Variable Definition ------------------------
---------------------------

lambda_1 <-           matrix(0,1,p)
lambdahat_1 <-        matrix(0,1,iteration1)

H_cov1 <-             list()
Bmatrix_cov1 <-       list()
Bmatrix_1 <-          list()
degreefreedom_cov1 <- rep(0,p)
degreefreedom_1 <-    rep(0,iteration1)

AIC_cov1 <-           rep(0,p)
AIC_1 <-              rep(0,iteration1)

s_1 <-                rep(0,iteration1)
```

```
beta_1 <-              matrix(0,p,iteration1)
beta_star_1 <-         rep(0,p)
probnew_1 <-           rep(0,(n*m))
ynewhat_1 <-           rep(0,(n*m))
Sigmahat_1 <-          list()

loss_func1 <-          rep(0,p)

# First Step Algorithm ---------------------------------
-----------------

for (k in 2:iteration1)
{
Bmatrix_1[[1]] <- diag(1,(n*m),(n*m))
Bmatrix_1[[k]] <- matrix(0,(n*m),(n*m))

for(i in 1:p)
{
x_1 <- x[,i,drop=F]
b_1 <- beta_1[,k-1,drop=F]
f1 <- function(lambda_1) -(1/(n*m))*sum((y*log(1/
(1+exp(-((x%*%b_1)+(x_1%*%lambda_1))))))+((1-y)*
log(1-(1/(1+exp(-((x%*%b_1)+(x_1%*%lambda_1)))))))))

lambda_1[,i] <- nlm(f1, lambda_1[,i])$estimate

H_cov1[[i]] <- matrix(0,(n*m),(n*m))
H_cov1[[i]] <- x_1%*%solve(t(x_1)%*%x_1)%*%(t(x_1))

Bmatrix_cov1[[i]] <- matrix(0,(n*m),(n*m))
Bmatrix_cov1[[i]] <- Bmatrix_1[[k-1]]%*%(diag(1,
(n*m),(n*m))-H_cov1[[i]])
```

```
degreefreedom_cov1[i] <- sum(diag(diag(1,(n*m),(n*m))
-Bmatrix_cov1[[i]])) #trace

AIC_cov1[i] <- -2*sum((y*log(1/(1+exp(-((x%*%
beta_1[,k-1])+(x_1%*%lambda_1[,i])))))))+((1-y)*
log(1-(1/(1+exp(-((x%*%beta_1[,k-1])+(x_1%*%
lambda_1[,i])))))))))+(2*degreefreedom_cov1[i])
}

s_1[k] <- min(which(AIC_cov1==min(AIC_cov1)))

x_select_1 <- x[,s_1[k]]

lambdahat_1[,k] <- lambda_1[,s_1[k]]

beta_1[,k] <- beta_1[,k-1]
beta_1[,k][s_1[k]] <- beta_1[,(k-1)][s_1[k]]+
v*lambdahat_1[,k]

Bmatrix_1[[k]] <- Bmatrix_1[[k-1]] %*%(diag(1,(n*m),
(n*m)) - v*H_cov1[[s_1[k]]])
degreefreedom_1[k] <- sum(diag(diag(1,(n*m),(n*m))
-Bmatrix_1[[k]]))

AIC_1[k] <- -2*sum((y*log(1/(1+
exp(-((x%*%beta_1[,k])))))))+((1-y)*log(1-
(1/(1+exp(-((x%*%beta_1[,k])))))))))+(2*degreefreedom_1[k])

}

khat_1 <- which.min((AIC_1)*NA^(AIC_1 <=0))
beta_star_1 <- beta_1[,khat_1]
```

```
probnew_1 <- 1/(1+exp(-(x%*%beta_star_1)))
ynewhat_1[probnew_1 >= 0.5] <- 1

library(caret)
conf_matrix_1 <- confusionMatrix
(table(ynewhat_1, y),
positive = "1", mode = "everything")

library(Matrix)
var_covar_betastar_1 <- solve(t(x)%*%Diagonal(n*m,
 probnew_1*(1-probnew_1))%*%x)
se_1 <- sqrt(diag(var_covar_betastar_1))

errorsquare_1 <- matrix(0,m,m) #temporal correlation
for (i in 1:n)
{
errorsquare_1 <- errorsquare_1+(sign(y-probnew_1)
[((i-1)*m+1):(i*m)]*ifelse(y[((i-1)*m+1):(i*m)]==1,
sqrt(-2*log(probnew_1)[((i-1)*m+1):(i*m)]),
sqrt(-2*log(1-probnew_1)[((i-1)*m+1):(i*m)])))%*%
t(sign(y-probnew_1)[((i-1)*m+1):(i*m)]
*ifelse(y[((i-1)*m+1):(i*m)]==1,sqrt(-2*
log(probnew_1)[((i-1)*m+1):(i*m)]),sqrt(-2*
log(1-probnew_1)[((i-1)*m+1):(i*m)])))
}
Sigmahat_1 <- errorsquare_1/n

TP1 <- 0

for (i in 1:num_sign_cov) {
TP1 <- TP1 + any(s_1[2:khat_1]==i)
}
```

```
FP1 <- length(unique(s_1[2:khat_1][s_1[2:khat_1]
>num_sign_cov]))
Size1 <- TP1+FP1

MIAE1 <- rep(0,num_sign_cov)

for (i in 1:num_sign_cov) {
MIAE1[i] <- abs(true_betas[i]-beta_star_1[i])
names(MIAE1)[i] <- paste("MIAE1", i, sep = "_")
}

BIAS1 <- rep(0,num_sign_cov)

for (i in 1:num_sign_cov) {
BIAS1[i] <- true_betas[i]-beta_star_1[i]
names(BIAS1)[i] <- paste("BIAS1", i, sep = "_")
}

MSE1 <- rep(0,num_sign_cov)

for (i in 1:num_sign_cov) {
MSE1[i] <- (true_betas[i]-beta_star_1[i])^2
names(MSE1)[i] <- paste("MSE1", i, sep = "_")
}

cp1 <- rep(0,num_sign_cov)

for (i in 1:num_sign_cov) {
cp1[i] <- coverage(beta_star_1[i], se_1[i],
true_betas[i], df = (n*m) - num_sign_cov)
$coverage.probability
names(cp1)[i] <- paste("cp1", i, sep = "_")
}
```

```
# Second Step Variable Definition ----------------------
---------------------------

library(Matrix)
w_2 <-
bdiag(replicate(n,Sigmahat_1,simplify=F))

lambda_2 <-           matrix(0,1,p)
lambdahat_2 <-        matrix(0,1,iteration2)

H_cov2 <-             list()
Bmatrix_cov2 <-       list()
Bmatrix_2 <-          list()
degreefreedom_cov2 <- rep(0,p)
degreefreedom_2 <-    rep(0,iteration2)

AIC_cov2 <-           rep(0,p)
AIC_2 <-              rep(0,iteration2)

s_2 <-                rep(0,iteration2)

beta_2 <-             matrix(0,p,iteration2)
beta_star_2 <-        rep(0,p)
probnew_2 <-          rep(0,(n*m))
ynewhat_2 <-          rep(0,(n*m))
Sigmahat_2 <-         list()

# Second Step Algorithm --------------------------------
------------------

for (k in 2:iteration2)
{
```

```
Bmatrix_2[[1]] <- Bmatrix_1[[khat_1]]
Bmatrix_2[[k]] <- matrix(0,(n*m),(n*m))

for(i in 1:p)
{
x_2 <- x[,i,drop=F]
b_2 <- beta_2[,k-1,drop=F]
f2 <- function(lambda_2) -(1/(n*m))*sum((y*log(1/(1+exp(-
((x%*%b_2)+(w_2%*%x_2%*%lambda_2))))))+(((1-y)*log(1-(1/
(1+exp(-((x%*%b_2)+(w_2%*%x_2%*%lambda_2))))))))))

lambda_2[,i] <- nlm(f2, lambda_2[,i])$estimate

H_cov2[[i]] <- matrix(0,(n*m),(n*m))
H_cov2[[i]] <- x_2%*%solve(t(x_2)%*%w_2%*%x_2)
%*%(t(x_2))%*%w_2

Bmatrix_cov2[[i]] <- matrix(0,(n*m),(n*m))
Bmatrix_cov2[[i]] <- Bmatrix_2[[k-1]] %*%(diag(1,(n*m),
(n*m))-H_cov2[[i]])
degreefreedom_cov2[i] <- sum(diag(diag(1,(n*m),(n*m))
-Bmatrix_cov2[[i]]))

AIC_cov2[i] <- -2*sum(w_2%*%(y*log(1/(1+exp(-((x%*%beta_2[,
k-1])+(x_2%*%lambda_2[,i])))))))+(w_2%*%((1-y)*log(1-(1/(1+
exp(-((x%*%beta_2[,k-1])+(x_2%*%lambda_2[,i])))))))))
+(2*degreefreedom_cov2[i])
}

s_2[k] <- min(which(AIC_cov2==min(AIC_cov2)))

x_select_2 <- x[,s_2[k]]
```

```
lambdahat_2[,k] <- lambda_2[,s_2[k]]

beta_2[,k] <- beta_2[,k-1]
beta_2[,k][s_2[k]] <- beta_2[,(k-1)][s_2[k]]
+v*lambdahat_2[,k]

hat_2 <- x_select_2%*%solve(t(x_select_2)%*%w_2
%*%x_select_2)%*%(t(x_select_2))%*%w_2

Bmatrix_2[[k]] <- Bmatrix_2[[k-1]] %*%(diag(1,(n*m),
(n*m)) - v*hat_2)
degreefreedom_2[k] <- sum(diag(diag(1,(n*m),(n*m))
-Bmatrix_2[[k]]))

AIC_2[k] <- -2*sum(w_2%*%(y*log(1/(1+exp(-((x%*%
beta_2[,k])))))+(w_2%*%((1-y)*log(1-(1/(1+exp(-
((x%*%beta_2[,k]))))))))))+(2*degreefreedom_2[k])

}

khat_2 <- which.min((AIC_2)*NA^(AIC_2 <=0))
beta_star_2 <- beta_2[,khat_2]

probnew_2 <- 1/(1+exp(-(x%*%beta_star_2)))
ynewhat_2[probnew_2 >= 0.5] <- 1

conf_matrix_2 <- confusionMatrix(table(ynewhat_2, y),
positive = "1", mode = "everything")

var_covar_betastar_2 <- solve(t(x)%*%Diagonal(n*m,
 probnew_2*(1-probnew_2))%*%x)
se_2 <- sqrt(diag(var_covar_betastar_2))
```

```
errorsquare_2 <- matrix(0,m,m) #temporal correlation
for (i in 1:n)
{
errorsquare_2 <- errorsquare_2+(sign(y-probnew_2)
[((i-1)*m+1):(i*m)]*ifelse(y[((i-1)*m+1):(i*m)]==1,
sqrt(-2*log(probnew_2)[((i-1)*m+1):(i*m)]),
sqrt(-2*log(1-probnew_2)[((i-1)*m+1):(i*m)]))%*%
t(sign(y-probnew_2)[((i-1)*m+1):(i*m)]*
ifelse(y[((i-1)*m+1):(i*m)]==1,sqrt(-2*
log(probnew_2)[((i-1)*m+1):(i*m)]),
sqrt(-2*log(1-probnew_2)[((i-1)*m+1):(i*m)])))
}
Sigmahat_2 <- errorsquare_2/n

TP2 <- 0

for (i in 1:num_sign_cov) {
TP2 <- TP2 + any(s_2[2:khat_2]==i)

}

FP2 <- length(unique(s_2[2:khat_2][s_2[2:khat_2]>
num_sign_cov]))
Size2 <- TP2+FP2

MIAE2 <- rep(0,num_sign_cov)

for (i in 1:num_sign_cov) {
MIAE2[i] <- abs(true_betas[i]-beta_star_2[i])
names(MIAE2)[i] <- paste("MIAE2", i, sep = "_")
}

BIAS2 <- rep(0,num_sign_cov)
```

```
for (i in 1:num_sign_cov) {
BIAS2[i] <- true_betas[i]-beta_star_2[i]
names(BIAS2)[i] <- paste("BIAS2", i, sep = "_")
}

MSE2 <- rep(0,num_sign_cov)

for (i in 1:num_sign_cov) {
MSE2[i] <- (true_betas[i]-beta_star_2[i])^2
names(MSE2)[i] <- paste("MSE2", i, sep = "_")
}

cp2 <- rep(0,num_sign_cov)

for (i in 1:num_sign_cov) {
cp2[i] <- coverage(beta_star_2[i], se_2[i],
true_betas[i], df = (n*m) - num_sign_cov)
$coverage.probability
names(cp2)[i] <- paste("cp2", i, sep = "_")
}

# Third Step Variable Definition ----------------------
-------------------------

w_3 <- Diagonal(n*m, probnew_1*(1-probnew_1))

lambda_3 <-          matrix(0,1,p)
lambdahat_3 <-       matrix(0,1,iteration3)

H_cov3 <-            list()
Bmatrix_cov3 <-      list()
Bmatrix_3 <-         list()
```

```
degreefreedom_cov3 <- rep(0,p)
degreefreedom_3 <-    rep(0,iteration3)

AIC_cov3 <-           rep(0,p)
AIC_3 <-              rep(0,iteration3)

s_3 <-                rep(0,iteration3)

beta_3 <-             matrix(0,p,iteration3)
beta_star_3 <-        rep(0,p)
probnew_3 <-          rep(0,(n*m))
ynewhat_3 <-          rep(0,(n*m))

# Third Step Algorithm ---------------------------------
-----------------

for (k in 2:iteration3)
{
Bmatrix_3[[1]] <- Bmatrix_1[[khat_1]]
Bmatrix_3[[k]] <- matrix(0,(n*m),(n*m))

for(i in 1:p)
{
x_3 <- x[,i,drop=F]
b_3 <- beta_3[,k-1,drop=F]
f3 <- function(lambda_3) -(1/(n*m))*sum((y*log(1/(1+exp(-
((x%*%b_3)+(w_3%*%x_3%*%lambda_3))))))+
(((1-y)*log(1-(1/(1+exp(-((x%*%b_3)+
(x_3%*%lambda_3)))))))))

lambda_3[,i] <- nlm(f3, lambda_3[,i])$estimate

H_cov3[[i]] <- matrix(0,(n*m),(n*m))
```

```
H_cov3[[i]] <- x_3%*%solve(t(x_3)%*%w_3%*%x_3)
%*%(t(x_3))%*%w_3

Bmatrix_cov3[[i]] <- matrix(0,(n*m),(n*m))
Bmatrix_cov3[[i]] <- Bmatrix_3[[k-1]] %*%(diag(1,
(n*m),(n*m))-H_cov3[[i]])
degreefreedom_cov3[i] <- sum(diag(diag(1,(n*m),
(n*m))-Bmatrix_cov3[[i]]))

AIC_cov3[i] <- -2*sum(w_3%*%(y*log(1/(1+
exp(-((x%*%beta_3[,k-1])+(x_3%*%lambda_3[,i])))))))+
(((1-y)*log(1-(1/(1+exp(-((x%*%beta_3[,k-1])+
(x_3%*%lambda_3[,i]))))))))))+
(2*degreefreedom_cov3[i])
}

s_3[k] <- min(which(AIC_cov3==min(AIC_cov3)))

x_select_3 <- x[,s_3[k]]

lambdahat_3[,k] <- lambda_3[,s_3[k]]

beta_3[,k] <- beta_3[,k-1]
beta_3[,k][s_3[k]] <- beta_3[,(k-1)][s_3[k]]
+v*lambdahat_3[,k]

hat_3 <- x_select_3%*%solve(t(x_select_3)%*%
w_3%*%x_select_3)%*%(t(x_select_3))%*%w_3

Bmatrix_3[[k]] <- Bmatrix_3[[k-1]] %*%(diag(1,
(n*m),(n*m)) - v*hat_3)
degreefreedom_3[k] <- sum(diag(diag(1,(n*m),
(n*m))-Bmatrix_3[[k]]))
```

```
AIC_3[k] <- -2*sum(w_3%*%(y*log(1/(1+exp(-((x%*%
beta_3[,k])))))+(((1-y)*log(1-(1/(1+exp(-((x%*%
beta_3[,k]))))))))))+(2*degreefreedom_3[k])

}

khat_3 <- which.min((AIC_3)*NA^(AIC_3 <=0))
beta_star_3 <- beta_3[,khat_3]

probnew_3 <- 1/(1+exp(-(x%*%beta_star_3)))
ynewhat_3[probnew_3 >= 0.5] <- 1

conf_matrix_3 <- confusionMatrix(table(ynewhat_3, y),
positive = "1", mode = "everything")

var_covar_betastar_3 <- solve(t(x)%*%Diagonal(n*m,
probnew_3*(1-probnew_3))%*%x)
se_3 <- sqrt(diag(var_covar_betastar_3))

