

ABSTRACT

JIANG, TAO. High Dimensional Linear and Tree-based Models in Bioinformatics. (Under the direction of Alison Motsinger-Reif and David Reif).

Dimensionality in statistics refers to the number of independent variables or features in data. High-dimensional data, including genetic data, are becoming increasingly available as data collection technology evolves. Scientists need powerful, effective analytic methods to glean maximum scientific insight from these data. There are a number of analytical challenges with high dimensional data, made even more challenging in the case of low sample sizes where the number of variables is larger than the number of observations. While such data are informative, it is also challenging to apply advanced analysis on them like variable selection and classification given their high dimensionality. In this thesis, we are motivated by the limitations of current analytical approaches for high dimensional genetic and genomic data and propose new methods for variable selection and classification purposes.

In Chapter 2, we propose a novel searching scheme for a tuning parameter in high-dimensional penalized regression methods to address variable selection and modeling when sample sizes are limited compared to the data dimensions. Our method is motivated by high-throughput biological data such as genome-wide association studies (GWAS) and epigenome-wide association studies (EWAS). We propose a new estimate of the regularization parameter λ in penalized regression methods based on an estimated lower bound of the proportion of false null hypotheses with confidence $(1 - \alpha)$. The bound is estimated by applying the empirical null distribution of the higher criticism statistic, a second-level significance test constructed by dependent p -values using a multi-split regression and aggregation method. A tuning parameter estimate in penalized regression, λ , corresponds with the lower bound of the proportion of false null hypotheses. Different penalized regression methods with varied signal sparsity and strength are compared in the multi-split method setting. We demonstrate the performance of our method using both simulation experiments and the applications of real data on (1) lipid-trait genetics from the Action to Control Cardiovascular Risk in Diabetes (ACCORD) clinical trial and (2) epigenetic analysis evaluating smoking's influence in differential methylation in the Agricultural Lung Health Study.

In Chapter 3, we propose a novel strategy for conducting variable selection without prior model topology knowledge using the knockoff method with boosted tree models.

Our method is inspired by the original knockoff method, where the differences between original and knockoff variables are used for variable selection with false discovery rate control. The original method uses Lasso regression and was limited by requiring more samples than variables. We extend this method to be both model-free and appropriate for high-dimensional data. We propose two new sampling methods for generating knockoffs, namely the sparse covariance and principal component knockoff methods. We test these methods and compare them with the original knockoff method in terms of their ability to control type I errors and power. The boosted tree model is a complex system and has more hyperparameters than models with simpler assumptions. In our framework, these hyperparameters are either tuned through Bayesian optimization or fixed at multiple levels for trend detection. In simulation tests, we also compare the properties and performance of importance test statistics of tree models. The results include combinations of different knockoffs and importance test statistics. We also consider scenarios that include main-effect, interaction, exponential, and second-order models while assuming the true model structures are unknown. We apply our algorithm for tumor purity estimation and tumor classification using the Cancer Genome Atlas (TCGA) gene expression data.

In Chapter 4, we introduce a machine learning algorithm to detect same species contamination in next generation sequence data using support vector machines. Our approach uniquely detects such contamination using variant calling information stored in the variant call format (VCF) files (either DNA or RNA), and importantly can differentiate between same species contamination and mixtures of tumor and normal cells. In the first stage of our approach, a change-point detection method is used to identify copy number variations or copy number aberrations (CNVs or CNAs) for filtering prior to testing for contamination. Next, single nucleotide polymorphism (SNP) data is used to test for same species contamination using a support vector machine model. Based on the assumption that alternative allele frequencies in next generation sequencing follow the beta-binomial distribution, the deviation parameter ρ is estimated by maximum likelihood method. All features of a radial basis function (RBF) kernel support vector machine (SVM) are generated using either publicly available or private training data. Lastly, the generated SVM is applied in the test data to detect contamination. If training data is not available, a default RBF kernel SVM model is used.

© Copyright 2020 by Tao Jiang

All Rights Reserved

High Dimensional Linear and Tree-based Models in Bioinformatics

by
Tao Jiang

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Bioinformatics and Statistics

Raleigh, North Carolina
2020

APPROVED BY:

Alison Motsinger-Reif
Co-chair of Advisory Committee

David Reif
Co-chair of Advisory Committee

Matthew Breen

Jung-Ying Tzeng

Daniel Rotroff
External Member

DEDICATION

To my beloved family.

BIOGRAPHY

The author was born and grew up in China. He came to Raleigh in August of 2015 to pursue the Ph.D. degree in Bioinformatics at North Carolina State University under the direction of Dr. Alison Motsinger-Reif. From June of 2016 to August of 2018, he worked at Q2 Solutions as a Bioinformatics intern with the graduate industry traineeship (GIT) from NC State University. In August of 2017, he passed his Statistics Ph.D. qualifying exam and started working on a comajor Ph.D. degree in Statistics. From December of 2018 to December of 2019, he worked at National Institute of Environmental Health Sciences as a predoctoral visiting fellow. He defended his Ph.D. degree in February of 2020.

ACKNOWLEDGEMENTS

During these years I spent at NC State, my advisor Dr. Alison Motsinger-Reif provided me directions not only in academic research, but also in career planning. She supported me to work on my co-major Ph.D. in statistics and to work as an intern in industry for more than two years. She has given me enough freedom to pick up the research topics that I am interested in. Meanwhile, she has also provided me a lot of opportunities to collaborate with researchers in other fields. Alison is more than an advisor to me.

Dr. David Reif helped me as my academic co-advisor since Dr. Motsinger-Reif became the Chief of Biostatistics & Computational Biology Branch at NIEHS. David is the one who introduced scientific programming to me. He has helped me a lot on my way to my degree.

Dr. Daniel Rotroff showed me how to design a research project. I learned how to write Bash code and use the computing cluster from Daniel. I will remember all Daniel has taught me in my future research.

Dr. Jung-Ying Tzeng is the first professor I met in the BRC. She helped me to translate my undergraduate diploma into English and showed me the first real statistics research project in my life.

Dr. Matthew Breen directed me to conduct data analysis in my first biological project. He is knowledgeable, friendly, and patient.

Dr. Spencer Muse and Ms. Dana Ripperton helped me with my student registration and other administrative needs. Given my complicated situation, I really appreciate their help. I would also like to thank Mr. Kevin Dudley and Mr. Chris Smith for all their help with IT support. Without them, my research would be much harder.

Dr. Chad Brown and Dr. Martin Buchkovich offered me my very first job. Their endless help during my internship will keep guiding me in the next stages of my career.

I appreciate my friends Caizhi Huang, Tanchumin Xu, Jun Ma, Jonathan Leirer, Yueyang Huang, Meng Yang, Xiang Ji, Kuncheng Song, Guozhu Zhang, Tao Hu, Xinhao Li, Yichen Si, Yi Zhang, Yue Hao, and Jeremy Ash for their friendship, help, and teaching.

Finally, I thank my family for their endless love to me. They are always my support system. I hope they will continue to be proud of me in future.

TABLE OF CONTENTS

List of Tables	viii
List of Figures	x
Chapter 1 Introduction	1
1.1 The Quantitative Side of Bioinformatics	2
1.2 High-dimensional Linear Regression	4
1.3 Tree-based Regression Models	6
1.4 Motivations	7
1.5 Outlines	8
Chapter 2 Higher Criticism Tuned Regression For Weak And Sparse Signals	9
2.1 Backgrounds	10
2.2 Methods and Materials	12
2.2.1 Higher Criticism Test Statistic and Proportion Estimator for Independent Multiple Tests	12
2.2.2 Proportion Estimator for Dependent Multiple Tests	14
2.2.3 p -values in High-dimensional Regression	16
2.2.4 Higher Criticism Tuned Regression Algorithm	17
2.3 Simulation Studies	20
2.3.1 Variable Selection with Different Signal Strengths under Block Dependence	20
2.3.2 Variable Selection with Different Signal Sparsity under Block Dependence	24
2.3.3 p -value Aggregation	24
2.3.4 Lower Bound Estimation Performance	26
2.3.5 Variable Selection Performance of Higher Criticism Tuned Regression	30
2.4 Real Data Applications	31
2.4.1 Discrete Predictor: Genome-Wide Association Study	31
2.4.2 Continuous Predictor: DNA Methylation Data	34
2.5 Conclusions	36
2.6 Supplementary Information	38
2.6.1 Covariates in Regression Model for DNA Methylation Data Applications	38
Chapter 3 Knockoff Boosted Tree for Model-Free Variable Selection	39
3.1 Introduction	40
3.2 Methods and Materials	42
3.2.1 Parametric Functions of a Single Regression Tree	42
3.2.2 Boosted Tree Methods and Regularization	44
3.2.3 Regularization Parameters and Bayesian Optimization	45
3.2.4 Knockoff Variables in Boosted Tree Models	46

3.2.5	Variable Importance Test Statistics in Tree Models	49
3.2.6	Knockoff Boosted Tree Algorithm	52
3.3	Simulation Studies	54
3.3.1	Incorrect but Related Variables in Boosted Tree Models	54
3.3.2	Power and Type I Error Control of Knockoff Variables	56
3.3.3	Comparison of Variable Importance Ranking Statistics	60
3.3.4	SEAD, Other Knockoff Test Statistics, and Their False Discovery Control	61
3.4	Real Data Applications	65
3.4.1	Tumor Sample Purity Estimation	65
3.4.2	Tumor Type Classification	67
3.5	Conclusions	71
3.6	Supplementary Information	73
3.6.1	Parameters in Real Data Applications	73
3.6.2	Selected Genes Using 10-principal Component Knockoff	74
3.6.3	Relative Gene Expression Level Comparison in Low and High Purity Samples	75
3.6.4	Survival Analysis for Other Selected Genes in SKCM	77
Chapter 4 Same-Species Contamination Detection with Variant Calling Informa-		
tion from Next Generation Sequencing		80
4.1	Backgrounds	81
4.2	Methods and Materials	84
4.2.1	Beta-binomial Model of Allele Frequency in Next Generation Se- quencing	84
4.2.2	Quality Control of Variant Call Format (VCF) files	85
4.2.3	Distribution and Likelihood based Features	87
4.2.4	Support Vector Machine	89
4.2.5	Variant Quality Investigation Helper	89
4.3	Simulation and Real Application Studies	90
4.3.1	Changepoint analysis for approximate copy number region detection	90
4.3.2	Beta-binomial parameter estimation for reference sample(s)	91
4.3.3	Features in the classification and regression model	93
4.3.4	Tune cost and gamma parameters in radial kernel SVM	93
4.3.5	Simulated data test results	93
4.3.6	Real data test results	96
4.4	Discussion	99
4.5	Supplementary Methods	100
4.5.1	Features in Support Vector Machine (SVM)	100
4.5.2	Tunable Hyperparameters	102
4.5.3	Suggestion for Variant Calling Tool	103
Chapter 5 Conclusions		104
5.1	Summary of Results	104

5.2 Future Work	106
Chapter 6 R Packages & Software	108
References	140

LIST OF TABLES

Table 2.1	Comparison of methods for different signal strengths in linear models	22
Table 2.2	Comparison of methods for different signal strengths in logistic models	23
Table 2.3	Comparison of methods for different signal sparsity in linear models	25
Table 2.4	Comparison of methods for different signal sparsity in logistic models	25
Table 2.5	Mean and standard deviation of proportion estimates over true proportion	30
Table 2.6	Method comparison between different penalized regression methods and higher criticism tuned regression	32
Table 2.7	HCR for GWAS on LDL from ACCORD	34
Table 2.8	Selected SNPs in $n = 4540$	35
Table 2.9	HCR for EWAS on DNA methylation data	37
Table 2.10	Selected CpGs in $n = 2286$	37
Table 2.11	Covariates in regression model for DNA methylation Data applications	38
Table 3.1	Available variable importance test statistics in tree models	50
Table 3.2	Means and standard errors of 100 10-fold cross-validation error scores	56
Table 3.3	Mean and standard error of signal percentage and ranking ratio for test statistics in main effect models	61
Table 3.4	Mean and standard error of signal percentage and ranking ratio for test statistics in interaction models	61
Table 3.5	Mean and standard error of signal percentage and ranking ratio for test statistics in exponential models	61
Table 3.6	Mean and standard error of signal percentage and ranking ratio for test statistics in second order models	62
Table 3.7	Mean and standard error of power from 50 normal distributed simulation tests	63
Table 3.8	Mean and standard error of FDR from 50 normal distributed simulation tests	64
Table 3.9	Mean and standard error of power from 50 Poisson-distributed simulation tests	64
Table 3.10	Mean and standard error of FDR from 50 Poisson-distributed simulation tests	64
Table 3.11	Genes whose expression is related to BRCA tumor purity	66
Table 3.12	Genes whose expression is related to SKCM tumor purity	67
Table 3.13	Genes whose expression is related to ESCA vs STAD tumor classification	69
Table 3.14	Genes whose expression is related to READ vs COAD tumor classification	70
Table 3.15	XGBoost parameters in real data applications	73
Table 3.16	Genes whose expression is related to tumor purity using 10-principal component knockoff	74
Table 4.1	Features for Classification Model	87

Table 4.2	Maximum Likelihood Estimator $\hat{\rho}$ of NA10855 Samples	91
Table 4.3	Monte Carlo test results for parameter tuning and performance test .	93
Table 4.4	Contamination Detection for a series of simulated data	96
Table 4.5	Contamination Detection for Real Data	97
Table 6.1	A summary of R packages in the dissertation.	108

LIST OF FIGURES

Figure 1.1	Estimation picture for Lasso and ridge regression	5
Figure 2.1	Flowchart of higher criticism tuned regression	18
Figure 2.2	An example of a Lasso (a) solution path and (b) cross-validation curve	21
Figure 2.3	Distributions of all p -values from multiple linear regression of the true model	26
Figure 2.4	Distributions of all p -values from multiple linear regression of weak signal detection	27
Figure 2.5	Distributions of all p -values from multiple linear regression of moderate signal detection	28
Figure 2.6	Distributions of all p -values from multiple linear regression of strong signal detection	29
Figure 3.1	Knockoff boosted tree flowchart	52
Figure 3.2	Examples of boosted tree iteration steps for all models and boosters.	57
Figure 3.3	MAAC and KMMD of knockoffs with Normal and Poisson original variables.	58
Figure 3.4	Relative gene expression level comparison for selected genes in low and high purity samples of BRCA	68
Figure 3.5	Relative gene expression level comparison for selected genes in low and high purity samples of SKCM	68
Figure 3.6	Relative gene expression level comparison for selected genes in ESCA and STAD samples	69
Figure 3.7	Relative gene expression level comparison for selected genes in COAD and READ samples	71
Figure 3.8	Relative gene expression level comparison for other genes in low and high purity samples of BRCA	75
Figure 3.9	Relative gene expression level comparison for other genes in low and high purity samples of SKCM	76
Figure 3.10	Plots of the Kaplan-Meier estimator	77
Figure 3.11	Plots of the Kaplan-Meier estimator	78
Figure 3.12	(Continued) Plots of the Kaplan-Meier estimator	79
Figure 4.1	Four regions for B-Allele frequency	88
Figure 4.2	Flowchart of contamination detection procedure	90
Figure 4.3	Change point analysis for copy number region detection	92
Figure 4.4	Heterozygous B-allele frequency patterns of pure and contaminated samples	94
Figure 4.5	Boxplots of all features for pure samples <i>vs</i> contaminated samples .	95
Figure 4.6	Contaminated or Pure	98
Figure 4.7	Formalin-fixed paraffin-embedded (FFPE) tissue with low quality . .	99

Figure 4.8 Average running time for change point detection and feature generation 101

Figure 5.1 Contents of dissertation 105

CHAPTER

1

INTRODUCTION

In 1970, Ben Hesper and Paulien Hogeweg used the term, bioinformatics, to refer to the study of information processes in biological systems (Hesper and Hogeweg 1970). Now, after about 50 years, bioinformatics has become an interdisciplinary research field that develops and applies statistical methods for exploring biological research topics. Bioinformatics is built on foundations from biology, computer science, statistics, and builds on other fields for specific problems.

Rapid advances in genetic and genomic technologies have revolutionized modern genetics. Large-scale systematic genomic datasets have been generated to inform our biological understanding of both the normal workings of organisms in biology and disrupted pathways that cause human diseases. Technologies such as genome-wide SNP chips, RNAseq data, and whole genome sequencing allow investigators to routinely collect millions of variables. The integrative analysis of these vast amounts of data poses many interesting statistical and computational problems for bioinformaticists to address. High dimensional statistics has become an active area of research, partially driven by the various types of genomic data. Many novel statistical methods have been developed for analysis of high-/ultra-high-dimensional genomic data, including powerful statistical methods for high-dimensional regression analysis and tree-based approaches.

In this thesis, we consider doing regression and classification in high-dimensional data. We work on developing appropriate statistical or machine learning methods to solve bioinformatics problems. This chapter includes background introduction and motivations.

1.1 The Quantitative Side of Bioinformatics

Bioinformatics has two sides, biological domain knowledge and quantitative methods. While often motivated recently, considerable statistical methods have been developed and used in bioinformatics. These include but are not limited to experimental design, supervised learning, unsupervised learning, Markov chains, and stochastic processes.

There are broad range of application in the field. Such applications include gene identification and functional annotation, characterization of polymorphisms across human populations, protein structure modeling, phylogenetic analysis, and others. Within human genetics, common applications focus on understanding the etiology of complex diseases. Current bioinformatics research increasingly relies on machine learning algorithms to both understand and predict complex biological processes in each of these application areas (Bhaskar et al. 2006).

In the last decade, together with the improvements of statistical methods for bioinformatics, some widely accepted software or software package have been developed to implement machine learning algorithms, such as scikit-learn (Pedregosa et al. 2011), and Weka (Frank et al. 2004). While there are a broad number of tools available, there is "no free lunch" in machine learning, and no single algorithm works best in all the cases. Here we briefly review some of the most commonly used machine learning approaches. We group selected supervised algorithms by their tasks, regression or classification, and discuss their relative strengths and weaknesses.

Regression is the statistical "work horse" used for estimating the relationship between dependent and independent variables. Benefiting from its straightforward idea and interpretation, linear regression is one of the most popular choices for model fitting. Regularization approaches can be applied for linear regression to avoid overfitting and to perform variable selection, such as Lasso (Tibshirani 1996) and adaptive Lasso (Zou 2006). However, linear regression is inappropriate for nonlinear relationships, which means it is not the best choice for complex patterns such as the expectation in genetics and genomics. Additionally, the model formula, such as whether interaction or polynomial terms are included in the model needs to be determined by the analyst before model fitting. Regression tree models

(Breiman et al. 1984) were an important breakthrough in addressing these key limitations for linear regression. Regression tree models keep splitting samples within a decision or regression tree framework to minimize its loss function (or maximize its information gain), which enables tree models to learn nonlinear relationships. An additional advantage is that, there is less data preprocessing required for tree models, such as standardization. While tree representation can handle more complex structures, it also brings more tuning parameters and calculation cost. This downside is magnified for some types of regression tree modeling, such as bagging and boosting trees that will be discussed in detail later in the dissertation. To learn extremely complex patterns, multi-layer neural networks are a preferred alternative in large data, though such methods lose the interpretability of regression tree models.

Like regression, classification is also a supervised machine learning task. Instead of numerical values, classification models predict classes or probabilities of classes. Logistic regression performs similar to linear regression, except that its output has a probabilistic interpretation. During optimization, the objective function of logistic regression can also be combined with a penalty term in order to avoid overfitting. However, logistic regression does not perform well for complex class boundaries. To handle complex decision boundaries, support vector machines (Cortes and Vapnik 1995) with different kernels were proposed. While SVMs handle complex decision boundaries well, there are important caveats and limitations. First, it is not straightforward to choose an appropriate kernel. Additionally, SVMs have high computation costs. The tree based models mentioned above are also appropriate for classification problem, with the same advantages and caveats mentioned above. Just as mentioned for regression trees, individual classification trees are prone to overfitting, so it is tricky to find when to stop growing trees. Neural network models can also be used for classification, again with the same caveats.

For those currently available algorithms, high-dimensional data and data with complex correlation among variables are challenging. In high-dimensional data, there is no unique solution for coefficients if there are more variables than samples and challenges with variable selection are magnified. In complex structure data, the existing correlation among variables will affect the modeling. In this dissertation, we will build upon highly successful regression and classification approaches, including LASSO regression, tree based models, and support vectore machines, inspired by important challenges in bioinformatics. We describe the methods we will focus on in more details below.

1.2 High-dimensional Linear Regression

The analysis of high-dimensional genetic data has been a popular field of bioinformatics in the last few years. Assume we have outcome measurements $\mathbf{y}_{n \times 1}$, and design matrix $\mathbf{X}_{n \times p}$. To model $\mathbf{X}_{n \times p}$ as a linear function of $\mathbf{y}_{n \times 1}$, for simplicity we define

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}, \quad (1.1)$$

where $\boldsymbol{\epsilon}_n \sim \mathbf{N}(\mathbf{0}, \sigma^2 \mathbf{I})$. We can use linear regression to solve a least squares problem

$$\hat{\boldsymbol{\beta}}_{p \times 1}^{LS} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2. \quad (1.2)$$

This linear regression estimate can be written as

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}. \quad (1.3)$$

In high-dimensional data, $p > n$, the matrix \mathbf{X} cannot have linearly independent columns, $\mathbf{X}^\top \mathbf{X}$ is not invertible. Therefore, the linear regression estimate is not well-defined.

As one of the methods solving multicollinearity, ridge regression (Hoerl and Kennard 1970) shrinks the estimated coefficients towards zero. The ridge coefficients are defined by

$$\hat{\boldsymbol{\beta}}_{p \times 1}^{Ridge} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \{ \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_2^2 \}, \quad (1.4)$$

where $\lambda \geq 0$ is a tuning parameter, which can control the strength of penalty term, $\|\boldsymbol{\beta}\|_2^2$. This makes ridge regression different from ordinary least squares regression. The ridge regression estimator is

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^\top \mathbf{y}. \quad (1.5)$$

Unlike $\hat{\boldsymbol{\beta}}^{LS}$ is an unbiased estimator of $\boldsymbol{\beta}$, $\hat{\boldsymbol{\beta}}^{Ridge}$ is biased. Based on equation (1.5) and definition (1.1), the variance-covariance matrix of $\hat{\boldsymbol{\beta}}^{Ridge}$ can be calculated as

$$\operatorname{Cov}(\hat{\boldsymbol{\beta}}) = \sigma^2 (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^\top \mathbf{X} (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I})^{-1}. \quad (1.6)$$

Since $\lambda \geq 0$, ridge regression estimates have smaller variance than ordinary least squares regression estimates.

A ridge solution is not sparse, so it cannot achieve dimension reduction or variable selection purpose. Lasso, the least absolute shrinkage and selection operator, proposed in

Tibshirani (1996), is the first regression method that can provide sparse solution. Similar to ridge regression, the Lasso coefficients are defined by

$$\hat{\beta}_{p \times 1}^{Lasso} = \underset{\beta}{\operatorname{argmin}} \{ \|\mathbf{y} - \mathbf{X}\beta\|_2^2 + \lambda \|\beta\|_1 \}, \quad (1.7)$$

where the penalty term is in $L1$ norm. The difference between ridge and Lasso regression is shown for the case $p = 2$ in Figure 1.1 (Tibshirani 1996). Because of the $L1$ geometry, Lasso performs variable selection in such that an estimated component can be exactly zero (see (a) in Figure 1.1).

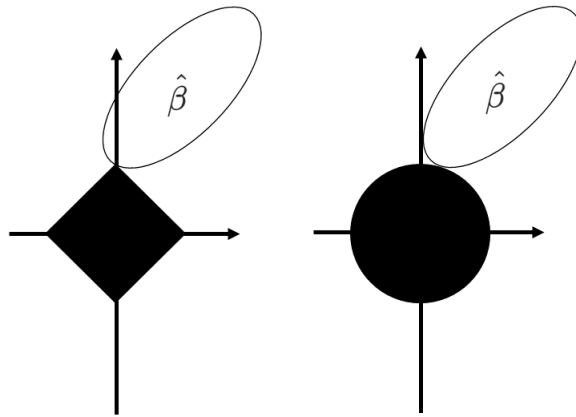


Figure 1.1: Estimation picture for (a) Lasso ($L1$ norm) and (b) ridge regression ($L2$ norm).

By modifying penalty term or loss function in objective function, other penalized regression models have been proposed besides ridge and Lasso regression, such as adaptive Lasso (Zou 2006), elastic net (Zou and Hastie 2005), group Lasso (Yuan and Lin 2006), SCAD (Fan and Li 2001) and MCP (Zhang et al. 2010a). We will revisit penalized regression models with other kinds of regularization in Chapter 2. All these methods require a formula to fit a model. However, it is tricky and time-consuming to decide whether interaction or polynomial terms should be included. Given the hierarchical structures of tree models, more complex data patterns are considered using tree models. We then take a look at regression tree models.

1.3 Tree-based Regression Models

Define the i -th pair of instance in a decision tree as $\{y_i, \mathbf{x}_i\}$, where \mathbf{x}_i is an item's predictor vector and y_i is its response value. When y_i has a continuous set, a decision tree is a regression tree. Breiman et al. (1984) proposed a procedure called Classification And Regression Tree (CART) in 1984, which is able to build regression trees for numeric attributes. In this chapter, we simply introduce bagging and boosting in tree models, and more details are included in Chapter 3.

Here in this chapter, we focus on bagging and random forest first, and then move on to boosting. Regression tree models suffer from high variance (Dietterich and Kong 1995). If we use two subsets of the same data set to fit regression tree models, these two models might be very different from each other. Bagging or Bootstrap aggregating is one of the solutions to reduce variance. Briefly speaking, bagging includes three steps:

1. Conduct bootstrap resampling B times;
2. Construct corresponding regression trees;
3. Combine all trees by averaging predictions.

Random forest originally proposed in Ho (1995) is one of special algorithms which shows improvements over traditional bagged tree models. The *random* in its title means it focuses on decorrelating trees. In order to split observation vector y_n , each time a random sample of m predictors are chosen as split candidates from whole p predictors, where $m < p$. A different sample of m predictors are selected for splitting each time. By doing this predictor subsampling, dominant predictors will not exist in all the tree models so that final tree regression results will not be highly correlated to other models. Note that when $m = p$, random forest is equal to bagging.

Boosting works in a similar way to bagging, except that boosted trees are trained adaptively or sequentially, where previously trained tree ensemble with new trained trees. There are two differences between bagging and boosting:

1. Each new tree is grown based on previously grown trees;
2. There is no bootstrap sampling in boosting.

Instead of learning data pattern in a single tree, boosting learns slowly, but keeps learning in multiple steps. Denote the prediction of a tree is $\hat{\mathbf{y}}_{n \times 1}^{(1)}$, then

$$\hat{\mathbf{y}}_{n \times 1}^{(1)} = \hat{f}_1(\mathbf{X}), \quad (1.8)$$

where f_1 is the projection function of the first grown regression tree. The idea is to fit a second regression tree on the residuals instead of \mathbf{y} , i.e.,

$$\hat{f}_2(\mathbf{X}) = \operatorname{argmin}_{f(\cdot)} L(\mathbf{y} - \hat{\mathbf{y}}^{(1)}, f(\mathbf{X})) = \operatorname{argmin}_{f(\cdot)} L(\mathbf{y} - \hat{f}_1(\mathbf{X}), f(\mathbf{X})), \quad (1.9)$$

where $L(\cdot)$ is a loss function. A third and fourth regression tree is grown in the same way but adding all previously grown trees. Thus, we have, for the k -th tree,

$$\hat{f}_k(\mathbf{X}) = \operatorname{argmin}_{f(\cdot)} L\left(\mathbf{y} - \sum_{i=1}^k \hat{f}_i(\mathbf{X}), f(\mathbf{X})\right). \quad (1.10)$$

The whole procedure is like turning multiple weak learner into a strong learner. There are different types of boosting tree algorithms available. Some famous ones among them are adaboost (Freund and Schapire 1997), gradient boosting (Friedman 2001), and extreme gradient boosting (Chen and Guestrin 2016). In Chapter 3, we will discuss more about extreme gradient boosting algorithm.

1.4 Motivations

We cover two research motivations here. One is for updating searching scheme of penalty in high-dimensional penalized model, and another one is for interpreting tree-based model.

In high-dimensional penalized model, choosing the optimal tuning parameter, λ is always tricky but important. The most widely used method is to find a λ that minimize the cross-validation error,

$$\hat{\lambda} = \operatorname{argmin}_{\lambda \in \{\lambda_1, \dots, \lambda_m\}} \operatorname{CVE}(\hat{\beta}(\lambda)), \quad (1.11)$$

where $\operatorname{CVE}()$ means cross-validation error, and m is the total number of λ 's attempted.

An alternative method is called the *one standard error rule*. The initial steps are similar and $\hat{\lambda}$ is calculated in the same way. Then we tune a parameter λ in the direction of increasing regularization until

$$\operatorname{CVE}(\hat{\beta}(\lambda)) \leq \operatorname{CVE}(\hat{\beta}(\hat{\lambda})) + \operatorname{SE}(\hat{\beta}(\hat{\lambda})), \quad (1.12)$$

where $\operatorname{SE}()$ is standard error. This results in the simplest model whose error is within one standard error of the minimal error. However, there is nearly nothing to interpret the choosing procedure of finding λ . A more meaningful searching algorithm for optimal λ is

desired.

Regression tree models are popular in data science because of their excellent prediction performance. On the other hand, inference of these models has not been equally developed as prediction. One of the reasons might be that most people working on tree-based regression are from computer science background, and inference is not their prior concern. Variable importance measures can be calculated when fitting regression tree models. Although these importance scores can be ranked, they cannot be used for variable selection. There is no clear threshold so that any larger values above the threshold should be selected. This can help to train a more interpretable tree-based model if any complex can between variables and responses can be detected.

Most current methods focus on mining available data to improve model prediction performance. However, it is possible that those variables in final fitted models are not meaningful enough. Therefore, generating appropriate variables from raw ones according to domain knowledge provides us more reasonable and interpretable final models. After making new variables, support vector machines (SVM) with nonlinear kernels are applied for classification tasks. Finally, the trained models rely heavily on domain knowledge of the problem where the data are related to.

1.5 Outlines

This dissertation mainly focuses on high dimensional linear and tree-based models. The rest of the dissertation is organized as follows. In Chapter 2, we propose a new tuning scheme for penalty terms in penalized regression. In Chapter 3, another new algorithm, called knockoff boosted tree, is proposed, where knockoff method and boosted tree models are combined. In Chapter 4, we show an application of using changepoint detection and support vector machine for same species contamination detection. For each chapter from 2 to 4, theoretical properties of the proposed methods are investigated. Simulation studies and real data applications are both contained to support our claims. Finally, we summarize our findings and suggest some future work.

CHAPTER

2

HIGHER CRITICISM TUNED REGRESSION FOR WEAK AND SPARSE SIGNALS

Here we propose a novel searching scheme for a tuning parameter in high-dimensional penalized regression methods to address variable selection and modeling when sample sizes are limited compared to the data dimensions. Our method is motivated by high-throughput biological data such as genome-wide association studies (GWAS) and epigenome-wide association studies (EWAS). We propose a new estimate of the regularization parameter λ in penalized regression methods based on an estimated lower bound of the proportion of false null hypotheses with confidence $(1 - \alpha)$. The bound is estimated by applying the empirical null distribution of the higher criticism statistic, a second-level significance test constructed by dependent p -values using a multi-split regression and aggregation method. A tuning parameter estimate in penalized regression, λ , corresponds with the lower bound of the proportion of false null hypotheses. Different penalized regression methods with varied signal sparsity and strength are compared in the multi-split method setting. We

demonstrate the performance of our method using both simulation experiments and the applications of real data on (1) lipid-trait genetics from the Action to Control Cardiovascular Risk in Diabetes (ACCORD) clinical trial and (2) epigenetic analysis evaluating smoking's influence in differential methylation in the Agricultural Lung Health Study. The proposed algorithm is included in the HCTR package, available at <https://cran.r-project.org/web/packages/HCTR/index.html>.

2.1 Backgrounds

High-throughput technologies in genetics and genomics present new challenges in high-dimensional data analysis. Genome-wide association studies (GWAS) have become a common tool to identify genetic loci associated with common complex diseases or disease-relevant phenotypes. It is understood that the etiology of many of the studied traits involves a large number of loci with small effects (Consortium et al. 2003). Most current methods, such as the mixed linear model approach (Yu et al. 2006; Zhang et al. 2010b) and genome-wide efficient mixed-model association (Zhou and Stephens 2012), require Bonferroni correction for multiple hypotheses after association analysis on a locus-by-locus basis (with millions of loci in each study). Assume α is the desired overall significance level and p is the number of hypotheses. Bonferroni correction tests each individual hypothesis at a significance level of α/p . However, in high-dimensional data, as $p \rightarrow \infty$ and $\alpha/p \rightarrow 0$, the Bonferroni correction is overly conservative (Perneger 1998) so that the power of GWAS is too low (even if correction is only for the effective number of loci after correcting for correlations). Therefore, important loci underlying complex traits are likely to be missed by traditional Bonferroni correction.

Multi-locus models of penalized regression, such as penalized logistic regression (Ayers and Cordell 2010) and elastic net (Cho et al. 2010), have been proposed as alternative strategies to single-locus GWAS methods because of their natural properties without Bonferroni correction. Since these methods are shrinkage approaches, tuning of the regularization parameter (λ) is extremely important (Tibshirani 1996). The accuracy of variable/predictor selection is highly related to the choice of the regularization parameter (Olson et al. 2017). The most widely accepted method for tuning the regularization parameter is described in Friedman et al. (2010). This algorithm computes solutions for a decreasing sequence of values for λ . In this decreasing sequence, λ_{\max} is the smallest value that can shrink the entire coefficient estimations to zero, and $\lambda_{\min} = c \lambda_{\max}$, where c is a small positive con-

stant. The final choice of λ in this sequence is data-driven and optimizes the λ value that minimizes the value of the loss function. However, the tuning range and final decision on the regularization parameter lacks theoretical support from domain knowledge, such as genetics in GWAS.

Another limit of penalized regression is the requirement for the signal strength. Here, we define $S_0 = \{i; \beta_i^0 \neq 0, i = 1, \dots, p\}$ as the set of true non-zero variables, where β_i^0 is the i -th variable in the true model, and the corresponding $s_0 = |S_0|$. van de Geer et al. (2011) proposed a beta-min condition to guarantee that all non-zero coefficients are sufficiently large

$$\min_{i \in S_0} |\beta_i^0| \geq \beta_{min}, \quad (2.1)$$

for a positive constant β_{min} . When $\beta_{min} = \phi^{-2} \sqrt{s_0 \log(p)/n}$, where ϕ^{-2} denotes a restricted eigenvalue of the design matrix. Under the beta-min condition (Bühlmann and Van De Geer 2011), this is the variable screening property:

$$P(\hat{S} \supseteq S_0) \rightarrow 1(p \geq n \rightarrow \infty). \quad (2.2)$$

Consistent variable selection is achieved when the irrepresentable condition (Zhao and Yu 2006) and beta-min condition are satisfied:

$$P(\hat{S} = S_0) \rightarrow 1(p \geq n \rightarrow \infty). \quad (2.3)$$

However, the beta-min condition is not always satisfied with real data (i.e., signals may be weaker than some thresholds). Thus, some signals can be ignored by variable selection in penalized regression methods. Accordingly, a method that can estimate the number of non-zero variables in a model is needed.

In this article, we propose an estimate of the regularization parameter λ based on p -values from the multi-split regression method. A second-level significance test, higher criticism (Donoho et al. 2004), is then used to estimate a lower bound of the proportion of false null hypotheses (Meinshausen et al. 2006) with confidence $(1 - \alpha)$. The estimate of λ corresponds to the lower bound of the proportion of false null hypotheses. Thus, the optimal choice of λ is determined based on not only the mean square error in cross-validation, but also an estimate of non-zero variables.

In the following sections, we describe our analytical framework, including (1) the generation of dependent p -values; (2) the derivation of the lower bound for the proportion of false null hypotheses; and (3) its relation to the choice of regularization parameter. We

validate our approach with simulation experiments that compare (1) the variable selection of different penalized regression methods in a multi-split setting under different signal sparsity and strength; (2) the consistency of the proportion estimator under different signal sparsity and strengths; and (3) the variable selection results for weak and sparse signals in high-dimensional data with higher criticism tuned parameters. We compare the performance of our analytic approach to traditional penalized regression methods. Finally, we demonstrate our approach using two datasets that were collected with different high throughput genetic technologies. In the first dataset we evaluate the genetic etiology of (1) lipid traits (low-density lipoprotein (LDL) values) in a cohort of patients with Type 2 Diabetes who were participants in the Action to Control Cardiovascular Risk in Diabetes (ACCORD) clinical trial (Group 2010) and have genome-wide single nucleotide polymorphism (SNP) data collected (Marvel et al. 2017; Rotroff et al. 2018a; Shah et al. 2016; Morieri et al. 2018; Shah et al. 2018; Rotroff et al. 2018b). In the second dataset, we evaluate epigenetic markers for association with epigenetic analysis evaluating smoking and pulmonary lung function in Agricultural Lung Health Study. Because of different data structures of SNP and epigenetic markers, we demonstrate the performance of our method in different data types. The particular data sets used we used were chosen because of strong, reproducible signals in the data. The etiology of lipid traits has been extensively studied in large cohorts and meta-analyses, which provides a baseline for the expected signals our method should be able to find (Dron and Hegele 2016). The methylation signature of smoking is also well validated (Joehanes et al. 2016; Sikdar et al. 2019).

2.2 Methods and Materials

2.2.1 Higher Criticism Test Statistic and Proportion Estimator for Independent Multiple Tests

Higher criticism is a multiple testing concept originally mentioned by Tukey. Donoho et al. (2004) proposed a generalized form of higher criticism. In a scenario with a large number of independent hypothesis tests, the higher criticism statistic performs as a global test statistic on the joint null hypothesis at significance level α . This test statistic is generated by standardizing the difference of the observed and expected fraction of significance under the global null. In this setting, there are p independent hypotheses tests, where the i -th test statistic T_i follows $H_{0,i} : T_i \sim N(0, 1)$ and $H_{a,i} : T_i \sim N(\mu_i, 1)$, where $\mu_i > 0$. In a special case,

assume all the non-zero μ_i are the same. If π is defined as the unknown proportion of false null hypotheses, then a global test can be described as $H_0 : T_i \sim N(0, 1), 1 \leq i \leq p, (i.i.d)$ and $H_a : T_i \sim (1 - \pi)N(0, 1) + \pi N(\mu, 1), 1 \leq i \leq p, (i.i.d)$.

Here, let the p -value of the i -th hypothesis test be

$$p_i = P\{N(0, 1) > T_i\}, \quad (2.4)$$

and denote $p_{(i)}$ as the i -th sorted p -value in increasing order. Note that the p -value follows a uniform distribution $\text{Unif}(0, 1)$ under the null hypotheses. Then, under the global null, $p_{(i)}$ is an order statistic from $\text{Unif}(0, 1)$. The higher criticism test statistic is defined as:

$$\text{HC}(i) = \max_{1 \leq i \leq ap} \frac{\sqrt{p}(\frac{i}{p} - p_{(i)})}{\sqrt{p_{(i)}(1 - p_{(i)})}}. \quad (2.5)$$

The higher criticism test statistic tests whether the proportion, π , is 0. Meinshausen et al. (2006) considered an estimator for the proportion of false null hypotheses among independent multiple hypotheses tests. Based on the empirical distribution of the p -values of those tests, a lower $100(1 - \alpha)\%$ confidence bound, $\hat{\pi}$, is proposed, such that

$$P(\hat{\pi} \leq \pi) \geq 1 - \alpha, \quad (2.6)$$

where $\pi = p^{-1} \sum_{i=1}^p \mathbf{1}\{P_i \sim G_i\}$, P_i is the distribution of the i -th p -value, G_i is an unknown distribution of the i -th p -value if the i -th null hypothesis is rejected. There is no distribution assumption for p -values from false null hypotheses, but it is certainly not $\text{Unif}(0, 1)$. Thus, the empirical distribution of the p -values is denoted as

$$F_p(t) = p^{-1} \sum_{i=1}^p \mathbf{1}\{P_i \leq t\} = \pi \hat{G}_{\pi p}(t) + (1 - \pi) \hat{U}_{(1-\pi)p}(t), \quad (2.7)$$

where $t \in (0, 1)$, and $\hat{U}_{(1-\pi)p}$ and $\hat{G}_{\pi p}$ denote the empirical distributions of the p -values from $(1 - \pi)p$ null hypotheses and πp false null hypotheses. When all p -values are from null hypotheses, $\pi = 0$, then define

$$V_p(t) = \sup_{t \in (0, 1)} \left\{ \frac{\hat{U}_p(t) - t}{\sqrt{t(1 - t)}} \right\} \quad (2.8)$$

and denote $\gamma_{p,\alpha}$ as a bounding sequence of $V_p(t)$ such that (a) $p\gamma_{p,\alpha}$ monotonically increases

with p , and (b) $P(V_p(t) > \gamma_{p,\alpha}) < \alpha$ for all p . The estimator of π is defined as:

$$\hat{\pi} = \sup_{t \in (0,1)} \frac{F_p(t) - t - \gamma_{p,\alpha} \sqrt{t(1-t)}}{1-t}, \quad (2.9)$$

where $\hat{\pi}$ satisfies the inequality in Equation (2.6).

2.2.2 Proportion Estimator for Dependent Multiple Tests

With the assumption that all variables, hypotheses tests, and p-values are independent, the estimator is determined with Equation (2.9). However, this independent assumption is not always true. Since the estimator is based on the higher criticism, the most straightforward way to handle correlation and dependence is by modifying the higher criticism and the corresponding estimator.

Some versions of modified higher criticism consider covariance structure, among other tests. Hall et al. (2010) proposed the innovated higher criticism (iHC), a modified higher criticism that detects sparse signals in correlated noise by paying attention to the correlation structure. Again, consider a p -dimensional Gaussian vector of test statistics from hypotheses tests,

$$T = \mu_p + Z, Z \sim N(\mathbf{0}, \Sigma_p), \quad (2.10)$$

where μ is a vector, and its i -th element $\mu_i = 0$ if the i -th null hypothesis is true. Let Σ_p be a positive definite matrix, and denote U_p such that $U_p \Sigma_p U_p^\top = I_p$. For simplicity, when choosing bandwidth $b_p = 1$, applying the standard higher criticism to $U_p T$ is a special case of innovated higher criticism, where

$$\text{iHC}(i) = \sup_{i: 1/p \leq p_{(i)} \leq 1/2} \frac{\sqrt{p}(\frac{i}{p} - p_{(i)})}{\sqrt{p_{(i)}(1-p_{(i)})}}. \quad (2.11)$$

Although innovated higher criticism has better performance than higher criticism under the dependence condition, we did not use it on our estimator for three reasons:

1. In most cases, Σ_p is unknown and must be estimated at high computational cost;
2. Some of the estimated Σ_p may not be **positive definite**, so U_p cannot be calculated directly;
3. The cost of Cholesky decomposition is $p^3/3$ flops and increases rapidly with p .

Another modified test statistic is generalized higher criticism proposed by Barnett et al. (2017). Unlike innovated higher criticism, generalized higher criticism does not transform the original test statistics. Instead, it estimates the variance of distribution to replace the binomial type denominator of standard higher criticism. Based on Equation (2.7), the empirical distribution of p -values can be defined using test statistics as

$$S_p(t) = \sum_{i=1}^p \mathbf{1}\{|T_i| \geq t\}, \quad (2.12)$$

where $t \geq 0$. Then, $\text{cov}\{S_p(t_i), S_p(t_j)\}$, as well as other estimates of $\text{var}(S_p(t))$, can be directly using Hermite polynomial. Generalized higher criticism is defined as

$$\text{GHC}(t) = \sup_{t \geq 0} \frac{S_p(t) - 2p\bar{\phi}(t)}{\sqrt{\hat{\text{Var}}(S_p(t))}}, \quad (2.13)$$

where $\bar{\phi}(t) = 1 - \phi(t)$ is the survival function of the standard normal distribution. Generalized higher criticism can handle correlation structure and signal sparsity, but the calculation complexity of estimation can be a problem.

This raises the question of whether there is any other solution since modifying standard higher criticism always results in more complex computations. Jeng et al. (2019) studied the consistency of the estimator using standard higher criticism in Meinshausen et al. (2006) in a scenario in which there was block dependence. Block dependence means that there is arbitrary dependence within each block of tests but not between blocks. For a sequence of ordered p -values, denote L as the total number of noise (true null hypotheses) variables ranked before the last signal (false null hypotheses) variable, and assume that L is bounded almost surely by a number \bar{l} . Let $p_{(1)}^0, \dots, p_{(p-\pi p)}^0$ be the ordered p -values from true null hypotheses, and assume that for any $r = 1, \dots, \bar{l}$,

$$P(p_{(r)}^0 \leq u | p_{(1)}^a, \dots, p_{(\pi p)}^a) \leq c_1 F_r(u), \quad (2.14)$$

where F_r is the left-side probability for independent p -values and $c_1 \geq 1$ is a constant. Theorem 2.4 of Jeng et al. (2019) stated that under block dependence and Equation (2.14), let the true but unknown $\pi = p^{-\eta}$ and $\eta \in [0, 1)$, then we have $P(1 - \delta < \hat{\pi}/\pi < 1) \rightarrow 1$ as $p \rightarrow \infty$ for an arbitrary small constant $\delta > 0$ if any of these conditions below are satisfied:

1. $\eta \in [0, (1 - \kappa)/2)$, $\inf_{t \in (0, 1)} G'(t) = 0$, and $1 \ll \bar{l} \log(\bar{l}) \ll p^{1-\eta/2}$.

2. $\eta \in [(1-\kappa)/2, 2/3)$, $G(p^{-\tau}) \rightarrow 1$ for some $\tau > 2\eta - (1-\kappa)$, and $1 \ll \bar{l} \log(\bar{l}) \ll p^{1-\eta/2}$.
3. $\eta \in [2/3, 1)$, $\kappa \in [0, 1)$, $G(p^{-\tau}) \rightarrow 1$ for some $\tau > 2\eta - (1-\kappa)$, and $1 \ll \bar{l} \log(\bar{l}) \ll p^{2(1-\eta)}$.

As these conditions are realistic and the consistency of the estimator from standard higher criticism is proven, we use a standard higher criticism estimator in our study.

2.2.3 p -values in High-dimensional Regression

Many prior works have focused on calculating p -values in high-dimension regression models. We discuss here a few that are related to our work. Before considering details, we first define our notations in a linear regression model:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \epsilon, \quad (2.15)$$

where \mathbf{y} is an $n \times 1$ vector, \mathbf{X} is an $n \times p$ design matrix and $p \gg n$, $\boldsymbol{\beta}$ is a $p \times 1$ parameter vector, and error term ϵ is an $n \times 1$ vector. To eliminate the intercept in the regression model, the observed variable and input variable are centered so that the observed mean is 0. Tibshirani (1996) defined the Lasso estimator, one of the most well-known ways to perform regression shrinkage and variable selection, as:

$$\hat{\boldsymbol{\beta}}_{p \times 1}^{\text{Lasso}}(\lambda) = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \{ \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_1 \}, \quad (2.16)$$

where $\|\cdot\|_1$ is vector $L1$ -norm, and $\lambda \geq 0$ is the tuning parameter. There are three main inferential tasks in high-dimensional models Bühlmann and Van De Geer (2011): given that $\boldsymbol{\beta}^0$ is the truth, (a) prediction accuracy, $\|\mathbf{X}\hat{\boldsymbol{\beta}} - \mathbf{X}\boldsymbol{\beta}^0\|_2^2/n$; (b) parameter estimation accuracy, $\|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^0\|_q$, $q \in \{1, 2\}$; and (c) variable selection by considering \hat{S} and S^0 , where $\hat{S} = \{i; \hat{\beta}_i \neq 0, i = 1, \dots, p\}$ and $S_0 = \{i; \beta_i^0 \neq 0, i = 1, \dots, p\}$.

A number of prior studies have focused on assigning significance in high-dimensional regression. This is a challenging task since asymptotically valid p -values are not available. Meinshausen et al. (2009) proposed randomly splitting samples into two groups multiple times, say B . The authors conducted variable selection on one subgroup of samples to choose $k_{i=1, \dots, B} < n \ll p$ variables and performed ordinary least squares for k_i variables on the other subgroup of samples. Subsequently, B groups of p -values were aggregated for all variables.

Post-selection inference is another widely accepted method of assigning significance in high-dimensional regression. Lockhart et al. (2014) originally described the method

and Tibshirani et al. (2016) extended the results of the asymptotic null distribution of the "covariance" test to test hypotheses up on adding new individual features to a selected model.

In the present study, we split the data multiple times. We apply Lasso, adaptive Lasso (Zou 2006), smoothly clipped absolute deviation (SCAD) (Fan and Li 2001), and minimax concave penalty (MCP) (Zhang et al. 2010a) to the first half of the data to perform variable selection. We chose these methods, because first, we would like to compare their penalties in regression; second, they satisfy the screening property when assuming the compatibility condition on the design matrix \mathbf{X} , the sparsity assumption, and a beta-min condition (Dezeure et al. 2015). We use the ordinary least squares method for the second half of the data to obtain corresponding p -values for the individual variables selected from the first half of the data. After sufficient Monte Carlo sampling, we aggregate the p -values for each individual variable. While this is a generic method, it requires a beta-min assumption. Details of our proposed method are described in Section 2.2.4.

2.2.4 Higher Criticism Tuned Regression Algorithm

To perform high-dimensional variable selection, we propose a robust multiple-stage algorithm called higher criticism tuned regression. Figure 2.1 is a flowchart that outlines details of the procedure. The algorithm contains two main parts: (a) p -value generation (in the green circle) and (b) a corresponding estimated lower bound $\hat{\pi}$ and higher criticism tuned regression solution $\hat{\beta}^{\text{HCR}}(\lambda)$ (in the red circle).

In the first part, p -values for each variable in the high-dimension model are generated according to single-split (Wasserman and Roeder 2009) and multi-split (Meinshausen et al. 2009) methods. Recall that Equation (2.15) is a linear regression model, where \mathbf{y} is an $n \times 1$ vector, and \mathbf{X} is an $n \times p$ design matrix. We randomly choose $\lfloor \frac{n}{2} \rfloor$ observations from \mathbf{y} and their corresponding rows in \mathbf{X} . We apply penalized regression methods, such as Lasso, adaptive Lasso, SCAD, or MCP, on the first half of data, $\mathbf{y}_{\lfloor \frac{n}{2} \rfloor}$ and $\mathbf{X}_{\lfloor \frac{n}{2} \rfloor \times p}$. Assume that we have k non-zero estimates out of p variables in penalized regression, where $k < (n - \lfloor \frac{n}{2} \rfloor)$. This creates a subset of the second half of \mathbf{X} , and so now we have $\mathbf{X}_{(n - \lfloor \frac{n}{2} \rfloor) \times k}$. The ordinary least squares method is applied for $\mathbf{y}_{n - \lfloor \frac{n}{2} \rfloor}$ and $\mathbf{X}_{(n - \lfloor \frac{n}{2} \rfloor) \times k}$. To control the family-wise error rate, the i -th p -value from t -test is corrected by Bonferroni correction as $p_i = \min(k p_i, 1)$. However, as mentioned in Meinshausen et al. (2006), using the family-wise error rate control results in a significant loss of power. If the signals are weak and the proportion of false null hypotheses is small, a more powerful tool is needed instead of stricter inference. Further,

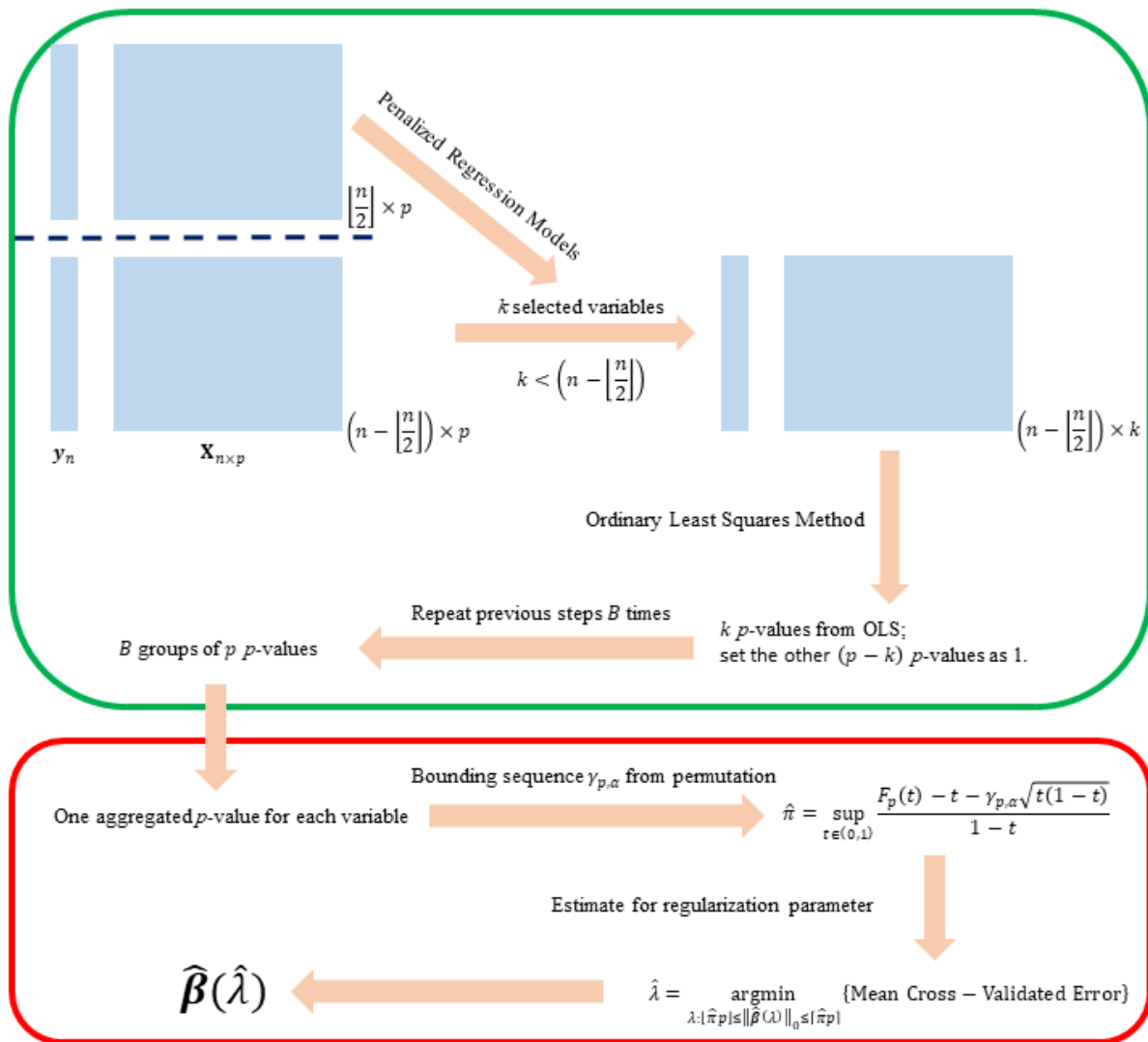


Figure 2.1: Flowchart of higher criticism tuned regression. p -value generation is in the green circle, and the estimated regression solution is in the red circle.

we empirically calculate the distribution of p -values under the null hypothesis and use this for subsequent estimator construction. We are interested in the difference between the null distribution and the observed values and thus we do not use family-wise error rate control for null distribution p -values or sample calculated p -values. We assign the k p -values from the t -test in ordinary least squares regression to the corresponding k selected variables. In contrast to Meinshausen et al. (2009), we denote the p -values of the rest ($p - k$) unselected variables as random variables following uniform distribution between (0, 1). We repeat random sample splitting and calculate the p -values B times, where B should be sufficiently large. For example, if $B = 100$, we will have B p -values for each variable, $p_i^1, \dots, p_i^B, (i = 1, \dots, p)$. These B p -values are aggregated by harmonic mean p -value Wilson (2019). Similar to Fisher's method, the harmonic mean p -value is a p -value used to test whether groups of p -values are statistically significant. Because the half samples used in ordinary least squares regression are part of the same full sample, there is dependence among the B p -values. The advantage of the harmonic mean p -value over Fisher's method is that independence among p -values is not required.

After generating p -values for all variables, we use them to build

$$\hat{\pi} = \sup_{t \in (0,1)} \frac{F_p(t) - t - \gamma_{p,\alpha} \sqrt{t(1-t)}}{1-t}$$

in equation 2.9. First, recall that $F_p(t)$ is the empirical distribution of p p -values from test data. Then, to get a valid bounding sequence $\gamma_{p,\alpha}$, Jeng et al. (2019) simulated the empirical distribution of $V_p(t)$ in equation 2.8 under global null, where the null hypothesis is realized by using permutation method proposed in Westfall et al. (1993). As a result, $\gamma_{p,\alpha}$ can be determined by finding the $(1 - \alpha_p)$ th quantile of the empirical distribution of $V_p(t)$ from simulations.

Previously, the choice of tuning parameter λ was always data-driven, for example, loss for linear regression or likelihood and misclassification rate for logistic regression. Here, based on $\hat{\pi}$, we propose a narrower tuning region for λ in penalized regression methods:

$$\hat{\lambda} = \underset{\lambda: \lfloor \hat{\pi} p \rfloor \leq \|\hat{\beta}(\lambda)\|_0 \leq \lceil \hat{\pi} p \rceil}{\operatorname{argmin}} \{ \text{Mean Cross-Validated Error} \}, \quad (2.17)$$

where p is the total number of variables. The widely accepted tuning region of λ is $[0, \lambda_0]$, where λ_0 is the minimal value that shrinks all variables to zero. Since $\pi \geq 0$ and $\hat{\pi} \geq 0$, we have $\hat{\lambda}(\hat{\pi} p) \leq \lambda_0$. Thus, $[\hat{\lambda}(\lfloor \hat{\pi} p \rfloor), \hat{\lambda}(\lceil \hat{\pi} p \rceil)] \subseteq [0, \lambda_0]$ gives a narrower tuning region of λ and a

better final choice of λ if $\pi > 0$.

For example, Figure 2.2 shows an example with a Lasso solution path using *glmnet*. The general tuning method for the regularization parameter is finding a minimum λ such that all the estimates are shrunk to zeros. The value of λ that gives the minimum mean cross-validated error is 0.242, which gives $\log(\lambda) = -1.419$, the left dashed line in Figure 2.2(b). Assume we have $\hat{\pi}p = 9.5$. Zooming into Figure 2.2(a), the gray area is the new tuning region, which is bounded by $\{\lambda_1; \|\hat{\beta}(\lambda)\|_0 = 9\}$ and $\{\lambda_2; \|\hat{\beta}(\lambda)\|_0 = 10\}$. When the new tuning region is shifted to Figure 2.2(b), we can find a new $\hat{\lambda}$ that minimizes cross-validated errors. In the following section, we demonstrate how this effects our final selected model.

We use simulation tests to answer the following topics: (a) whether a combination of different methods will perform better than a single method for multi-split variable selection; (b) the performance of p -value aggregation; (c) the consistency of the lower bound estimator; and (d) the accuracy of variable selection using higher criticism tuned regression.

2.3 Simulation Studies

2.3.1 Variable Selection with Different Signal Strengths under Block Dependence

We first consider sparse linear and logistic regression models with different signal strengths and compare the variable selection performance of penalized regression methods: Lasso, adaptive Lasso, SCAD, and MCP. For the simulation studies, we simulate $n = 100$ samples and $p = 1000$ variables. Each row of design matrix $X_{n \times p}$ is generated as $x_i \sim N_p(\mathbf{0}, \Sigma)$ with block dependence structure matrix $\Sigma = \text{diag}(\Sigma_1, \Sigma_2, \Sigma_3, \dots)$, where each Σ_i is a $\pi p \times \pi p$ matrix with element $\sigma_{j,k} = (0.5^{|j-k|})$. Note that our simulations focus on dependent multiple tests using a block correlation structure, which is a more realistic and challenging scenario than an independent variable design. To implement the multi-split algorithm, we conduct penalized regression in $B = 100$ loops. In each loop, 50 samples are randomly chosen from a total 100 samples for penalized regression. Then, 5-fold cross-validation tests are applied for Lasso, adaptive Lasso, SCAD, and MCP to search for the corresponding best model using two R packages *glmnet* (Friedman et al. 2009) and *ncvreg* (Breheny and Breheny 2019). The parameter γ is fixed at 3.7 in SCAD and 3.0 in MCP. Tables 2.1 and 2.2 summarize the average true positive (TP, a variable in the true model is chosen correctly), false positive

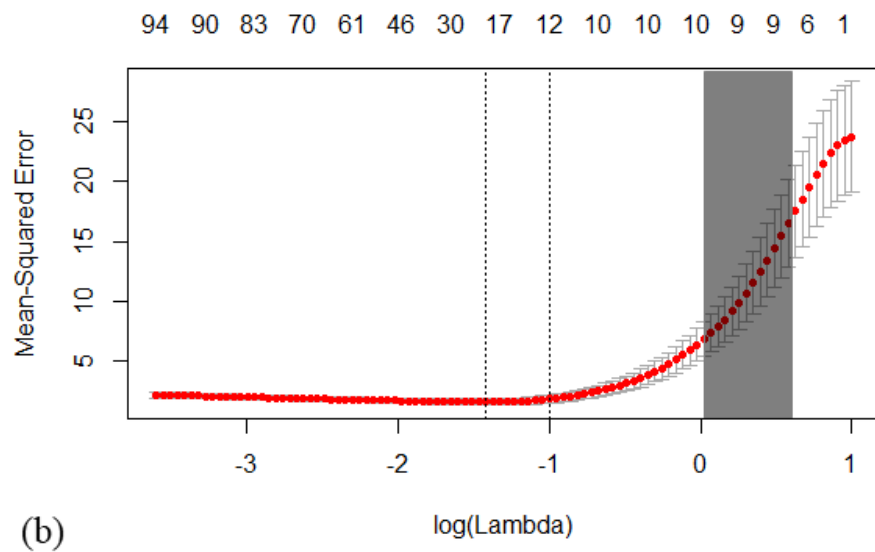
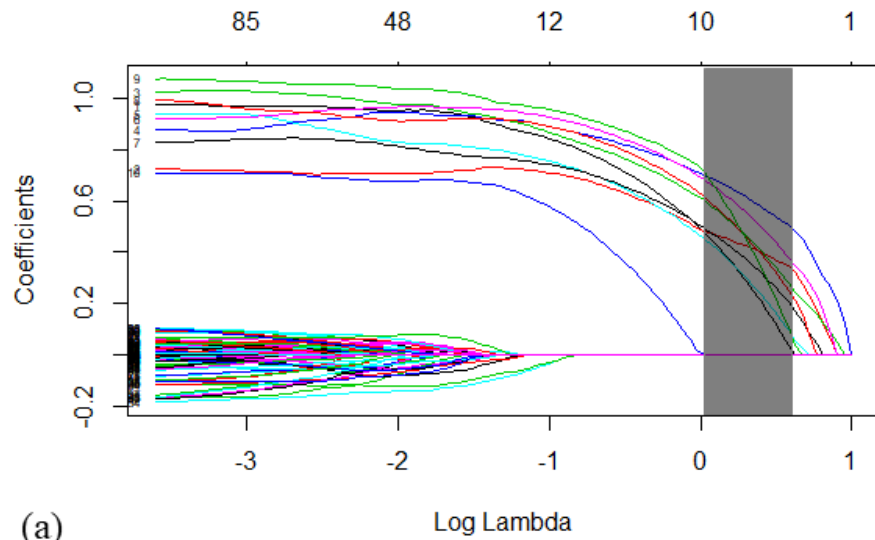


Figure 2.2: An example of a Lasso (a) solution path and (b) cross-validation curve (drawn with *glmnet*). An updated tuning region (in gray) for the regularization parameter in Lasso is suggested by an estimated value of non-zero variable proportion.

Table 2.1: Comparison of methods for different signal strengths in linear models

Methods	$\beta_i \neq 0$, where $i = 1, \dots, 10$	TP	FP	TN	FN	DOR	F ₁ score
Lasso	0.2	2.51	9.09	980.91	7.49	36.162	0.232
	0.5	8.06	19.50	970.50	1.94	206.773	0.429
	1	9.83	25.80	964.20	0.17	2160.986	0.431
	2	9.97	27.05	962.95	0.03	11830.7	0.424
Adaptive Lasso	0.2	4.46	36.05	953.95	5.54	21.303	0.177
	0.5	8.51	27.31	962.69	1.49	201.330	0.371
	1	9.78	23.16	966.84	0.22	1855.805	0.456
	2	9.87	17.61	972.39	0.13	4192.325	0.527
SCAD ($\gamma = 3.7$)	0.2	2.17	6.28	983.72	7.83	43.412	0.235
	0.5	5.80	10.25	979.75	4.20	131.999	0.445
	1	6.17	9.99	980.01	3.83	158.034	0.472
	2	6.50	9.80	980.20	3.50	185.752	0.494
MCP ($\gamma = 3$)	0.2	1.08	1.94	988.06	8.92	61.665	0.166
	0.5	3.55	2.76	987.24	6.45	196.871	0.435
	1	4.38	3.96	986.04	5.62	194.061	0.478
	2	4.83	3.55	986.45	5.17	259.599	0.526

(FP), true negative (TN), and false negative (FN) values. Besides, diagnostic odds ratio (Glas et al. 2003) and F₁ score (Chinchor 1992) are calculated for each method, where

$$\text{DOR} = \frac{\text{TP} \cdot \text{TN}}{\text{FP} \cdot \text{FN}}, \quad (2.18)$$

$$F_1 = \left(\frac{2}{\text{recall}^{-1} + \text{precision}^{-1}} \right) = \frac{2\text{TP}}{2\text{TP} + \text{FP} + \text{FN}}. \quad (2.19)$$

From Equation 2.18 and 2.19, we can see F₁ score focuses more on sensitivity of a test, while diagnostic odds ratio is a more general test. It is possible that they do not give the same ranking of tests.

In sparse linear regression, the response is $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$, where $\boldsymbol{\beta} = (\boldsymbol{\beta}_{\pi p}^\top, \mathbf{0}_{p-\pi p}^\top)^\top$, $\boldsymbol{\beta}_{\pi p}$ is a vector with equal elements, and π is fixed at 0.01, $\boldsymbol{\epsilon} \sim N_p(\mathbf{0}, \mathbf{I}_p)$. For all four methods, more signals are chosen as the signal strengths increases. Among the four methods, Lasso chooses more false positive signals when the signal is strong, while adaptive Lasso results in less false positive values when the signal is strong. However, both Lasso and adaptive Lasso have lower specificity than SCAD and MCP. The diagnostic odds ratio (DOR) and F₁ score are more general indicators of test performance. We say a completely random test will have DOR = 1, so a larger DOR indicates a more significant classification test. Both

Table 2.2: Comparison of methods for different signal strengths in logistic models

Methods	$\beta_i \neq 0$, where $i = 1, \dots, 10$	TP	FP	TN	FN	DOR	F ₁ score
Lasso	0.2	0.43	13.15	976.85	9.57	3.338	0.036
	0.5	1.82	11.41	978.59	8.18	19.082	0.157
	1	3.39	13.55	976.45	6.61	36.958	0.252
	2	3.33	10.69	979.31	6.67	45.736	0.277
Adaptive Lasso	0.2	0.47	21.35	968.65	9.53	2.238	0.030
	0.5	2.72	18.93	971.07	7.28	19.166	0.172
	1	3.86	15.79	974.21	6.14	38.787	0.260
	2	4.28	13.76	976.24	5.72	53.087	0.305
SCAD ($\gamma = 3.7$)	0.2	0.11	3.42	986.58	9.89	3.209	0.016
	0.5	1.28	4.99	985.01	8.72	28.976	0.157
	1	2.51	7.60	982.40	7.49	43.318	0.250
	2	3.09	8.77	981.23	6.91	50.032	0.283
MCP ($\gamma = 3$)	0.2	0.02	1.56	988.44	9.98	1.270	0.003
	0.5	0.64	1.46	988.54	9.36	46.296	0.106
	1	1.48	2.28	987.72	8.52	75.253	0.215
	2	1.95	2.43	987.57	8.05	98.447	0.271

DOR and F₁ score show us the fact that it is more challenging to identify signals when their strengths are weak. Table 2.1 indicates two points that could be improved: (1) the power of SCAD and MCP, as they are less likely to produce type I errors than the other two methods; and (2) the power of all methods when signals are weak. We later compare our proposed methods with the results.