TP3 <- 0

for (i in 1:num_sign_cov) {
TP3 <- TP3 + any(s_3[2:khat_3]==i)

}


FP3 <- length(unique(s_3[2:khat_3][s_3[2:khat_3]>
num_sign_cov]))
Size3 <- TP3+FP3

MIAE3 <- rep(0,num_sign_cov)
```

```
for (i in 1:num_sign_cov) {
MIAE3[i] <- abs(true_betas[i]-beta_star_3[i])
names(MIAE3)[i] <- paste("MIAE3", i, sep = "_")
}

BIAS3 <- rep(0,num_sign_cov)

for (i in 1:num_sign_cov) {
BIAS3[i] <- true_betas[i]-beta_star_3[i]
names(BIAS3)[i] <- paste("BIAS3", i, sep = "_")
}

MSE3 <- rep(0,num_sign_cov)

for (i in 1:num_sign_cov) {
MSE3[i] <- (true_betas[i]-beta_star_3[i])^2
names(MSE3)[i] <- paste("MSE3", i, sep = "_")
}

cp3 <- rep(0,num_sign_cov)

for (i in 1:num_sign_cov) {
cp3[i] <- coverage(beta_star_3[i], se_3[i],
true_betas[i], df = (n*m) - num_sign_cov)
$coverage.probability
names(cp3)[i] <- paste("cp3", i, sep = "_")
}

# Output ------------------------------------------------
--

TP <- c(TP1, TP2, TP3)
```

```
TP <- setNames(TP, c("TP1", "TP2", "TP3"))
FP <- c(FP1, FP2, FP3)
FP <- setNames(FP, c("FP1", "FP2", "FP3"))
BIAS <- c(BIAS1, BIAS2, BIAS3)
MIAE <- c(MIAE1, MIAE2, MIAE3)
MSE <- c(MSE1, MSE2, MSE3)
cp <- c(cp1, cp2, cp3)

output1 <- c(p,n,m,rho_temp,rho_spat,iteration1,
iteration2,iteration3,khat_1,khat_3)
output1 <- setNames(output1, c("p","n","m","rho_temp",
"rho_spat","iteration1","iteration2","iteration3",
"khat_1","khat_2","khat_3"))
output2 <- c(TP,FP,BIAS,MIAE,MSE,cp)
output <- c(output1,output2)


beta_star <- rbind(beta_star_1,beta_star_3,se_1,se_3)

AICs <- rbind(AIC_1,AIC_3)

Sigma_estimate <- as.matrix(var_covar_betastar_3)

conf_matrices <- rbind(conf_matrix_1, conf_matrix_3)

probs <- rbind(probnew_1, probnew_3)

write.table(output, file = "results_0808.txt",
sep = "\t", row.names = FALSE, col.names =
!file.exists("results_0808.txt"), append = TRUE)
write.table(beta_star, file = "beta_star0808.txt",
sep = "\t", row.names = TRUE, col.names = FALSE,
append = TRUE)
```

```
write.table(AICs, file = "aics0808.txt", sep = "\t",
row.names = TRUE, col.names = FALSE, append = TRUE)
write.table(as.data.frame(as.matrix(Sigma_estimate)),
file = "Sigma_estimate0808.txt", sep = "\t",
row.names = FALSE, col.names = FALSE, append = TRUE)
write.table(conf_matrices, file = "conf_matrices0808
.txt", sep = "\t", row.names = TRUE, col.names = FALSE,
append = TRUE)
write.table(probs, file = "probs0808.txt",
sep = "\t", row.names = TRUE, col.names = FALSE,
append = TRUE)

gc() #free up memory and report the memory usage.

list()