Note that in a genomics situation, detection of any one variable, such as single-nucleotide polymorphism (SNP), in a block would usually be sufficient to trigger detailed fine mapping to understand the inter-variable correlation or linkage disequilibrium (LD) in genetics. In a word, if one of variables in a block is detected, then you have effectively detected all correlated variables. The power of this method is underestimated if this is a genetic simulation.

In sparse logistic regression, each element in response \mathbf{y} follows a Bernoulli distribution with the success probability as $\frac{\exp(\mathbf{x}_i \beta)}{(1 + \exp(\mathbf{x}_i \beta))}$, where \mathbf{x}_i is the i -th row of design matrix \mathbf{X} and β is the same as denoted in the linear regression models. Local solutions of SCAD and MCP are implemented in logistic regression models. Table 2.2 shows that it is more challenging to find the true model in logistic regression models than in linear regression models. The results also indicate that power can be improved when signals are weak. This is a high-dimensional classification problem and is the focus of other projects.

2.3.2 Variable Selection with Different Signal Sparsity under Block Dependence

We now look at variable selection performance of the four methods at different sparsity values, $\pi = 0.001, 0.002, 0.005, 0.01$. The setup for all models is similar to that outlined in Section 3.1. Note that all row vectors in the design matrix are generated randomly following multivariate normal distribution, so increasing the number of variables in the true model may not increase the number in the final chosen model for some algorithms.

Again, we first fit linear regression models. Table 2.3 shows that at all sparsity levels, adaptive Lasso is always the most powerful method, followed by Lasso. However, these models result in more false positive values than SCAD and MCP. Among all the methods, MCP has the highest specificity. Table 2.4 summarizes the logistic regression models and shows the same conclusions as Table 2.3. Now we revisit the penalties of these four penalized regression methods and the reason why we choose them for comparison. Lasso is used as a baseline here since it is the most basic shrinkage regression methods while generating biased estimators. In an orthonormal case, we have

$$\begin{cases} E|\hat{\beta}_j - \beta_j| = 0, & \beta_j = 0; \\ E|\hat{\beta}_j - \beta_j| \approx \beta_j, & 0 < |\beta_j| \leq \lambda; \\ E|\hat{\beta}_j - \beta_j| \approx \lambda, & |\beta_j| > \lambda, \end{cases} \quad (2.20)$$

where $\hat{\beta}_j$ is the j -th Lasso estimator, and β_j is the corresponding true value. Based on Lasso, Adaptive Lasso has a two-stage algorithm. In order to reduce bias, Adaptive Lasso assigns smaller penalties to variables with larger initial regression coefficients from OLS or ridge regression. As alternatives to Adaptive Lasso, some single-stage algorithms have been proposed where the penalty tapers off when $|\beta_j|$ gets larger. Tapering penalties cannot be convex. Famous examples of them are SCAD and MCP, both with nonconvex penalties. This is where the γ in the tables from, which controls how rapidly the penalty tapers off.

2.3.3 p -value Aggregation

After conducting variable selection on the first-half samples with penalized regression models (see Section 3.1 and 3.2), we use classic ordinary least squares regression to calculate p -values for those selected variables and assign $\text{Unif}(0, 1)$ to the unselected variables. After generating B groups of p -values, we implement *harmonicmeanp*, as described in Wilson (2019), to calculate the corresponding asymptotically exact harmonic mean p -values. For

Table 2.3: Comparison of methods for different signal sparsity in linear models, $p = 1000$ and $\beta_i = 0.5$, where $i = 1, \dots, \pi p$

πp	Methods	TP	FP	TN	FN	DOR	F ₁ score
1	Lasso	0.65	9.06	989.94	0.35	202.921	0.121
	Adaptive Lasso	0.94	39.60	959.40	0.06	379.561	0.045
	SCAD ($\gamma = 3.7$)	0.58	6.99	992.01	0.42	195.983	0.135
	MCP ($\gamma = 3$)	0.49	1.98	997.02	0.51	483.799	0.282
2	Lasso	1.91	10.08	987.92	0.09	2079.946	0.273
	Adaptive Lasso	1.94	35.16	962.84	0.06	885.433	0.099
	SCAD ($\gamma = 3.7$)	1.74	7.71	990.29	0.26	859.575	0.304
	MCP ($\gamma = 3$)	1.32	2.05	995.95	0.68	943.080	0.492
5	Lasso	4.49	15.09	979.91	0.51	571.706	0.365
	Adaptive Lasso	4.63	29.41	965.59	0.37	410.844	0.237
	SCAD ($\gamma = 3.7$)	3.81	7.69	987.31	1.19	411.060	0.462
	MCP ($\gamma = 3$)	2.67	2.05	992.95	2.33	555.046	0.549
10	Lasso	8.06	19.50	970.50	1.94	206.773	0.429
	Adaptive Lasso	8.51	27.31	962.69	1.49	201.330	0.371
	SCAD ($\gamma = 3.7$)	5.80	10.25	979.75	4.20	131.999	0.445
	MCP ($\gamma = 3$)	3.55	2.76	987.24	6.45	196.871	0.435

Table 2.4: Comparison of methods for different signal sparsity in logistic models, $p = 1000$ and $\beta_i = 1$, where $i = 1, \dots, \pi p$

πp	Methods	TP	FP	TN	FN	DOR	F ₁ score
1	Lasso	0.32	10.70	988.30	0.68	43.466	0.053
	Adaptive Lasso	0.43	22.43	976.57	0.57	32.845	0.036
	SCAD ($\gamma = 3.7$)	0.13	2.16	996.84	0.87	68.960	0.079
	MCP ($\gamma = 3$)	0.08	1.26	997.74	0.92	68.857	0.068
2	Lasso	1.52	10.01	987.99	0.48	312.551	0.225
	Adaptive Lasso	1.79	15.71	982.29	0.21	532.963	0.184
	SCAD ($\gamma = 3.7$)	1.62	7.29	990.71	0.38	579.363	0.297
	MCP ($\gamma = 3$)	1.30	2.28	995.72	0.70	811.050	0.466
5	Lasso	2.50	9.72	985.28	2.50	101.366	0.290
	Adaptive Lasso	2.80	12.32	982.68	2.20	101.517	0.278
	SCAD ($\gamma = 3.7$)	2.70	10.66	984.34	2.30	108.399	0.294
	MCP ($\gamma = 3$)	1.80	3.16	991.84	3.20	176.554	0.361
10	Lasso	3.39	13.55	976.45	6.61	36.958	0.252
	Adaptive Lasso	4.12	16.20	973.80	5.88	42.119	0.272
	SCAD ($\gamma = 3.7$)	2.61	7.60	982.40	7.39	45.653	0.258
	MCP ($\gamma = 3$)	1.35	1.91	988.09	8.65	80.739	0.204

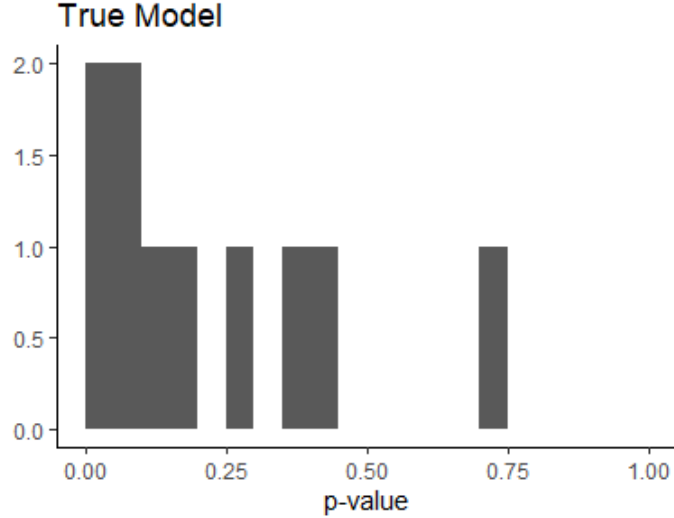


Figure 2.3: Distributions of all p -values from multiple linear regression of the true model, where the proportion of signal is 0.01, $p = 1000$, the signal strength $\beta_i = 0.2$.

comparison, we fit the true model with all signals and plot the distribution of p -values from the fit in Figure 2.3, where, for model $\beta_i = 0.2$, $\sigma^2 = 1$ and i is the index of signals. Because of multicollinearity and the weakness of signals, the distribution of p -values is highly uniform. Figure 2.4, 2.5, and 2.6 show the distributions of the aggregated p -values, with p -values of signals in black and p -values of noise in gray. When the signal is weak, such as in Figure 2.4, $\beta_i = 0.2$, where i is the location of a signal, and not all signals have p -values smaller than 0.05. This indicates that using p -values for variable selection is not desirable and that the beta-min condition must be satisfied Bühlmann and Van De Geer (2011). However, as signal strength increases to 0.5 and 1, the p -values of signals become increasingly significant. Further, although there is a clear difference in variable selection performance in Section 2.3.1 and 2.3.2, the distributions of the p -values of signals are not significantly different. Finally, we conclude that our modified multi-split algorithm works better for strong signals. We use the permutation method to repeat this procedure and generate a valid higher criticism test statistic under the null hypothesis.

2.3.4 Lower Bound Estimation Performance

Using the same models in Section 2.3.1 through 2.3.3, we follow the permutation method introduced in Jeng et al. (2019) to shuffle outcomes \mathbf{y} and re-assign each element of \mathbf{y} to a random individual. For each permuted data set, we calculate a new group of p -values

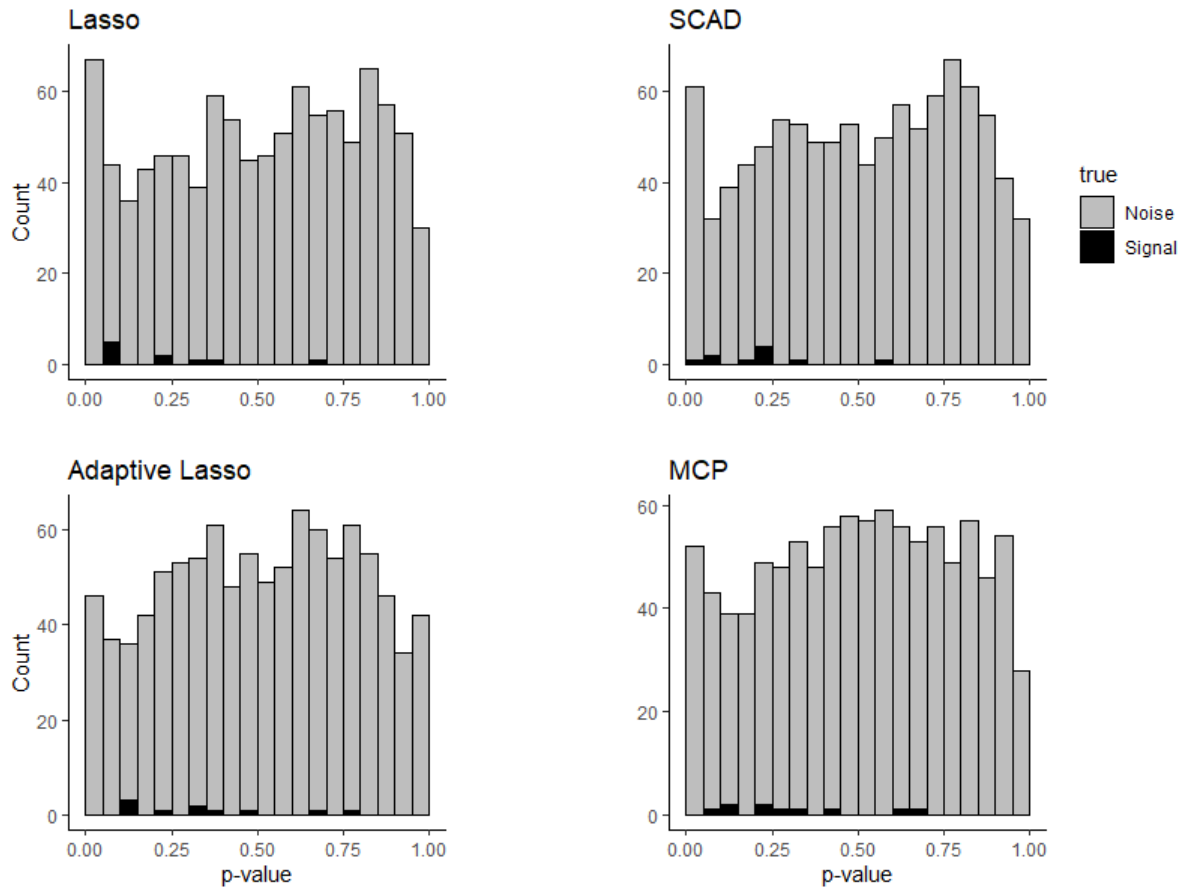


Figure 2.4: Distributions of all p -values from multiple linear regression of weak signal detection using 4 penalized regression methods, where proportion of signal is 0.01, $p = 1000$, and signal strength $\beta_i = 0.2$.

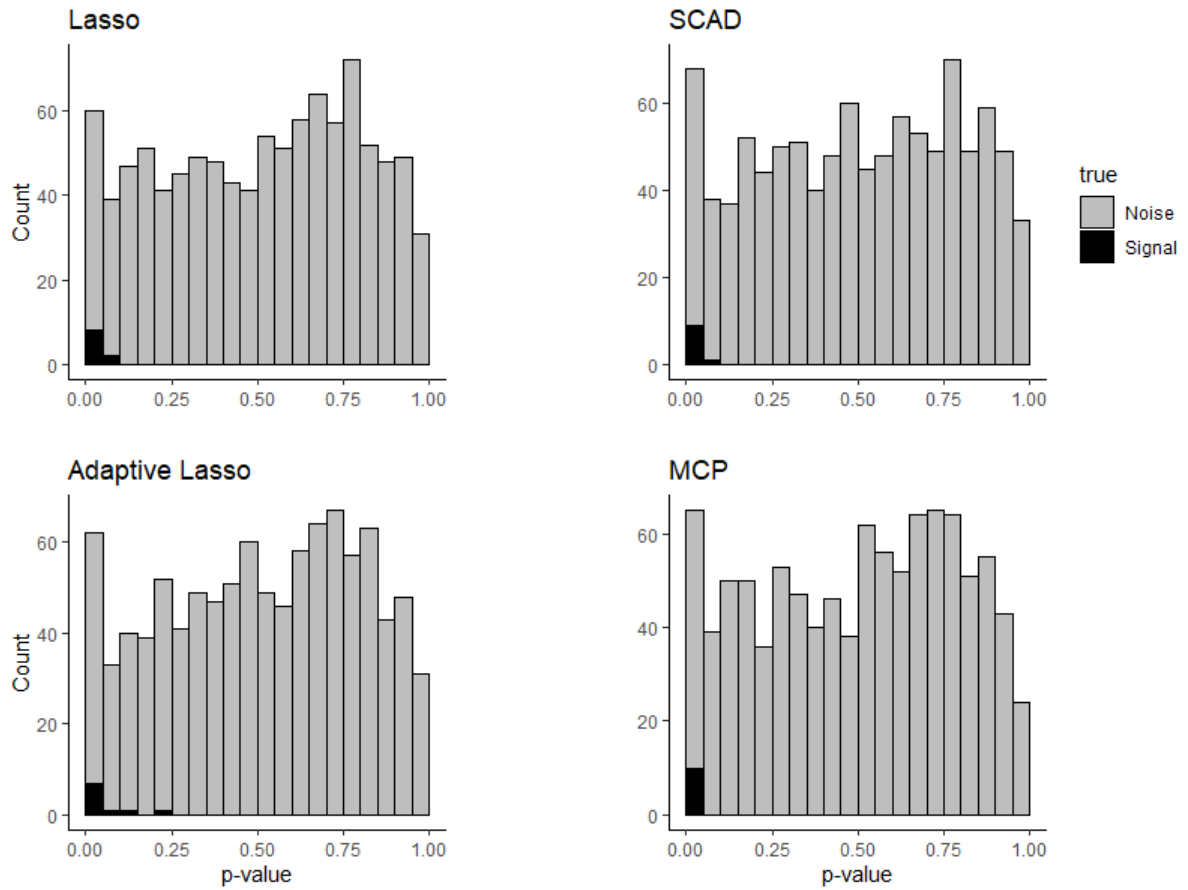


Figure 2.5: Distributions of all p -values from multiple linear regression of moderate signal detection using 4 penalized regression methods, where the proportion of signal is 0.01, $p = 1000$, and signal strength $\beta_i = 0.5$.

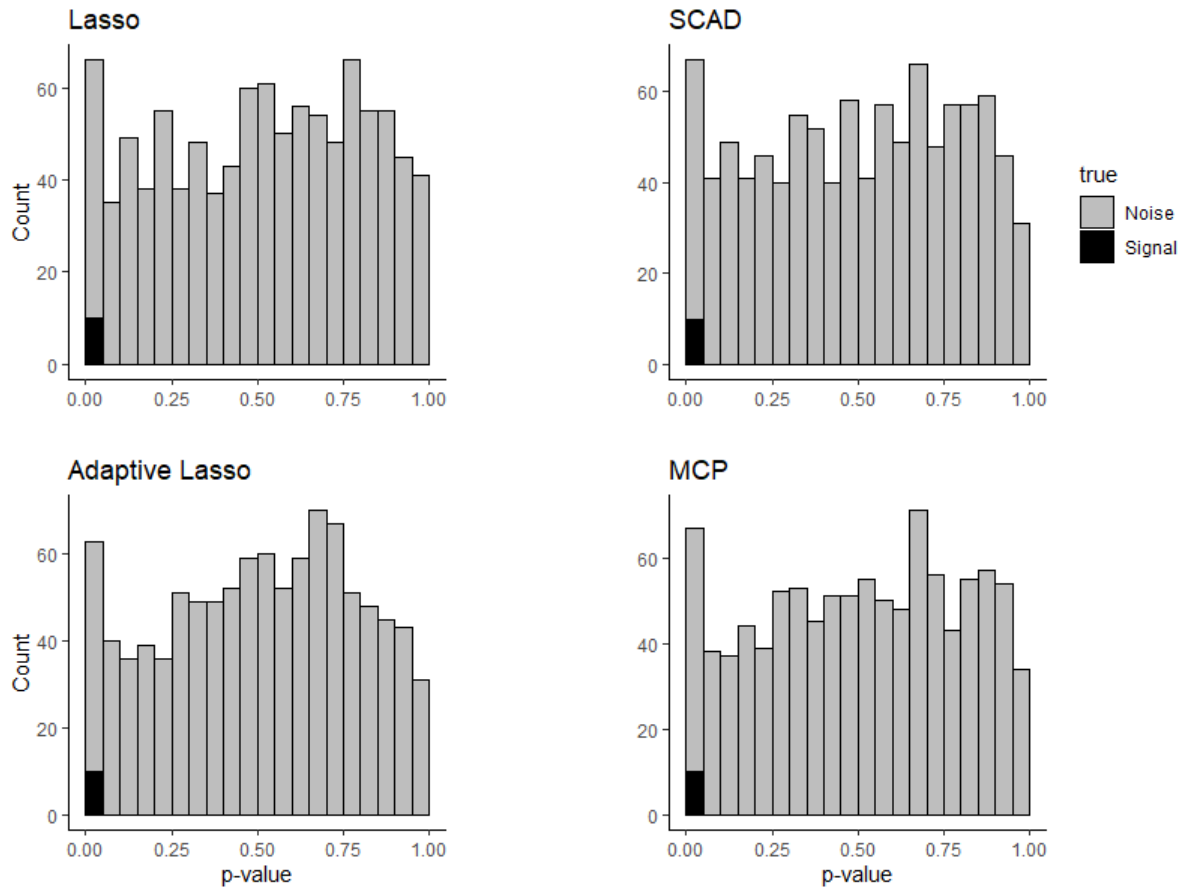


Figure 2.6: Distributions of all p -values from multiple linear regression of strong signal detection using 4 penalized regression methods, where the proportion of signal is 0.01, $p = 1000$, and signal strength $\beta_i = 1$.

Table 2.5: Eight groups of $\hat{\pi}/\pi$ (SE), mean and standard error of mean for *proportion estimates over true proportion*, where π is the true signal proportion, $\beta_i \neq 0$ is the signal strength, and the sample size is 10.

π	β_i	Lasso	Adaptive Lasso	SCAD ($\gamma = 3.7$)	MCP ($\gamma = 3$)	Ensemble
0.01	0.5	0.425 (0.150)	0.318 (0.114)	0.796 (0.101)	0.872 (0.079)	0.563 (0.072)
	1	1.142 (0.143)	0.952 (0.057)	1.026 (0.036)	1.230 (0.213)	0.994 (0.031)
0.005	0.2	0.149 (0.074)	0.402 (0.402)	0.583 (0.280)	0.310 (0.251)	0.185 (0.124)
	0.5	0.831 (0.192)	0.387 (0.109)	1.089 (0.302)	1.197 (0.364)	0.649 (0.053)
	1	1.226 (0.158)	1.005 (0.039)	1.328 (0.231)	1.087 (0.083)	1.025 (0.033)
0.002	0.2	0.764 (0.418)	0.324 (0.204)	0.654 (0.341)	0.273 (0.260)	0.174 (0.123)
	0.5	1.133 (0.440)	0.594 (0.452)	1.222 (0.450)	1.039 (0.440)	0.782 (0.192)
	1	1.555 (0.199)	1.087 (0.101)	1.449 (0.255)	1.443 (0.294)	1.142 (0.073)

and a corresponding higher criticism test statistic. We repeat this process 1,000 times and construct an empirical distribution of higher criticism test statistic under the null hypothesis. As suggested in Jeng et al. (2019), we use $\alpha = 1/\sqrt{\log(p)}$ for our test, where $p = 1000$. Using Equation (2.9), we calculate an estimate of signal proportion, $\hat{\pi}$, and compare it with the true signal proportion, π . Table 2.5 provides details. We try different sparsity values, $\pi = 0.002, 0.005, 0.01$, and different signal strengths. We conduct the penalized regression method 10 times for each sparsity and signal strength condition. Table 2.5 summarizes mean values and standard deviations of ratios $\hat{\pi}/\pi$. The table also includes a column titled *ensemble*. For each simulation, the maximum and minimum estimated proportion is dropped from the four penalized regression methods, and the mean is calculated from the remaining two estimates. Our *ensemble* method provides a robust estimate of signal proportion and its standard deviation is always the lowest among all five methods across all conditions. While we generally use variance and bias to evaluate the estimator of a parameter, here, given the low standard deviation and its closeness to 1, we use the *ensemble* method estimator.

2.3.5 Variable Selection Performance of Higher Criticism Tuned Regression

Finally, to summarize the simulation studies, we combine all the steps in our proposed algorithm. We conduct variable selection performance between four current penalized regression methods and their corresponding higher criticism tuned regression. Table 2.6

shows the results. Since the variance of the error term in our simulation model is 1, $\beta_i = 0.2$ or 0.5 is relatively weak. This setting is for weak and sparse signals. In this scenario, it is challenging to find the true variables of a model. In Table 2.6, it is clear that for the original tuning region, $[0, \lambda_0]$, all four penalized regression methods select more variables than the true model. Especially for adaptive Lasso, the numbers of false positive variable selection are large for all four settings. As a comparison, when we update the tuning region as a higher criticism tuned region, $\{\lambda : \lceil \hat{\pi} p \rceil \leq \|\hat{\beta}(\lambda)\|_0 \leq \lceil \hat{\pi} p \rceil\}$, all four methods make tremendously fewer false positive variable selections. Some of the methods have similar performance because the most significant variables in a model are highly likely to be the same as those in another model. It is clear that after higher criticism tuning, the odds ratios are increased compared with original values. This means that both original and tuned methods work, while the tuned method provides more significant classifications. On the other hand, the F_1 score depends a lot on true positive choices, so in the weak signal environment, where $\beta = 0.2$, tuned methods returns lower scores. However, in the moderate signal environment, where $\beta = 0.5$, tuned methods show better classification performance again.

2.4 Real Data Applications

We demonstrate our approach on two high dimensional use cases in human genetics. The first is single-nucleotide polymorphism (SNP) data from a genome-wide association study evaluating low density lipoprotein (LDL) levels in patients with Type 2 diabetes (Marvel et al. 2017). The second is differential methylation data from an epigenome wide study evaluating differential methylation related to smoking pack years (Sikdar et al. 2019). The two different datatypes were chosen because they demonstrate the method on both discrete and continuous predictors. While both studies have genome-wide data available, candidate regions of substantial dimensionality were chosen based on previously established biological associations to aid in interpreting the results of our new approach.

2.4.1 Discrete Predictor: Genome-Wide Association Study

The Action to Control Cardiovascular Risk in Diabetes (ACCORD) clinical trial investigated whether intensive therapy to target normal glycated hemoglobin levels would reduce cardiovascular events in patients with type 2 diabetes (Gerstein et al. 2008). Genome wide SNP data was collected on a large proportion of patients within the trial, and the data has been used mapping a number of clinically important phenotypes (Tang et al. 2019;

Table 2.6: Method comparison between different penalized regression methods and higher criticism tuned regression methods, with a sample size of 10 across all methods, $p = 1000$, $\sigma^2 = 1$.

π	β_i	Method	TP	FP	TN	FN	DOR	F ₁ score
0.002	0.2	Lasso	0.6	13.5	984.5	1.4	31.254	0.075
		HC-Lasso	0.1	0.2	997.8	1.9	262.579	0.087
		Adaptive Lasso	1.1	74.9	923.1	0.9	15.063	0.028
		HC-Adaptive Lasso	0.1	0.2	997.8	1.9	262.579	0.087
		SCAD ($\gamma = 3.7$)	0.4	4.6	993.4	1.6	53.989	0.114
		HC-SCAD ($\gamma = 3.7$)	0.1	0.1	997.9	1.9	525.211	0.091
		MCP ($\gamma = 3$)	0.2	1.1	996.9	1.8	100.697	0.121
		HC-MCP ($\gamma = 3$)	0.1	0.1	997.9	1.9	525.211	0.091
0.002	0.5	Lasso	2.0	19.4	978.6	0.0	246.005	0.171
		HC-Lasso	1.2	0.0	998.0	0.8	2611.462	0.750
		Adaptive Lasso	2.0	47.7	950.3	0.0	98.631	0.077
		HC-Adaptive Lasso	1.1	0.0	998.0	0.9	2282.286	0.710
		SCAD ($\gamma = 3.7$)	1.9	13.0	985.0	0.1	1439.615	0.225
		HC-SCAD ($\gamma = 3.7$)	1.2	0.0	998.0	0.8	2611.462	0.750
		MCP ($\gamma = 3$)	1.6	2.5	995.5	0.4	1592.800	0.525
		HC-MCP ($\gamma = 3$)	0.9	0.1	997.9	1.1	8164.636	0.600
0.005	0.2	Lasso	2.9	14.6	980.4	2.1	92.732	0.258
		HC-Lasso	0.4	0.0	995.0	4.6	351.353	0.148
		Adaptive Lasso	3.6	65.0	930.0	1.4	36.791	0.098
		HC-Adaptive Lasso	0.7	0.0	995.0	4.3	497.750	0.246
		SCAD ($\gamma = 3.7$)	2.6	16.9	978.1	2.4	62.699	0.212
		HC-SCAD ($\gamma = 3.7$)	0.5	0.0	995.0	4.5	398.200	0.182
		MCP ($\gamma = 3$)	1.8	3.5	991.5	3.2	159.348	0.350
		HC-MCP ($\gamma = 3$)	0.4	0.0	995.0	4.6	351.353	0.148
0.005	0.5	Lasso	5.0	17.9	977.1	0.0	584.435	0.358
		HC-Lasso	2.4	0.0	995.0	2.6	1862.548	0.649
		Adaptive Lasso	5.0	36.3	958.7	0.0	286.717	0.216
		HC-Adaptive Lasso	2.4	0.0	995.0	2.6	1862.548	0.649
		SCAD ($\gamma = 3.7$)	4	9.7	985.3	1.0	406.309	0.428
		HC-SCAD ($\gamma = 3.7$)	2.6	0.0	995.0	2.4	2128.31	0.684
		MCP ($\gamma = 3$)	3.5	7.2	987.8	1.5	320.120	0.446
		HC-MCP ($\gamma = 3$)	2.3	0.0	995.0	2.7	1742.125	0.630

Morieri et al. 2018; Rotroff et al. 2018a; Shah et al. 2018; Rotroff et al. 2018b; Shah et al. 2016). Baseline lipid levels were the first trait mapped in ACCORD because the genetic etiology of lipid levels is well studied in non-diabetic people, with numerous studies and meta-analyses on hundreds of thousands of individuals replication (Marvel et al. 2017). We use the data for low density lipoprotein (LDL) levels to demonstrate our approach. Data from chromosome 19 was chosen for the current analysis because there are confirmed SNPs that are consistently associated with LDL levels.

Details of the data are described in Marvel et al. (2017). Prior to analysis, quality control procedures were performed. For SNP level quality control, only SNPs with high variant call rates (95%) and minor allele frequency (5%) are kept. SNPs are also required to follow the expected proportions of Hardy-Weinberg equilibrium. A z-test for Hardy-Weinberg equilibrium is conducted for each SNP. If $|z_{\text{HWE}}| < |F^{-1}(0.5 \cdot 10^{-6})|$, the SNP will be filtered. Following Waldmann et al. (2013), we replace missing alleles with the average allele frequency. For sample-level quality control, coefficients of inbreeding are calculated to filter samples with common ancestors. After quality control and genotype imputation, association of common genetic variants is tested using penalized multiple linear regression models and our proposed higher criticism tuned penalized regression models. In GWAS, the choice of covariates can significantly affect results. We use the 26 phenotype observations employed in Marvel et al. (2017) as covariates of multiple linear regression models, including the top three principal components that summarize population substructure (Price et al. 2006). Complete-case analysis was used. Finally, we have $n = 4540$ individuals, $p = 5746$ SNPs and 29 covariates. There is no penalty effect put on these 29 covariates, which means they will always remain in the model. All four approaches were run on the data, and complete results are shown in Table 2.7-2.8.

In Table 2.7, we calculate the estimated proportion to be $\hat{\pi} = 0.00536 = 0.536\%$. Thus the estimated number of significant SNPs is 30.807. Applying our estimation, we narrow the region for tuning parameter λ to $[\lambda_{\min}, \lambda_{\max}]$, as shown in Table 2.7. While each method found a slightly different list of SNPs, with highly consistent results presented in bold. Here we highlight that all approaches identify rs445925, which is within the Apolipoprotein C1 (APOC1) gene. This SNP has been reported to be associated with LDL in previous publications, but was not identified when we tested for associations using traditional regression implemented locus-by-locus and then multiple testing procedures (Shatwan et al. 2018; Tang et al. 2015; Deshmukh et al. 2012; Trompet et al. 2011; Marvel et al. 2017). APOC1 encodes a member of the apolipoprotein C1 family. It has been reported that its encoded protein plays a central role in high density lipoprotein (HDL) and very low density

Table 2.7: Estimated proportion $\hat{\pi}$ and new tuning region $\{\lambda_{\min}, \lambda_{\max}\}$ for GWAS on baseline LDL and Chr 19 from ACCORD data, where $p = 5746$ and the number of covariate $q = 29$.

Estimated Proportion	$n = 4540$
$\hat{\pi}$	0.00536
$\{\lceil \hat{\pi} p \rceil, \lfloor \hat{\pi} p \rfloor\}$	{30, 31}
Models	$\{\lambda_{\min}, \lambda_{\max}\}$
Lasso	{1.218, 1.276}
Adaptive Lasso	{22.931, 24.022}
SCAD	{1.264, 1.303}
MCP	{1.226, 1.264}

lipoprotein (VLDL) metabolism (Jong et al. 1999; O’Leary et al. 2015). This protein has also been shown to inhibit cholesteryl ester transfer protein in plasma (de Barros et al. 2009; O’Leary et al. 2015). Our discovery demonstrates the potential of the method to identify genetic association while considering correlation among biomarkers while higher power as compared to more traditional approaches. All selected SNPs are listed in Table 2.8. The selection differences among four methods are from the differences of solution paths for each respective penalized regression method.

2.4.2 Continuous Predictor: DNA Methylation Data

In contrast to the categorical nature of the SNP data, differential methylation at Cytosine-phosphate-Guanine (CpG) sites across the genome is measured as a continuous trait that ranges from 0 to 1. As an application on continuous predictors, we evaluated data from an epigenome-wide association study (EWAS) for smoking related DNA methylation data in the Agricultural Lung Health Study, a case-control study of adult asthma nested within an agricultural cohort. After data cleaning, our analysis includes IlluminaEPIC data from 2286 individuals. We investigate associations between pack-years, (packs smoked per day) \times (years as a smoker), and methylation β values at CpG sites in Chromosome 5. This region was selected because it included 153 CpG sites mapped to the Aryl hydrocarbon receptor repressor (*AHRR*) and the PDZ and LIM domain protein 7 (*PDLIM7*) genes (R package *IlluminaHumanMethylationEPICmanifest*) that have validated associations across multiple studies (Joehanes et al. 2016). Additionally, to demonstrate our method on a similar level of dimensionality as the SNP data, we included selected CpG sites on the chromosome

Table 2.8: Selected SNPs in different regression models, $n = 4540$. Overlapped SNPs in four models are in **bold**.