}

stopCluster(cl)
```

APPENDIX B

DETAILED RESULTS OF DIFFERENT CORRELATION STRUCTURES

B.1 The case of $\rho_{spat} = 0.8, \rho_{temp} = 0.6$

As presented in Table 4.1, the results in this scenario is as follows:

Table B.1: Variable selection results - $\rho_{spat} = 0.8, \rho_{temp} = 0.6$

Case			**p=29**								
			CS			MS			Size		
	ρ_{temp}	ρ_{spat}	Step I	Step II	Step III	Step I	Step II	Step III	Step I	Step II	Step III
II	0.8	0.6	6	3	6	3	0	2	9	3	8

The contribution of final covariates that are identified as significant is illustrated using SHAP values as explained above. Figure B.1 and Figure B.2 represent SHAP values for the 4^{th} observation, whose observed response is 0, and for the 598^{th} observation, for whom the observed response is 1, respectively.

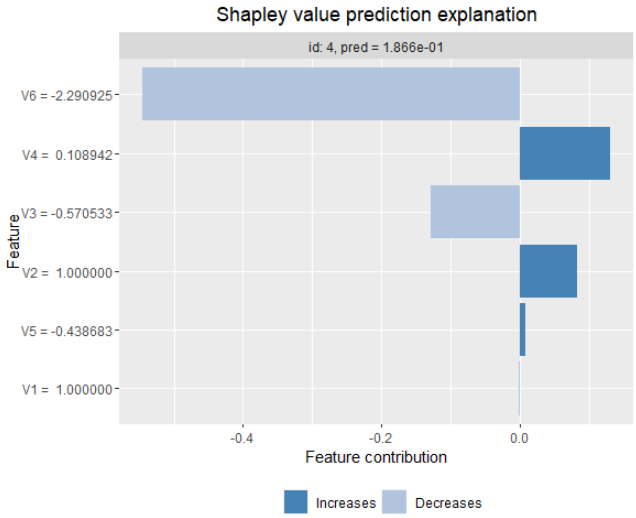


Figure B.1: Contribution of Each Covariate for Class 0 - $\rho_{spat} = 0.8, \rho_{temp} = 0.6$

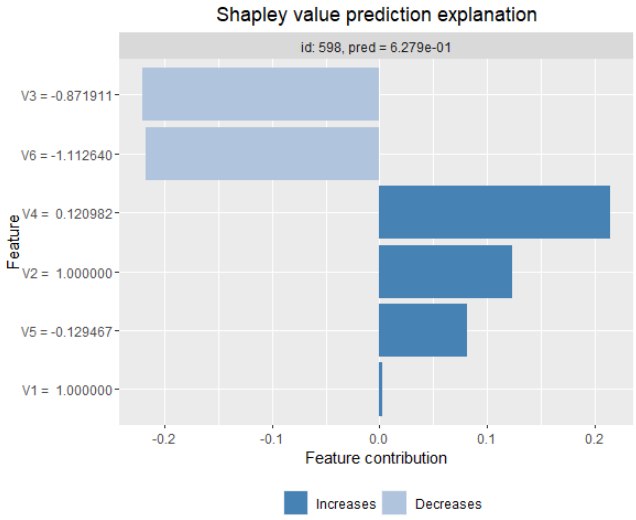


Figure B.2: Contribution of Each Covariate for Class 1 - $\rho_{spat} = 0.8, \rho_{temp} = 0.6$

The loss function results at the end of each iteration and in each step is presented as AIC values in Figure B.3. It can be seen that AICs after each iteration and step gets smaller.

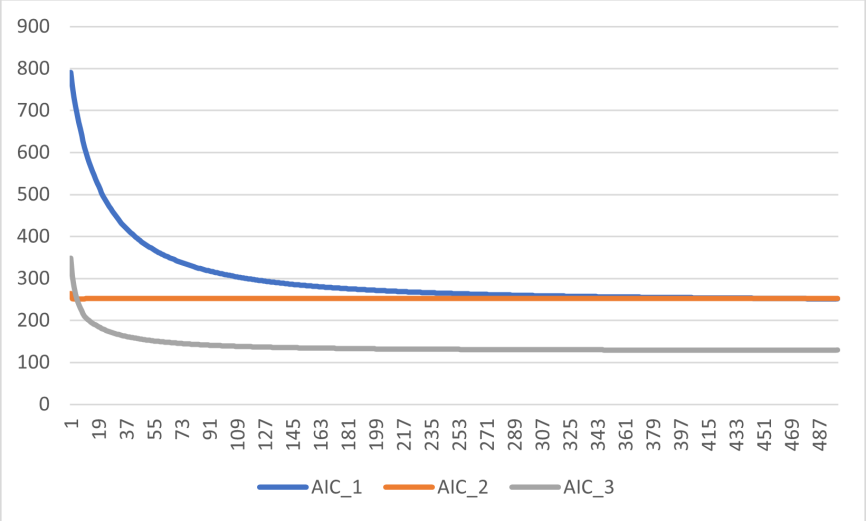


Figure B.3: AIC's in Each Step and Iteration - $\rho_{spat} = 0.8, \rho_{temp} = 0.6$

B.2 The case of $\rho_{spat} = 0.8, \rho_{temp} = 0.2$

As presented in Table 4.1, the results in this scenario is as follows:

Table B.2: Variable selection results - $\rho_{spat} = 0.8, \rho_{temp} = 0.2$

Case			**p=29**								
			CS			MS			Size		
	ρ_{temp}	ρ_{spat}	Step I	Step II	Step III	Step I	Step II	Step III	Step I	Step II	Step III
III	0.8	0.2	6	6	6	3	0	1	9	6	7

The contribution of final covariates that are identified as significant is illustrated using SHAP values as explained above. Figure B.4 and Figure B.5 represent SHAP values for the 3^{rd} observation, whose observed response is 0, and for the 600^{th} observation, for whom the observed response is 1, respectively.

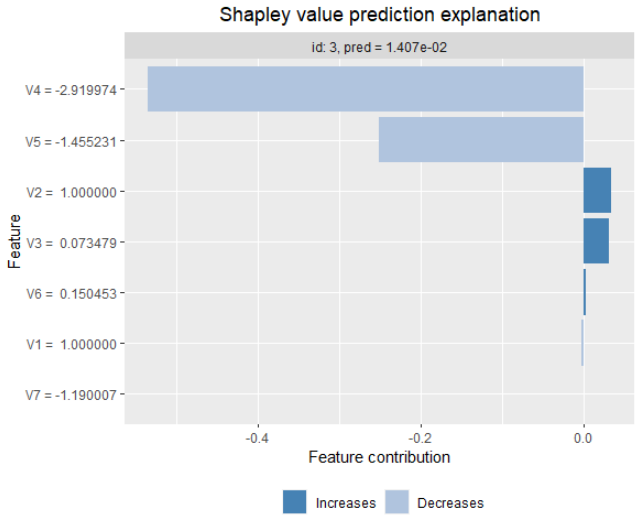


Figure B.4: Contribution of Each Covariate for Class 0 - $\rho_{spat} = 0.8, \rho_{temp} = 0.2$

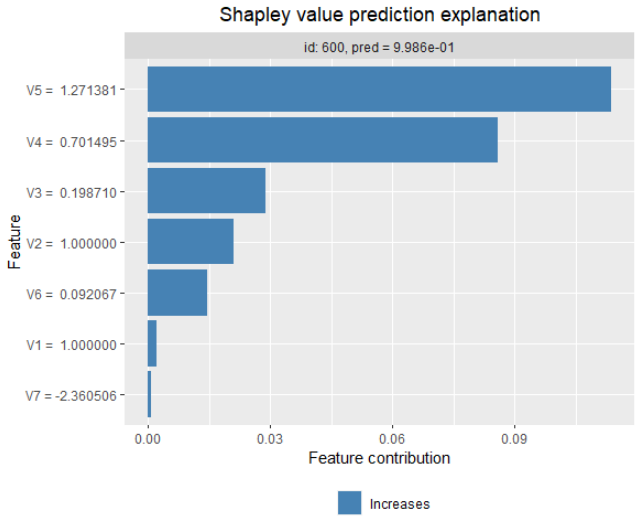


Figure B.5: Contribution of Each Covariate for Class 1 - $\rho_{spat} = 0.8, \rho_{temp} = 0.2$

The loss function results at the end of each iteration and in each step is presented as AIC values in Figure B.6. It can be seen that AICs after each iteration and step gets smaller.

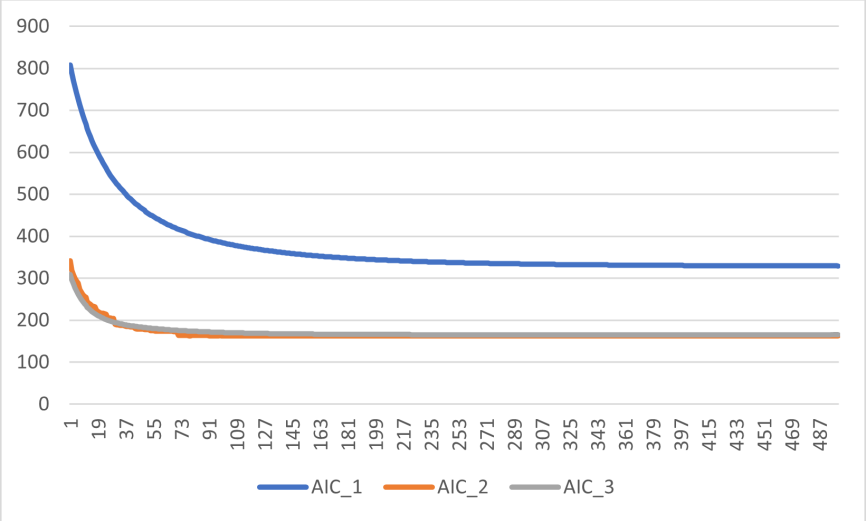


Figure B.6: AIC's in Each Step and Iteration - $\rho_{spat} = 0.8, \rho_{temp} = 0.2$

B.3 The case of $\rho_{spat} = 0.6, \rho_{temp} = 0.8$

As presented in Table 4.1, the results in this scenario is as follows:

Table B.3: Variable selection results - $\rho_{spat} = 0.6, \rho_{temp} = 0.8$

Case			**p=29**								
			CS			MS			Size		
	ρ_{temp}	ρ_{spat}	Step I	Step II	Step III	Step I	Step II	Step III	Step I	Step II	Step III
IV	0.6	0.8	6	2	6	2	0	1	8	2	7

The contribution of final covariates that are identified as significant is illustrated using SHAP values as explained above. Figure B.7 and Figure B.8 represent SHAP values for the 3^{rd} observation, whose observed response is 0, and for the 600^{th} observation, for whom the observed response is 1, respectively.

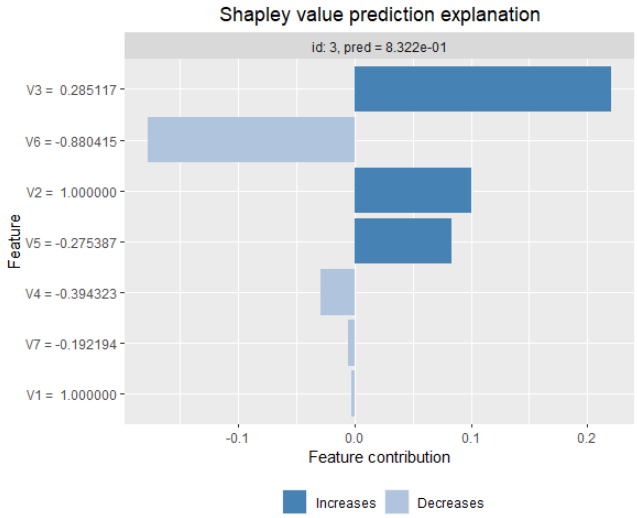


Figure B.7: Contribution of Each Covariate for Class 0 - $\rho_{spat} = 0.6, \rho_{temp} = 0.8$

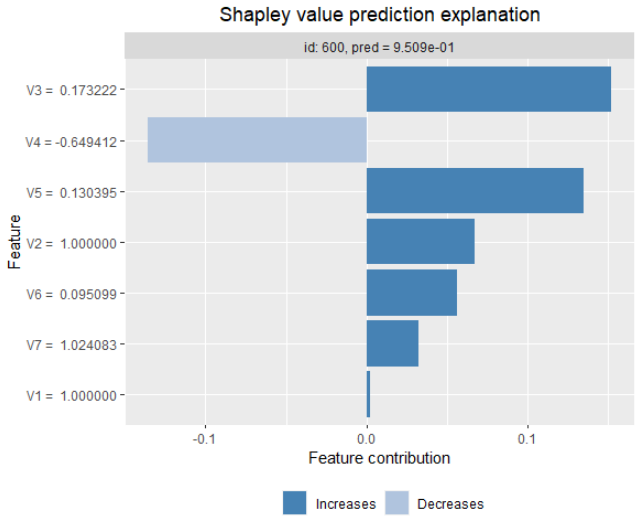


Figure B.8: Contribution of Each Covariate for Class 1 - $\rho_{spat} = 0.6, \rho_{temp} = 0.8$

The loss function results at the end of each iteration and in each step is presented as AIC values in Figure B.9. It can be seen that AICs after each iteration and step gets smaller.

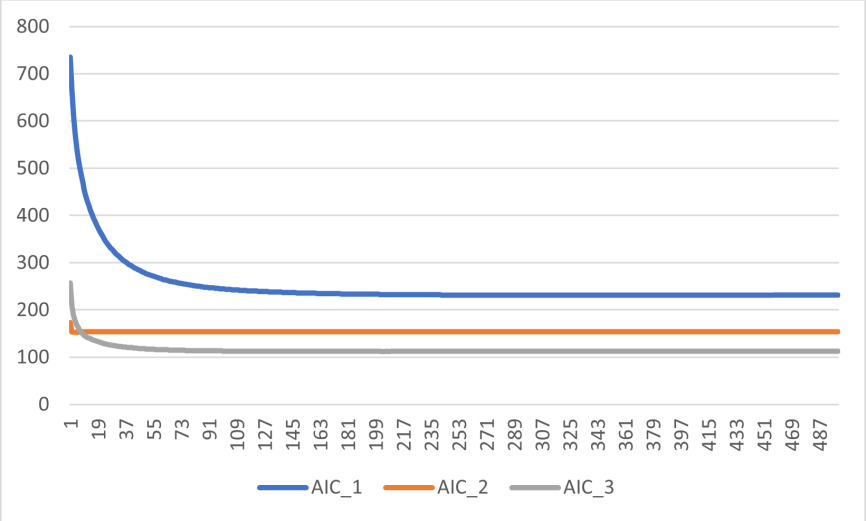


Figure B.9: AIC's in Each Step and Iteration - $\rho_{spat} = 0.6, \rho_{temp} = 0.8$

B.4 The case of $\rho_{spat} = 0.5, \rho_{temp} = 0.5$

As presented in Table 4.1, the results in this scenario is as follows:

Table B.4: Variable selection results - $\rho_{spat} = 0.5, \rho_{temp} = 0.5$

Case			**p=29**								
			CS			MS			Size		
	ρ_{temp}	ρ_{spat}	Step I	Step II	Step III	Step I	Step II	Step III	Step I	Step II	Step III
V	0.5	0.5	6	3	6	3	0	1	9	3	7

The contribution of final covariates that are identified as significant is illustrated using SHAP values as explained above. Figure B.10 and Figure B.11 represent SHAP values for the 2^{nd} observation, whose observed response is 0, and for the 594^{th} observation, for whom the observed response is 1, respectively.

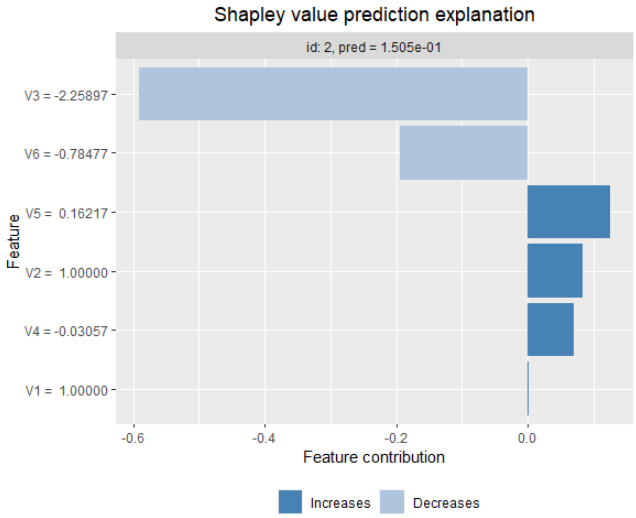


Figure B.10: Contribution of Each Covariate for Class 0 - $\rho_{spat} = 0.5, \rho_{temp} = 0.5$

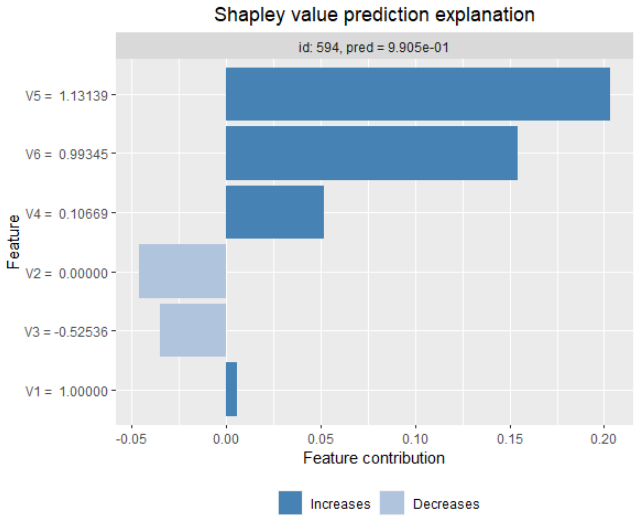


Figure B.11: Contribution of Each Covariate for Class 1 - $\rho_{spat} = 0.5, \rho_{temp} = 0.5$

The loss function results at the end of each iteration and in each step is presented as AIC values in Figure B.12. It can be seen that AICs after each iteration and step gets smaller.

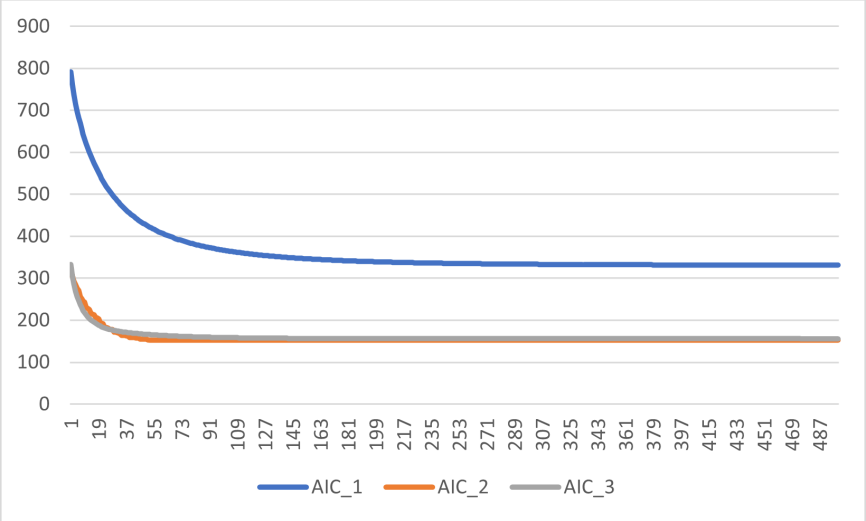


Figure B.12: AIC's in Each Step and Iteration - $\rho_{spat} = 0.5, \rho_{temp} = 0.5$

B.5 The case of $\rho_{spat} = 0.4, \rho_{temp} = 0.2$

As presented in Table 4.1, the results in this scenario is as follows:

Table B.5: Variable selection results - $\rho_{spat} = 0.4, \rho_{temp} = 0.2$

Case			**p=29**								
			CS			MS			Size		
	ρ_{temp}	ρ_{spat}	Step I	Step II	Step III	Step I	Step II	Step III	Step I	Step II	Step III
VI	0.4	0.2	6	6	6	3	0	2	9	6	8

The contribution of final covariates that are identified as significant is illustrated using SHAP values as explained above. Figure B.13 and Figure B.14 represent SHAP values for the 1^{st} observation, whose observed response is 0, and for the 600^{th} observation, for whom the observed response is 1, respectively.

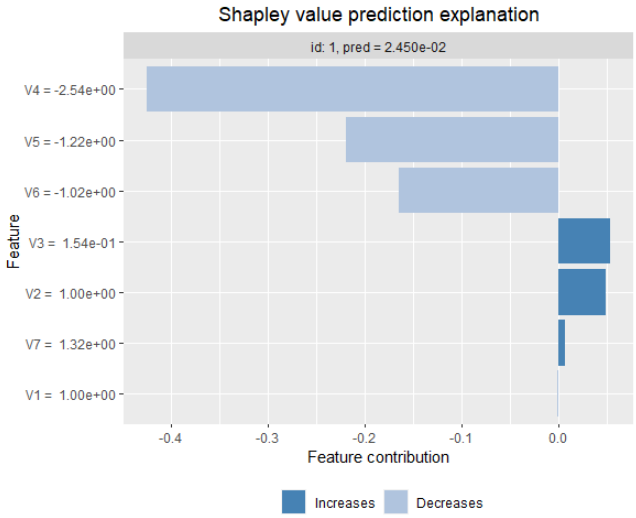


Figure B.13: Contribution of Each Covariate for Class 0 - $\rho_{spat} = 0.4, \rho_{temp} = 0.2$

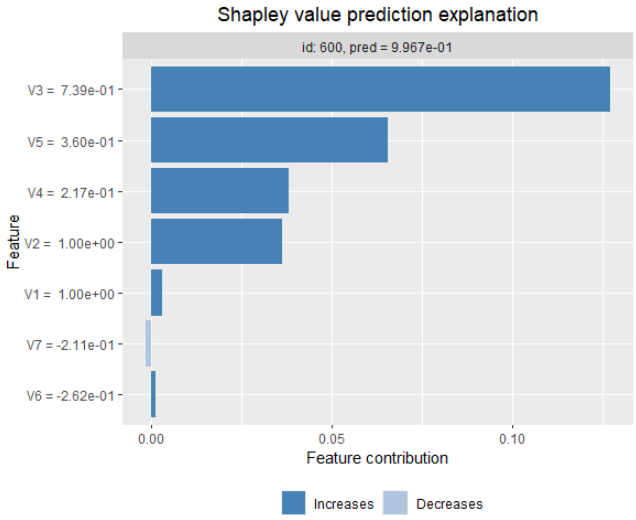


Figure B.14: Contribution of Each Covariate for Class 1 - $\rho_{spat} = 0.4, \rho_{temp} = 0.2$

The loss function results at the end of each iteration and in each step is presented as AIC values in Figure B.15. It can be seen that AICs after each iteration and step gets smaller.

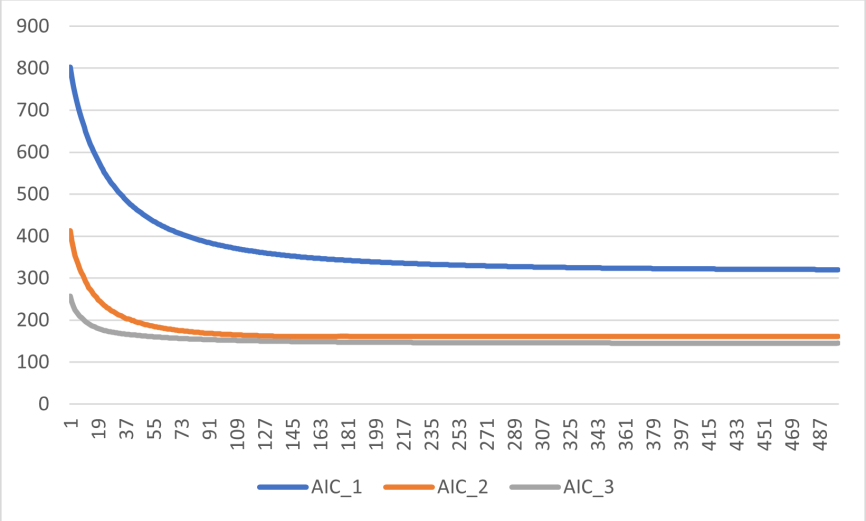


Figure B.15: AIC's in Each Step and Iteration - $\rho_{spat} = 0.4, \rho_{temp} = 0.2$

B.6 The case of $\rho_{spat} = 0.2, \rho_{temp} = 0.8$

As presented in Table 4.1, the results in this scenario is as follows:

Table B.6: Variable selection results - $\rho_{spat} = 0.2, \rho_{temp} = 0.8$

Case			**p=29**								
			CS			MS			Size		
	ρ_{temp}	ρ_{spat}	Step I	Step II	Step III	Step I	Step II	Step III	Step I	Step II	Step III
VII	0.2	0.8	6	2	6	2	0	1	8	2	7

The contribution of final covariates that are identified as significant is illustrated using SHAP values as explained above. Figure B.16 and Figure B.17 represent SHAP values for the 6^{th} observation, whose observed response is 0, and for the 600^{th} observation, for whom the observed response is 1, respectively.

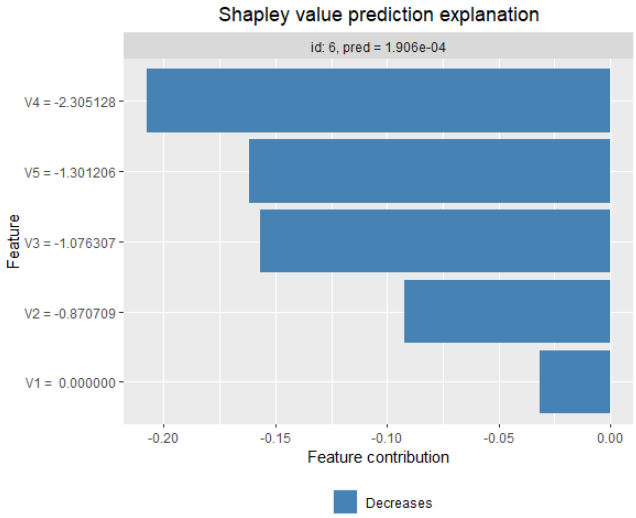


Figure B.16: Contribution of Each Covariate for Class 0 - $\rho_{spat} = 0.2, \rho_{temp} = 0.8$

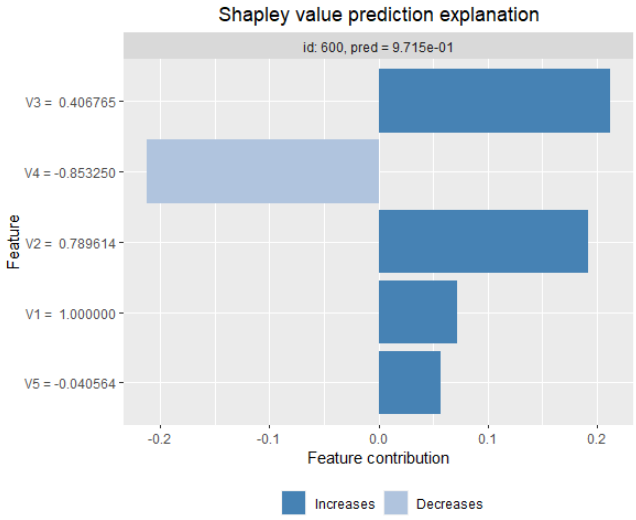


Figure B.17: Contribution of Each Covariate for Class 1 - $\rho_{spat} = 0.2, \rho_{temp} = 0.8$

The loss function results at the end of each iteration and in each step is presented as AIC values in Figure B.18. It can be seen that AICs after each iteration and step gets smaller.

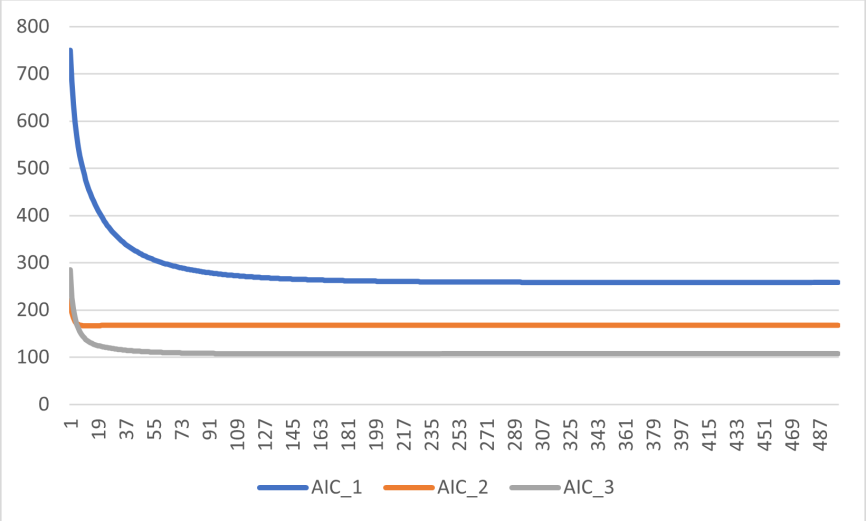


Figure B.18: AIC's in Each Step and Iteration - $\rho_{spat} = 0.2, \rho_{temp} = 0.8$

B.7 The case of $\rho_{spat} = 0.2, \rho_{temp} = 0.4$

As presented in Table 4.1, the results in this scenario is as follows:

Table B.7: Variable selection results - $\rho_{spat} = 0.2, \rho_{temp} = 0.4$

Case			**p=29**								
			CS			MS			Size		
	ρ_{temp}	ρ_{spat}	Step I	Step II	Step III	Step I	Step II	Step III	Step I	Step II	Step III
VIII	0.2	0.4	6	5	6	3	0	1	9	5	7

The contribution of final covariates that are identified as significant is illustrated using SHAP values as explained above. Figure B.19 and Figure B.20 represent SHAP values for the 4^{th} observation, whose observed response is 0, and for the 600^{th} observation, for whom the observed response is 1, respectively.

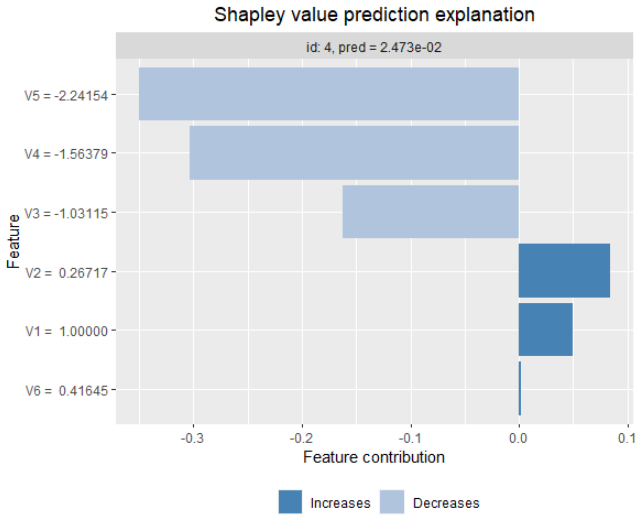


Figure B.19: Contribution of Each Covariate for Class 0 - $\rho_{spat} = 0.2, \rho_{temp} = 0.4$

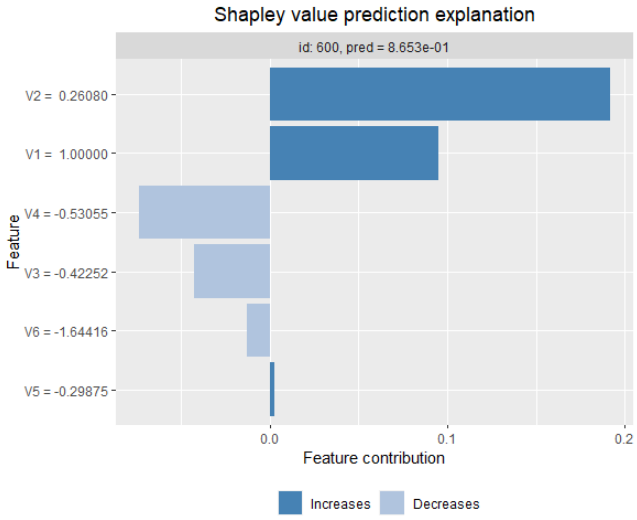


Figure B.20: Contribution of Each Covariate for Class 1 - $\rho_{spat} = 0.2, \rho_{temp} = 0.4$

The loss function results at the end of each iteration and in each step is presented as AIC values in Figure B.21. It can be seen that AICs after each iteration and step gets smaller.

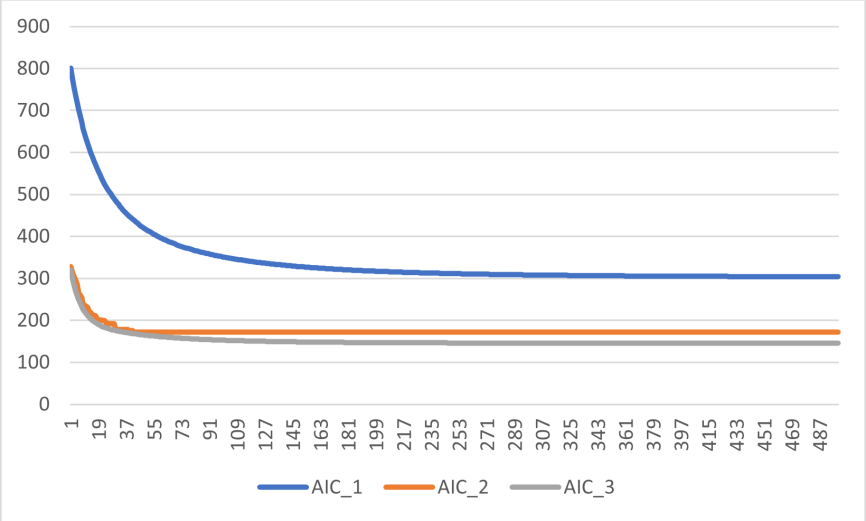


Figure B.21: AIC's in Each Step and Iteration - $\rho_{spat} = 0.2, \rho_{temp} = 0.4$