Model	λ	Selected SNPs
Lasso	1.222	rs475192, rs62106026 , rs72999871 , rs77071715 , rs7254996 , rs34002820 , rs7255589 , rs2106917 , rs10414987, rs75981480 , rs1687983 , rs4806078, rs73037271 , rs17265865 , rs892594 , rs6859, rs445925 , rs11671132 , rs251683 , rs77059600 , rs3745495 , rs11084080, rs2288868, rs10420138, rs62110397 , rs200657736 , rs3745902, rs28506185 , rs8102873, rs1548476 , rs7257872
Adaptive Lasso	22.931	rs263057, rs62106026 , rs60357057, rs72999871 , rs77071715 , rs7254996 , rs34002820 , rs7255589 , rs2106917 , rs4806230, rs75981480 , rs11668916, rs1687983 , rs73037271 , rs17265865 , rs892594 , rs445925 , rs10426962, rs11671132 , rs251683 , rs77059600 , rs3745495 , rs73067324, rs10405559, rs62110397 , rs200657736 , rs28506185 , rs1548476 , rs10413455, rs55655541, rs7257872
SCAD	1.264	rs475192, rs62106026 , rs72999871 , rs77071715 , rs7254996 , rs34002820 , rs7255589 , rs2106917 , rs10414987, rs75981480 , rs1687983 , rs4806078, rs73037271 , rs17265865 , rs892594 , rs6859, rs445925 , rs11671132 , rs251683 , rs77059600 , rs3745495 , rs11084080, rs2288868, rs10420138, rs62110397 , rs200657736 , rs3745902, rs28506185 , rs8102873, rs1548476 , rs7257872
MCP	1.231	rs475192, rs62106026 , rs79038264, rs72999871 , rs77071715 , rs7254996 , rs34002820 , rs7255589 , rs2106917 , rs10414987, rs75981480 , rs1687983 , rs4806078, rs73037271 , rs17265865 , rs892594 , rs445925 , rs11671132 , rs251683 , rs77059600 , rs3745495 , rs11084080, rs2288868, rs10420138, rs62110397 , rs200657736 , rs3745902, rs28506185 , rs8102873, rs1548476 , rs7257872

so that the number of sites, p , is 5000.

Prior to analysis, quality control procedures were performed. We limit extreme values by winsorizing top and bottom 5% β values at each CpG site. Missing DNA methylation β values are imputed by using observed average values. After quality control and imputation, association between pack year and DNA methylation β values is tested using penalized multiple linear regression models and our proposed higher criticism tuned penalized regression models. Similar to GWAS, the choice of covariates also plays an important role in EWAS. We use 20 variables (see Table 2.11 for details) as covariates in multiple linear regression models, such as age, gender, principal components, and cell-type proportion. In summary, we have $n = 2286$ individuals, $p = 5000$ β values and 20 covariates. There is no penalty effect put on these 20 covariates, which means they will always remain in the model. We conduct multiple EWAS with the cleaned sample $n = 2286$ to evaluate our proposed approach. In Table 2.9, we calculate the estimated proportion to be $\hat{\pi} = 0.000807 = 0.0807\%$. Thus the estimated number of significant CpG sites is 4.036. Applying our estimation, we narrow the region for tuning parameter λ to $[\lambda_{\min}, \lambda_{\max}]$, as shown in Table 2.9. All detected CpG sites are listed in Table 2.10. Those CpG sites in at least three models are in bold. The **cg05575921** site in the aryl hydrocarbon receptor repressor (AHRR) gene, a CpG gene consistently differentially methylated in relation to smoking across many studies, along with other CpGs in this gene (Monick et al. 2012; Joehanes et al. 2016). We detected another well established CpG associated with smoking: **cg23576855** (also from AHRR gene). The association of **cg23576855** and smoking has been previously reported by Philibert et al. (2012) and in the large meta-analysis of Joehanes et al. (2016). This demonstrates the ability of our method, which considers correlation, to jointly detect well established associations in quantitative data.

2.5 Conclusions

For high-dimensional data, variable selection is always interesting but challenging. Penalized regression methods are preferred in high-dimensional setting because (1) they are more efficient than stepwise information criteria methods; and (2) they can consider effects of all variables simultaneously. The challenge is to find an appropriate penalty. In this article, we propose a new searching scheme for the regularization parameter λ in penalized regression. We develop an algorithm to calculate p -values in high-dimensional data through data multi-splitting and p -value combination. We demonstrate that our approach

Table 2.9: Estimated proportion $\hat{\pi}$ and new tuning region $\{\lambda_{\min}, \lambda_{\max}\}$ for EWAS from DNA Methylation Data on Chr 5, where $p = 5000$ and the number of covariate $q = 20$.

Estimated Proportion	
$\hat{\pi}$	$n = 2286$
$\{\lfloor \hat{\pi} p \rfloor, \lceil \hat{\pi} p \rceil\}$	0.0807%
	{4, 5}
Models	
	$\{\lambda_{\min}, \lambda_{\max}\}$
Lasso	{1.133, 1.498}
Adaptive Lasso	{1651.226, 1988.908}
SCAD	{1.123, 1.307}
MCP	{1.158, 1.268}

Table 2.10: Selected CpG sites in different regression models, $n = 2286$. Overlapped CpG sites in at least three models are in **bold**.

Model	λ	Selected CpGs
Lasso	1.133	cg05575921 , cg08916839 , cg11554391, cg23576855 , cg13032951
Adaptive Lasso	1651.226	cg05575921 , cg23953133, cg24965308, cg06016466, cg18174928
SCAD	1.123	cg05575921 , cg08916839 , cg23576855 , cg20075683, cg13039251
MCP	1.158	cg05575921 , cg08916839 , cg23576855 , cg20075683, cg13039251

is applicable to multiple penalized regression approaches. These results also compare the performance of each of the different penalized regression methods on genetic applications.

The estimator constructed with p -values are used as a key step in our algorithm. We demonstrate the performance of our algorithm using both simulation tests and real data applications. In simulation tests, we prove that our method is robust for both weak and sparse signals, especially compared to other approaches. Using real data, we demonstrate the utility of our method in a real life setting. We analyze Chromosome 19 in ACCORD data and detect rs44592 whose association with LDL has been reported in previous publications. In epigenome-wide association study (EWAS), we detect association between smoking and 5,000 CpG sites in Chromosome 5 and report some sites that were reported in other publications. Finally, we provide an R package on CRAN to implement our proposed approaches, which is available at <https://cran.r-project.org/web/packages/HCTR/index.html>.

2.6 Supplementary Information

2.6.1 Covariates in Regression Model for DNA Methylation Data Applications

Table 2.11: A list of covariates in regression model for DNA methylation data applications in Section 2.4.2

Covariate	Unit	Note
Age	Years	Numerical
Gender	Male/Female	Categorical
Body mass index	kg/m^2	Numerical
Asthma status	Case/Noncase	Categorical
10 ancestry principal components		Numerical
Estimated cell type proportions: CD8+ and CD4+ T cells, NK cells, B cells, monocytes, granulocytes		Numerical

The cell type proportions were estimated using the method described by Houseman et al. (2012) with the reference panel introduced by Reinius et al. (2012).

CHAPTER

3

KNOCKOFF BOOSTED TREE FOR MODEL-FREE VARIABLE SELECTION

In this article, we propose a novel strategy for conducting variable selection without prior model topology knowledge using the knockoff method with boosted tree models. Our method is inspired by the original knockoff method, where the differences between original and knockoff variables are used for variable selection with false discovery rate control. The original method uses Lasso for regression models and assumes there are more samples than variables. We extend this method to both model-free and high-dimensional variable selection. We propose two new sampling methods for generating knockoffs, namely the sparse covariance and principal component knockoff methods. We test these methods and compare them with the original knockoff method in terms of their ability to control type I errors and power. The boosted tree model is a complex system and has more hyperparameters than models with simpler assumptions. In our framework, these hyperparameters are either tuned through Bayesian optimization or fixed at multiple levels for trend detection. In simulation tests, we also compare the properties and performance of importance test statistics of tree models. The results include combinations of different knockoffs and importance test statistics. We also consider scenarios that include main-effect, interaction, exponential,

and second-order models while assuming the true model structures are unknown. We apply our algorithm for tumor purity estimation and tumor classification using the Cancer Genome Atlas (TCGA) gene expression data. The proposed algorithm is included in the KOBT package, available at <https://cran.r-project.org/web/packages/KOBT/index.html>.

3.1 Introduction

In nearly all existing variable selection methods, a set of predictor candidates must be specified before selection (Li et al. 2005). Predictors in the final selected model are a subset of candidate set. However, it can be challenging to determine an appropriate set of predictor candidates, especially considering the order (e.g., quadratic, cubic) of main effects and interactions with a large number of variables. Li et al. (2005) stated the difference between model selection and variable selection is whether to assume a model for comparison. One of the advantages of model-free variable selection is that its performance is not limited by the choice of model formulation. Model-free methods such as Random Forest (Breiman 2001), AdaBoost (Rätsch et al. 2001), and convolutional neural networks (LeCun et al. 1995) are widely used in research and industry for regression and classification purposes. These methods recognize data patterns without the need to impose specific model structures on regression functions. This independence allows model-free methods more flexibility, thus making them highly appropriate in many cases. For example, Gao et al. (2018) reported that compared with other methods, model-free methods provide more reliable clinical outcomes when forecasting falls of Parkinson's patients.

Among the widely used model-free methods, tree-based models have demonstrated greater effectiveness and interpretability than most other learning algorithms. In a standard regression model, an interaction term is not estimated if it is not specified in the model. In contrast, tree-based methods automatically include interactions during tree growing, without the requirement of an *a priori* formula preconception (Su et al. 2011). Further, tree-based models are more naturally associated with nonlinear interactions (Schiltz et al. 2018), and the branches in a tree model are original interaction variables. Friedman (2001) stated that the number of terminal nodes of a boosted tree model should depend on the highest order of the dominant interactions among the variables. This implies that (1) terminal (leaf) nodes in tree models cannot be interpreted simply as main effects of original variables and (2) growth of a tree model is selective.

Variable selection in tree models is affected by the number of categories of features (Kim

and Loh 2001), with variables with more categories preferred. To eliminate bias in variable selection, Loh (2002) proposed the GUIDE algorithm, which is used to conduct chi-square analysis of residuals and bootstrap calibration of significance probabilities. Gini importance in random forest methods (Breiman 2001) is used to measure improvement in the splitting criterion produced by each corresponding variable. To generate an unbiased importance index, Breiman and Cutler (2008) proposed permutation importance by statistical permutation. In addition to regression tree and random forest models, attention has also been paid to variable selection in boosted tree models, given their superior performance in data pattern recognition. Miller et al. (2016) extended gradient boosted tree to multivariate tree boosting and performed nonparametric regression to identify important variables. Gain, weight, and cover are three common importance ranking measures in the highly successful XGBoost (Chen and Guestrin 2016). SHapley Additive exPlanation (SHAP) (Lundberg and Lee 2017b) has also been proposed to interpret model predictions, and its performance has been highly consistent when applied in tree models (Lundberg et al. 2018).

While most data scientists are interested in model prediction accuracy, inference from analysis is also important. Originally described for Random Forest (Breiman 2001), the permutation method has been widely used to generate a null distribution of test statistics to detect significance in tree models. Unfortunately, some statistical inference, such as accurate false discovery rate (FDR) control, is not possible with permutation-based methods. For example, in a linear regression problem, a permutation-based construction may underestimate the false discovery proportion (FDP) in cases where the design matrix displays nonvanishing correlations (Barber et al. 2015). Under a global null, p -values from permutation tests are exact. However, under an alternative hypothesis, rejecting the null based on permutation tests does not necessarily imply a valid rejection (Chung et al. 2013). To control the FDR in variable selection, Barber et al. (2015) proposed a novel statistical framework, the *knockoff* framework, for low-dimensional ($n \geq p$) fixed design. The *knockoff* framework was later extended to high-dimensional ($n < p$) and random design scenarios (Candes et al. 2018). The method generates knockoff variables without the need to collect new data or consider response variables while retaining the correlation structure of the original variables. These knockoff variables are used as a negative control group, and a variable selection procedure is applied for both the original and knockoff variables. Test statistics measuring the importance of both kinds of variables can be compared to compute symmetrized knockoff statistics. Finally, a threshold for these statistics is determined according to the (desired) target FDR level. The goal of using the *knockoff* method is discovering relevant variables while controlling the FDR.

In this article, we propose the knockoff boosted tree (KOBT) algorithm, a model-free variable selection algorithm that includes boosted tree (Friedman 2001; Chen and Guestrin 2016) and model-X knockoffs (Candes et al. 2018). Model-X knockoff variables are generated with the same correlation structure as the original variables. Test statistics, such as tree SHAP (Lundberg et al. 2018), Cover, Frequency, Gain, and Saabas are generated for each pair of original and knockoff variables, and the consistency and accuracy of the produced test statistics are compared. The FDR is controlled using the difference between original and knockoff test statistics. Further, regularization parameters in boosted tree models are tuned through Bayesian optimization (Snoek et al. 2012).

In the following sections, we describe our analytical framework, including (1) the generation of knockoff variables; (2) fitting and Bayesian optimization in boosted tree regression; and (3) consistency, accuracy, and FDR control in model-free variable selection. We validate our approach with simulation experiments. Finally, we demonstrate our approach on tumor purity estimation and tumor classification using the Cancer Genome Atlas (TCGA) gene expression data.

3.2 Methods and Materials

3.2.1 Parametric Functions of a Single Regression Tree

Denote $\mathbf{y}_{n \times 1}$ as n responses and $\mathbf{X}_{n \times p}$ as p independent variables each with n values. In regression trees, $\mathbf{y}_{n \times 1}$ can be continuous or ordered discrete. The goal is to minimize an objective function, which may contain a loss function component and a regularization component, for a combination of all leaves (terminal nodes) on a tree. Given a tree with m terminal nodes (i.e., the whole sample contains m partitions), if we model the response as a constant in each region, the estimated regression tree function is defined as

$$\hat{y}_i = \hat{f}(\mathbf{x}_i) = \sum_{k=1}^m \mathbf{I}\{\mathbf{x}_i \in \mathcal{R}_k^p\} \beta_k, \quad (3.1)$$

where $\mathbf{I}(\cdot)$ is a function of \mathbf{x} , and $\mathbf{I}\{\mathbf{x}_i \in \mathcal{R}_k^p\} = 1$, if the i -th row vector of \mathbf{X} , $\mathbf{x}_i \in \mathcal{R}_k^p$, the k -th disjoint partition region and otherwise $\mathbf{I}\{\mathbf{x}_i \in \mathcal{R}_k^p\} = 0$. \mathcal{R}_k^p is a split region or a so-called leaf, $k = 1, \dots, m$. With the squared $L2$ norm loss function, we have estimator

$$\hat{\beta} = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^n \left\{ y_i - \sum_{k=1}^m \mathbf{I}\{\mathbf{x}_i \in \mathcal{R}_k^p\} \beta_k \right\}^2, \quad (3.2)$$

and

$$\partial \sum_{i=1}^n \left\{ y_i - \sum_{k=1}^m \mathbf{I}\{\mathbf{x}_i \in \mathcal{R}_k^p\} \beta_k \right\}^2 / \partial \beta_k = 0. \quad (3.3)$$

Solving Equation 3.3, the k -th estimated parameter of β is

$$\hat{\beta}_k = \frac{\sum_{i=1}^n y_i \mathbf{I}_k \{\mathbf{x}_i \in \mathcal{R}_k^p\}}{\sum_{i=1}^n \mathbf{I}_k \{\mathbf{x}_i \in \mathcal{R}_k^p\}}, \quad (3.4)$$

where $\mathbf{I}_k \{\mathbf{x}_i \in \mathcal{R}_k^p\} = 0$ for a fixed \mathcal{R}_k^p . This is the sample mean of \mathbf{y} on that leaf of the regression tree. Meanwhile, to grow a tree model, the data need to be partitioned or split. For the j -th variable \mathbf{x}_j , we define a split on \mathbf{x}_j with some value c as

$$\begin{aligned} \mathcal{R}_{k,<}^p &= \{i : x_{ij} < c | \mathbf{x}_i \in \mathcal{R}_k^p\}, \\ \mathcal{R}_{k,>}^p &= \{i : x_{ij} \geq c | \mathbf{x}_i \in \mathcal{R}_k^p\}. \end{aligned} \quad (3.5)$$

Based on Equation 3.4, we have two estimates

$$\begin{aligned} \hat{\beta}_{k,<} &= \frac{\sum_{i=1}^n y_i \mathbf{I}_k \{\mathbf{x}_i \in \mathcal{R}_{k,<}^p\}}{\sum_{i=1}^n \mathbf{I}_k \{\mathbf{x}_i \in \mathcal{R}_{k,<}^p\}}, \\ \hat{\beta}_{k,>} &= \frac{\sum_{i=1}^n y_i \mathbf{I}_k \{\mathbf{x}_i \in \mathcal{R}_{k,>}^p\}}{\sum_{i=1}^n \mathbf{I}_k \{\mathbf{x}_i \in \mathcal{R}_{k,>}^p\}}. \end{aligned} \quad (3.6)$$

Thus, an optimal split is defined as

$$\hat{c} = \operatorname{argmin}_c \left\{ \sum_{i=1}^n (y_{i \in \mathcal{R}_{k,<}^p} - \hat{\beta}_{k,<})^2 + \sum_{i=1}^n (y_{i \in \mathcal{R}_{k,>}^p} - \hat{\beta}_{k,>})^2 \right\}. \quad (3.7)$$

To avoid overfitting, similar to penalized regression, a penalty term is added to the cost-complexity criterion of a regression tree model

$$\sum_{k=1}^m \|\mathbf{y}_{i \in \mathcal{I}_k} - \mathbf{1} \hat{\beta}_k\|_2^2 + \gamma |T|, \quad (3.8)$$

where $T(\mathbf{X}_{n \times p}; \hat{\cdot}) = f(\mathbf{X}_{n \times p})$ is a regression tree model, $\hat{\cdot} = \{\mathcal{R}, \beta\}$ contains tree structure parameters, $|T|$ is the cardinality of terminal nodes in T , and $\gamma \geq 0$ is a tuning parameter.

3.2.2 Boosted Tree Methods and Regularization

To overcome the bias and high-variance problem in a single regression tree, ensemble tree models, such as bagging or boosting structures, have been proposed. Defining a boosted tree model as a sum of B trees in Section 3.2.1,

$$f^B(\mathbf{X}) = \sum_{b=1}^B T(\mathbf{X}_{n \times p}; \Theta_b) = \sum_{b=1}^B f_b(\mathbf{X}_{n \times p}). \quad (3.9)$$

In a forward stagewise algorithm (Hastie et al. 2005), the B -th tree structure is found by solving

$$\hat{\Theta}_B = \underset{\Theta_B}{\operatorname{argmin}} L(\mathbf{y}, f^{B-1}(\mathbf{X}) + T(\mathbf{X}, \Theta_B)), \quad (3.10)$$

where $L(\Theta_B; \mathbf{y}, f^{B-1})$ is a loss function based on the B -th tree structure, given response \mathbf{y} and previously fitted $B - 1$ tree models f^{B-1} .

In boosted tree model fitting, the objective function usually contains a loss function and penalty function for tree complexity (i.e., regularization for tree models). The complexity depends on the number of trees in a sequence, the depth of each tree, and the number of terminal nodes in each tree. One way to control the complexity of tree model fitting is to set a minimum number of instances in a single terminal node. For example, the *min_child_weight* parameter in XGBoost (Chen and Guestrin 2016) controls the minimum sum of instance weight (Hessian) in a terminal node, which is equivalent to a minimum number of instances if all instances have a weight of 1. For simplicity, here, we assume all weights are 1. Combining Equation 3.8 and Equation 3.10, the B -th objective function is defined as

$$\begin{aligned} \text{Obj}(f_B; \mathbf{y}, f^{B-1}) &= L(\mathbf{y}, f^{B-1}(\mathbf{X}) + T(\mathbf{X}, \Theta_B)) + \sum_{b=1}^B \Omega(f_b) \\ &\propto L(\mathbf{y}, f^{B-1}(\mathbf{X}) + T(\mathbf{X}, \Theta_B)) + \Omega(f_B) \\ &= L(\mathbf{y}, f^{B-1}(\mathbf{X}) + T(\mathbf{X}, \Theta_B)) + \gamma |T(\mathbf{X}, \Theta_B)|, \end{aligned} \quad (3.11)$$

where $|T(\mathbf{X}, \Theta_B)|$ is the cardinality of terminal nodes in the B -th tree, and $\gamma \geq 0$ is a tuning parameter. Similar to elastic net (Zou and Hastie 2005), XGBoost introduces $L1$ and $L2$ -norm penalty terms into objective functions. Therefore, the objective function for the B -th tree can be updated as

$$\text{Obj}(f_B; \mathbf{y}, f^{B-1}) \propto L(\mathbf{y}, f^{B-1}(\mathbf{X}) + T(\mathbf{X}, \Theta_B)) + \gamma |T| + \lambda \|\beta_B\|_2^2 + \alpha \|\beta_B\|_1, \quad (3.12)$$

where β_B is a vector of leaf weights in the B -th tree, and $\gamma \geq 0$, $\lambda \geq 0$, and $\alpha \geq 0$ are tuning parameters.

Besides setting the minimum number of instances in terminal nodes and applying penalization on node weights, regularization can be achieved by limiting the maximum depth of a tree, applying the maximum number of trees in a boosting sequence, or defining an early stopping criteria for boosting (Zhang et al. 2005). Given the numerous parameters and hyperparameters of regularization and tree structure, it is tedious to find the optimal group of parameters by grid search. The parameters and their optimization are summarized in the following section.

3.2.3 Regularization Parameters and Bayesian Optimization

As reviewed by Nielsen (2016), there are three kinds of regularization parameters: boosting parameters, randomization parameters, and tree parameters. Boosting parameters include the number of trees B (i.e., the number of boosting iterations) and the learning rate η . A small step-size of the learning rate has been found to play an important role in the convergence of boosting procedures (Zhang et al. 2005). In simulation studies, decreasing the learning rate increased the performance of boosted models (Friedman et al. 2000). Using a smaller learning rate and thus a relatively larger number of trees has been suggested, given the relationship between the learning rate and the number of boosted models. A drawback of this procedure is increased running time for model fitting. The randomization parameters refer to the row subsampling parameter and the column subsampling parameter. In the random forest algorithm (Breiman 2001), only a random subsample of the training data set is used for building model. Friedman (2002) built on this idea and introduced stochastic gradient boosting for the boosting tree algorithm. Basically, the row subsampling parameter controls the ratio for a subset of the data, and the column subsampling parameter determines the ratio for a subgroup of features in each tree fitting. Both boosting parameters and randomization parameters are used for general regularization since they are used in the optimization of all kinds of boosting models. However, tree parameters are for tree models only. Therefore, in this article, the discussion is focused on tree parameters and their optimization.

Note that all the tree parameters implicate a tradeoff between bias and variance. In each individual tree, besides the penalization parameters γ , λ , and α in Equation 3.12, the tree parameters also include structure parameters, the maximum depth of the tree, and the minimum sum of observation weights required for each leaf. Given the maximum depth of

a tree, D , using the decomposition of target function introduced in Friedman et al. (1991), the b -th individual tree function in Equation 3.9 can be defined as

$$f_b(\mathbf{X}_{n \times p}) = \sum_j g^{(1)}(\mathbf{x}_j) + \sum_{j,k} g^{(2)}(\mathbf{x}_j, \mathbf{x}_k) + \sum_{j,k,l} g^{(3)}(\mathbf{x}_j, \mathbf{x}_k, \mathbf{x}_l) + \dots + \sum_{j,k,l,\dots} g^{(D)}(\mathbf{x}_j, \mathbf{x}_k, \mathbf{x}_l, \dots), \quad (3.13)$$

where $g^{(1)}(\mathbf{x}_j)$ is the first order interaction (main effect) of the j -th variable, \mathbf{x}_j , $g^{(2)}(\mathbf{x}_j, \mathbf{x}_k)$ is the second order interaction between \mathbf{x}_j and \mathbf{x}_k , and so on. From Equation 3.13, it is clear that the highest order of interaction is the maximum depth of a tree, D . The minimum sum of observation weights in a leaf decides the variance of $\hat{\beta}$'s, estimated weights of leaves in a tree. If the minimum sum is large, the growing of a tree will be conservative, which means fewer leaves will be grown and thus a smaller variance of $\hat{\beta}$'s.

Finally, we discuss how Bayesian optimization (Snoek et al. 2012) is applied for tuning γ , λ , and α in Equation 3.12. We choose Bayesian optimization because brute-force search such as grid search or random search is time-consuming in a scenario where there are three hyperparameters. Here we treat hyperparameter optimization as a black-box process. We define a configuration of tuning hyperparameters, $\theta = (\gamma, \lambda, \alpha)$, for the function we want to minimize,

$$\text{CVTE}(\gamma, \lambda, \alpha | \mathbf{y}, \mathbf{X}) = \frac{1}{n} \sum_{i=1}^n (y_i - f^B(\mathbf{x}_i))^2, \quad (3.14)$$

where $\text{CVTE}(\cdot)$ is a K -fold cross-validation error score, \mathbf{x}_i is a row vector containing p features of an observation, and $f^B(\cdot)$ is the fitted boosted tree model with B individual trees. An early stopping criteria states that if adding new trees does not decrease cross-validation error within five trees, the boosting iteration should be stopped before the maximum number of trees is reached. The value of B is equal to the number of trees in the sequence, where the combination of trees provides the lowest cross-validation error score. This error score is the output value from $\text{CVTE}(\cdot)$ in Equation 3.14. The support sets of γ , λ , and α are located within a tuning region, $[0, 20]$. The optimal combination is used in the final model for comparison. Details are provided in Section 3.3.1.

3.2.4 Knockoff Variables in Boosted Tree Models

Barber et al. (2015) proposed the knockoff filter, a new variable selection method that controls the FDR. The knockoff filter generates knockoff variables that mimic the correlation structure of original variables but are not associated with the response. These knockoff variables are used as controls for the original variables, so only the original variables that

are highly associated with the response are selected. The knockoff filter has been shown to provide accurate FDR control, which cannot be realistically achieved using permutation methods. Barber et al. (2015)'s filter works under the following conditions, where for a fixed design, $\mathbf{X}_{n \times p}$, $n > p$, and \mathbf{y} follows a linear Gaussian model. Candès et al. (2018) introduced model-X knockoffs, which extended the model assumption to high-dimensional and covariate selection from random known distributions. Following the definition of model-X knockoffs in Candès et al. (2018), we restate the definition for boosted tree models here

Definition 3.2.1 For a row vector of p random variables $\mathbf{x}_{1 \times p} = (x_1, x_2, \dots, x_p)$, where each x_j is a random variable that represents a feature, the corresponding model-X knockoff variables $\mathbf{z}_{1 \times p} = (z_1, z_2, \dots, z_p)$ are constructed such that:

1. for a combined random vector $(\mathbf{x}, \mathbf{z})_{1 \times 2p}$, if its j -th random variable is switched with its $(j + p)$ -th random variable ($j = 1, \dots, p$) (i.e, an original variable is switched with its knockoff counterpart), the distribution of the new random vector is invariant to $(\mathbf{x}, \mathbf{z})_{1 \times 2p}$; and
2. $\mathbf{z}_{1 \times p} \perp \mathbf{y} | \mathbf{x}_{1 \times p}$, where \mathbf{y} is the response.

Based on the definition, to achieve high-dimensional FDR-controlled variable selection, qualified knockoff variables should satisfy two properties: (1) distribution equality with original variables and (2) independence of response. Two sampling methods, namely exact constructions and approximate construction, have been introduced to generate knockoffs (Candès et al. 2018). For exact construction, a new knockoff variable z_j is sampled from the conditional distribution $L(x_j | x_{-j}, z_{1:(j-1)})$, where $j = 1, \dots, p$, and $x_{-j} = (x_{1:(j-1)}, x_{(j+1):p})$. Approximate construction focuses on whether $(\mathbf{x}, \mathbf{z})_{1 \times 2p}$ retains its first two moments of a distribution after swapping. Given this summary of the two methods, it is apparent that the approximate construction method requires less complex computations than the exact construction method.

In this article, we compare three algorithms for knockoff generation. The first algorithm, *Approximate Construction* (AC), is available in the *knockoff* R package (Candès et al. 2018), where the covariance matrix of original variables, $\text{cov}(\mathbf{X}_{n \times p})$, is estimated directly. The estimated covariance matrix is shrunk to the identity matrix if it is not positive definite (Opgen-Rhein and Strimmer 2007). In addition, we propose two other algorithms: one that is similar to the AC algorithm described above and one without Gaussian assumption. The second algorithm (i.e., the one that is similar to the AC algorithm) is a Sparse Constructions

(SC) algorithm. Instead of estimating the covariance matrix directly, we conduct sparse estimation of the covariance matrix (Bien and Tibshirani 2011), which generates a sparse matrix with a simpler structure. The rest of the algorithm is the same as the AC algorithm. The third algorithm (i.e., the one without Gaussian assumption) is a *Principal Component Constructions* (PCC) algorithm, which is inspired by Algorithm A.1 in Shen et al. (2019). In accordance with Definition 3.2.1, we describe our proposed PCC algorithm in Algorithm 3.2.1 below. Unlike the AC and SC algorithms, the PCC algorithm does not require data with a Gaussian assumption.

Algorithm 3.2.1 *Principal Component Construction (PCC)*

For each column vector of original variable \mathbf{x}_j , where $j = 1, \dots, p$,

1. *Conduct principal component analysis on matrix $(\mathbf{X}_{-j}, \mathbf{Z}_{1:j-1})_{n \times (p+j-2)}$, where \mathbf{X}_{-j} is matrix \mathbf{X} without the j -th column and $\mathbf{Z}_{1:j-1}$ is the first $(j-1)$ columns in matrix \mathbf{Z} . When $j = 1$, $\mathbf{Z}_{1:j-1}$ is empty.*
2. *Denote K as the number of principal components chosen for a regression model, $K = 1, \dots, n-1$. For a fixed K , fit \mathbf{x}_j on K PCs. There is a tradeoff in that the larger the K , the more akin the knockoff will be to the original variables. This results in a smaller type 1 error but weaker power of the test.*
3. *Compute a residual vector $\boldsymbol{\varepsilon}_n = (\mathbf{x}_j - \hat{\mathbf{x}}_j)$.*
4. *Permute $\boldsymbol{\varepsilon}_n$ randomly. Denote the permuted vector as $\boldsymbol{\varepsilon}_n^*$.*
5. *Set $\mathbf{z}_j = \hat{\mathbf{x}}_j + \boldsymbol{\varepsilon}_n^*$, and combine it with the current knockoff matrix $\mathbf{Z}_{1:j-1}$.*

This algorithm was designed in accordance with our definition of knockoff. Using linear regression models, the empirical conditional distribution of \mathbf{x}_j , $L(\mathbf{x}_j | \mathbf{X}_{-j}, \mathbf{Z}_{1:j-1})$ can be estimated. Using permutation, we eliminate the Gaussian assumption. Meanwhile, the generated \mathbf{z}_j is independent of the response \mathbf{y} , since \mathbf{y} is ignored in our algorithm. Note that this is a sequential algorithm, so computation could be expensive. In Shen et al. (2019), the residuals are assumed to be approximately independently and identically distributed. We keep this assumption for our algorithm.

To evaluate knockoffs generated by these methods, we propose the *Mean Absolute Angle of Columns* (MAAC) metric for checking vector independence, and use the *Kernel Maximum Mean Discrepancy* (KMMD) metric to test distribution similarity (Gretton et al. 2007).

Definition 3.2.2 *Mean Absolute Angle of Columns (MAAC)*

For two column vectors with the same length, \mathbf{x} and \mathbf{z} , we define

$$MAAC(\mathbf{x}, \mathbf{z}) = \arccos \left| \frac{\mathbf{x}^\top \mathbf{z}}{\|\mathbf{x}\| \cdot \|\mathbf{z}\|} \right|. \quad (3.15)$$

For two matrices with the same dimensions, \mathbf{X} and \mathbf{Z} , with p columns, we define

$$MAAC(\mathbf{X}, \mathbf{Z}) = \frac{1}{p} \sum_{j=1}^p \arccos \left| \frac{\mathbf{x}_j^\top \mathbf{z}_j}{\|\mathbf{x}_j\| \cdot \|\mathbf{z}_j\|} \right|. \quad (3.16)$$

MAAC indicates the correlation between corresponding columns in two matrices. For knockoff variable selection, we know that the weaker the correlation, the more powerful the test. The KMMD metric is used to perform a non-parametric distribution test. The null hypothesis test, H_0 , is that the row vectors \mathbf{x}_i in \mathbf{X} and \mathbf{z}_i in \mathbf{Z} come from the same distribution. In summary, MAAC is for type 2 error control, and KMMD is for type 1 error control. In Section 3.3.2, simulation tests are described, and permutation tests are used for comparison.

3.2.5 Variable Importance Test Statistics in Tree Models

For each original and knockoff variable pair, Barber et al. (2015) suggested the *Lasso signed max* (LSM) statistic as the test statistic for variable selection. Candès et al. (2018) later proposed the *Lasso coefficient difference* (LCD) statistic for the same purpose. While both of these test statistics were designed specifically for Lasso, there are also test statistics designed for tree models. Table 3.1 provides a list of common variable importance test statistics for boosted tree models. Given that different test statistics will provide different results for importance ranking, it is difficult to determine which test statistic is the best choice in a specific scenario.

Among these statistics, SHAP, which is based on game theory and local explanations, has the demonstrated advantage of retaining both consistency and local accuracy (Lundberg and Lee 2017b). Lundberg and Lee (2017a) introduced feature attribution for tree ensembles using the *additive feature attribution methods* defined by Lundberg and Lee (2017b). They also proposed the Tree SHAP algorithm, which reduces the complexity of computing exact SHAP values from $\mathcal{O}(BL2^p)$ to $\mathcal{O}(BLD^2)$, where B is the number of trees, $L = \max(|T_1|, \dots, |T_B|)$ is the maximum number of leaves in any tree, p is the number of vari-

Table 3.1: Available variable importance test statistics in boosted tree models.

Test Statistic	Definition
Cover	The number of times a variable is used, weighted by the amount of training data in the corresponding nodes
Gain (Breiman et al. 1984)	The total loss reduction gained when using a variable
Permutation	The difference of test errors between original variables and corresponding permuted variables
Saabas (Saabas 2014)	The change in the model's expected outputs
Tree SHAP (Lundberg et al. 2018)	The sum of weighted difference of conditional expectations with and without a variable
Weight	The number of times a variable is used

ables, and D is the maximum depth of any tree. Further, Lundberg et al. (2018) extended SHAP values to SHAP interaction values. In *additive feature attribution methods*, $f(\cdot)$ in Equation 3.9 is the original model, and an explanation model, $h(\cdot)$, is defined for $f(\cdot)$ such that

$$h(\mathbf{u}) = \phi_0 + \phi^\top \mathbf{u}, \quad (3.17)$$

where $\mathbf{u} \in \{0, 1\}^p$ is a binary p dimensional column vector, $u_j = 1$ if the j -th variable is observed, $u_j = 0$ if it is missing, and ϕ stands for feature attribution values. The exact value of each ϕ_j can be calculated as

$$\phi_j = \sum_{S \subseteq P \setminus \{j\}} \frac{|S|!(p - |S| - 1)!}{p!} [f_x(S \cup \{j\}) - f_x(S)], \quad (3.18)$$

where P is the set of all variables, S is a subset of P with non-zero indices in \mathbf{u} , and $f_x(S) = E[f(X)|X_S]$ is defined as a conditional expectation. This ϕ_j is the SHAP value for the j -th variable.

Following the concept outlined by Barber et al. (2015) and Candes et al. (2018), we move on to define a test statistic for knockoff variables in tree models that can control the FDR. For simplicity, we use the SHAP value as an example, but other test statistics can be applied to the knockoff variable using a similar procedure. We denote a pair of original and knockoff variables as \mathbf{x}_j and \mathbf{z}_j , which is the same notation used in Section 3.2.4. Accordingly, their respective SHAP values are ϕ_j^x and ϕ_j^z . The hypothesis tests in which we are interested

are H_0 : The original variable, \mathbf{x}_j , and the knockoff variable, \mathbf{z}_j , are equally associated with response; and H_a : the original variable, \mathbf{x}_j , is more closely associated with response than the knockoff variable, \mathbf{z}_j .

Here, the *Shapley explanation absolute difference* (SEAD) statistic is defined as

$$T_j = |\phi_j^x(\gamma, \lambda, \alpha)| - |\phi_j^z(\gamma, \lambda, \alpha)|, \quad (3.19)$$

where γ , λ , and α are hyperparameters used in Equation 3.14. A larger positive T_j indicates that the response is more likely to be associated with the original variable than its knockoff. Under the null hypothesis, T_j is equally likely to be positive or negative. Given that null T_j s are symmetric, for any fixed $t > 0$, the false discovery proportion is

$$\text{FDP}(t) = \frac{\#\{\text{null } j : T_j \geq t\}}{\#\{j : T_j \geq t\}}. \quad (3.20)$$

This can be estimated by

$$\hat{\text{FDP}}(t) = \min \left\{ \frac{\#\{j : T_j \leq -t\}}{\#\{j : T_j \geq t\}}, 1 \right\}, \quad (3.21)$$

where $\hat{\text{FDP}}(t) = 1$ when there are more T_j s below $-t$ than above t . Candès et al. (2018) provided equations for both the modified FDR and the typical FDR. Here, we accept the latter as it is more conservative. A threshold τ is defined such that

$$\tau(t) = \min \left\{ \frac{\#\{j : T_j \leq -t\} + 1}{\#\{j : T_j \geq t\}} \leq \delta \right\}, \quad (3.22)$$

where δ is the level under which we would like to keep the FDR. A set of selected variables is determined as

$$\hat{S}(\tau) = \{j : T_j \geq \tau\}. \quad (3.23)$$

Then, the FDR is controlled as

$$E \left(\frac{|\hat{S} \cap S_0|}{|\hat{S}| \vee 1} \right) \leq \delta, \quad (3.24)$$

where $S_0 \subseteq \{1, 2, \dots, p\}$ is a set of noise variables. The performance of the SEAD statistic in various models is evaluated in the following sections, and test statistics constructed using other methods are tested for comparison.

3.2.6 Knockoff Boosted Tree Algorithm

To perform model-free variable selection with FDR control, we propose a robust multiple-stage KnockOff Boosted Tree (KOBT) algorithm. Figure 3.1 is a flowchart that outlines details of the procedure. The algorithm contains four main steps and one optional step (circled in red): (1) sample knockoff matrix \mathbf{Z} according to original matrix \mathbf{X} ; (2) grow boosted trees using combined predictors (\mathbf{X}, \mathbf{Z}) ; (3) calculate the test statistic, ϕ , for each variable from the fitted boosted tree model, $f_B(\mathbf{X}, \mathbf{Z})$; and (4) conduct steps (1) to (3) q times and get $T_j = \frac{1}{q} \sum_{m=1}^q |\phi_{j,m}^x| - \frac{1}{q} \sum_{m=1}^q |\phi_{j,m}^z|$ for the j -th original variable. A larger positive T_j indicates a variable is more closely associated with response. The optional step is taken in the scenario in which some variables (shown as \mathbf{W} in Figure 3.1) need to be retained in the final model.

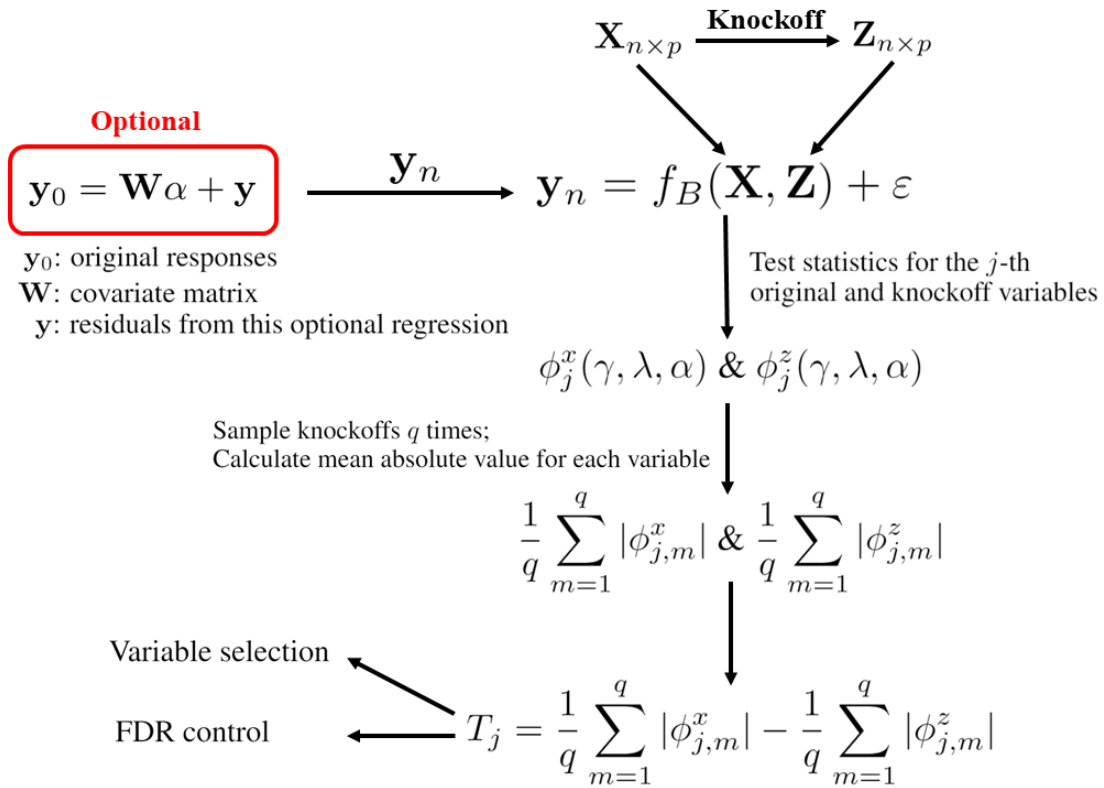


Figure 3.1: Knockoff boosted tree flowchart. An optional step is circled in red.

The first part of the algorithm is optional, and a regression model is added before the

boosted tree model. Given that we want to retain some covariate variables in our final model, we conduct a regression,

$$\mathbf{y}_0 = \mathbf{W}\alpha + \mathbf{y}, \quad (3.25)$$

where \mathbf{y}_0 is the original responses, \mathbf{W} is the covariate matrix, and \mathbf{y} stands for residuals from this optional regression. This \mathbf{y} is treated as new responses for the subsequent boosted tree fitting.

For the first required step of KOBT, we generate knockoff variables \mathbf{Z} conditional on original variables \mathbf{X} . With the assumption that the data in \mathbf{X} follow a Gaussian distribution, the column mean and covariance matrix of \mathbf{X} are estimated. As discussed in Section 3.2.4, there are two choices for covariance matrix estimation: shrinking the estimation to the identity matrix (Ogpen-Rhein and Strimmer 2007) and sparse estimation (Bien and Tibshirani 2011). Without a Gaussian assumption, we propose a strategy based on principal components and permutation (see Section 3.2.4 for details). The properties of knockoff variables are compared in Section 3.3.2.

During the process of model fitting, hyperparameters γ , λ , and α are tuned through Bayesian optimization. Once a boosted model is fitted, test statistics showing importance are calculated. Multiple statistics are considered, such as gain, cover, weight, SHAP value, and Saabas value. Their values are discussed in Section 3.3.3. The above steps, from generating knockoff variables to calculating test statistics, are repeated q times to achieve the mean absolute value for each variable, where q should be sufficiently large. According to the strategy of applying knockoff variables (Barber et al. 2015; Candès et al. 2018), for each pair of original and knockoff variables, a test statistic is calculated,

$$T_j = \frac{1}{q} \sum_{m=1}^q |\phi_{j,m}^x| - \frac{1}{q} \sum_{m=1}^q |\phi_{j,m}^z|, \quad (3.26)$$

which is the difference between the mean absolute test statistic values of each variable pair. It is straightforward that a larger test statistic proves that the original variable is more important than its corresponding knockoff. Finally, the FDR of variable selection is controlled through values of all generated T s, as shown in Equation 3.22.

We use simulation tests to address the following topics: (a) control of type I and type II errors for knockoffs generated by different methods; (b) the properties of different variable importance statistics in boosted tree models; and (c) power and false discovery control using various combinations of knockoffs and statistics. Further, the performance of boosted tree fitting for different model structures is compared.

3.3 Simulation Studies

3.3.1 Incorrect but Related Variables in Boosted Tree Models

First, we describe the models and the boosted tree framework used in the following simulation tests. For the design matrix $\mathbf{X}_{n \times p}^0$, we simulate $n = 200$ samples and $p = 1000$ variables. Each row vector in design matrix $\mathbf{X}_{n \times p}^0$ is generated as $\mathbf{x}_i \sim N_p(\mathbf{0}, \Sigma)$ with block dependence structure matrix $\Sigma = \text{diag}(\Sigma_1, \Sigma_2, \Sigma_3, \dots)$, where each Σ_i is a $\pi p \times \pi p$ matrix with matrix element $\sigma_{j,k} = (\rho^{|j-k|})$. We set $\pi = 0.01$ and $\rho = 0.1$. Because we want to test model-free variable selection, we generate the response as

$$\mathbf{y} = \mathbf{X}\beta + \varepsilon, \quad (3.27)$$

where $\beta = (\beta_{\pi p}^\top, \mathbf{0}_{p-\pi p}^\top)^\top$, $\beta_{\pi p}$ is a vector with equal elements, $\varepsilon \sim N_p(\mathbf{0}, \mathbf{I}_p)$, and \mathbf{X} is transformed from initial \mathbf{X}^0 with four different structures, namely:

1. Main-effect model: $\mathbf{X} = \mathbf{X}^0$.
2. Interaction effect model: each $x_{i,j} = x_{2i-1,j}^0 x_{2i,j}^0$, for $i = 1, \dots, \pi p/2$. This means we keep $\pi p/2$ signal variables the same as the main-effect models.
3. Exponential effect model: each $x_{i,j} = \exp(x_{i,j}^0)$, for $i = 1, \dots, \pi p$.
4. Quadratic effect model: each $x_{i,j} = (x_{i,j}^0)^2$, for $i = 1, \dots, \pi p$.

While we generate the response \mathbf{y} using the transformed \mathbf{X} , we work under the notion of having no information about the transformed \mathbf{X} and fit the models with the initial design matrix \mathbf{X}^0 . Thus, we use incorrect but related variables for fitting boosted tree models. While this is challenging, there is no guarantee that the correct variables will be used for real data modeling, so it is possible to see boosted tree modeling performance with incorrect but related variables.

After defining the simulated data values, we move on to our boosted tree models. The booster used for growing our tree are the gradient boosted tree (GBRT in Table 3.2) and dropouts meet multiple additive regression tree (DART in Table 3.2) (Rashmi and Gilad-Bachrach 2015). As suggested in Friedman et al. (2000) and Zhang et al. (2005), a small learning rate is preferred in boosted model fitting. For our simulation, we fix the learning rate $\gamma = 0.01$. In this scenario, the trees previously added to the boosted model are more important. This overfits the model with trees added earlier. In iterations of DART model

training, some trees are dropped randomly to avoid overfitting while in traditional GBRT, all trees are kept once they join a model. We compare these two algorithms in the following simulations.

The tuning of hyperparameters is tricky, especially for a complex system such as a boosted tree with many parameters. The purpose of our simulation is to compare the performance of boosted trees under a series of conditions and thus we choose 10 initial hyperparameter values of $(\gamma, \lambda, \alpha)$ and conduct Bayesian optimization iterations 20 times for each fitting. We use the average 10-fold cross-validation error score for model evaluation. An early stopping criteria is set: if the average 10-fold cross-validation error score has not improved after five training iterations, no new trees are added. The maximum depth of each tree is fixed at 2, 3, 4, 5, and 6. The minimum weight of each single leaf is 10. Note that

1. We can increase the number of initial $(\gamma, \lambda, \alpha)$ hyperparameter values and Bayesian optimization iterations to find new $(\gamma, \lambda, \alpha)$ that can improve the performance of the final model. However, this is not within the scope of our simulation tests.
2. For real data, we can choose a larger number for the maximum depth of a boosted tree model. Here, we use only 2 to 6 because this is a simple simulation test.

For each combination of the above four models, two boosters, and five maximum tree depths, 100 boosted tree models are fitted. Table 3.2 lists the means and standard errors of 100 10-fold cross-validation error scores. We use the same design matrices $\mathbf{X}_1^0, \dots, \mathbf{X}_{100}^0$ for generating transformed matrices. For each model structure, $\mathbf{X}_1, \dots, \mathbf{X}_{100}$ are created according to our definitions. Among all four model structures, the main-effect model, which is the correct variable model, has the lowest cross-validation prediction error. The incorrect but related models, ranked from the lowest to highest cross-validation prediction error, are interaction, exponential, and then quadratic. This is reasonable since the structure of tree models is formed naturally by interaction among variables. From the standard errors of means, we conclude that predictions for the main-effect and interaction models are more stable than predictions for the exponential and quadratic models. The results indicate that although prediction ability differs by the various true models, it is realistic to use incorrect but related variables in boosted tree modeling. As for maximum tree depth, since the correlation between two variables is chosen as 0.1^d , where d is the index difference of each pair of variables, the correlation decreases rapidly as we choose two variables that are farther away from each other. This is why, in Table 3.2, the cross-validation error score is the lowest when depth = 2. This implies that prior knowledge of the strength of variable

Table 3.2: Means and standard errors of 100 10-fold cross-validation error scores, mean (SD), where sample size $n = 200$, feature size $p = 1000$, signal proportion $\pi = 0.01$, signal strength $\beta = 1$, learning rate $\eta = 0.01$, minimum child weight $w_{min} = 10$, and all samples and features are considered in each iteration step.

Models	Boosters	Max Depth = 2	3	4	5	6
Main Effect	GBRT	2.394 (0.012)	2.590 (0.012)	2.709 (0.014)	2.741 (0.014)	2.743 (0.014)
	DART	2.389 (0.012)	2.591 (0.013)	2.710 (0.013)	2.741 (0.014)	2.743 (0.014)
Interaction	GBRT	2.853 (0.018)	2.862 (0.018)	2.878 (0.018)	2.883 (0.018)	2.887 (0.018)
	DART	2.852 (0.018)	2.861 (0.018)	2.877 (0.018)	2.882 (0.018)	2.861 (0.023)
Exponential	GBRT	5.346 (0.059)	5.369 (0.055)	5.446 (0.054)	5.461 (0.058)	5.551 (0.057)
	DART	5.344 (0.058)	5.350 (0.052)	5.461 (0.056)	5.512 (0.056)	5.548 (0.057)
Quadratic	GBRT	5.704 (0.033)	6.063 (0.032)	6.269 (0.034)	6.366 (0.034)	6.424 (0.035)
	DART	5.713 (0.034)	6.059 (0.032)	6.275 (0.033)	6.368 (0.034)	6.401 (0.034)

correlation is important, as we need to choose an appropriate value for tree depth. Finally, under the conditions of this simulation, there is no obvious difference between GBRT and DART boosters. Figure 3.2 shows examples of boosted tree iteration steps for all models and boosters.

3.3.2 Power and Type I Error Control of Knockoff Variables

As proposed in Section 3.2.4, there are three strategies to generate knockoff variables: Gaussian assumption with shrunk covariance matrix, Gaussian assumption with sparse covariance matrix, and permuted residuals from principal component regression. We are interested in evaluating the power and type I error performance using different knockoff variables for testing. We apply the MAAC metric, which is defined in Section 3.2.4, to test power evaluation. A larger positive MAAC value shows that two tested matrices have less related column space, so the difference between a real signal and its knockoff is more significant. As for type I errors, we use KMMD to test if row vectors in the original design matrix \mathbf{X} and knockoff matrix \mathbf{Z} are drawn from the same distribution. A larger positive test statistic implies that it is more likely to reject the null hypothesis, which assumes that these two row vectors are from the same distribution. If knockoff variables are highly different from the original variables, this leads to large type I errors. In other words, we want knockoff variables to be drawn from the same distribution as the original variables, but with sufficiently different values. There is a tradeoff between power and type I error control.

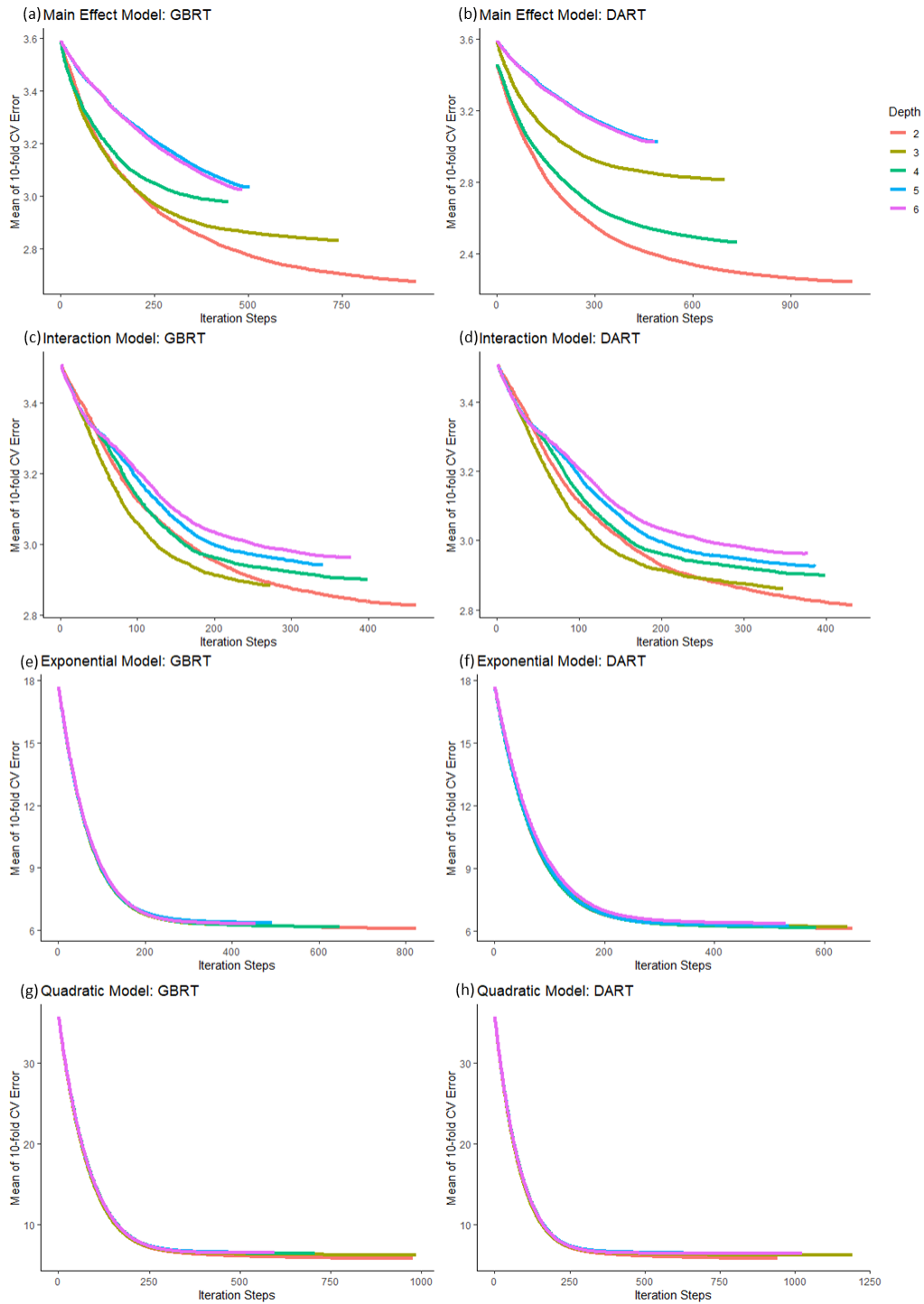


Figure 3.2: Examples of boosted tree iteration steps for all models and boosters. (a) Main-effect model with GBRT; (b) Main-effect model with DART; (c) Interaction model with GBRT; (d) Interaction model with DART; (e) Exponential model with GBRT; (f) Exponential model with DART; (g) Quadratic model with GBRT; and (h) Quadratic model with DART.

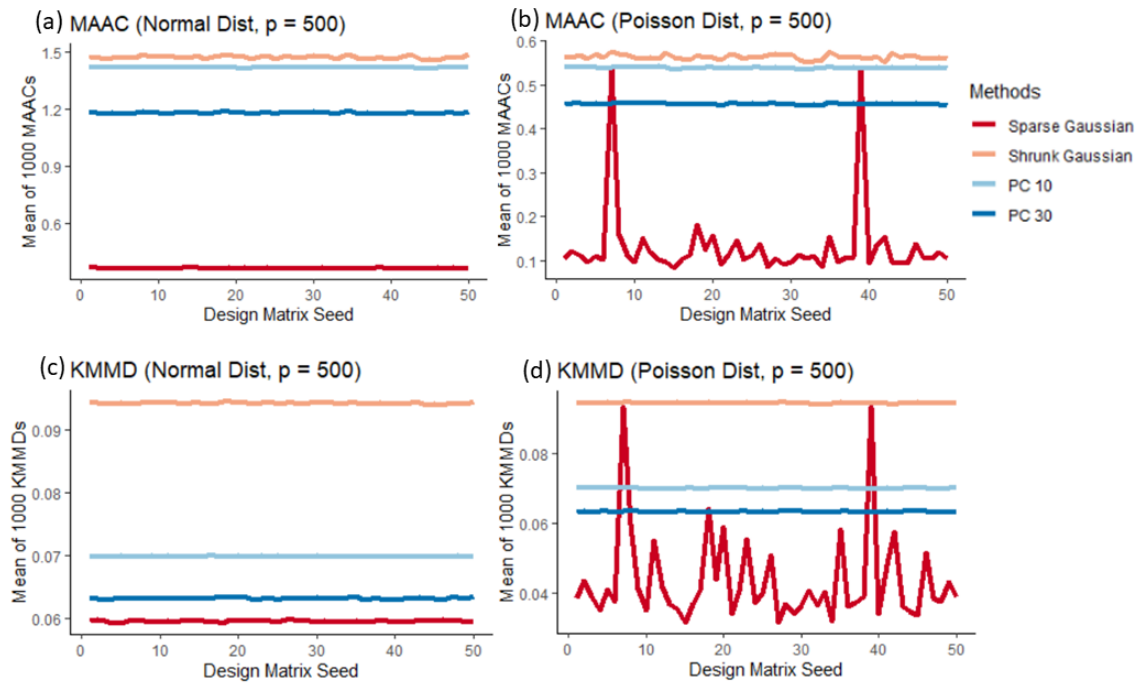


Figure 3.3: MAAC and KMMD of knockoffs with Normal and Poisson original variables. (a) MAAC with normally distributed \mathbf{X} ; (b) MAAC with Poisson-distributed \mathbf{X} ; (c) KMMD with normally distributed \mathbf{X} ; and (d) KMMD with Poisson-distributed \mathbf{X} .

We first assume the row vector in design matrix \mathbf{X} is from a normal distribution. We simulate $n = 100$ samples and $p = 500$ variables. Each row vector in design matrix $\mathbf{X}_{n \times p}$ is generated as $\mathbf{x}_i \sim N_p(\mathbf{0}, \mathbf{\Sigma})$ with block dependence structure matrix $\mathbf{\Sigma} = \text{diag}(\Sigma_1, \Sigma_2, \Sigma_3, \dots)$, where each Σ_i is a $\pi p \times \pi p$ matrix with matrix element $\sigma_{j,k} = (\rho^{|j-k|})$. We set $\pi = 0.01$ and $\rho = 0.1$. In Figure 3.3, we simulate 50 original design matrices and generate 1,000 corresponding knockoff matrices for each original design matrix. Four kinds of knockoffs are created for comparison: shrunk Gaussian, sparse Gaussian, permuted principal component 10, and permuted principal component 30. The y -axis is calculated as the mean value of each 1,000 knockoff samples from one original matrix. The x -axis is the seed used to sample the original matrix. Therefore, y values of different methods at the same x location are comparable. It is clear from Figure 3.3 (a) and (c) that all methods are consistent under normal assumption. The MAAC values in Figure 3.3 (a) show that the sparse method provides the most similar knockoffs, which also means they are the least powerful. On the other hand, the shrunk method has the most different knockoffs. Principal component methods provide knockoffs in between. This makes sense since the shrunk Gaussian method directly estimate the covariance matrix of \mathbf{X} and shrinks it to the identity matrix if it is not definitely positive, while the sparse Gaussian method has zeros in the estimated matrix given its sparse assumption. In Figure 3.3 (c), the KMMD plot has the same order as MAAC. However, the sparse method has the lowest type I error while the shrunk method has the highest.

Since both the shrunk Gaussian and sparse Gaussian methods assume \mathbf{X} follows a normal distribution, it is interesting to see what would happen if a design matrix does not follow a normal distribution. Figure 3.3 (b) and (d) show that the sparse Gaussian method is not as consistent for a Poisson distribution as it is for normal distribution. The principal component method is consistent, however, because it does not depend on normal assumption. Despite some exceptions, for most design matrices, the y -axis order of these four curves is the same as for normal distribution. In summary, the sparse Gaussian method is the most conservative, the shrunk Gaussian method is the most powerful, and the principal component method lies in between. When we increase the number of principal components, the performance of the generated knockoff is similar to the performance of the shrunk method.

3.3.3 Comparison of Variable Importance Ranking Statistics

For each variable in a tree model, feature importance can be used to show its importance in the model. As discussed in Section 3.2.5, a few test statistics can represent this importance. In this section, we conduct simulation tests on *gain*, *cover*, *frequency*, *SHAP*, and *Saabas*. Since we are only interested in the ranking ability of these test statistics for selected variables, not the performance of variable selection, which is conducted by tree growing itself, we define a score called the ranking ratio (RR) as

$$\text{RR} = \frac{\# \text{ Noises ranked above the last signal in ranking}}{\# \text{ Noises}}. \quad (3.28)$$

The RR contains two parts: the number of noises selected by the boosted tree and ranked above the lowest ranked signal among all selected signals, and the total number of mis-selected noises. This score, which ranges within (0, 1), indicates how many false positive decisions must be made, if, for a given set of variables chosen during tree growing, we want to include all signals for variable selection. The score is undefined if no signal or noise are selected in a tree model. It evaluates the ranking ability of each variable importance ranking statistic. A small RR score indicates better ranking ability of variable importance.

We keep the model described in Section 3.3.2 and simulate $n = 100$ samples and $p = 500$ variables. Each row vector in design matrix $\mathbf{X}_{n \times p}$ is generated as $\mathbf{x}_i \sim N_p(\mathbf{0}, \Sigma)$, with block dependence structure matrix $\Sigma = \text{diag}(\Sigma_1, \Sigma_2, \Sigma_3, \dots)$, where each Σ_i is a $\pi p \times \pi p$ matrix with matrix element $\sigma_{j,k} = (\rho^{|j-k|})$. We set $\pi = 0.01$ and $\rho = 0.1$. Following tree model hyperparameters in Section 3.3.1, the maximum depth of each tree is fixed at 6. We use two boosters, GBRT and DART, four model structures, and two signal strengths. In Table 3.3, the correct variables are used for modeling. Each mean and standard error are calculated from 1,000 simulations. In each signal strength-booster combination, means for the test statistic are close, and the standard errors are almost the same. This indicates these variable importance statistics have similar consistency even under different signal strengths and boosters. There is no obvious difference between the two boosters. Among the five variable importance statistics, *frequency* has the weakest ranking ability given its highest RR score, regardless of strength or booster. In Tables 3.4, 3.5, and 3.6, incorrect but related variables are used for modeling. For these models, *cover* has the best performance as it returns the lowest RR scores in most cases. Note that we are only comparing five variable importance statistics under each combination of model structure, strength, and booster (i.e., each row in Tables 3.3 to 3.6). Different rows have different boosted tree models and are thus

Table 3.3: Mean and standard error of signal percentage and ranking ratio (RR) from 1,000 simulation tests for ranking test statistics in **main-effect** models, where $n = 100$, $p = 500$.

Strength	Booster	Gain	Cover	Frequency	SHAP	Saabas
0.5	GBRT	0.705 (0.008)	0.685 (0.008)	0.716 (0.008)	0.699 (0.008)	0.700 (0.008)
	DART	0.714 (0.009)	0.700 (0.010)	0.729 (0.009)	0.708 (0.010)	0.711 (0.010)
1.5	GBRT	0.699 (0.008)	0.697 (0.008)	0.723 (0.008)	0.700 (0.008)	0.700 (0.008)
	DART	0.694 (0.010)	0.693 (0.010)	0.713 (0.010)	0.700 (0.010)	0.703 (0.010)

Table 3.4: Mean and standard error of signal percentage and ranking ratio (RR) from 1,000 simulation tests for ranking test statistics in **interaction** models, where $n = 100$, $p = 500$.

Strength	Booster	Gain	Cover	Frequency	SHAP	Saabas
0.5	GBRT	0.564 (0.010)	0.561 (0.010)	0.586 (0.010)	0.566 (0.010)	0.565 (0.010)
	DART	0.562 (0.009)	0.550 (0.010)	0.581 (0.009)	0.561 (0.010)	0.562 (0.010)
1.5	GBRT	0.626 (0.009)	0.634 (0.009)	0.654 (0.009)	0.633 (0.009)	0.635 (0.009)
	DART	0.625 (0.010)	0.627 (0.010)	0.649 (0.010)	0.630 (0.010)	0.632 (0.010)

incomparable.

3.3.4 SEAD, Other Knockoff Test Statistics, and Their False Discovery Control

In this final subsection describing the simulation, we combine all of the previous steps and perform the whole framework of the knockoff boosted tree (KGBT) algorithm. We are interested in the power and FDR of different combinations of ranking statistics and knockoff

Table 3.5: Mean and standard error of signal percentage and ranking ratio (RR) from 1,000 simulation tests for ranking test statistics in **exponential** models, where $n = 100$, $p = 500$.

Strength	Booster	Gain	Cover	Frequency	SHAP	Saabas
0.5	GBRT	0.802 (0.006)	0.783 (0.006)	0.800 (0.006)	0.799 (0.006)	0.799 (0.006)
	DART	0.810 (0.006)	0.784 (0.006)	0.810 (0.006)	0.808 (0.006)	0.808 (0.006)
1.5	GBRT	0.817 (0.005)	0.809 (0.005)	0.818 (0.005)	0.821 (0.005)	0.821 (0.005)
	DART	0.817 (0.005)	0.808 (0.005)	0.818 (0.005)	0.820 (0.005)	0.821 (0.005)

Table 3.6: Mean and standard error of signal percentage and ranking ratio (RR) from 1,000 simulation tests for ranking test statistics in **second-order** models, where $n = 100$, $p = 500$.