B.8 The case of $\rho_{spat} = 0.2, \rho_{temp} = 0.2$

As presented in Table 4.1, the results in this scenario is as follows:

Table B.8: Variable selection results - $\rho_{spat} = 0.2, \rho_{temp} = 0.2$

Case			**p=29**								
			CS			MS			Size		
	ρ_{temp}	ρ_{spat}	Step I	Step II	Step III	Step I	Step II	Step III	Step I	Step II	Step III
IX	0.2	0.2	6	6	6	3	0	1	9	6	7

The contribution of final covariates that are identified as significant is illustrated using SHAP values as explained above. Figure B.22 and Figure B.23 represent SHAP values for the 1^{st} observation, whose observed response is 0, and for the 600^{th} observation, for whom the observed response is 1, respectively.

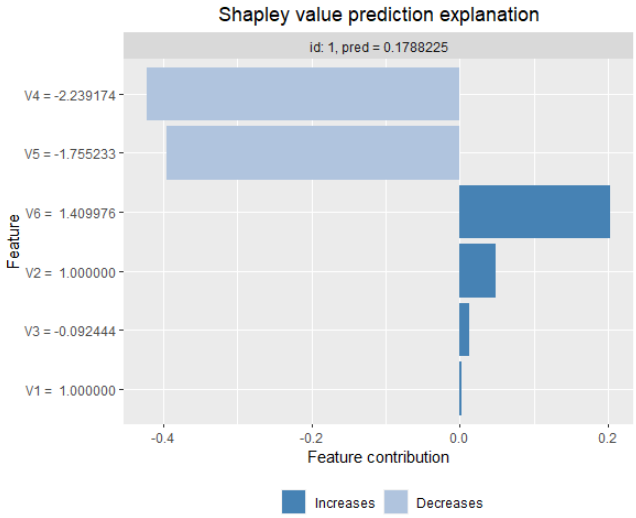


Figure B.22: Contribution of Each Covariate for Class 0 - $\rho_{spat} = 0.2, \rho_{temp} = 0.2$

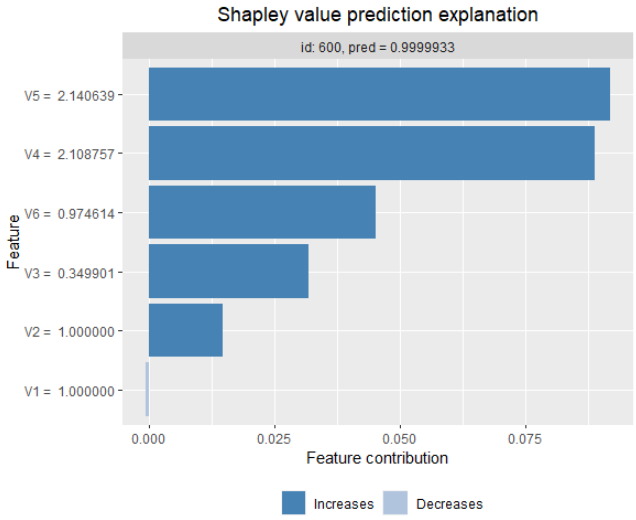


Figure B.23: Contribution of Each Covariate for Class 1 - $\rho_{spat} = 0.2, \rho_{temp} = 0.2$

The loss function results at the end of each iteration and in each step is presented as AIC values in Figure B.24. It can be seen that AICs after each iteration and step gets smaller.

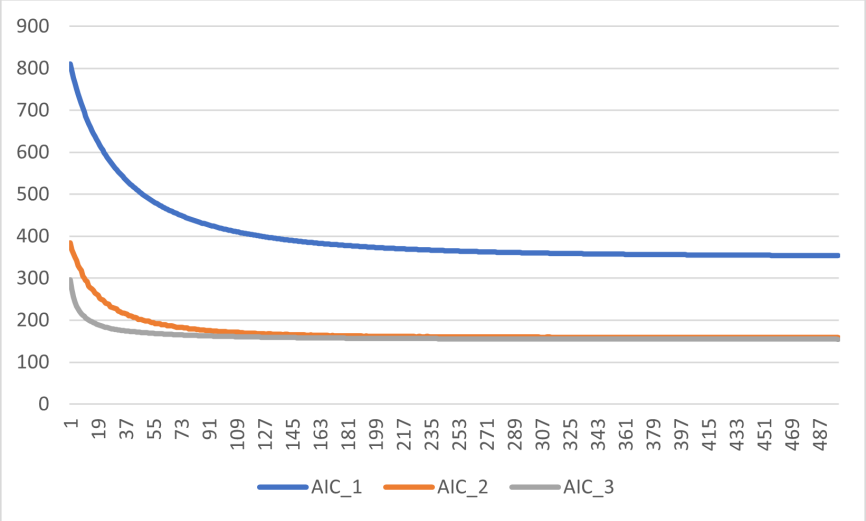


Figure B.24: AIC's in Each Step and Iteration - $\rho_{spat} = 0.2, \rho_{temp} = 0.2$

APPENDIX C

CONFUSION MATRICES OF THREE-STEP SPARSE BOOSTING

C.1 The case of $\rho_{spat} = 0.8, \rho_{temp} = 0.8$

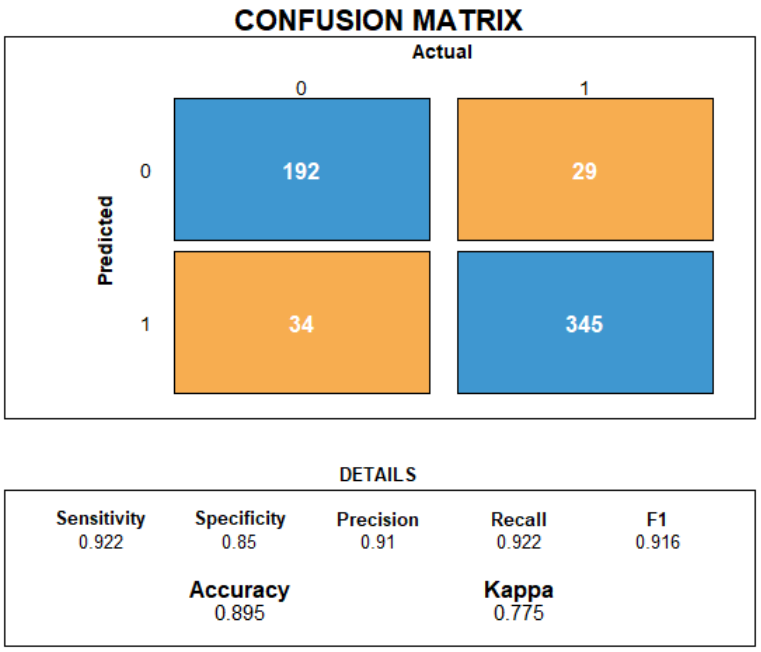


Figure C.1: Confusion Matrix for The First Step - $\rho_{spat} = 0.8, \rho_{temp} = 0.8$

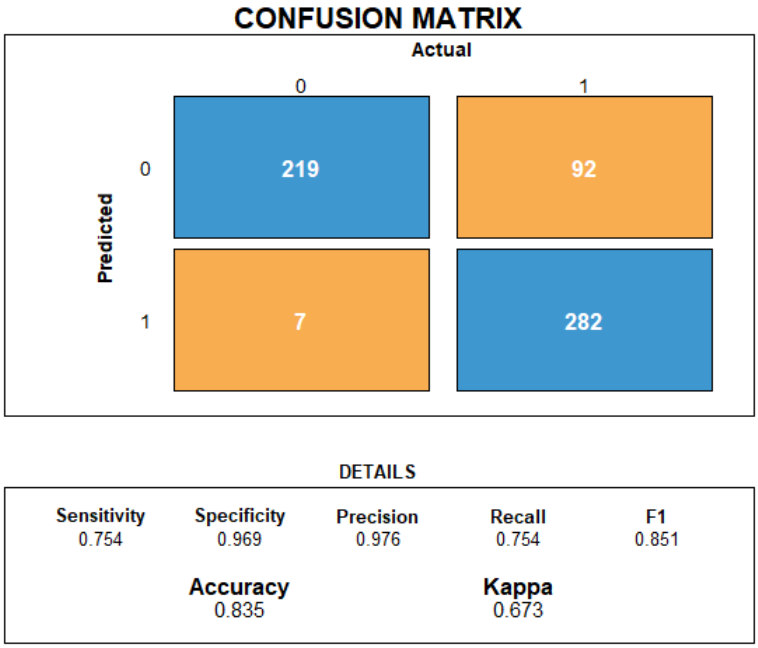


Figure C.2: Confusion Matrix for The Second Step - $\rho_{spat} = 0.8, \rho_{temp} = 0.8$

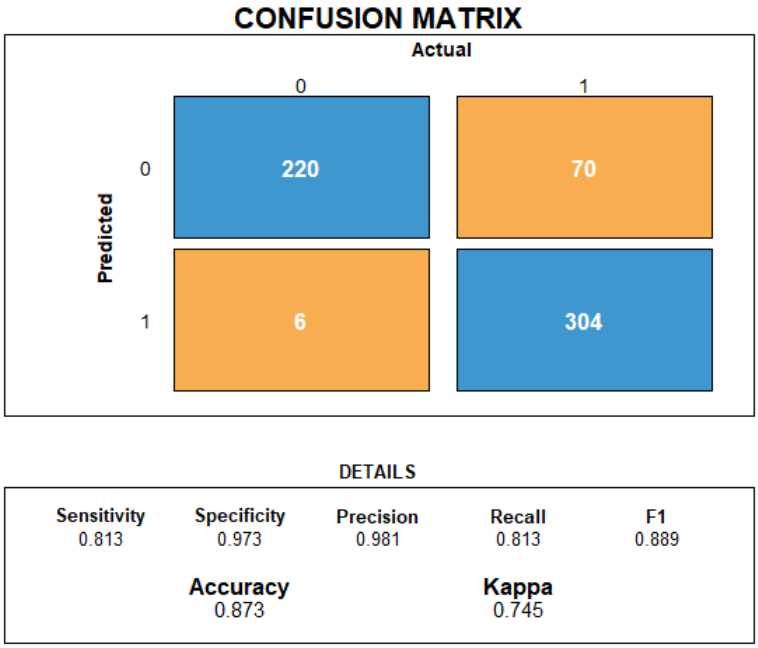


Figure C.3: Confusion Matrix for The Third Step - $\rho_{spat} = 0.8, \rho_{temp} = 0.8$

C.2 The case of $\rho_{spat} = 0.8, \rho_{temp} = 0.6$

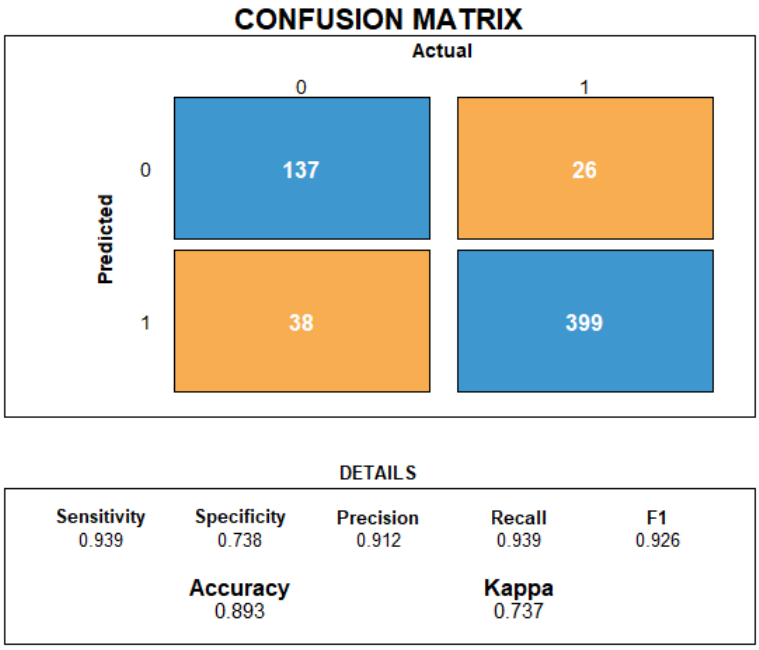


Figure C.4: Confusion Matrix for The First Step - $\rho_{spat} = 0.8, \rho_{temp} = 0.6$

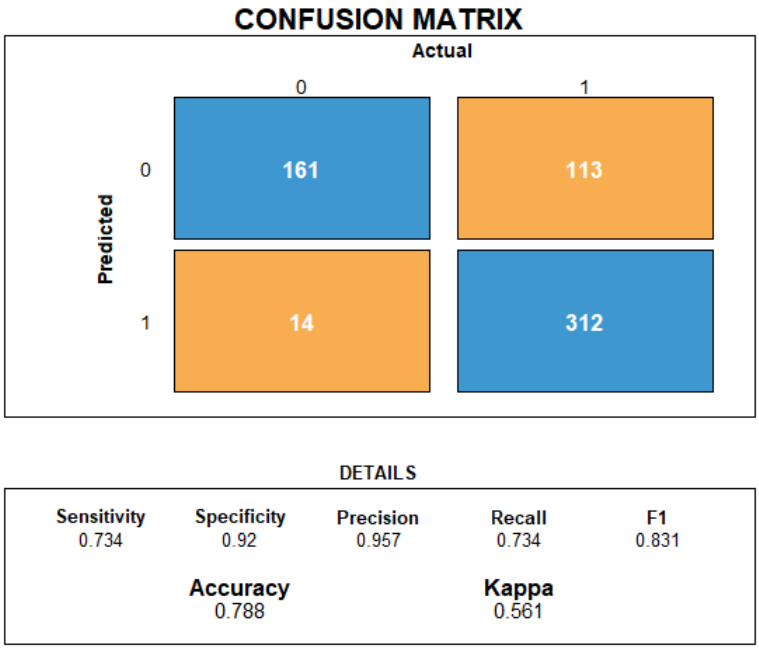


Figure C.5: Confusion Matrix for The Second Step - $\rho_{spat} = 0.8, \rho_{temp} = 0.6$

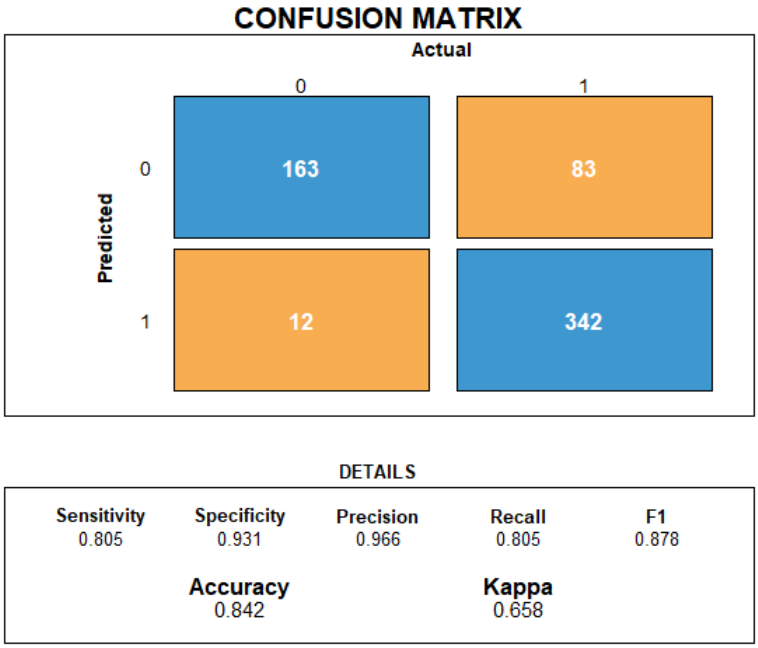


Figure C.6: Confusion Matrix for The Third Step - $\rho_{spat} = 0.8, \rho_{temp} = 0.6$

C.3 The case of $\rho_{spat} = 0.8, \rho_{temp} = 0.2$

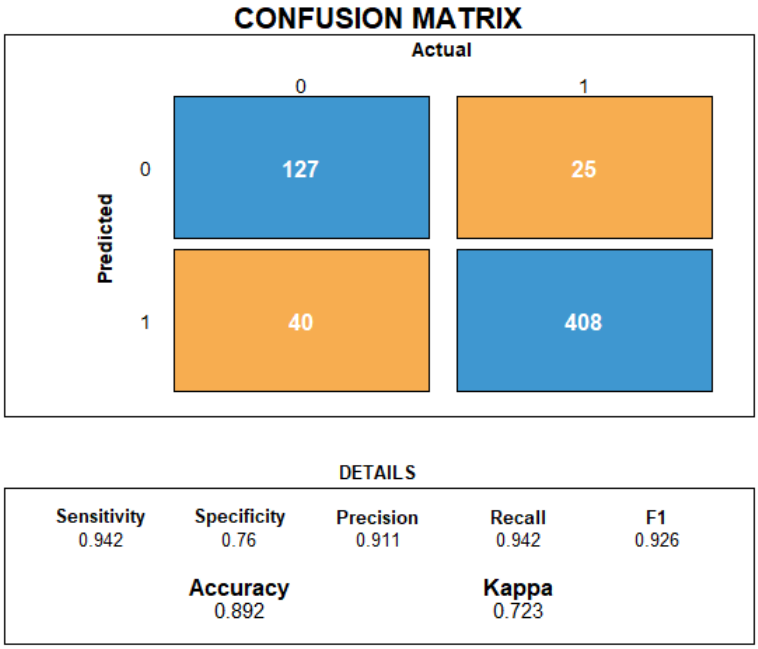


Figure C.7: Confusion Matrix for The First Step - $\rho_{spat} = 0.8, \rho_{temp} = 0.2$

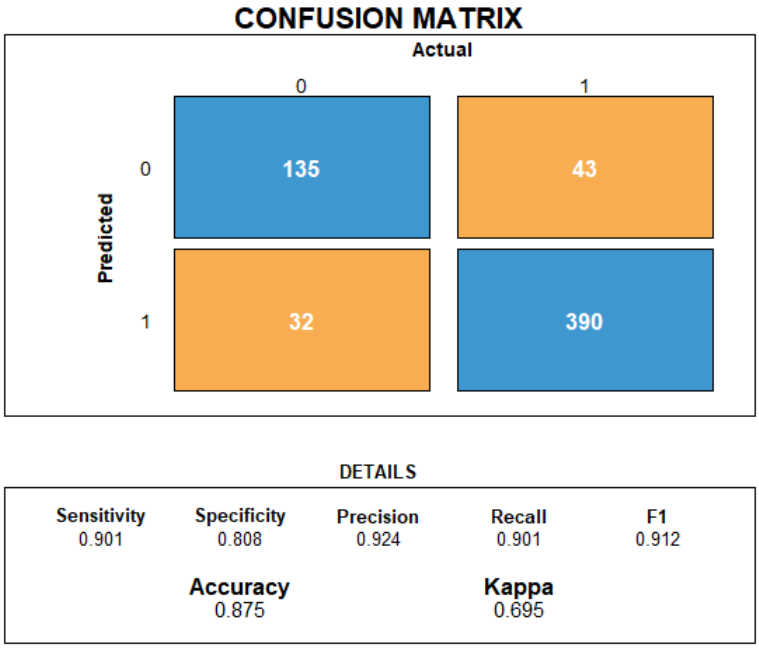


Figure C.8: Confusion Matrix for The Second Step - $\rho_{spat} = 0.8, \rho_{temp} = 0.2$

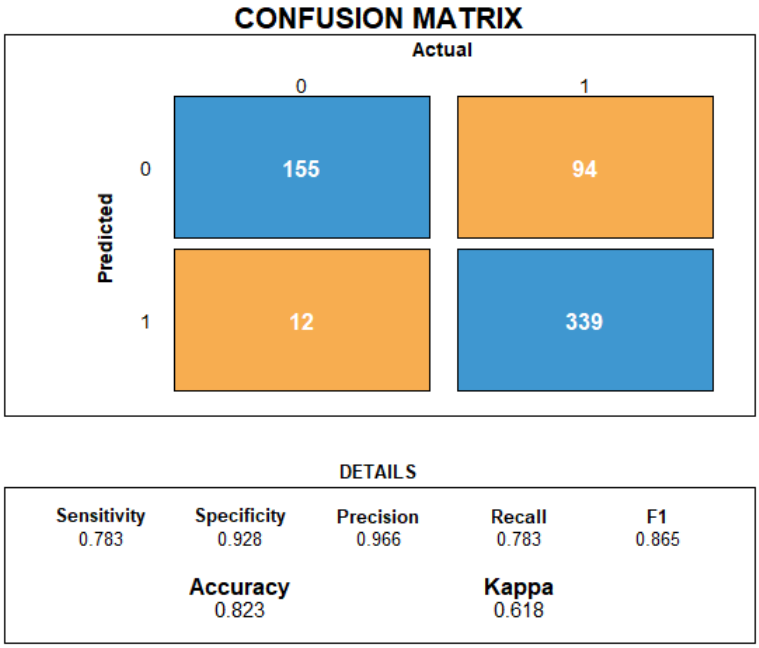


Figure C.9: Confusion Matrix for The Third Step - $\rho_{spat} = 0.8, \rho_{temp} = 0.2$

C.4 The case of $\rho_{spat} = 0.6, \rho_{temp} = 0.8$

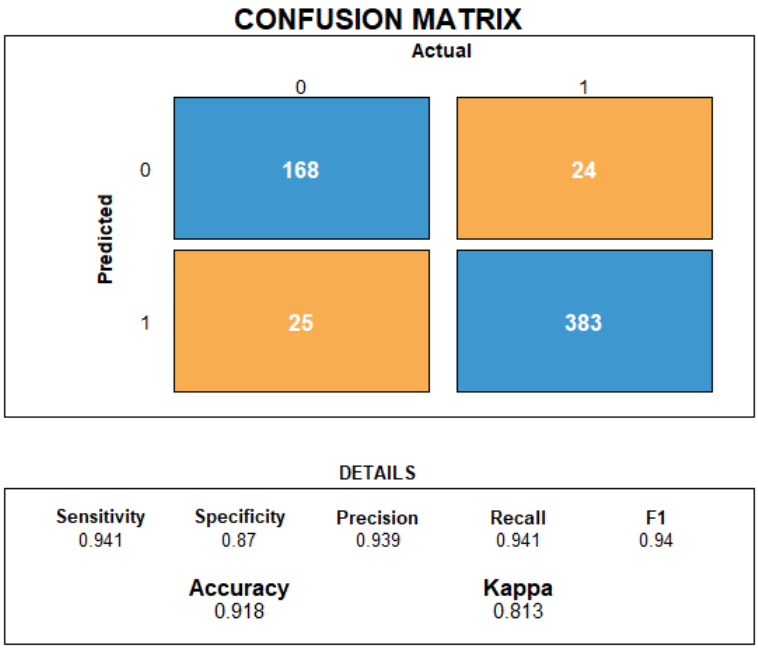


Figure C.10: Confusion Matrix for The First Step - $\rho_{spat} = 0.6, \rho_{temp} = 0.8$

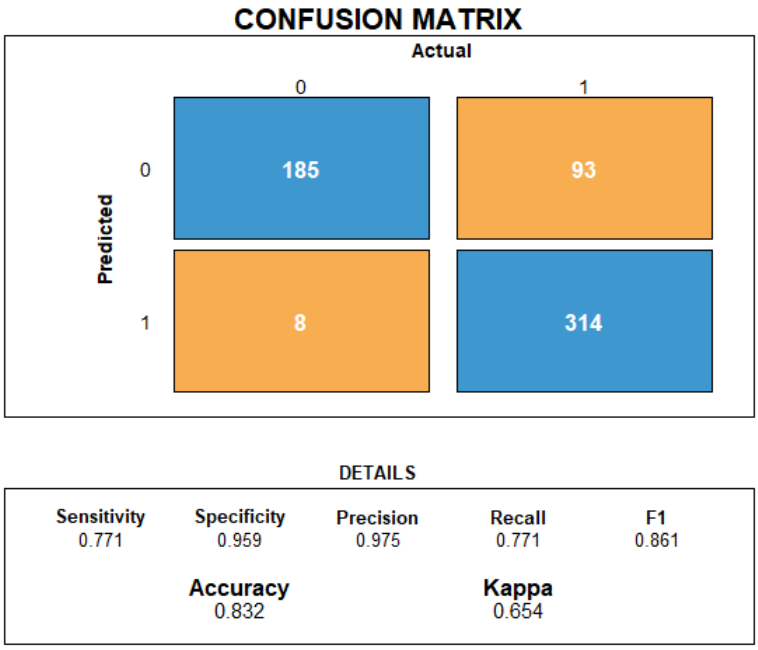


Figure C.11: Confusion Matrix for The Second Step - $\rho_{spat} = 0.6, \rho_{temp} = 0.8$

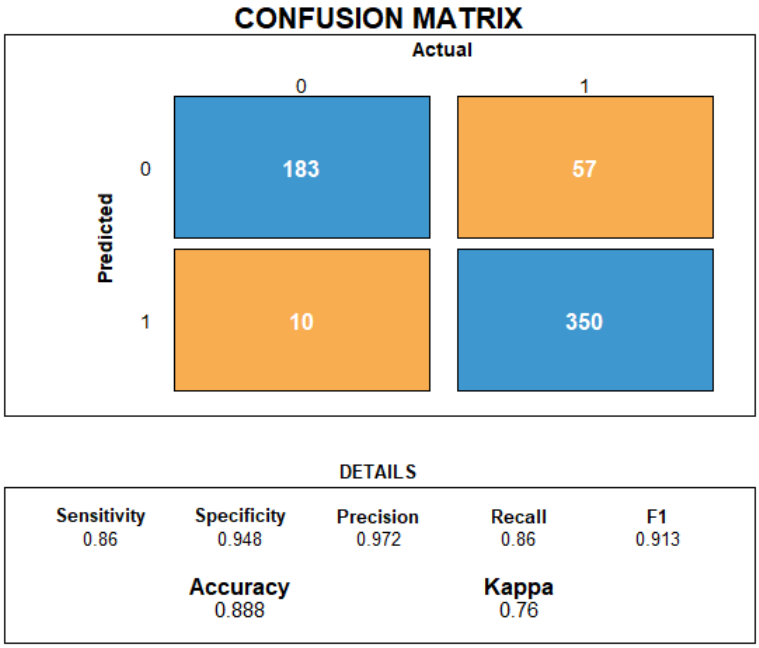


Figure C.12: Confusion Matrix for The Third Step - $\rho_{spat} = 0.6, \rho_{temp} = 0.8$

C.5 The case of $\rho_{spat} = 0.5, \rho_{temp} = 0.5$

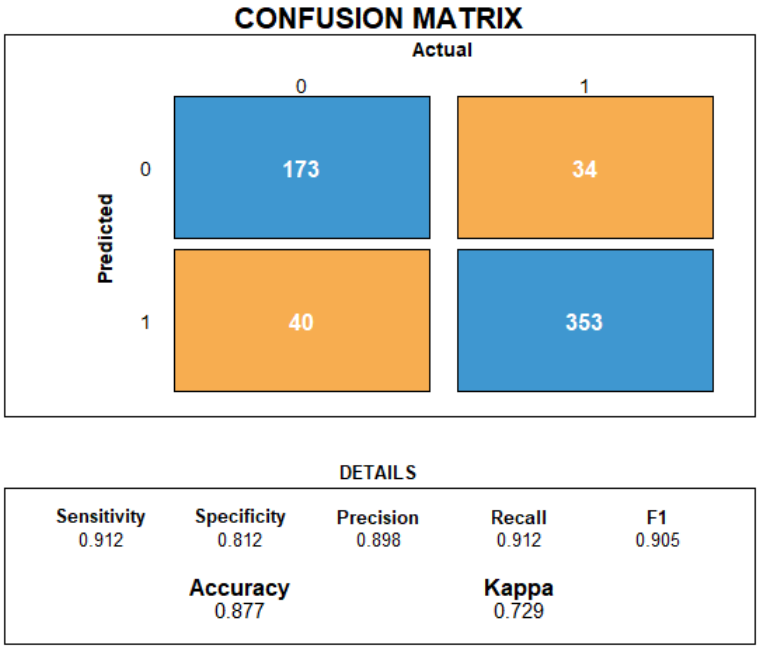


Figure C.13: Confusion Matrix for The First Step - $\rho_{spat} = 0.5, \rho_{temp} = 0.5$

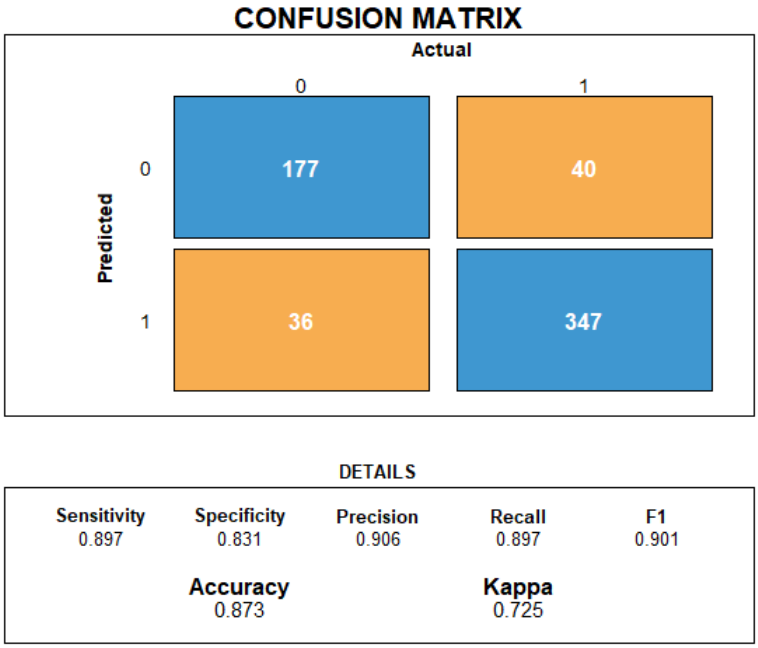


Figure C.14: Confusion Matrix for The Second Step - $\rho_{spat} = 0.5, \rho_{temp} = 0.5$

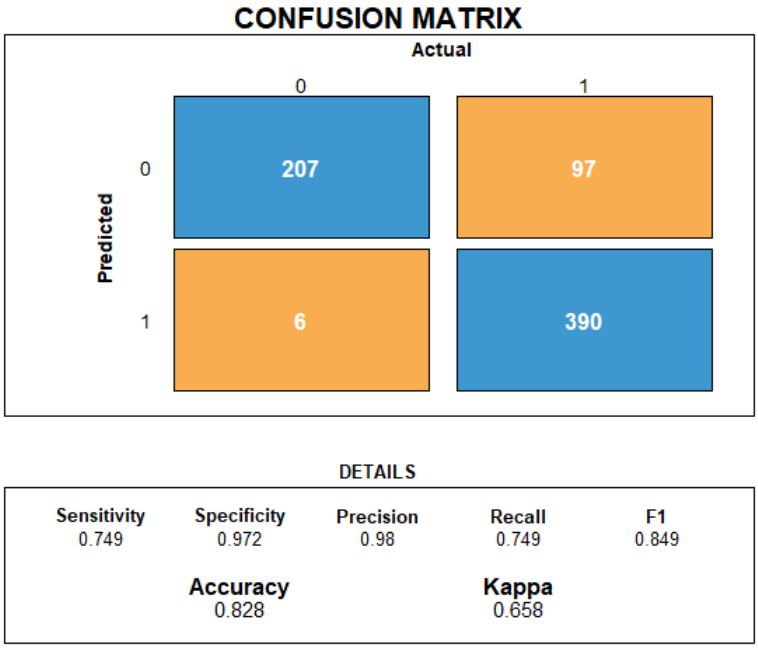


Figure C.15: Confusion Matrix for The Third Step - $\rho_{spat} = 0.5, \rho_{temp} = 0.5$

C.6 The case of $\rho_{spat} = 0.4, \rho_{temp} = 0.2$

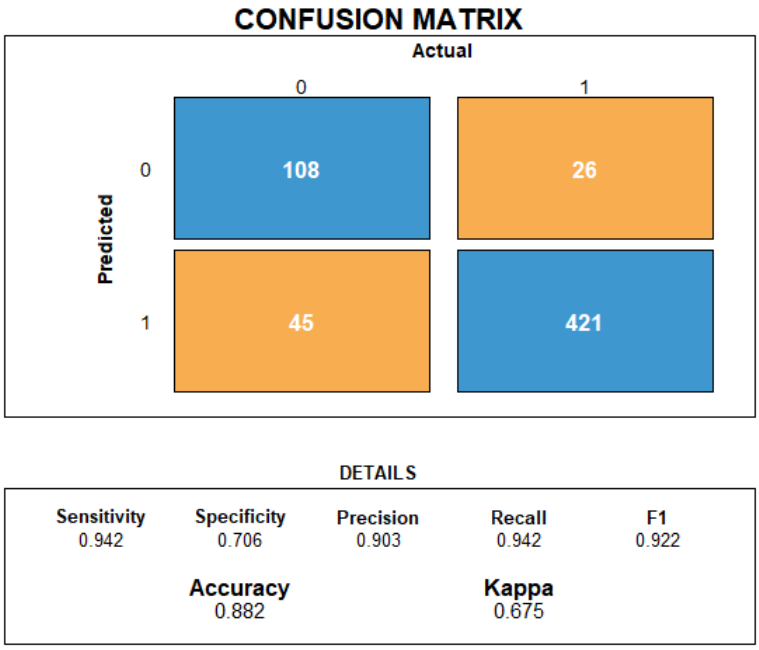


Figure C.16: Confusion Matrix for The First Step - $\rho_{spat} = 0.4, \rho_{temp} = 0.2$

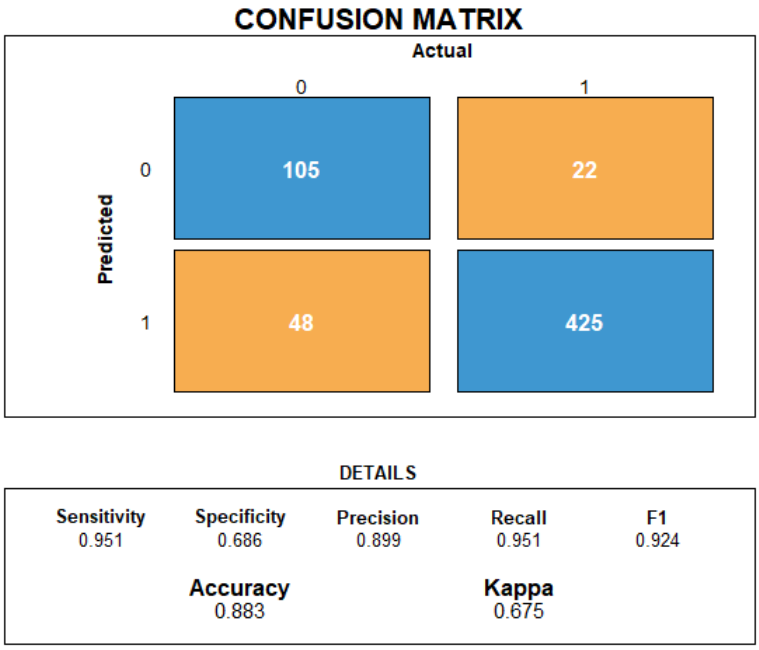


Figure C.17: Confusion Matrix for The Second Step - $\rho_{spat} = 0.4, \rho_{temp} = 0.2$

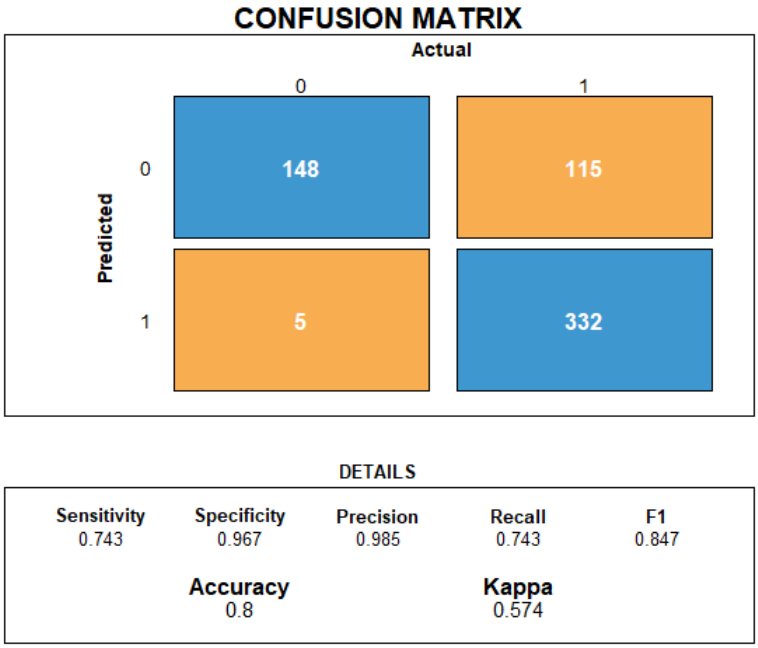


Figure C.18: Confusion Matrix for The Third Step - $\rho_{spat} = 0.4, \rho_{temp} = 0.2$

C.7 The case of $\rho_{spat} = 0.2, \rho_{temp} = 0.8$

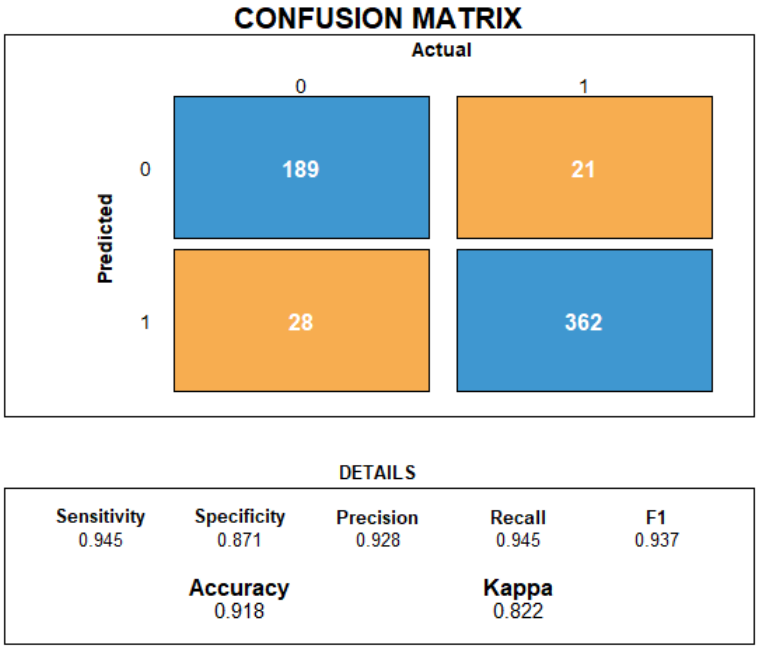


Figure C.19: Confusion Matrix for The First Step - $\rho_{spat} = 0.2, \rho_{temp} = 0.8$

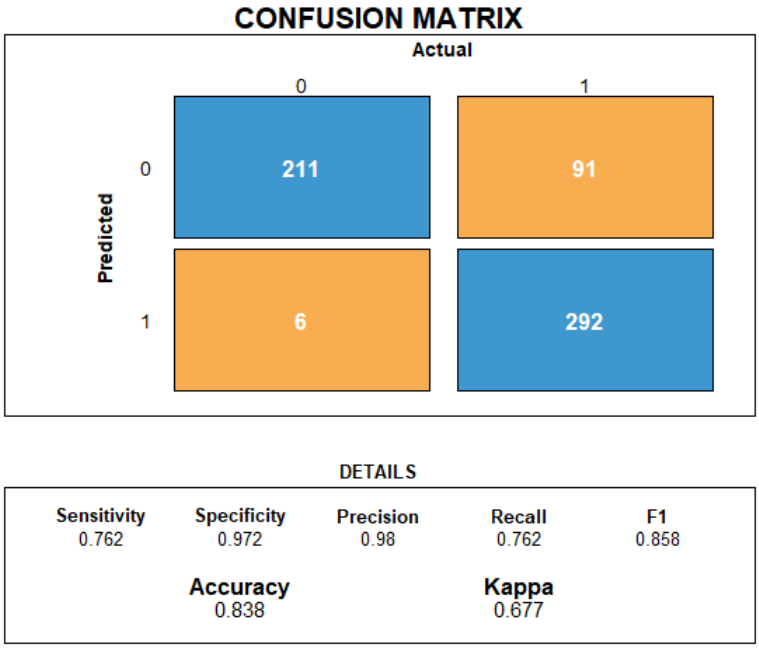


Figure C.20: Confusion Matrix for The Second Step - $\rho_{spat} = 0.2, \rho_{temp} = 0.8$

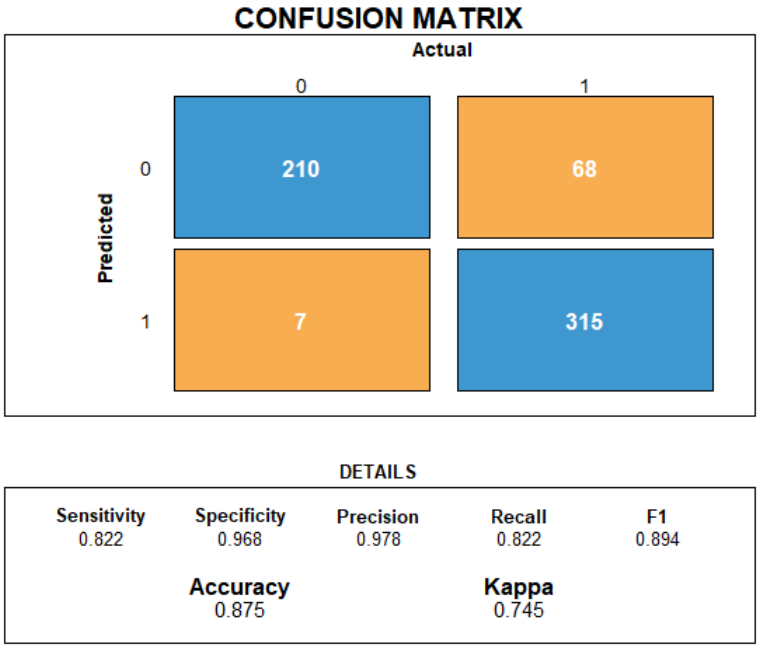


Figure C.21: Confusion Matrix for The Third Step - $\rho_{spat} = 0.2, \rho_{temp} = 0.8$