Strength	Booster	Gain	Cover	Frequency	SHAP	Saabas
0.5	GBRT	0.784 (0.007)	0.759 (0.007)	0.762 (0.007)	0.786 (0.007)	0.786 (0.007)
	DART	0.772 (0.009)	0.747 (0.010)	0.749 (0.010)	0.768 (0.009)	0.768 (0.009)
1.5	GBRT	0.832 (0.005)	0.834 (0.005)	0.823 (0.005)	0.846 (0.005)	0.846 (0.005)
	DART	0.834 (0.005)	0.828 (0.005)	0.821 (0.006)	0.839 (0.005)	0.840 (0.005)

types. We simulate $n = 100$ samples and $p = 500$ variables. Each row vector in design matrix $\mathbf{X}_{n \times p}$ is generated as $\mathbf{x}_i \sim N_p(\mathbf{0}, \Sigma)$ or $\mathbf{x}_i \sim \text{Poisson}_p(\mathbf{5}, \Sigma)$ with block dependence structure matrix $\Sigma = \text{diag}(\Sigma_1, \Sigma_2, \Sigma_3, \dots)$, where each Σ_i is a $\pi p \times \pi p$ matrix with matrix element $\sigma_{j,k} = (\rho^{|j-k|})$. We set $\pi = 0.01$ and $\rho = 0.1$. The maximum depth of each tree is fixed at 6. For simplicity, we choose GBRT as the booster and the main-effect model as the true model. In Tables 3.7 to 3.10, power and FDR for different combinations of knockoff types and ranking statistics are listed for comparison. We choose a condition where signals are sparse and weak so that the performance of each method is considerably distinct.

In Table 3.7, 50 normally distributed design matrices are simulated, each with 1,000 shrunk Gaussian knockoffs, 1,000 sparse Gaussian knockoffs, 1,000 10-principal component knockoffs, and 1,000 30-principal component knockoffs. We set the targeted FDR as 0.1. The means and corresponding standard errors of power for each combination are listed. Among these four types of knockoffs, the shrunk Gaussian knockoff has the highest power, which is the same conclusion reached in Figure 3.3 (a). This is followed by 10- and 30-principal component knockoffs. The sparse Gaussian knockoff has the lowest power. Among the five importance statistics we test, *frequency* is always the most powerful statistic for all knockoffs, while *gain* is the least powerful statistic. The other three are moderately powerful and fall between *frequency* and *gain*. Table 3.8 shows the means and corresponding standard errors of the FDR for each combination. Since the targeted FDR is 0.1, it is clear that the sparse Gaussian knockoff is the most conservative method and is the only one that can ensure the FDR stays under the targeted level. At the same order of power, the shrunk Gaussian knockoff has the highest FDR, followed by the two principal component knockoffs.

Table 3.9 shows the power of the statistics when the normality assumption is invalid. Fifty Poisson-distributed design matrices are simulated, each with 1,000 shrunk Gaussian

knockoffs, 1,000 sparse Gaussian knockoffs, 1,000 10-principal component knockoffs, and 1,000 30-principal component knockoffs. Again, the targeted FDR is set as 0.1. The power of the knockoffs is the same as for normal distribution. For the importance statistics, the order for Poisson-distributed matrices is different from that for normally distributed matrices, where *cover* has the highest power, followed by *frequency* and then the other three. Both the sparse Gaussian and 30-principal component knockoffs can control the FDR to stay close to the targeted level. The shrunk Gaussian and 10-principal component knockoffs have higher FDRs. The order of importance statistics for FDR is the same as their power ranking. In summary, the ranges in tables below show the trends of knockoffs and importance statistics.

More Conservative			More Powerful
Sparse Gaussian	30-Principal Component	10-Principal Component	Shrunk Gaussian

	More Conservative		More Powerful
Normal	Gain	Cover, Saabas, and SHAP	Frequency
Non-Normal	Cover	Frequency	Gain, Saabas, and SHAP

Table 3.7: Mean and standard error of **power** from 50 **normal** distributed simulation tests for ranking test statistics and knockoff types, where $n = 100$, $p = 500$, $\pi = 0.04$, signal strength = 1.5, design matrix variance $\sigma^2 = 1$, and $q = 1000$.

	Sparse Gaussian	Shrunk Gaussian	10-PC	30-PC
Cover	0.042 (0.014)	0.460 (0.018)	0.448 (0.018)	0.297 (0.026)
Frequency	0.054 (0.016)	0.480 (0.017)	0.463 (0.017)	0.355 (0.021)
Gain	0.007 (0.007)	0.423 (0.016)	0.398 (0.017)	0.211 (0.021)
Saabas	0.037 (0.013)	0.471 (0.017)	0.448 (0.017)	0.283 (0.024)
SHAP	0.037 (0.013)	0.467 (0.017)	0.448 (0.017)	0.284 (0.024)

Table 3.8: Mean and standard error of **FDR** from 50 **normal** distributed simulation tests for ranking test statistics and knockoff types, where $n = 100$, $p = 500$, $\pi = 0.04$, signal strength = 1.5, design matrix variance $\sigma^2 = 1$, and $q = 1000$.

	Sparse Gaussian	Shrunk Gaussian	10-Principal Component	30-Principal Component
Cover	0.091 (0.031)	0.721 (0.008)	0.693 (0.011)	0.481 (0.035)
Frequency	0.139 (0.038)	0.751 (0.008)	0.739 (0.009)	0.640 (0.023)
Gain	0.007 (0.007)	0.684 (0.010)	0.660 (0.012)	0.407 (0.039)
Saabas	0.070 (0.026)	0.720 (0.008)	0.694 (0.011)	0.450 (0.035)
SHAP	0.070 (0.026)	0.719 (0.009)	0.694 (0.011)	0.448 (0.035)

Table 3.9: Mean and standard error of **power** from 50 **Poisson**-distributed simulation tests for ranking test statistics and knockoff types, where $n = 100$, $p = 500$, $\pi = 0.04$, signal strength = 1.5, design matrix variance $\sigma^2 = 1$, and $q = 1000$.

	Sparse Gaussian	Shrunk Gaussian	10-Principal Component	30-Principal Component
Cover	0.010 (0.010)	0.380 (0.015)	0.359 (0.016)	0.105 (0.023)
Frequency	0.008 (0.008)	0.377 (0.015)	0.338 (0.017)	0.086 (0.021)
Gain	0.008 (0.008)	0.332 (0.015)	0.262 (0.023)	0.053 (0.017)
Saabas	0.008 (0.008)	0.349 (0.015)	0.276 (0.023)	0.040 (0.015)
SHAP	0.008 (0.008)	0.351 (0.015)	0.275 (0.023)	0.041 (0.015)

Table 3.10: Mean and standard error of **FDR** from 50 **Poisson**-distributed simulation tests for ranking test statistics and knockoff types, where $n = 100$, $p = 500$, $\pi = 0.04$, signal strength = 1.5, design matrix variance $\sigma^2 = 1$, and $q = 1000$.

	Sparse Gaussian	Shrunk Gaussian	10-Principal Component	30-Principal Component
Cover	0.007 (0.007)	0.551 (0.016)	0.488 (0.021)	0.142 (0.031)
Frequency	0.007 (0.007)	0.545 (0.015)	0.474 (0.024)	0.116 (0.029)
Gain	0.005 (0.005)	0.496 (0.022)	0.396 (0.034)	0.084 (0.027)
Saabas	0.005 (0.005)	0.514 (0.019)	0.388 (0.032)	0.066 (0.024)
SHAP	0.005 (0.005)	0.514 (0.019)	0.392 (0.032)	0.065 (0.024)

3.4 Real Data Applications

3.4.1 Tumor Sample Purity Estimation

Tumor sample purity refers to the percentage of cancer cells in a tumor tissue sample. It is used to discover the roles of cancerous and non-cancerous cells in the tumour microenvironment, which mainly comprises immune cells (Aran et al. 2015; Turley et al. 2015). To estimate tumor purity, Carter et al. (2012) proposed ABSOLUTE, a method used to perform analysis of somatic DNA alterations. In addition, it has been reported that DNA methylation data and expression data from selected stromal genes (Houseman et al. 2012; Yoshihara et al. 2013) have been successfully used for estimation. Recently, Aran et al. (2015) and Li et al. (2019) used RNA-seq gene expression data to determine tumor purity and found several gene signatures of individual cancer types. As it has been shown reasonable to estimate tumor sample purity using gene expression data, we apply our KOBT algorithm so that we can compare our results with those of previous reports. Here, our interest is not the accuracy of estimation but the selection of genes that are important in estimation with no topology assumptions. If a gene's expression level is positively correlated with tumor purity, then it is highly likely that this gene is expressed primarily by cancer cells in tumor samples.

Processed TCGA RNA-seq gene expression data was downloaded from the Pan-Cancer Atlas Publication website (Hoadley et al. 2018). Among all 33 available tumor types, breast invasive carcinoma (BRCA) and skin cutaneous melanoma (SKCM) were chosen for tumor purity estimation. There are 1,017 BRCA and 459 SKCM samples with 17,176 expressed genes for each sample. The genes with missing values or zero variance were filtered out. We used the tumor purity estimates in Hoadley et al. (2018) as the responses, which were obtained using ABSOLUTE. The responses are thus bounded between 0 and 1.

To reduce the original dimensionality to a lower value, we apply Lasso (Tibshirani 1996) on original data sets before performing our KOBT steps. After tuning the penalty terms in Lasso objective functions, 486 and 496 genes of BRCA and SKCM respectively are selected for further analysis. We generated $q = 300$ knockoffs for each gene. Here, we compare two knockoff types: shrunk Gaussian and 10-principal component knockoffs. We conducted Bayesian optimization and 10-fold cross-validation to choose the optimal parameters for the boosted tree algorithm (see Section 3.6.1 for details of the parameters). We used the SHAP for importance evaluation. Therefore, our test statistic is the average difference of SHAP values between each gene and its own knockoff, as shown in Equation 3.2.6. Finally, FDR control is applied to these test statistics of genes with a targeted FDR level of 10%.

Table 3.11: Genes whose expression is related to BRCA tumor purity, where targeted FDR = 0.1. Genes are selected using the Shrunk Gaussian knockoff. Overlapping genes with ESTIMATE models are in **bold**.

Detected Genes
C1S, C2orf48, CCDC69 , CSF2RB, CXorf65, DNAJC12, EDN3, FAM163A, FAM65B, FGR , GBP3, HGFAC, HTR2A, IL7R , KIAA0087, NCRNA00175, NRADDP, NRG2, PM20D1, PNOC, SLAIN1, UNQ6494

Table 3.11 reports the selected genes whose expression is related to BRCA tumor purity, where the targeted FDR is 0.1. Genes are selected using the Shrunk Gaussian knockoff. Genes selected by the 10-principal component knockoff are listed in Table 3.16. Among all detected genes, CSF2RB (colony stimulating factor 2 receptor beta common subunit), C1S (complement C1s), CCDC69 (coiled-coil domain containing 69), and FGR are also reported among the top 10-ranked important genes for pan-cancer tumor purity prediction in Li et al. (2019). CSF2RB is an immune-related gene. It has been reported that in BRCA, the high expression of CSF2RB is positively correlated with patient survival (Liu et al. 2019). ESTIMATE (Yoshihara et al. 2013) uses stromal and immune gene expression to predict tumor purity. Our detected immune genes CCDC69, FGR, and IL7R are included in Yoshihara et al. (2013). The box plots of gene expression levels for selected genes are presented in Figure 3.4. These genes are selected because they are detected by both types of knockoffs. Other genes are plotted in Figure 3.8. All samples are grouped according to their purity: samples with the top 1/3 purity are labeled as high (blue); and samples with the bottom 1/3 purity are labeled as low (yellow). The non-parametric Wilcoxon signed-rank test is conducted for each low-high pair. Their corresponding p -values are included at the top of the plot. It is shown that genes expression levels of almost all selected genes are highly correlated with tumor purity. Samples are grouped according to their gene expression levels. The top 1/3 samples are grouped into the high-level group, and the bottom 1/3 samples are grouped into the low-level group. Survival analysis has been conducted for these genes. Among them, CCDC69, CXORF65, and FGR have significant p -values for coefficients in Cox regression. Plots of the Kaplan-Meier estimators and p -values are in Figure 3.10.

Table 3.12 reports more genes associated with SKCM tumor purity. Li et al. (2019) reports CSF2RB, RHOH, C1S, CCDC69, and CCL22 as among the top 10-ranked important genes for pan-cancer tumor purity prediction. Most of these detected genes are from stromal cells and not cancer cells. For example, the expression level of CCDC69 is negatively correlated

Table 3.12: Genes whose expression is related to SKCM tumor purity, where targeted FDR = 0.1. Genes are selected using the Shrunk Gaussian knockoff. Overlapping genes with ESTIMATE models are in **bold**.

Detected Genes
CSF2RB, RHOH , C1S, CCDC69 , CCL22, FGR , ALPK2, ANKRD40, CAPN13, CASP5, CCDC30, CTCE, DEFA1B, FAM78A, GAL3ST2, GSTM5, HERPUD1, IGF2, KCNRG, KIR2DL3, LOC100129066, LOC91450, MYBPC3, NAP1L2, OLAH, OSTalpha, PADI4, PHC1, PLXNB3, PROCA1, PRRG4, PTK7, RAP1GAP, RGL4, SOAT1, SRPRB, TXNDC3 , VNN3

with tumor purity, which can not be true if it mainly comes from cancer cells. Immune genes **CCDC69**, **FGR**, and **RHOH**, and stromal gene **TXNDC3** are included in ESTIMATE models. The box plots of gene expression levels for selected genes are presented in Figure 3.5. Other genes are plotted in Figure 3.9. Same as it is for BRCA, samples with the top 1/3 purity are labeled as high (blue); and samples with the bottom 1/3 purity are labeled as low (yellow). Related p -values from Wilcoxon signed-rank tests are attached to the plots. We see that genes expression levels of almost all selected genes are highly correlated with tumor purity. All samples are grouped according to their gene expression levels. Plots of their Kaplan-Meier estimators and p -values are in Figure 3.11 and 3.12. In summary, for tumor purity estimation, we propose that our KOBT algorithm can detect a few genes that are largely expressed in stromal cells. Our results reproduce previously reported work using different algorithms.

3.4.2 Tumor Type Classification

For application of KOBT on classification, we focus on its performance of assigning tumors to known classes and identifying those genes whose expression levels are related to the classification. In 1999, Golub et al. (1999) used gene expression monitoring by DNA microarrays to conduct cancer classification. Their test on classifying acute myeloid leukemia (AML) and acute lymphoblastic leukemia (ALL) showed that it was realistic to distinguish different cancer subtypes only based on gene expression data. Li et al. (2017) performed a pan-cancer classification with RNA-seq data from TCGA using boosted tree and k -nearest neighbours methods. Since KOBT algorithm can detect signals with false discovery rate control, we would like to see how it works on gene selection for tumor type classification.

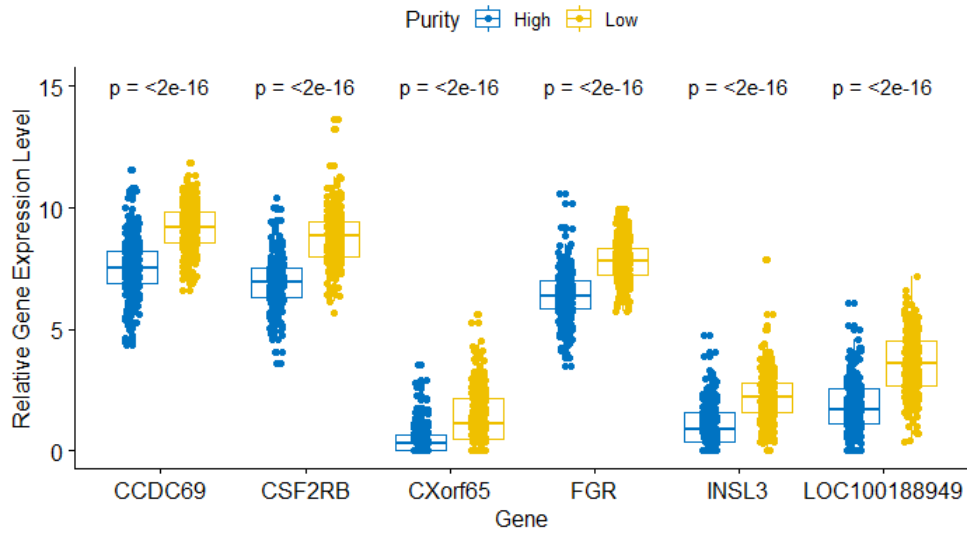


Figure 3.4: Relative gene expression level comparison for selected genes in low (yellow) and high (blue) purity samples of BRCA. The non-parametric Wilcoxon signed-rank test is conducted for each low-high pair. Their corresponding p -values are included at the top of the plot. The relative gene expression level is the \log_2 -transformed normalized expression read counts mapped (unit: million reads).

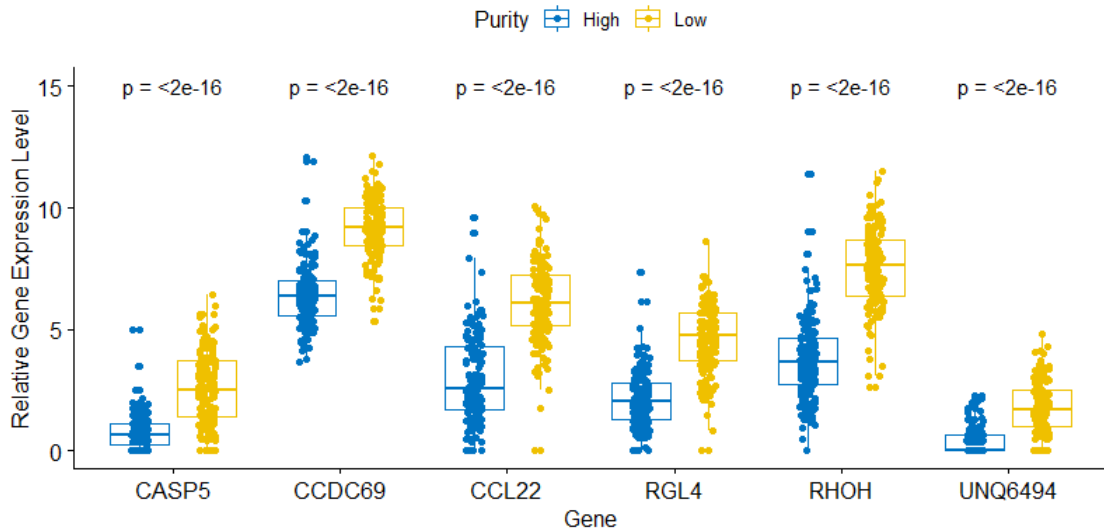


Figure 3.5: Relative gene expression level comparison for selected genes in low (yellow) and high (blue) purity samples of SKCM. The non-parametric Wilcoxon signed-rank test is conducted for each low-high pair. Their corresponding p -values are included at the top of the plot. The relative gene expression level is the \log_2 -transformed normalized expression read counts mapped (unit: million reads).

Table 3.13: Genes whose expression is related to ESCA vs STAD tumor classification, where targeted FDR = 0.1.

Detected Genes
BARX1, HAND2, KRT14, NVL

The TCGA RNA-seq gene expression data is the same as in Subsection 3.4.1. Instead of a pan-cancer classification, we focus on two challenging binary classifications for (1) esophageal carcinoma (ESCA) *vs* stomach adenocarcinoma (STAD) and (2) rectum adenocarcinoma (READ) *vs* colon adenocarcinoma (COAD). We choose these two pairs of cancers because of their similarities. Esophageal carcinoma (ESCA) and stomach adenocarcinoma (STAD) are both malignant tumors in the digestive tract. In Li et al. (2017), it has been reported that almost all READ samples were mis-assigned to COAD. We have 499 samples for the ESCA vs STAD classification test and 529 samples for the READ vs COAD one. The steps are the same as in Subsection 3.4.1 except that the responses are binary now: 0 for one cancer and 1 for another cancer.

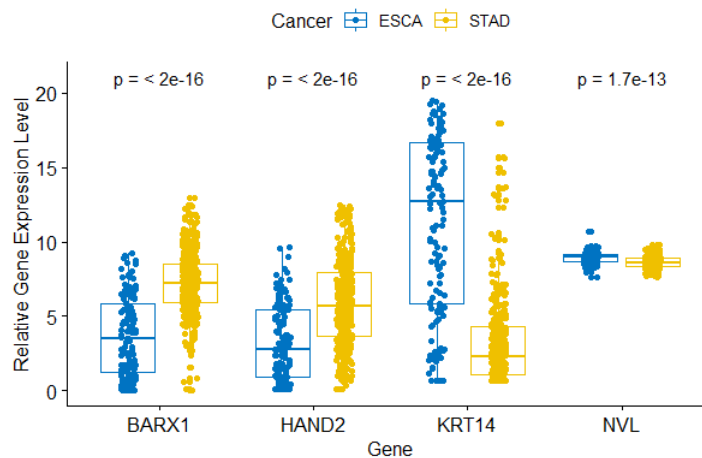


Figure 3.6: Relative Gene expression level comparison for selected genes in ESCA (blue) and STAD (yellow) samples. The non-parametric Wilcoxon signed-rank test is conducted. The corresponding p -values are included at the top of the plot. The relative gene expression level is the \log_2 -transformed normalized expression read counts mapped (unit: million reads).

In Table 3.13, the selected genes that can distinguish ESCA and STAD are presented.

Table 3.14: Genes whose expression is related to READ vs COAD tumor classification, where targeted FDR = 0.1.

Detected Genes
EMB, HOXC4, HOXC8, ZNF595

Among them, BARX1 is a stomach mesenchymal transcription factor. Kim et al. (2005) has shown that BARX1 loss in the mesenchyme prevents stomach epithelial differentiation of overlying endoderm and induces intestine-specific genes instead. Their results defined a transcriptional and signaling pathway of inductive cell interactions in vertebrate organogenesis. Kim et al. (2011) proved that BARX1 controls mouse stomach morphogenesis and is required to specify stomach-specific epithelium in adjacent endoderm. HAND2 is an RNA Gene, which is affiliated with the long non-coding RNA class. Tsubokawa et al. (2002) stated that KRT14 is often expressed by tumor cells in the trabecular nests of the primary carcinoma. The box plots of gene expression levels for all detected genes are presented in Figure 3.6. All samples are grouped according to their cancer types: ESCA samples are in blue, and STAD samples are in yellow. The non-parametric Wilcoxon signed-rank test is conducted for each gene. Their corresponding p -values are included at the top of the plot. It is shown that genes expression levels of all selected genes are highly correlated with the cancer type.

In Table 3.14, we show the detected genes that can used for READ and COAD classification. HOXC4 and HOXC8 are members of Homeobox genes, which is a large family of transcription factors that direct the formation of many body structures during early embryonic development. It has been observed that HOXC family gene expression is upregulated in most solid tumor types (Bhatlekar et al. 2014). The box plots of gene expression levels for all detected genes are presented in Figure 3.7. All samples are grouped according to their cancer types: COAD samples are in blue, and READ samples are in yellow. The non-parametric Wilcoxon signed-rank test is conducted for each gene. Their corresponding p -values are included at the top of the plot. It is shown that genes expression levels of all selected genes are highly correlated with the cancer type.

Unlike in tumor purity estimation, cancer genes play a more important role in tumor type classification. We detect some protein coding genes and find previous published literature to support our findings in both classification tests. Given the fact that the mechanism of cancers is complicated, and the cancer types we choose have been reported to be hard to distinguish, we show that our proposed KOB algorithm to be robust in real data with

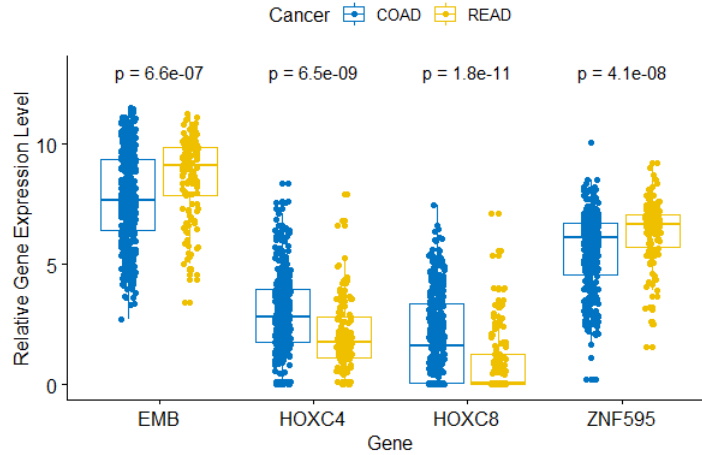


Figure 3.7: Relative gene expression level comparison for selected genes in COAD (blue) and READ (yellow) samples. The non-parametric Wilcoxon signed-rank test is conducted. The corresponding p -values are included at the top of the plot. The relative gene expression level is the \log_2 -transformed normalized expression read counts mapped (unit: million reads).

unknown models.

3.5 Conclusions

In this article, we introduce a new model-free variable selection method, called knockoff boosted tree, KOBT. Given the nature of boosted tree models, no prior model topology knowledge is required. We extend the current application of knockoff methods to tree models. Meanwhile, we proposed two new sampling methods to generate knockoffs, the principal component construction knockoff and the sparse Gaussian knockoff. Unlike currently available methods, the principal component construction knockoff does not depend on Gaussian assumption of design matrix. To evaluate the power of our generated knockoffs in simulation tests, we define a new test statistic, *mean absolute angle of columns*, which stands for the average distance between column vectors in two matrices. We apply *kernel maximum mean discrepancy* to test whether our proposed knockoffs are drawn from the same distributions as the original vectors. In our boosted tree framework, we fix most hyperparameters at multiple reasonable levels and leave regularization parameters to be tuned by Bayesian optimization. We consider different importance scores in tree models. Combinations of different knockoffs and importance test statistics are listed in

our results. We test our strategy on different unknown true models, including main effect, interaction, exponential, and second order models. Finally, we apply our algorithm on tumor purity estimation and tumor type classification using TCGA gene expression data. We see that KOBT performs well and our results match previously published literature. Finally, we provide an R package to implement our proposed approaches, which is available at <https://cran.r-project.org/web/packages/KOBT/index.html>.

3.6 Supplementary Information

3.6.1 Parameters in Real Data Applications

Table 3.15: XGBoost parameters in real data applications.

Parameter	Value
Booster	gbtree
Maximum number of trees	5000
Learning rate	0.01
Maximum tree depth	6
Minimum leaf weight	10
Sub-sample	80%
Sub-feature	80%
γ	[0, 20] (a tuning region for Bayesian optimization)
λ	[0, 20] (a tuning region for Bayesian optimization)
α	[0, 20] (a tuning region for Bayesian optimization)
Objective function	reg:squarederror & binary:logistic
Evaluation metric	root mean square error & classification error

The early stopping rule is stopping adding new trees when average test loss in cross validation does not improve in five additional trees.

3.6.2 Selected Genes Using 10-principal Component Knockoff

Table 3.16: Genes whose expression is related to tumor purity using 10-principal component knockoff, where targeted FDR = 0.1.

Cancer	Detected Genes
BRCA	CSF2RB, CCDC69, FGR, CXorf65, INSL3, UNQ6494
SKCM	RHOH, CCDC69, CCL22, ALPK2, ANKRD40, BBS1, BEST4, C11orf70, C2orf27A, CAPN13, CASP5, CCDC30, CHDH, CHRM4, CTAGE9, CTCE, CXCL6, DEFA1B, EIF1B, FAHD1, FAM78A, FBXW8, FERMT1, GAL3ST2, GRM4, GSTM5, IGF2, KCNRG, KDM4D, KHDC1, KIR2DL3, KLF11, LAMA1, LOC100129034, LOC100129066, LOC91450, MIA, MYBPC3, NAP1L2, OLAH, OSTalpha, PADI4, PCCB, PER1, PER2, PHC1, PLXNB3, PROCA1, PRRG4, PTK7, PTPN5, PTRH2, PYROXD2, RAP1GAP2, RGL4, RIC3, RNF126P1, RYBP, SEC61A1, SGSM2, SMAD2, SOAT1, SRPRB, TXNDC3, USP30, VNN3, ZIC1

3.6.3 Relative Gene Expression Level Comparison in Low and High Purity Samples

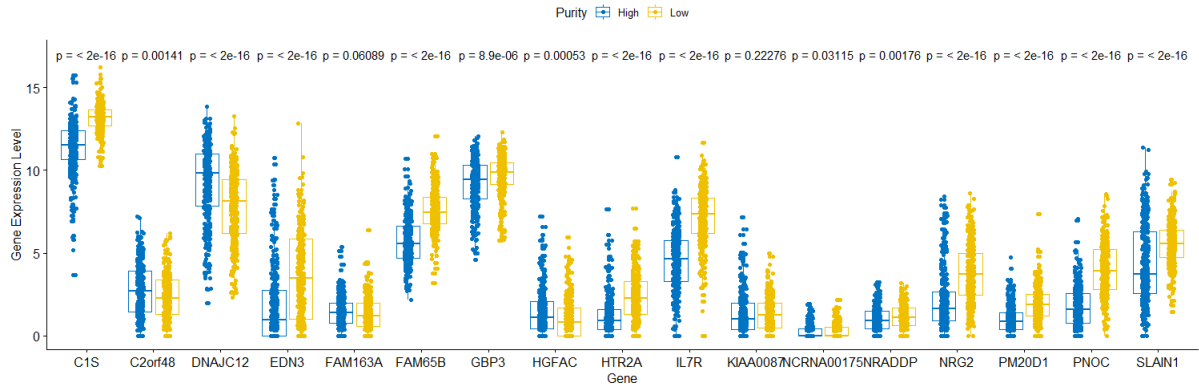


Figure 3.8: Relative gene expression level comparison for other genes in low (yellow) and high (blue) purity samples of BRCA. The non-parametric Wilcoxon signed-rank test is conducted for each low-high pair. Their corresponding p -values are included at the top of the plot. The relative gene expression level is the \log_2 -transformed normalized expression read counts mapped (unit: million reads).

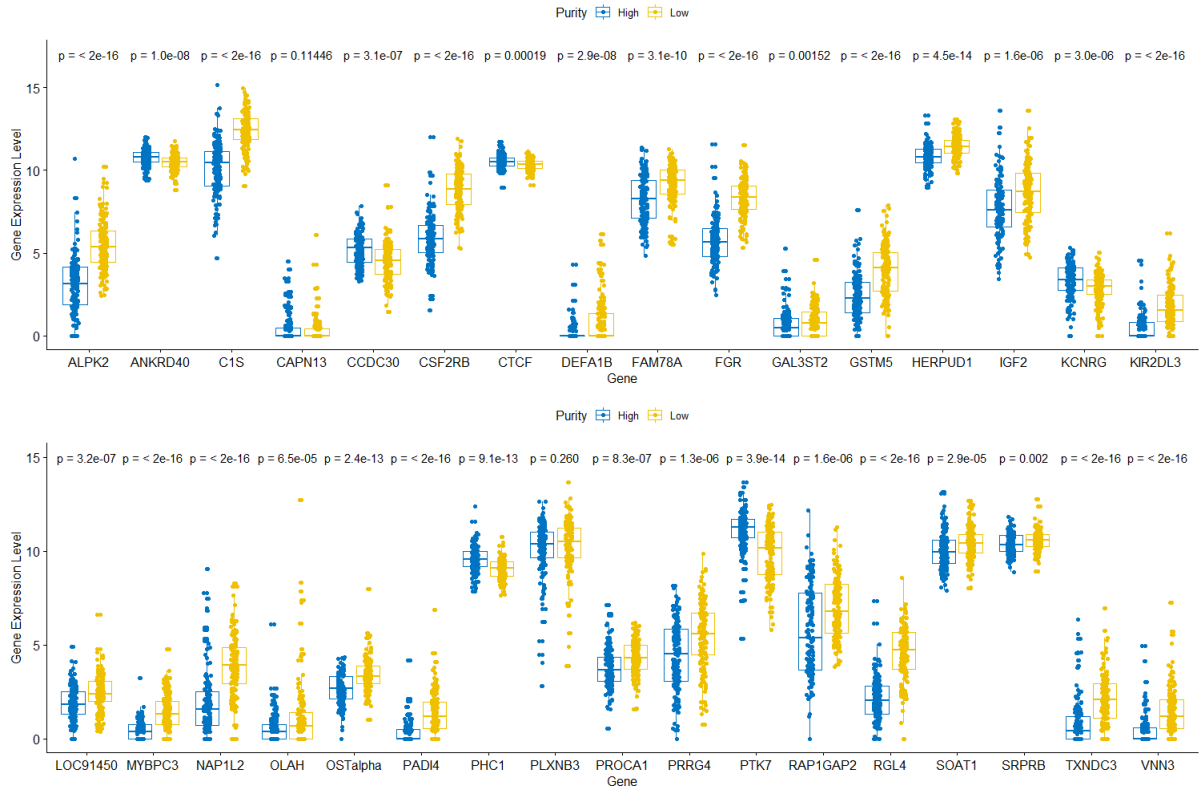


Figure 3.9: Relative gene expression level comparison for other genes in low (yellow) and high (blue) purity samples of SKCM. The non-parametric Wilcoxon signed-rank test is conducted for each low-high pair. Their corresponding p -values are included at the top of the plot. The relative gene expression level is the \log_2 -transformed normalized expression read counts mapped (unit: million reads).

3.6.4 Survival Analysis for Other Selected Genes in SKCM

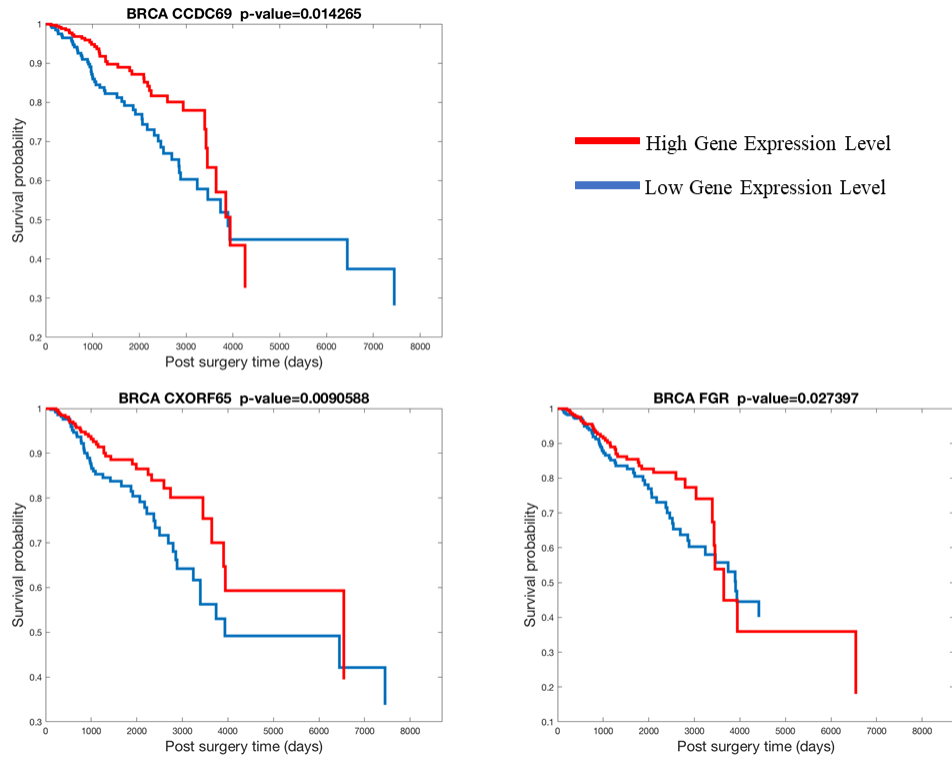


Figure 3.10: Plots of the Kaplan-Meier estimators and p -values for coefficients in Cox regression of BRCA. Samples with high gene expression levels are in red, and samples with low gene expression levels are in blue.

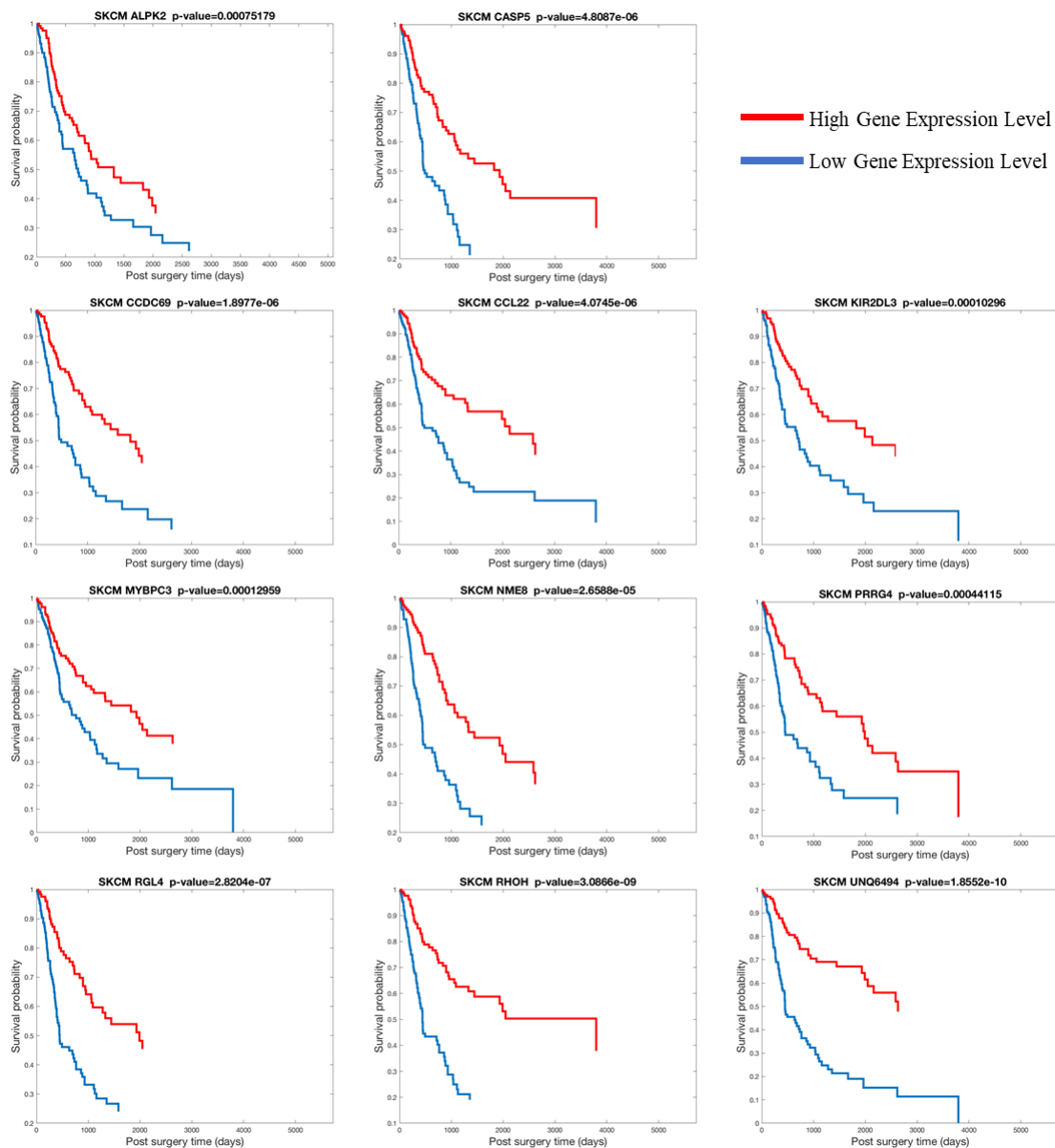


Figure 3.11: Plots of the Kaplan-Meier estimators and p -values for coefficients in Cox regression of SKCM.

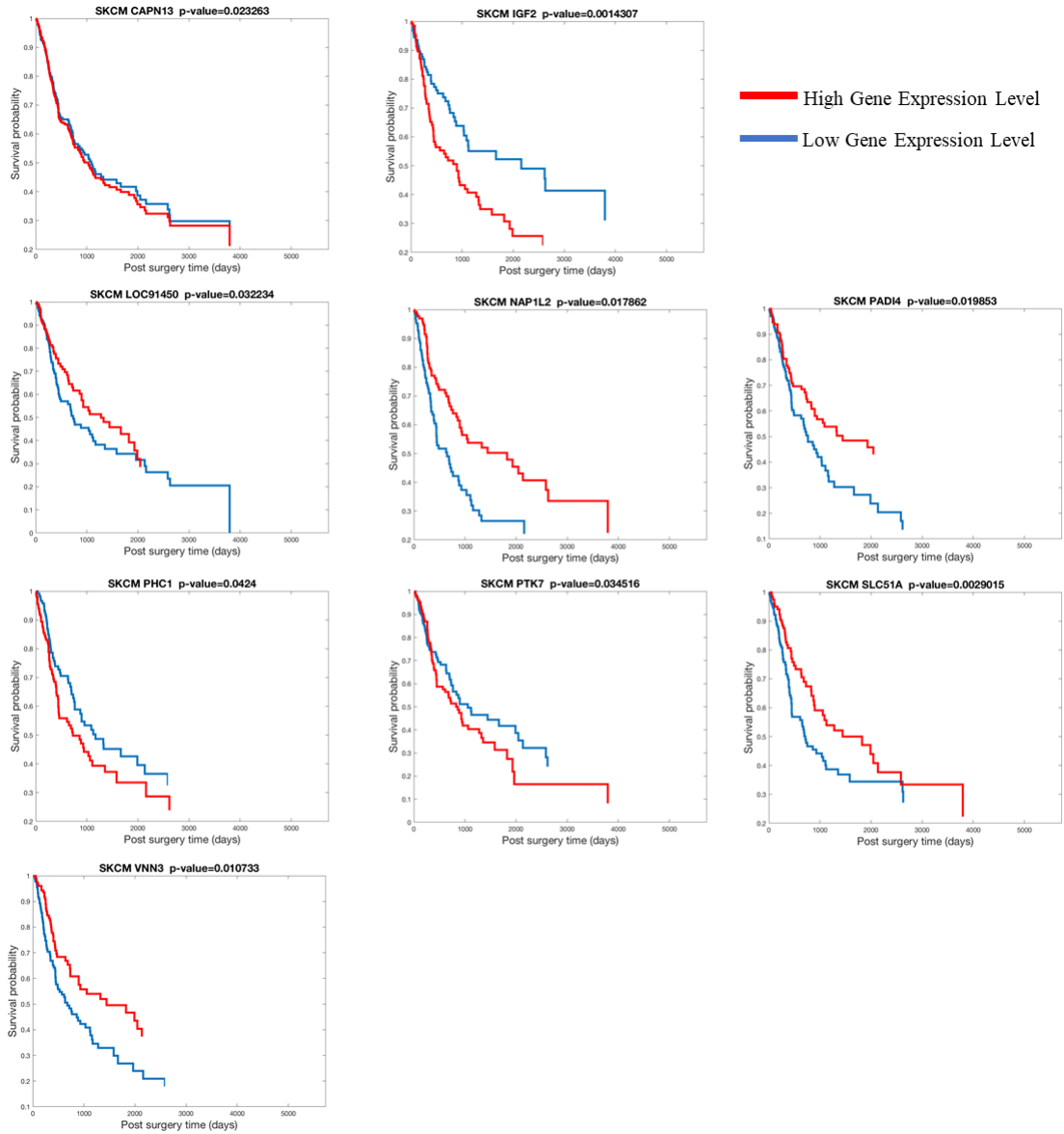


Figure 3.12: (Continued) Plots of the Kaplan-Meier estimators and p -values for coefficients in Cox regression of SKCM.

CHAPTER

4

SAME-SPECIES CONTAMINATION DETECTION WITH VARIANT CALLING INFORMATION FROM NEXT GENERATION SEQUENCING

Same-species contamination detection is an important quality control step in genetic data analysis. Compared with widely discussed cross-species contamination, same-species contamination is more challenging to detect, and there is a scarcity of methods to detect and correct for this quality control issue. Same-species contamination may be due to contamination by lab technicians or samples from other contributors. Here, we introduce a novel machine learning algorithm to detect same species contamination in next generation sequence data using support vector machines. Our approach uniquely detects such contamination using variant calling information stored in the variant call format (VCF) files (either DNA or RNA), and importantly can differentiate between same species contamination and mixtures of tumor and normal cells.

In the first stage of our approach, a change-point detection method is used to identify copy number variations or copy number aberrations (CNVs or CNAs) for filtering prior to testing for contamination. Next, single nucleotide polymorphism (SNP) data is used to test for same species contamination using a support vector machine model. Based on the assumption that alternative allele frequencies in next generation sequencing follow the beta-binomial distribution, the deviation parameter ρ is estimated by maximum likelihood method. All features of a radial basis function (RBF) kernel support vector machine (SVM) are generated using either publicly available or private training data. Lastly, the generated SVM is applied in the test data to detect contamination. If training data is not available, a default RBF kernel SVM model is used.

We demonstrate the potential of our approach using simulation experiments, creating datasets with varying levels of contamination. The datasets combine, *in silico*, exome sequencing data of DNA from two lymphoblastoid cell lines (NA12878 and NA10855). We generated VCF files using variants identified in these data, and then evaluated the power and false positive rate of our approach to detect same species contamination. Our simulation experiments show that our method can detect levels of contamination as low as 5% with reasonable false positive rates. Results in real data have sensitivity above 99.99% and specificity at 90.24%, even in the presence of DNA degradation that has similar features to contaminated samples. Additionally, the approach can identify the difference between mixture of tumor-normal cells and contamination. We provide an R software implementation of our approach using the `defcon()` function in the `vanquish: Variant Quality Investigation Helper R package` on CRAN.

4.1 Backgrounds

High throughout next generation sequencing (NGS), has shown its advantages over traditional Sanger sequencing and microarrays in terms of accuracy, cost and speed (Merchant et al. 2014; Van Dijk et al. 2014). As NGS technologies have matured, best practices for quality control and data processing procedures have also developed (Patel and Jain 2012). One key step in a robust pipeline is the detection of sample contamination detection. Sample contamination detection is a necessary data quality control step of NGS data analysis pipeline since contamination might be introduced during sample preparation and sequencing analysis. Sample contamination affects all downstream sample analysis and may even generate misleading results, which in turn lead to false positive associations and genotype

misclassification (Jun et al. 2012).

In this paper, our working definition of contamination is when any sample contains tissues from more than one contributing source. There are two major types of contamination: (1) cross-species contamination; (2) same-species or within-species contamination. Additionally, it is very common for a single cancer sample to contain both normal and tumor cells from the same patient or cell line.

Contamination can emerge in next generation sequencing samples for various reasons. Despite best practices, unclean lab devices are one of the most common causes, with the introduction of unexpected materials like *Mycoplasma* (Schmidt et al. 1995). This same problem exists in even more recent, large scale research projects, such as the 1000 Genomes Project (Langdon 2014). Contamination can also arise from sample handling, sample extraction, library preparation and amplification, sample multiplexing and inaccurate barcode sequencing (Simion et al. 2018). Available methods are mainly based on the sequencing and allele frequency information of the testing sample and can be categorized into two groups, according to the source of contaminants: cross-species contamination and same-species contamination.

Cross-species contamination has been well studied, with several methods that have emerged to detect this type of contamination (Korneliussen et al. 2014; Laurence et al. 2014; Schmieder and Edwards 2011; Strong et al. 2014). In fact, modern metagenomics approaches are extensions of cross species contamination detection approaches. For example, Schmieder and Edwards developed DeconSeq, a framework for identification and removal of human contamination from microbial metagenomes (Schmieder and Edwards 2011), providing a framework for cross-species contamination detection during sequencing alignment. Later in 2014, Merchant *et al.* used microbiome analysis software to scan a sample of domestic cow, *Bos taurus*, and found small contigs from microbial contaminants (Merchant et al. 2014). Generally, data is assembled from available Sanger reads to the known species, then the unmapped contigs within the assembly are classified by k -mer matching where a database containing all bacteria, archaea, and viruses from the RefSeq database. As a result, contigs aligning to other genomes are found as a sign of contamination.

By contrast, same-species or within-species contamination is relatively more challenging, with fewer methods to implement valid and robust approaches. Arguably the most commonly implemented approach, and the earliest developed is ContEst (Cibulskis et al. 2011), a module within the Genome Analysis ToolKit (GATK) software (McKenna et al. 2010). ContEst uses a Bayesian method to calculate the posterior probability of each contamination level and find the maximum a posterior probability (MAP) estimate of the

contamination level at homozygous loci. Assuming a uniform prior distribution, $Unif(0, 1)$, on the contamination level, the posterior distribution of contamination level is proportional to the joint distribution of seeing observed alleles given the qualities of base calling and the probabilities of observing true alleles in contaminated sample. Thus, ContEst requires VCF and BAM format input and general population frequency information, i.e., base identities and quality scores from sequencing data.

Later in 2012, Jun *et al.* developed the VerifyBamID package for detecting same-species contamination of human DNA samples in both sequence and array-based data (Jun *et al.* 2012). VerifyBamID implements both likelihood-based and regression-based approaches that assume no more than one contaminant contained in a testing DNA sample. The likelihood of a contamination level is maximized by using a grid search over each contamination level for a maximizer. While VerifyBamID has demonstrated good sensitivity in real normal data experiments, copy number alterations (CNAs) in tumor samples will shift allele frequencies away from those outside of CNA regions, which leads to misinterpreting copy number-driven shift as contamination (Bergmann *et al.* 2016).

The statistical model introduced in VerifyBamID, was further developed by Bergmann *et al.* in their Conpair method for detecting another source of same-species contamination - the mixture of tumor and normal cells from the same patient sample Bergmann *et al.* (2016). Conpair focuses on homozygous loci within tumor-normal paired samples. Given that homozygous markers are invariant to copy number changes; pre-selected highly informative genomic homozygous markers are provided with Conpair to perform contamination detection.

Other more recent methods developments have used haplotype structure for contamination detection in NGS data Sehn *et al.* (2015). Closely spaced SNP pairs within sequencing region are identified from the 1000 Genomes database Clarke *et al.* (2012). Then read haplotypes for these selected SNP pairs are inferred. Human-human admixture is suggested if more than two read haplotypes are observed at a given locus in a sample. The estimated level of contamination for each testing sample are twice the mean frequency of the minor haplotype.

While the current approaches have been successful in a broad range of applications, there are major limitations that we address with our current approach. These developments represent substantial improvements in both the practical implementation of the quality control procedures and the statistical model used. Current approaches rely on the sizeable human reference genome data, as well as at least two large, memory intensive files. They each require either tumor and normal BAM files (Conpair), or VCF file and BAM file

(VerifyBamID and ContEst). By using a combination of a Beta-Binomial assumption and support vector machines within our algorithm to detect same species contamination, it works directly from information in VCF file. Even for a tumor-normal paired sample, no other information is needed to perform contamination detection. The change points of B allele frequencies (from the VCF file) are detected and then all chromosomes are separated into shorter sequences. Each sequence overlapping any copy number variation or aberration region is detected and filtered. In the enclosed study, we demonstrate our method in both real and simulated data, and show that it demonstrates excellent sensitivity and specificity for both simulated and real data. We also describe an R package implementation of the method.

4.2 Methods and Materials

4.2.1 Beta-binomial Model of Allele Frequency in Next Generation Sequencing

The proposed method is designed for human applications and assumes a diploid genome. For each locus that contains a single nucleotide variant (SNV) called from next generation sequencing (NGS) data, we define the allele frequency as the number of counts for the alternative (non-reference genome) allele over the total number of depth. For any diploid genome, if an individual is homozygous for the alternative allele (denoted as alternative/alternative, 1/1), the expected allele frequency is 1; meanwhile, if an individual is heterozygous (denoted as reference/alternative, 0/1) at a locus, 0.5 is the expected allele frequency. Using these theoretical expectations motivates using the Binomial distribution for the number of reads at each locus,

$$P(x; n, p) = \binom{n}{x} p^x (1-p)^{n-x}, \quad (4.1)$$

where n is the total number of depth at the locus; p is the theoretical allele frequency; x is the number of counts for the alternative allele.

While such a simple model is intuitively appealing, previous studies have discovered extra binomial dispersion, specifically, overdispersion of allele frequency distributions (Esteve-Codina et al. 2011; Pickrell et al. 2010; Skelly et al. 2011; Zhang et al. 2014). This overdispersion results in a higher variability than the Binomial distribution, so a distribution

that models such large variance is needed. Previous studies proposed and demonstrated the Beta-binomial distribution as an appropriate model for allele frequencies at a particular locus in a subpopulation. The Beta-binomial distribution is a discrete hierarchical model containing the Beta distribution and Binomial distribution, where the probability follows the Beta distribution; and the response follows the Binomial distribution. Hence the probability mass function of the Beta-binomial distribution is

$$P(x; n, a, b) = \binom{n}{x} \frac{B(x+a, n-x+b)}{B(a, b)}, \quad (4.2)$$

where n is the total number of reads at the locus; $B(a, b)$ is the beta function theoretical allele frequency; x is the number of counts for the alternative allele. This model has been applied in a number of studies since, and the advantages of the Beta-binomial over the Binomial distribution in dealing with overdispersion has been repeatedly demonstrated (Chen et al. 2016; Mayba et al. 2014). This work motivates our use of the Beta-binomial distribution.

4.2.2 Quality Control of Variant Call Format (VCF) files

The input format for our method is the well-established Variant Call Format Danecsek et al. (2011). To our knowledge, this is the first method to detect same-species contamination from the VCF format. The VCF format contains all single nucleotide variation (SNV) information that is needed in the next steps. Before a VCF file is used in our software, quality control is needed to filter noise and unnecessary information. The recommended quality control and processing steps are outlined below, and are additional quality control steps beyond the processing to produce the VCF file itself.

Step 1. Indel (insertion/deletion) Filtering

The SNVs used in our method must be substitution variants, not copy number variations like insertions and deletions. Only substitution mutations result in heterozygous and homozygous genotypes that are appropriately modeled by the Beta-binomial distribution. Insertions and deletions are identified as any mutation segments with a length of more than one base pair, and are then filtered/dropped in this step.

Step 2. Homozygous and Heterozygous Genotype Calling

The next step of processing is to call the genotypes for modeling. As mentioned above, there are two genotypes for alternative allele at any SNV: homozygous and heterozygous. Suggested genotypes are listed in the GT field of VCF file where 0/0 means homozygous reference; 0/1 means heterozygous; and 1/1 means homozygous alternative. In step 2 of processing, new information is generated that summarizes the genotype in reference to the alternative allele, in which "Het" reflects 0/1 in GT, and "Hom" reflects 1/1 in GT. This results in two categories of called variants, each corresponding to its own Beta-binomial model. Homozygous reference (0/0) and other heterozygous genotypes (1/2, 2/3, and so on) are not considered to simplify computations, and are labeled as "Complex" and are not included in further calculations.

Step 3. Low and High Depth Filtering

NGS data is not completely error free, so it is important to identify whether a sequence is a true call or a sequencing error. One of the efficient methods is to set thresholds for coverage depth (Morgan et al. 2010). A previous study suggested read depths of > 50 provide acceptable sensitivity and specificity in mutation detection. A reasonable read depth threshold should be chosen according to the average read depths of a testing sample.

Step 4. Change point Detection for CNV

In a pure sample, when there is a copy number variation (CNV) region, its features look very similar to the region with more than one contributors (same species contamination). Hence it is necessary to filter CNV region before generating features. If there is CNV information already generated for testing sample, the function `vanquish::defcon()` can filter the CNV region directly. Otherwise, a change point detection method is used to detect the CNV region. It has been reported that the variances of B-Allele Frequency (BAF, alternative allele frequency) at heterozygous loci are different among normal, duplication, deletion and loss of heterozygous (Ku et al. 2013). Therefore, change point analysis can be employed to detect change point of variance, *i.e.*, border of copy number region. The package *changepoint* is applied on only heterozygous positions for multiple changepoint searching of variance change (Killick and Eckley 2014).

Table 4.1: All features for classification model and their descriptions.

Name	Description
LOH	Het/Hom, the ratio of heterozygous and homozygous markers
HomRate	The percentage of the loci in HomRate region
HighRate	The percentage of the loci in HighRate region
HetRate	The percentage of the loci in HetRate region
LowRate	The percentage of the loci in LowRate region
HomVar	The variance of allele frequencies in HomRate region
HetVar	The variance of allele frequencies in HetRate region
AvgLL	Average likelihood of all loci assuming Beta-binomial distribution

4.2.3 Distribution and Likelihood based Features

The next step of our approach is to generate variables/features that will be used for model building to predict whether a sample contains same-species contamination. Briefly, there are two types of features that are generated and used in model building: distribution-based feature and likelihood-based feature.

Distribution based features

By its definition, allele frequency is a real number between 0 and 1. To generate distribution-based features, allele frequency is binned into 4 regions as shown in Figure 4.1: Low Alternative Allele Frequency (LowRate), Heterozygous Alternative Allele Frequency (HetRate), High Alternative Allele Frequency (HighRate), and Homozygous Alternative Allele Frequency (HomRate). Here 0, 0.3, 0.7, and 0.99 are their corresponding cut-off values.

From the bins in Table 4.1, there are 8 distribution-based features generated for the following model-building steps. These features reflect the distribution of allele frequencies in the whole file, instead of at each variant calling position. Therefore, each input sample/VCF file has one set of features to represent itself.

Average log-likelihood, the likelihood based feature

The only likelihood based feature is the average likelihood of all loci in VCF file. The likelihood is calculated by applying the Beta-binomial distribution. In the current study, NA10855 and other available pure samples are selected as a reference genome to calculate maximum likelihood estimator for both parameters, p and ρ , in the Beta-binomial distribution (Sequenced at Q2 Solutions). With \hat{p} and $\hat{\rho}$, log likelihood of all loci are calculated so that the

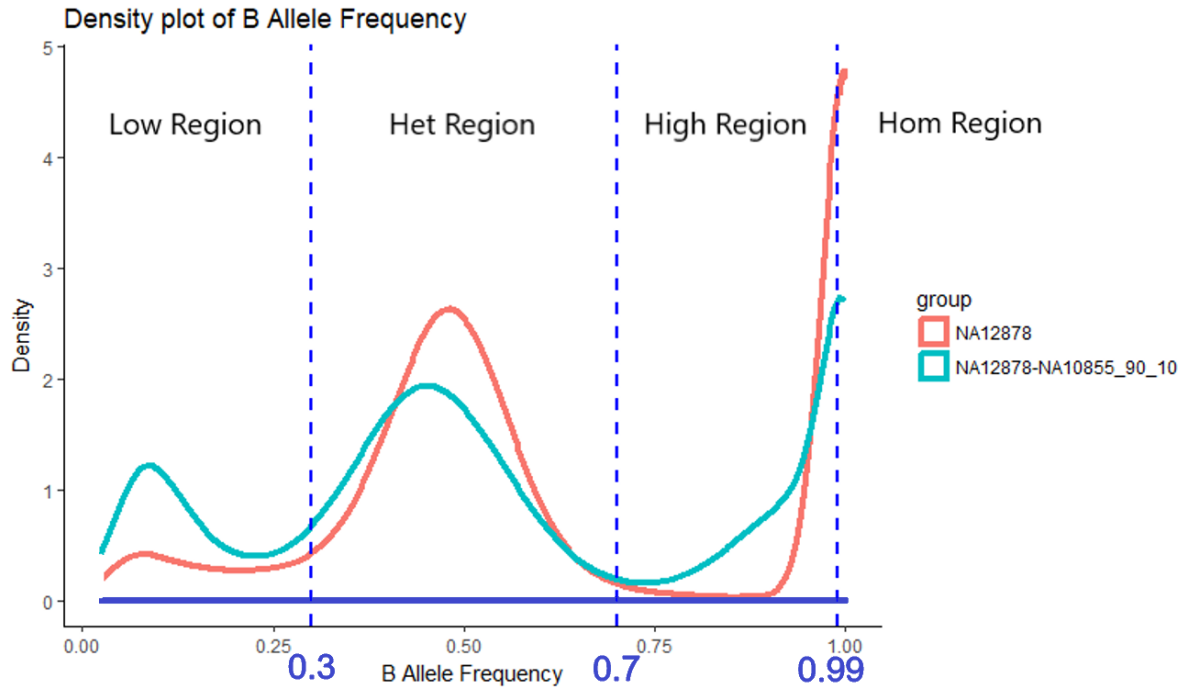


Figure 4.1: Four regions for B-Allele frequency: LowRate, HetRate, HighRate, and HomRate and their corresponding cut-off values. LowRate is the region where B-Allele frequency is below 0.3; HetRate is the region where B-Allele frequency is between 0.3 and 0.7; HighRate is the region where B-Allele frequency is between 0.7 and 0.99; HomRate is the region where B-Allele frequency is above 0.99. B Allele frequency density plots of pure (NA12878) and contaminated (NA12878 and NA10855) samples. The features used in SVM classifier are based on the difference between pure and contaminated curves.

last feature, their average value is also generated.

4.2.4 Support Vector Machine

After generating features, a classification method is used to identify whether a sample is from single contributor or multiple contributors. In the current study, we apply a Support Vector Machine (SVM) model because of the complexity in pattern recognition within feature space (Cortes and Vapnik 1995). The SVM method fits a hyperplane between single and multiple contributor regions to optimally discriminate between classifications. In our approach, we implement the SVM method using the `e1071` R package (Meyer et al. 2019). Since a linear model is not guaranteed, the Gaussian (Radial Basis Function) kernel is used to avoid such parametric assumptions. There are two parameters in SVM analysis that need to be tuned: cost and gamma. They are tuned using the parallel searching method, where a grid search is conducted on an exponentially growing sequence of cost and gamma to look for optimized paired values. It is possible for the estimated parameter to be different, given another training data set.

4.2.5 Variant Quality Investigation Helper

In this study, we introduced a novel strategy that detected same-species or within-species contamination by using B-allele frequency from variant call information only. An R package, *vanquish*: Variant Quality Investigation Helper, was produced to conduct this analysis. A brief flowchart (Figure 4.2) introducing contamination detection procedure shows the necessary steps as following:

1. Variant call format (VCF) generated by a variant caller was read into R by `vanquish::read_vcf` function. So far, the supported variant callers are GATK, VarDict and strelka2;
2. Copy number variation regions existing in VCF file were approximately detected and filtered by `vanquish::update_vcf` function;
3. Features for radial kernel SVM model were extracted from each sample by `vanquish::generate_feature` function;
4. Parameter cost and gamma for kernel SVM were tuned;
5. A test sample was predicted whether it was contaminated.

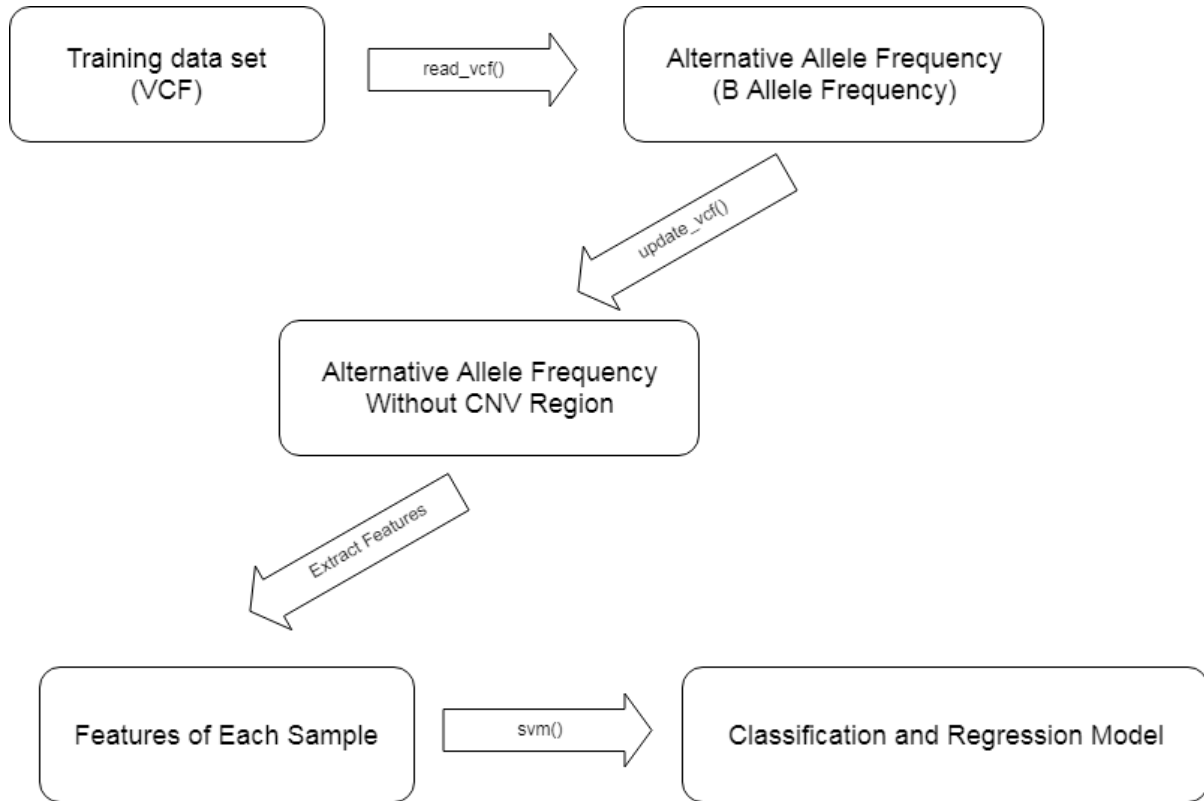


Figure 4.2: A brief flowchart of contamination detection procedure.

Additionally, two scenarios were found where the performance of this approach may be affected. First, if the sample is a mixture of tumor and normal cells from the sample individual, which is not considered contamination by our definition. The second scenario is when the test sample has very low quality so that no clear B-allele frequency pattern can be found.

4.3 Simulation and Real Application Studies

4.3.1 Changepoint analysis for approximate copy number region detection

In the scenario where the copy number information of a sample is not provided, change point analysis was conducted to find copy number regions of a sample. The `rmChangePoint()` function within our *vanquish* package imports `cpt.var()` from *changepoint* package (Killick and Eckley 2014). The pruned exact linear time (PELT) method (Killick et al. 2012)

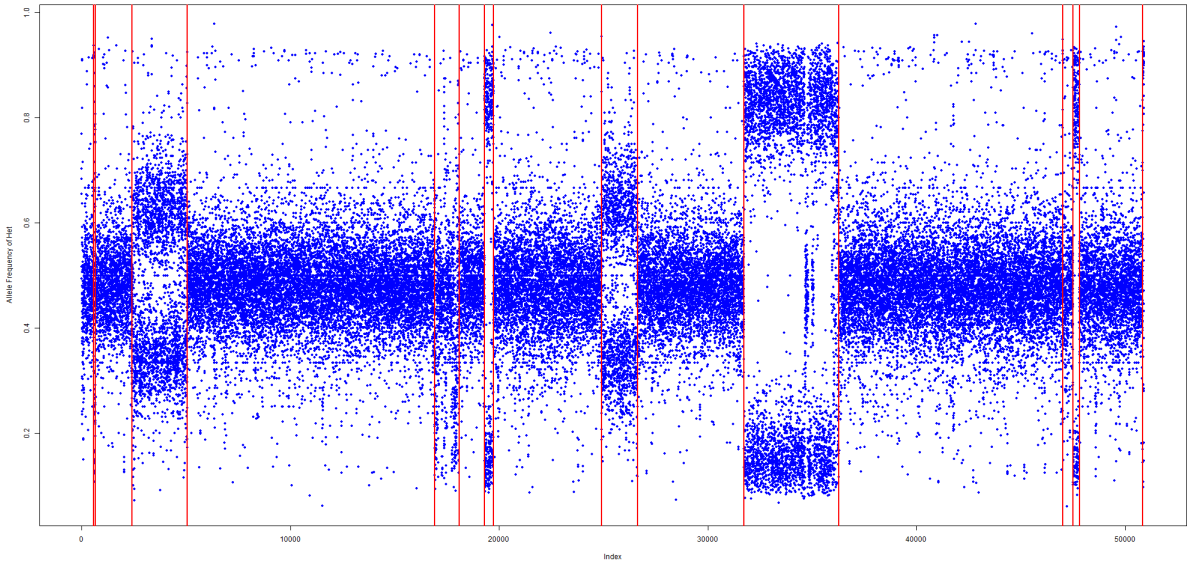
Table 4.2: Maximum Likelihood Estimator $\hat{\rho}$ of NA10855 Samples. $\hat{\rho}$ of heterozygous and homozygous models were estimated for each sequencing replicate. The sample mean can be used for generating features from training data set.