C.8 The case of $\rho_{spat} = 0.2, \rho_{temp} = 0.4$

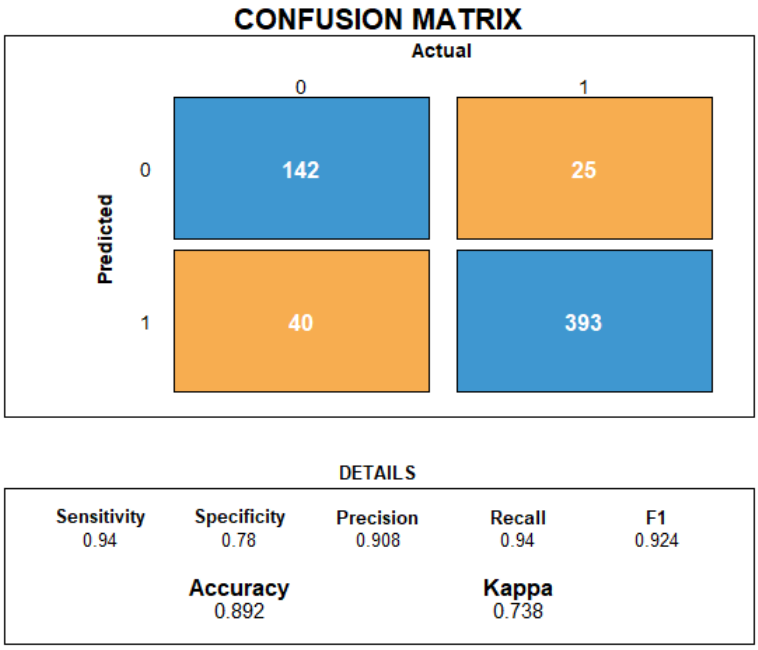


Figure C.22: Confusion Matrix for The First Step - $\rho_{spat} = 0.2, \rho_{temp} = 0.4$

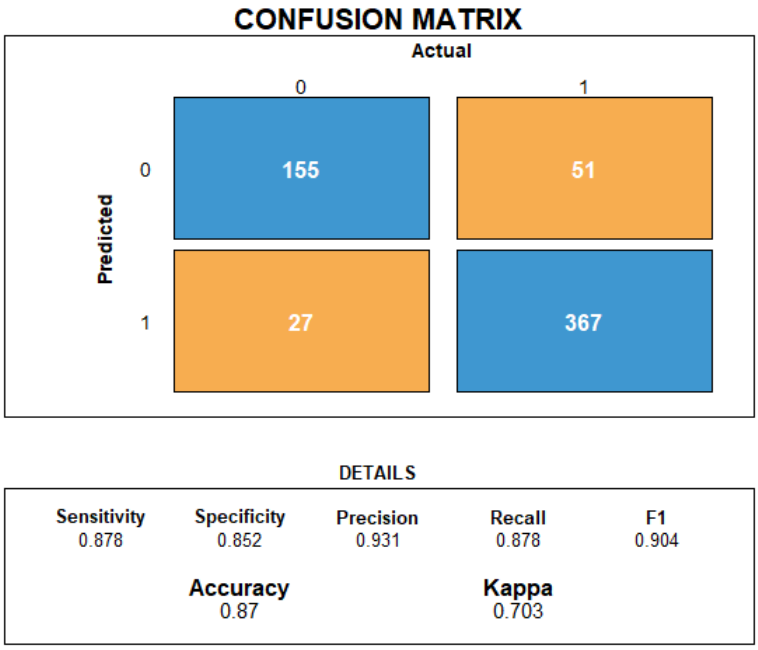


Figure C.23: Confusion Matrix for The Second Step - $\rho_{spat} = 0.2, \rho_{temp} = 0.4$

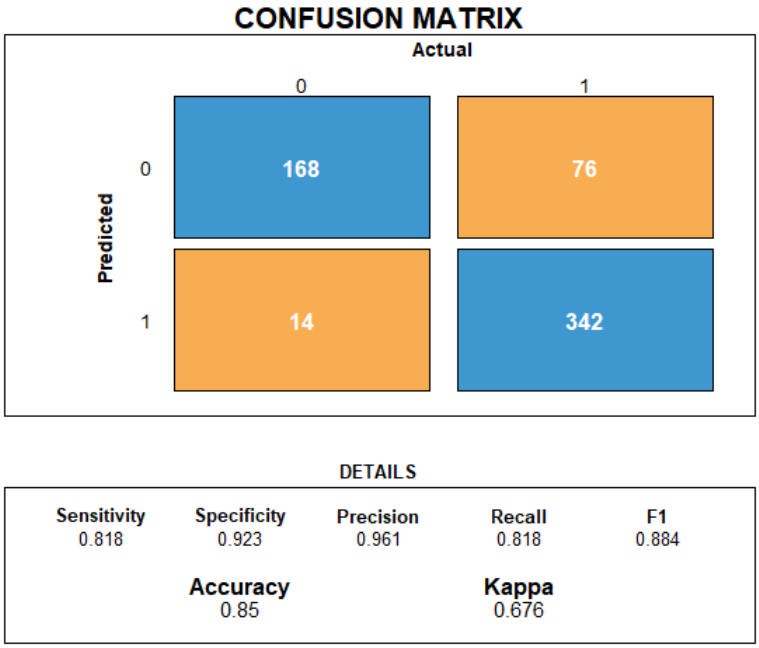


Figure C.24: Confusion Matrix for The Third Step - $\rho_{spat} = 0.2, \rho_{temp} = 0.4$

C.9 The case of $\rho_{spat} = 0.2, \rho_{temp} = 0.2$

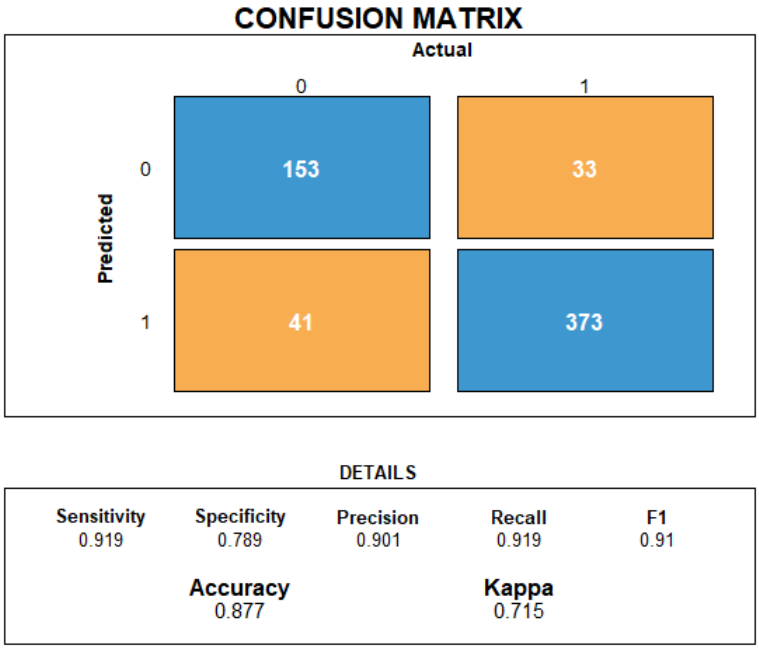


Figure C.25: Confusion Matrix for The First Step - $\rho_{spat} = 0.2, \rho_{temp} = 0.2$

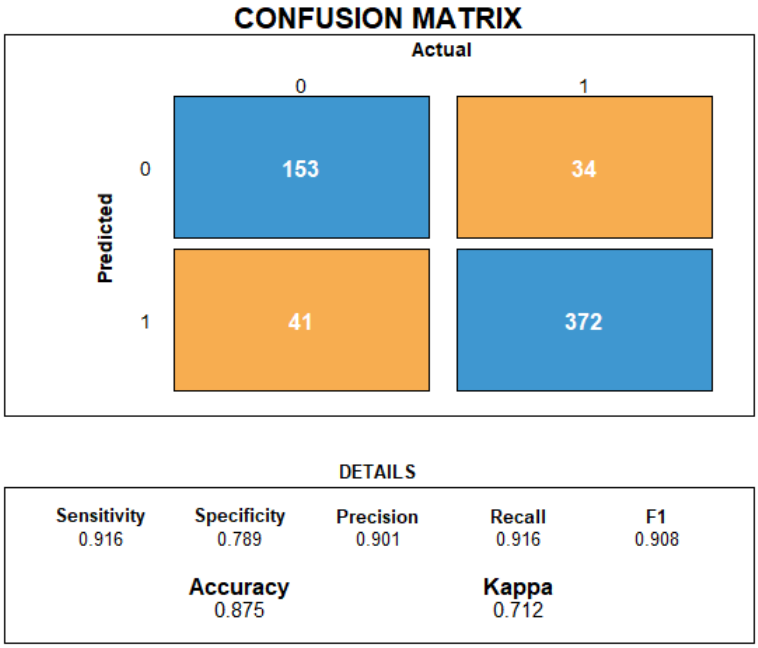


Figure C.26: Confusion Matrix for The Second Step - $\rho_{spat} = 0.2, \rho_{temp} = 0.2$

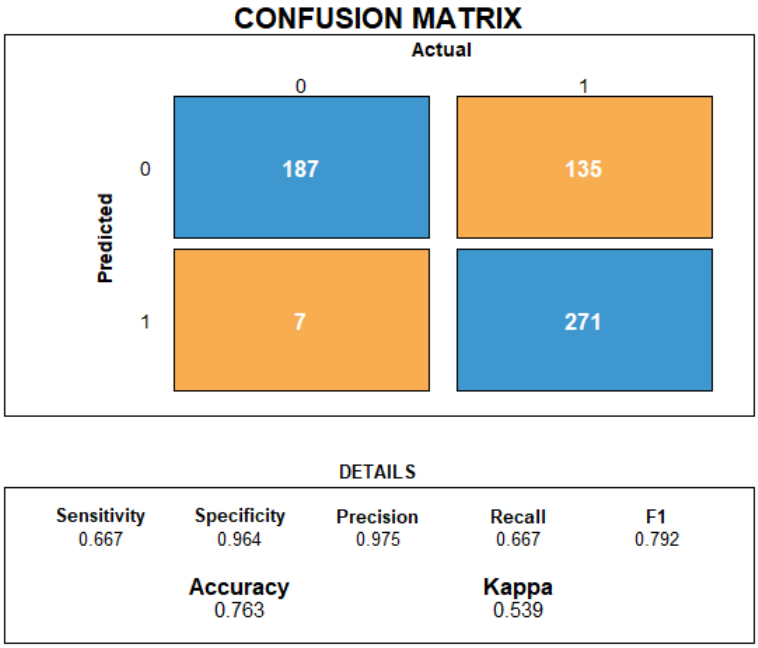


Figure C.27: Confusion Matrix for The Third Step - $\rho_{spat} = 0.2, \rho_{temp} = 0.2$

APPENDIX D

ADDITIONAL SIMULATION STUDY - 2 STEP SPARSE BOOSTING

After analyzing the results of simulation study, due to lower results of Step 2 compared to the other steps, an additional simulation study was conducted omitting the Step 2. In this way, the contribution of Step 2 can be measured. The same structure of the data generation, of setup and of 9 different spatial and temporal correlation structure scenarios were applied. The only difference was the maximum number of boosting iterations, set as 100, since we did not observe any significant change in AIC after 100^{th} iterations in each step. In this setup, total completion time of 100 runs of Monte Carlo Simulation was reduced to 17 hours from 160 hours, due to omitting one step and reducing the number of iteration to 100 in each step.

In the following subsections, first variable selection results are summarized with performance measures. Then, for each different correlation structure, the comparison of loss functions' results with Three Step algorithm are presented separately. After that, the classification performances of our algorithm are discussed.

D.1 Variable Selection Results

In Table D.1 the performance of variable selection is summarized with correctly identified significant variables (CS), falsely identified significant variable (MS).

Table D.1: Variable selection results in simulation for different correlation structures

Case			**p=29**			
			CS		MS	
	ρ_{temp}	ρ_{spat}	Step I	Step III	Step I	Step III
I	0.8	0.8	6	6	0	0
II	0.8	0.6	6	6	0	0
III	0.8	0.2	6	6	0	0
IV	0.6	0.8	6	5	0	0
V	0.5	0.5	6	6	0	0
VI	0.4	0.2	6	6	0	0
VII	0.2	0.8	6	6	0	0
VIII	0.2	0.4	6	6	0	0
IX	0.2	0.2	6	6	0	0

When we analyze the results of the above nine scenarios, the followings can be observed.

- There exists only one scenario, $\rho_{temp} = 0.6$ and $\rho_{spat} = 0.8$, where 2 step algorithm did not identify all the significant variables even thought the first step captured all of them.
- In other scenarios, there is no improvement from Step 1 to Step 3.
- The major difference from 3 step algorithm is that 2 step algorithm did not identify any covariate as mistakenly significant.

In Tables D.2-D.7, the bias, absolute bias and MSE for each estimated coefficients are presented.

Table D.2: Bias, Absolute Bias and MSE for $\widehat{\beta_0}$ - 2 Step Algorithm

Case			**p=29**					
			BIAS($\widehat{\beta_0}$)		AB($\widehat{\beta_0}$)		MSE($\widehat{\beta_0}$)	
	ρ_{temp}	ρ_{spat}	Step I	Step III	Step I	Step III	Step I	Step III
I	0.8	0.8	0.549	1.609	0.553	1.609	0.342	2.622
II	0.8	0.6	0.638	1.594	0.638	1.594	0.433	2.556
III	0.8	0.2	0.708	1.684	0.708	1.684	0.525	2.847
IV	0.6	0.8	0.543	1.601	0.543	1.601	0.321	2.587
V	0.5	0.5	0.644	1.584	0.644	1.584	0.432	2.519
VI	0.4	0.2	0.713	1.695	0.713	1.695	0.522	2.882
VII	0.2	0.8	0.551	1.614	0.551	1.614	0.330	2.636
VIII	0.2	0.4	0.693	1.651	0.693	1.651	0.499	2.742
IX	0.2	0.2	0.703	1.696	0.703	1.696	0.507	2.886

Table D.3: Bias, Absolute Bias and MSE for $\widehat{\beta_1}$ - 2 Step Algorithm

Case			**p=29**					
			BIAS($\widehat{\beta_1}$)		AB($\widehat{\beta_1}$)		MSE($\widehat{\beta_1}$)	
	ρ_{temp}	ρ_{spat}	Step I	Step III	Step I	Step III	Step I	Step III
I	0.8	0.8	0.680	0.730	0.680	0.730	0.544	0.612
II	0.8	0.6	0.727	0.748	0.727	0.748	0.582	0.595
III	0.8	0.2	0.813	0.869	0.813	0.869	0.707	0.781
IV	0.6	0.8	0.629	0.624	0.630	0.626	0.459	0.461
V	0.5	0.5	0.797	0.814	0.797	0.814	0.667	0.693
VI	0.4	0.2	0.808	0.850	0.808	0.850	0.684	0.744
VI	0.2	0.8	0.633	0.654	0.637	0.655	0.467	0.509
VIII	0.2	0.4	0.795	0.818	0.795	0.818	0.667	0.693
IX	0.2	0.2	0.828	0.874	0.828	0.874	0.718	0.785

Table D.4: Bias, Absolute Bias and MSE for $\widehat{\beta}_2$ - 2 Step Algorithm

Case			**p=29**					
			BIAS($\widehat{\beta}_2$)		AB($\widehat{\beta}_2$)		MSE($\widehat{\beta}_2$)	
	ρ_{temp}	ρ_{spat}	Step I	Step III	Step I	Step III	Step I	Step III
I	0.8	0.8	0.647	0.309	0.648	0.374	0.465	0.197
II	0.8	0.6	0.746	0.419	0.746	0.427	0.578	0.219
III	0.8	0.2	0.927	0.679	0.927	0.679	0.869	0.474
IV	0.6	0.8	0.660	0.304	0.660	0.362	0.479	0.181
V	0.5	0.5	0.830	0.516	0.830	0.516	0.703	0.297
VI	0.4	0.2	0.907	0.649	0.907	0.649	0.832	0.434
VII	0.2	0.8	0.651	0.304	0.651	0.380	0.473	0.221
VIII	0.2	0.4	0.836	0.550	0.836	0.550	0.712	0.323
IX	0.2	0.2	0.893	0.634	0.893	0.634	0.808	0.418

Table D.5: Bias, Absolute Bias and MSE for $\widehat{\beta}_3$ - 2 Step Algorithm

Case			**p=29**					
			BIAS($\widehat{\beta}_3$)		AB($\widehat{\beta}_3$)		MSE($\widehat{\beta}_3$)	
	ρ_{temp}	ρ_{spat}	Step I	Step III	Step I	Step III	Step I	Step III
I	0.8	0.8	0.398	0.383	0.424	0.492	0.236	0.324
II	0.8	0.6	0.521	0.431	0.521	0.435	0.296	0.243
III	0.8	0.2	0.693	0.506	0.693	0.506	0.491	0.272
IV	0.6	0.8	0.354	0.337	0.382	0.414	0.214	0.260
V	0.5	0.5	0.537	0.392	0.537	0.401	0.308	0.195
VI	0.4	0.2	0.669	0.479	0.669	0.479	0.456	0.241
VII	0.2	0.8	0.364	0.371	0.382	0.473	0.220	0.309
VIII	0.2	0.4	0.615	0.460	0.615	0.460	0.388	0.228
IX	0.2	0.2	0.696	0.514	0.696	0.514	0.493	0.277

Table D.6: Bias, Absolute Bias and MSE for $\widehat{\beta_4}$ - 2 Step Algorithm

Case			**p=29**					
			BIAS($\widehat{\beta_4}$)		AB($\widehat{\beta_4}$)		MSE($\widehat{\beta_4}$)	
	ρ_{temp}	ρ_{spat}	Step I	Step III	Step I	Step III	Step I	Step III
I	0.8	0.8	0.500	0.432	0.514	0.527	0.333	0.384
II	0.8	0.6	0.613	0.388	0.615	0.403	0.410	0.216
III	0.8	0.2	0.830	0.601	0.830	0.601	0.700	0.376
IV	0.6	0.8	0.502	0.387	0.511	0.478	0.327	0.315
V	0.5	0.5	0.695	0.471	0.695	0.471	0.511	0.267
VI	0.4	0.2	0.842	0.622	0.842	0.622	0.715	0.397
VII	0.2	0.8	0.478	0.321	0.493	0.489	0.315	0.345
VIII	0.2	0.4	0.742	0.518	0.742	0.518	0.566	0.298
IX	0.2	0.2	0.831	0.610	0.831	0.610	0.699	0.388

Table D.7: Bias, Absolute Bias and MSE for $\widehat{\beta_5}$ - 2 Step Algorithm

Case			**p=29**					
			BIAS($\widehat{\beta_5}$)		AB($\widehat{\beta_5}$)		MSE($\widehat{\beta_5}$)	
	ρ_{temp}	ρ_{spat}	Step I	Step III	Step I	Step III	Step I	Step III
I	0.8	0.8	0.565	0.298	0.567	0.386	0.388	0.234
II	0.8	0.6	0.649	0.380	0.649	0.389	0.446	0.190
III	0.8	0.2	0.755	0.531	0.755	0.531	0.580	0.293
IV	0.6	0.8	0.567	0.302	0.567	0.349	0.363	0.205
V	0.5	0.5	0.664	0.392	0.664	0.392	0.454	0.177
VI	0.4	0.2	0.759	0.532	0.759	0.532	0.583	0.294
VII	0.2	0.8	0.580	0.364	0.580	0.409	0.378	0.246
VIII	0.2	0.4	0.696	0.455	0.696	0.455	0.495	0.223
IX	0.2	0.2	0.758	0.544	0.758	0.544	0.581	0.305

For the intercept and binary covariate, 2 step algorithm seems to increase the biases

and MSEs, for the normal covarites, the metrics get lower at the end of the 2 step algorithm.

Also, when we compare the results with 3 step algorithm, all the metrics are higher in 2 step case. This is probably due to the smaller number of boosting iterations.

The loss function results at the end of each iteration and in each step is for both algorithms are presented as AIC values in the following figures.

$\underline{\rho_{temp} = 0.8, \rho_{spat} = 0.8}$

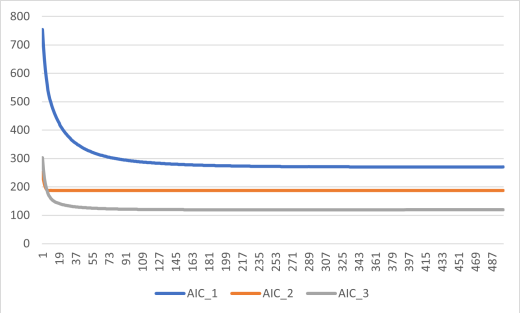


Figure D.1: AIC - 3 Step Boosting - $\rho_{spat} = 0.8, \rho_{temp} = 0.8$

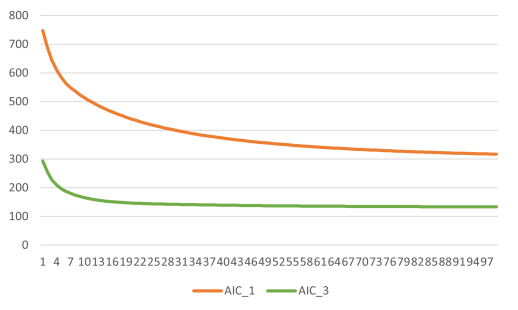


Figure D.2: AIC - 2 Step Boosting - $\rho_{spat} = 0.8, \rho_{temp} = 0.8$

$\underline{\rho_{temp} = 0.8, \rho_{spat} = 0.6}$

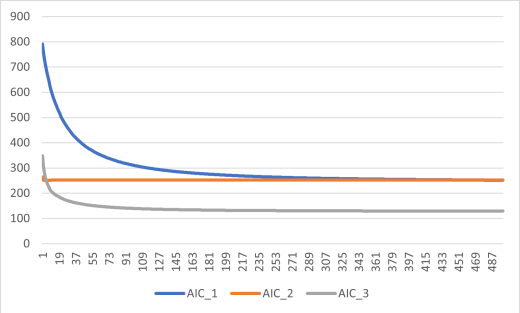


Figure D.3: AIC - 3 Step Boosting - $\rho_{spat} = 0.8, \rho_{temp} = 0.6$

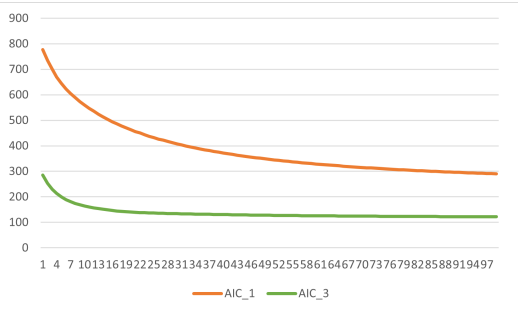


Figure D.4: AIC - 2 Step Boosting - $\rho_{spat} = 0.8, \rho_{temp} = 0.6$

$\underline{\rho_{temp} = 0.8, \rho_{spat} = 0.2}$

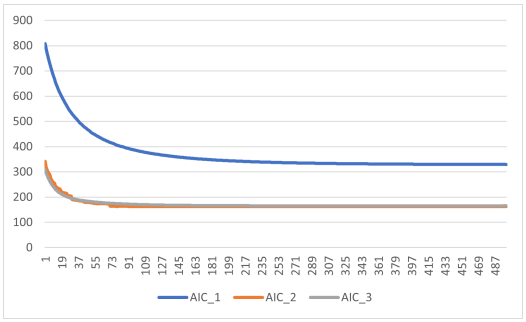


Figure D.5: AIC - 3 Step Boosting - $\rho_{spat} = 0.8, \rho_{temp} = 0.2$

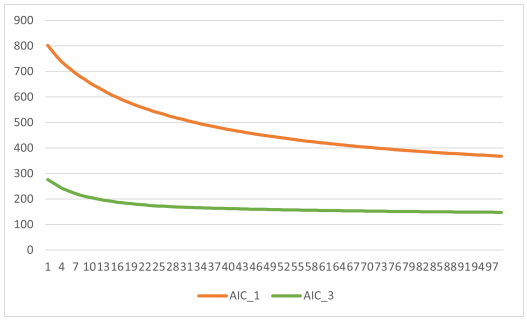


Figure D.6: AIC - 2 Step Boosting - $\rho_{spat} = 0.8, \rho_{temp} = 0.2$

$\rho_{temp} = 0.6$,$\rho_{spat} = 0.8$

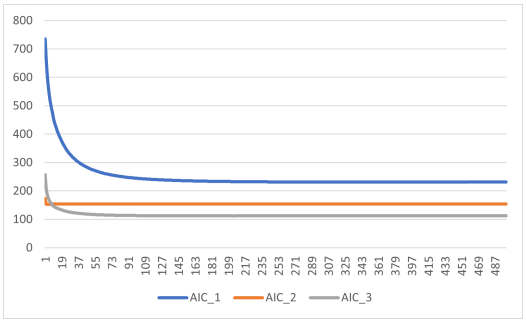


Figure D.7: AIC - 3 Step Boosting - $\rho_{spat} = 0.6, \rho_{temp} = 0.8$

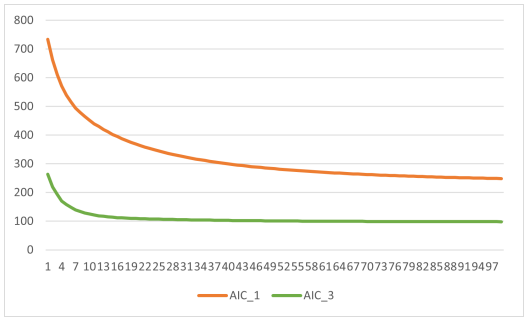


Figure D.8: AIC - 2 Step Boosting - $\rho_{spat} = 0.6, \rho_{temp} = 0.8$

$\rho_{temp} = 0.5$,$\rho_{spat} = 0.5$

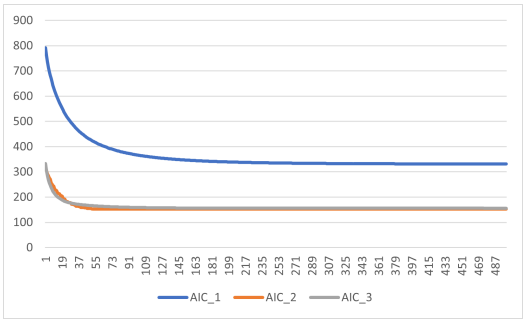


Figure D.9: AIC - 3 Step Boosting - $\rho_{spat} = 0.5, \rho_{temp} = 0.5$

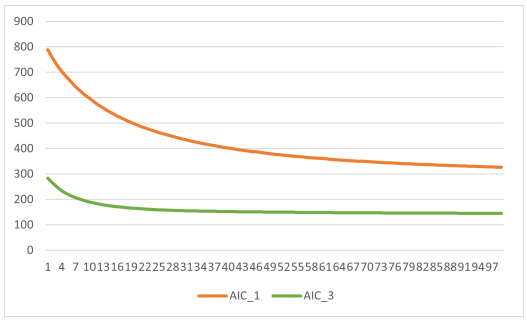


Figure D.10: AIC - 2 Step Boosting - $\rho_{spat} = 0.5, \rho_{temp} = 0.5$

$\rho_{temp} = 0.4, \rho_{spat} = 0.2$

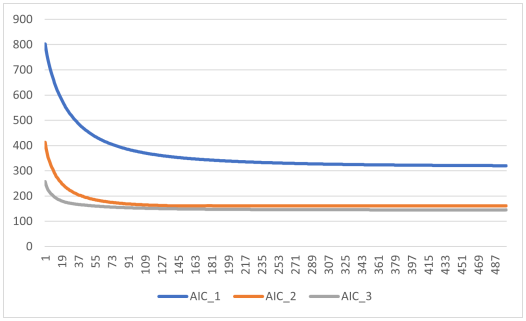


Figure D.11: AIC - 3 Step Boosting - $\rho_{spat} = 0.4, \rho_{temp} = 0.2$

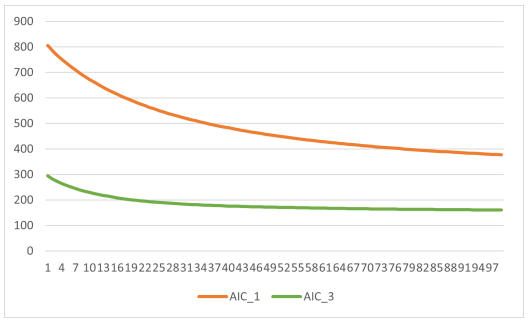


Figure D.12: AIC - 2 Step Boosting - $\rho_{spat} = 0.4, \rho_{temp} = 0.2$

$\rho_{temp} = 0.2, \rho_{spat} = 0.8$

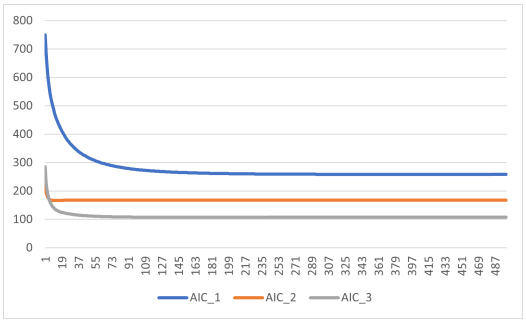


Figure D.13: AIC - 3 Step Boosting - $\rho_{spat} = 0.2, \rho_{temp} = 0.8$

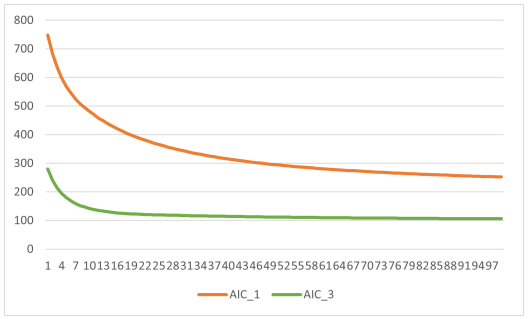


Figure D.14: AIC - 2 Step Boosting - $\rho_{spat} = 0.2, \rho_{temp} = 0.8$

$\rho_{temp} = 0.2, \rho_{spat} = 0.4$

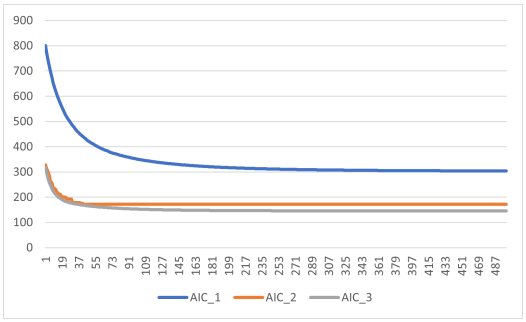


Figure D.15: AIC - 3 Step Boosting - $\rho_{spat} = 0.2, \rho_{temp} = 0.4$

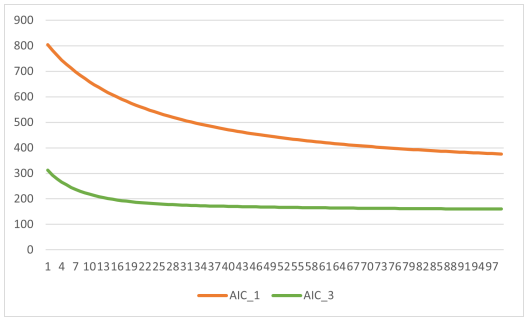


Figure D.16: AIC - 2 Step Boosting - $\rho_{spat} = 0.2, \rho_{temp} = 0.4$

$\rho_{temp} = 0.2$,$\rho_{spat} = 0.2$

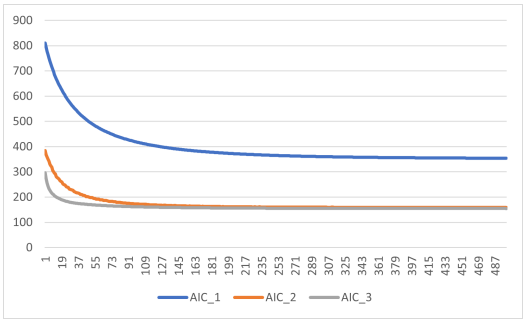


Figure D.17: AIC - 3 Step Boosting - $\rho_{spat} = 0.2, \rho_{temp} = 0.2$

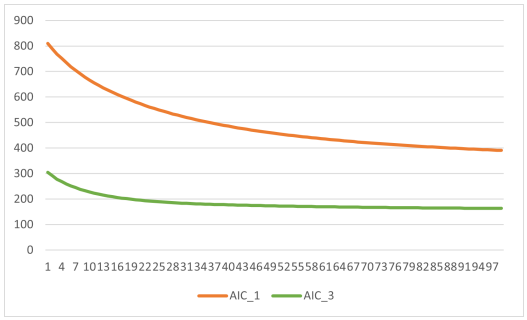


Figure D.18: AIC - 2 Step Boosting - $\rho_{spat} = 0.2, \rho_{temp} = 0.2$

In all cases, after 100 iteration, AIC give very similar results for both algorithms.

D.2 Classification Results

In Tables D.8 and D.9 the following measures calculated at the end of each steps are presented.

Table D.8: Classification metrics in simulation for different correlation structures

Metric	ρ_{temp}	ρ_{spat}	Step I	Step III
Sensitivity	0.8	0.8	0.919	0.760
	0.8	0.6	0.936	0.800
	0.8	0.2	0.951	0.756
	0.6	0.8	0.931	0.843
	0.5	0.5	0.941	0.755
	0.4	0.2	0.959	0.713
	0.2	0.8	0.937	0.830
	0.2	0.4	0.930	0.705
	0.2	0.2	0.930	0.724
Specificity	0.8	0.8	0.852	0.977
	0.8	0.6	0.872	0.964
	0.8	0.2	0.739	0.967
	0.6	0.8	0.896	0.986
	0.5	0.5	0.761	0.960
	0.4	0.2	0.702	0.950
	0.2	0.8	0.883	0.976
	0.2	0.4	0.747	0.957
	0.2	0.2	0.734	0.948
Precision	0.8	0.8	0.917	0.983
	0.8	0.6	0.938	0.979
	0.8	0.2	0.914	0.985
	0.6	0.8	0.943	0.991
	0.5	0.5	0.905	0.979
	0.4	0.2	0.898	0.975
	0.2	0.8	0.939	0.985
	0.2	0.4	0.891	0.973
	0.2	0.2	0.896	0.972

Table D.9: Classification metrics in simulation for different correlation structures - cont

Metric	ρ_{temp}	ρ_{spat}	Step I	Step III
F1	0.8	0.8	0.918	0.858
	0.8	0.6	0.937	0.880
	0.8	0.2	0.932	0.856
	0.6	0.8	0.937	0.911
	0.5	0.5	0.923	0.852
	0.4	0.2	0.927	0.824
	0.2	0.8	0.938	0.901
	0.2	0.4	0.910	0.818
	0.2	0.2	0.913	0.830
Accuracy	0.8	0.8	0.895	0.838
	0.8	0.6	0.915	0.853
	0.8	0.2	0.897	0.810
	0.6	0.8	0.918	0.893
	0.5	0.5	0.888	0.815
	0.4	0.2	0.890	0.777
	0.2	0.8	0.918	0.880
	0.2	0.4	0.783	0.918
	0.2	0.2	0.873	0.788
Kappa	0.8	0.8	0.772	0.678
	0.8	0.6	0.807	0.696
	0.8	0.2	0.717	0.591
	0.6	0.8	0.822	0.780
	0.5	0.5	0.723	0.616
	0.4	0.2	0.702	0.538
	0.2	0.8	0.819	0.752
	0.2	0.4	0.567	0.819
	0.2	0.2	0.683	0.566

The following arguments can be made according to the above tables.

- Specificity and precision increased in Step 3, and others decreased.
- When the temporal correlation is high, 2 step algorithm performs better than 3 step algorithm in all metrics, while when the spatial correlation is high, 3 step algorithm produces higher results.

In conclusion, if the maximum number of boosting iterations are taken as the same, then for the variable selection performance of 2 step algorithm seems better by identifying no covariate as significant mistakenly. On the other hand, for the classification part, as stated above, when the temporal correlation is high, 2 step algorithm performs better than 3 step algorithm in all metrics.

CURRICULUM VITAE

PERSONAL INFORMATION

Surname, Name: Emer, Deniz Esin
Nationality: Turkish (TC)
Date and Place of Birth: 29/06/1987, Ankara
Phone: 0555 507 21 48

EDUCATION

Degree	Institution	Year of Graduation
M.S.	Middle East Technical University (METU) Department of Industrial Engineering	2014
B.S.	Middle East Technical University (METU) Department of Mathematics	2010
High School	Milli Piyango Anatolian High School	2005

PROFESSIONAL EXPERIENCE

Year	Place	Enrollment
Sep 2017 - Present	MRC Turkey	Executive Consultant
Dec 2016 - Sep 2017	Neurodis	Business Development Analyst
Sep 2014 - Nov 2016	EnerjiSA Elec. Retail Sales	Business Application Specialist
Apr 2014 - Aug 2014	ETİ	System Development Engineer

PUBLICATIONS

1. Bakal, I.S., Bayindir, Z.P., & Emer, D.E. (2017). Value of disruption information in an EOQ environment. *European Journal of Operational Research*, 263, 446-460. DOI:10.1016/j.ejor.2017.04.045