	Heterozygous $\hat{\rho}$	Homozygous $\hat{\rho}$
NA10855-1	0.154	0.0269
NA10855-2	0.223	0.0253
NA10855-3	0.177	0.0210
NA10855-4	0.187	0.0310
NA10855-5	0.169	0.0274
Sample Mean	0.182	0.0263

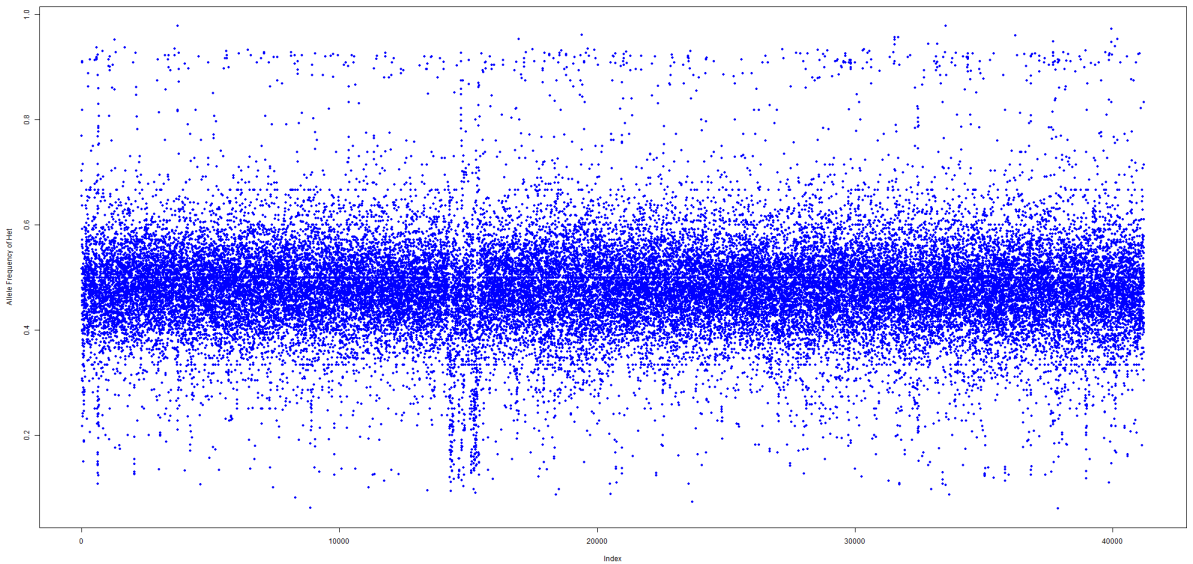
and the Changepoints for a Range of Penalties (CROPS) algorithm (Haynes et al. 2014) are employed to search for variance changepoints. In Figure 4.3a, the B-allele frequencies (only those between 0.05 and 0.95) of corresponding loci contained within the input VCF files are plotted. It is clear that there exist some CNV patterns. Red vertical lines are where the variance changes detected. The whole plot is separated into multiple parts by these change points. For each part, if the percentage of loci of the B-allele frequency is between 0.45 and 0.55 is larger than 10% and the skewness is larger than 0.5, the part will be kept in further analysis. The result after filtering is shown in Figure 4.3b. See the documentation of the *vanquish* package for more details.

4.3.2 Beta-binomial parameter estimation for reference sample(s)

In order to calculate likelihood-based features for further analysis, maximum likelihood estimators of ρ for Beta-binomial distribution of heterozygous and homozygous models are estimated. For the B-allele frequency, the theoretical value of the parameter p is 0.5 in heterozygous model and 1 in homozygous model. p is fixed at 0.5 and 0.999 to search for ρ in corresponding model. L-BFGS-B (Byrd et al. 1995) is applied for maximum value searching. For instance, NA10855 was chosen as a reference sample, and five replicates were sequenced by Q2 Solutions. The maximum likelihood estimator of ρ in each sample was estimated by ρ estimating function in the *vanquish* package (Table 4.2) and the sample averages were achieved for further analysis. The value the of estimator highly depends on the variant caller, so it is suggested to keep using the same variant caller for the reference sample, training sample, and test samples.



(a) Before copy number region filtering



(b) After copy number region filtering

Figure 4.3: Change point analysis for copy number region detection. The Y-axis shows B-Allele frequency; X-axis is shows the location number of each variant from chromosome 1 to 22. (A) is before copy number region filtering. Red lines indicate where the variance change based on detection results. It is clear that there exist some copy number patterns. (B) is after copy number region filtering and in which most copy number patterns are removed.

Table 4.3: Monte Carlo test results for parameter tuning and performance test.

Median Cost	Median Gamma	Average Sensitivity	Average Specificity
16	0.25	97.65%	96.27%

4.3.3 Features in the classification and regression model

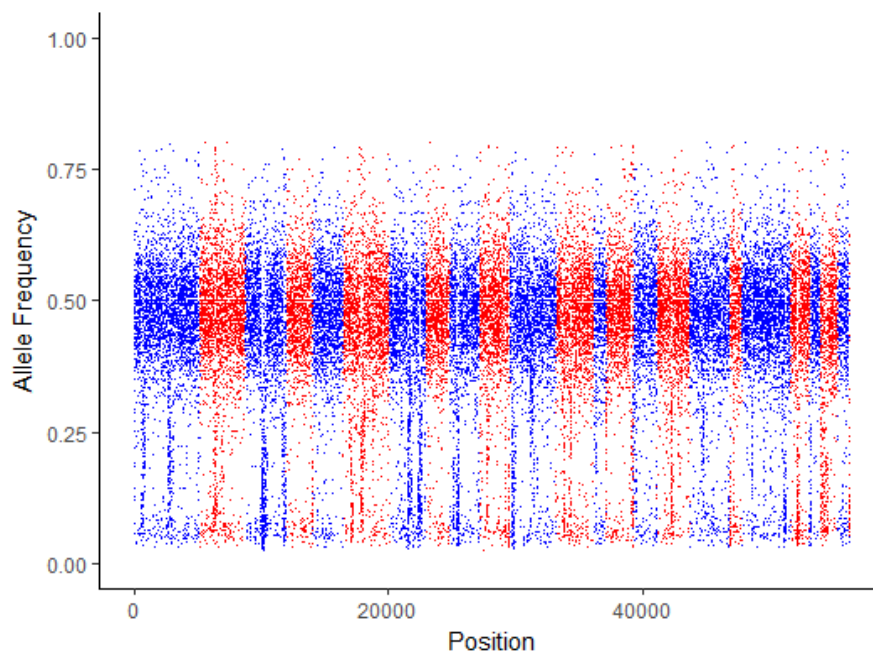
To train the classification and regression model, 238 samples were sequenced at Q2 Solutions as a training data set. 124 out of the 238 samples were pure; and the rest of the training set were contaminated. Some of the contaminated samples were mixed on purpose in wet lab; and others were simulated by two pure FASTQ format files (Cock et al. 2009). Pure and contaminated samples show different B-allele frequency patterns. As shown in Figure 4.4, only heterozygous loci detected in samples are plotted. The pure sample (Figure 4.4a) had a narrow horizontal band; while the contaminated sample (Figure 4.4b) had a relatively uniform distribution for B-allele frequency. Eight boxplots and t -tests (null hypothesis of no differences) were conducted to show the difference of pure and contaminated samples considering each feature (Figure 4.5). Among all eight features after t -tests, HomVar, HetVar and HighRate have significant p -values 1.786^{-9} , 1.750^{-6} , and 4.540^{-20} correspondingly. See Supplementary data for details of each feature.

4.3.4 Tune cost and gamma parameters in radial kernel SVM

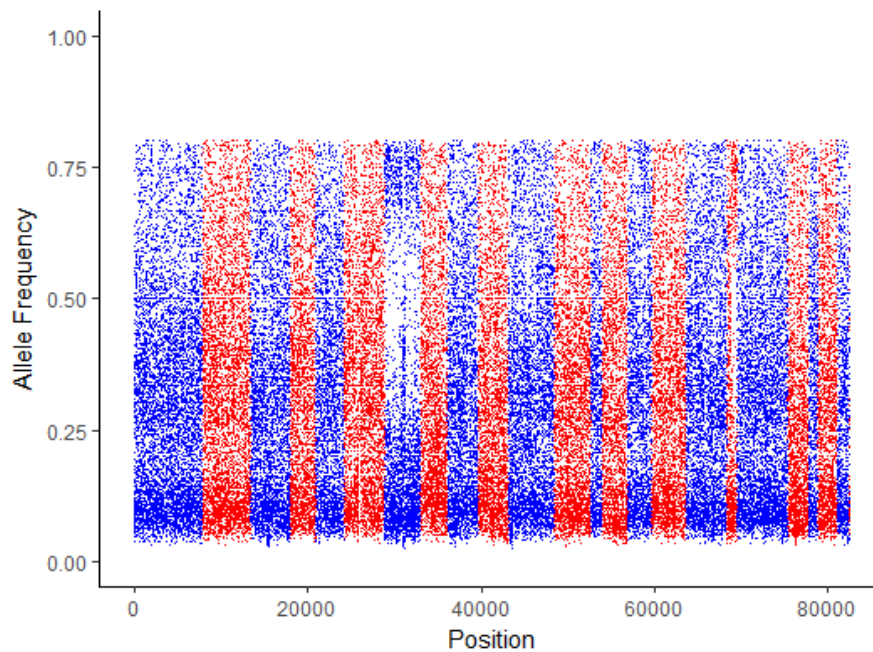
Monte Carlo method (1000 times) and `tune()` from R package *e1071* were used to tune parameters *cost* and *gamma* in support vector machine. All 238 samples were split into training (70%, 167 samples) and test (30%, 71 samples) sets. For the training set, we used grid search to tune parameter *cost* in the range of $(2^{-4}, 2^{12})$, and *gamma* in the range of $(2^{-4}, 2^4)$. Then sensitivity and specificity were calculated from the test set by using tuned *cost* and *gamma*. The results of Monte Carlo simulation are in Table 4.3. Median values of *cost* (16) and *gamma* (0.25), mean values of sensitivity (97.65%) and specificity (96.27%) were presented and the tuned *cost* and *gamma* were used in radial kernel SVM model for prediction.

4.3.5 Simulated data test results

Two reference samples, NA12878 and NA10855, were sequenced at Q2 Solutions. Two pairs of FASTQ format files were obtained from sequencing results, and where re-sampled



(a) Example of a sequenced pure sample: NA24143



(b) Example of a sequenced contaminated sample: Multiplex Reference

Figure 4.4: Heterozygous B-allele frequency patterns of pure (A) and contaminated (B) samples. Blue and red bands indicate different chromosomes. Sequenced at Q2 Solutions | EA Genomics, a Quintiles Quest Joint Venture.

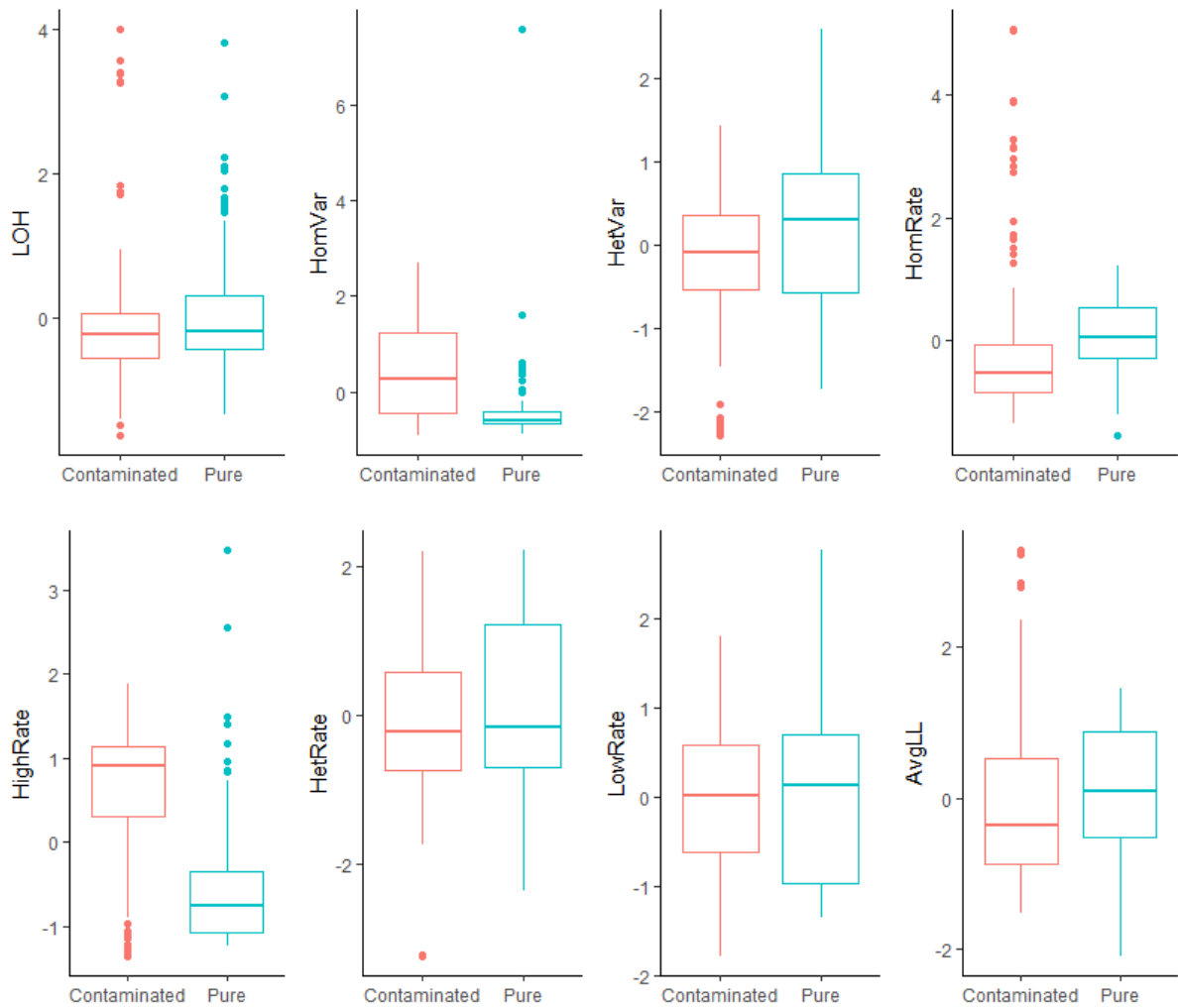


Figure 4.5: Boxplots of all features for pure samples (124) *vs* contaminated samples (114). Among all eight t-tests (null hypothesis of no difference hypothesis), HomVar, HetVar and HighRate have significant p -values 1.786^{-9} , 1.750^{-6} , and 4.540^{-20} correspondingly.

Table 4.4: Contamination Detection for a series of simulated data. (M: million)

Sample Component	Reads (NA12878)	Reads (NA10855)	Test Results
NA12878 (80%) + NA10855 (20%)	40M	10M	Contaminated
NA12878 (90%) + NA10855 (10%)	45M	5M	Contaminated
NA12878 (95%) + NA10855 (5%)	47.5M	2.5M	Contaminated
NA12878 (97.5%) + NA10855 (2.5%)	48.75M	1.25M	Pure
NA12878 (99%) + NA10855 (1%)	49.5M	0.5M	Pure
NA12878 (99.5%) + NA10855 (0.5%)	49.75M	0.25M	Pure

and mixed according to different proportions as shown in Table 4.4 by seqtk (Li 2012). In this simulated test, NA12878 was treated as the sample, while NA10855 was assumed to be mixed into the NA12878 sample. The percentages of the artificial contaminated mixtures range from 0.5% to 20%, and the detection rate at different levels of contamination were calculated. The total number of reads were 50 million for all 6 mixture samples. After running contamination analysis, contamination percentages above 5% were readily detected, while lower percentages were not detectable (Table 4.4). Based on this result, the contamination detection analysis has a detection sensitivity down to 5% in this case. If contaminant is less similar to the sample that it is mixed with, the detection sensitivity will be lower; on the other hand, if the similarity is larger, then contamination detection analysis will be more challenging.

4.3.6 Real data test results

After quantitative simulation test, the trained model was applied on a real data set (22 samples in all); and the results are shown in Table 4.5. The samples are ranked by regression values from `e1071::svm()`. Twenty out of twenty-two samples had the same prediction as prior identification. However, a pair of Human-T-Lymphoblast samples (see bold text in Table 4.5) were considered as contaminated but predicted as pure. Therefore, we zoomed in to check the distribution of B-allele frequency in this pair of samples (Figure 4.6). The copy number variation pattern (two bands and three bands in chromosomes) was shifted below (middle area of CNV pattern shifted from 0.5 to 0.3) since the sample was a mixture of tumor normal cells from the same person. The pattern was clear: the source of sample was uniform. Based on the distance of shift, the percentage of tumor cells and normal cells of the sample is even predictable.

A second data set (contains 53 samples in all) was also used to test the model (See

Table 4.5: Contamination Detection for a series of real data. Twenty out of twenty-two samples had the same prediction as prior identification. However, a pair of Human T-Lymphoblast samples (bold text) were considered as contaminated but predicted as pure.

Sample Name	Classification	Regression	Prior Identification
Human B-Lymphocyte L8	1	1.9243094	1
Human B-Lymphocyte 2 L20	1	1.9209875	1
Human Breast 2 L16	0	1.483925	0
Human Breast L4	0	1.463376	0
Human T-Lymphoblast 2 L21*	0	1.3622305	1
Human T-Lymphoblast L9	0	1.3472358	1
Human Brain L3	0	1.3147938	0
Human Brain 2 L15	0	1.303287	0
Human Testis L12	0	1.245767	0
Human Cervix 2 L17	0	1.2429423	0
Human Testis 2 L24	0	1.2424441	0
Human Cervix L5	0	1.203943	0
Human Macrophage L10	0	1.158416	0
Human Macrophage 2 L22	0	1.1582528	0
Human Liver 2 L18	0	1.1442246	0
Human Liposarcoma L7	0	1.1406007	0
Human Liposarcoma 2 L19	0	1.132044	0
Human Skin 2 L23	0	1.1209464	0
Human Skin L11	0	1.1194772	0
Human Liver L6	0	1.1170909	0
Human Reference DNA Male L1	0	1.0945151	0
Human Reference DNA Male 2 L13	0	1.0906951	0

* This is a mixture of tumor and normal cells. See Figure 4.6 for distribution of B-allele frequency of this sample.

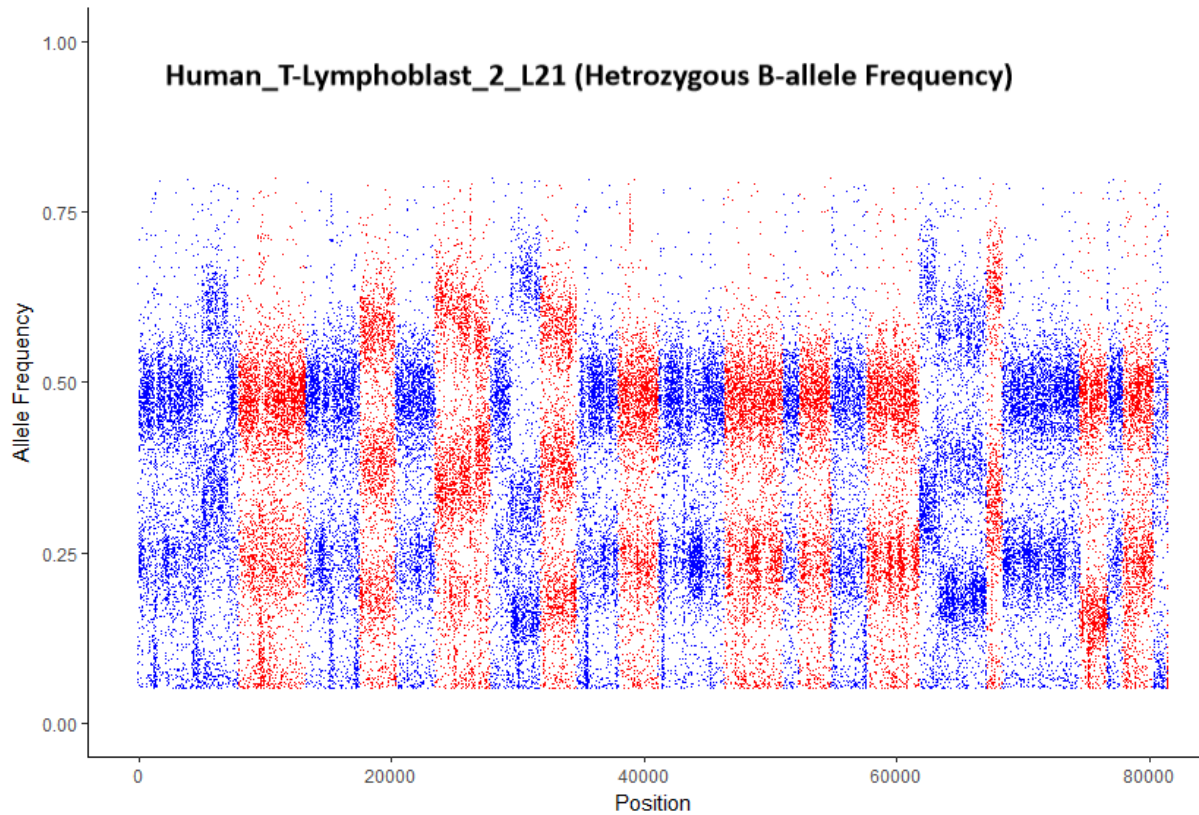


Figure 4.6: A sample that was previously considered as contaminated, but was detected as pure by `vanquish::defcon()`. The copy number variation pattern was shifted lower (middle area of CNV pattern shifted from 0.5 to 0.3) since the sample was a mixture of tumor normal cells from the same person.

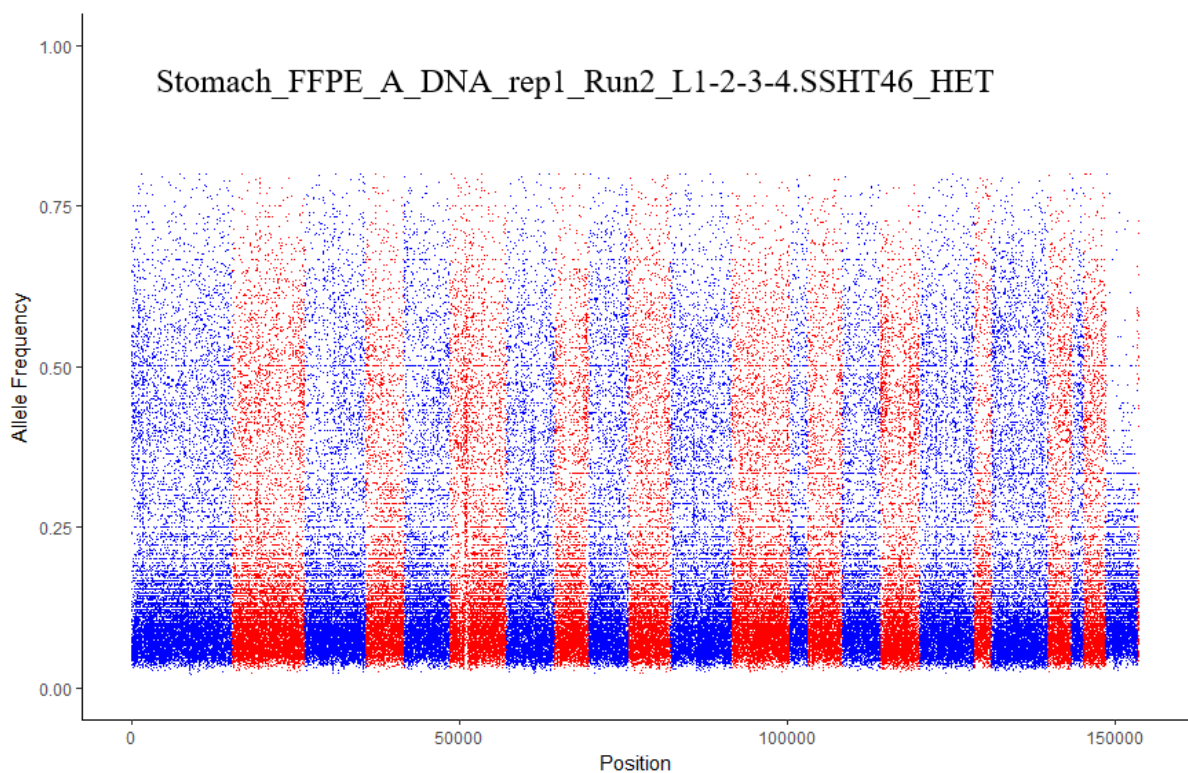


Figure 4.7: Formalin-fixed paraffin-embedded (FFPE) tissue with low quality. This sample was a false positive prediction result because of the similarity between its low quality and contaminants.

Section 4.5 for more details). Within this data set, 12 samples were mixed with contaminant on purpose; 41 samples were pure. The test results show that the sensitivity is $> 99.99\%$, and the specificity is 90.24% . All four false positive samples were formalin-fixed paraffin-embedded (FFPE) tissues, and were very likely to be degraded (Figure 4.7). When generating features from this degraded sample in Figure 4.7, its features looked very similar to features from a contaminated sample.

4.4 Discussion

In this study, we introduced a novel strategy that detected same-species or within-species contamination by using B-allele frequency from variant call information only. An R package, *vanquish: Variant Quality Investigation Helper*, was produced to conduct this analysis. A brief flowchart (Figure 4.2) introducing contamination detection procedure shows the

necessary steps as following:

- Variant call format (VCF) generated by a variant caller was read into R. So far, the supported variant callers are GATK, VarDict and strelka2.
- Copy number variation regions existing in VCF file were approximately detected and filtered.
- Features for radial kernel SVM model were extracted from each sample.
- Parameter cost and gamma for kernel SVM were tuned.
- A test sample was predicted whether it was contaminated.

Additionally, two scenarios were found where the performance of this approach may be affected. First, if the sample is a mixture of tumor and normal cells from the sample individual, it will be misclassified as contamination. However, it is actually not contaminated. The second scenario is when the test sample has very low quality, there is no clear B-allele frequency pattern can be found.

Finally, we have produced a user friendly R package to enable rapid analysis for same species analysis. Our tool uniquely performs this important step of QC from VCF files, improving the performance, memory requirements, etc. The running time of CNA region removal and feature generating is summarized in Figure 4.8 (Hardware: Dell R820, 512GB of RAM). Five samples without any known change point are run 10 times each to achieve their average running time, keeping their maximum numbers of runs of algorithm are uniform. Same as our expectation, larger samples need more time to do change point detection and feature generation. Besides, if a sample contains more change points, it also needs a longer running time.

4.5 Supplementary Methods

4.5.1 Features in Support Vector Machine (SVM)

All SNPs distributed across chromosomes are classified either homozygous (1/1) or heterozygous (0/1). LOH is the ratio of the number of heterozygous SNP loci over the number of homozygous SNP loci. A large LOH value means that there are more heterozygous SNP loci in a sample.

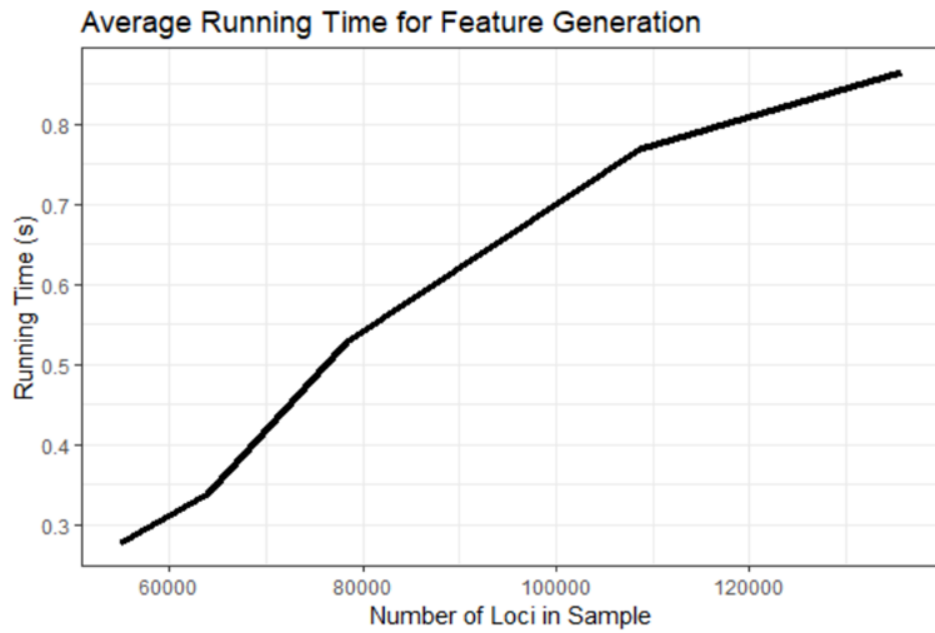
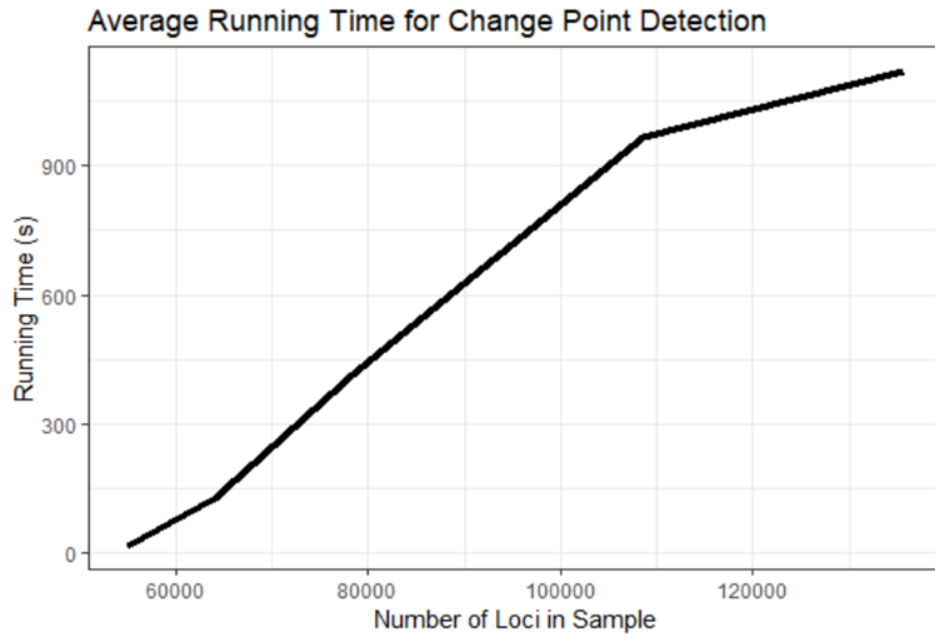


Figure 4.8: Average running time (of 10 runs) for change point detection (Top) and feature generation (Bottom). Five samples with different numbers of loci but the same maximum number of runs of algorithm (= 2).

Each SNP locus has a respective B-Allele frequency (BAF), which is percentage of the depth alternative allele out of the total depth at each SNP locus. Here $BAF \in [0, 1]$, and three cut-off values have been applied to separate the support set of BAF, $[0, 1]$, into 4 sub-regions: HomRate region $[0.99, 1]$, HighRate region $[0.7, 0.99)$, HetRate region $[0.3, 0.7)$, and LowRate region $[0, 0.3)$. A pure sample is expected to have higher HomeRate and HetRate values than a contaminated sample.

1. HomRate is the number of loci with $BAF \in [0.99, 1]$ over the total number of SNP loci in a sample.
2. HighRate is the number of loci with $BAF \in [0.7, 0.99)$ over the total number of SNP loci in a sample.
3. HetRate is the number of loci with $BAF \in [0.3, 0.7)$ over the total number of SNP loci in a sample.
4. LowRate is the number of loci with $BAF \in [0, 0.3)$ over the total number of SNP loci in a sample.

For SNP loci distributed within the HomRate region as defined above, the variance of BAF values of these SNP loci is calculated and defined as HomVar. HetVar is calculated in a similar procedure. A pure sample is expected to have lower HomeVar and HetVar values than a contaminated sample.

BAF of a SNP locus is assumed to follow the Beta-Binomial distribution. A reference sample (NA10855 sequenced at Q2 Solutions) is assumed as a pure sample. The chosen reference sample is used to calculate the maximum likelihood estimators for parameters p and ρ in the Beta-Binomial distribution. After that, log-likelihood of all SNP loci in testing sample are summed up. For comparability purpose, the sum of log-likelihood is then divided by the number of loci in each sample so that the final outcome is the average value of log-likelihood across all loci in a sample. A pure sample is expected to have higher AvgLL value than a contaminated sample.

4.5.2 Tunable Hyperparameters

Two hyper-parameters: the soft margin constant C and the inverse-width parameter of Gaussian kernel γ are optimized by using grid search and cross validation. Grid search is applied on exploring the two dimensional space (C, γ) . The grid points of C are chosen

on an exponential scale of $(2^{-4}, 2^{12})$; and grid points of γ are chosen between $(2^{-4}, 2^4)$. The sensitivity and specificity are estimated for each point on the grid.

4.5.3 Suggestion for Variant Calling Tool

Using various algorithms, different variant calling tools generate different allele frequency pattern. It is strongly suggested that the software used for training and testing data should be the same so that the features in models can be consistent and the classification or regression results can be accurate.

CHAPTER

5

CONCLUSIONS

5.1 Summary of Results

In this dissertation, we advanced machine learning methods for important applications in genetics, epigenetics, and genomics. We proposed two new variable selection algorithms in high-dimensional linear and tree-based models. Additionally, we developed an efficient machine learning method to detect same-species contamination detection. The overlapping and unique concepts between the projects presented are demonstrated in a Venn diagram in Figure 5.1.

In Chapter 2, we focused on how to find an appropriate penalty for high-dimensional regression. We proposed a new searching scheme, named higher criticism tuned regression, for the regularization parameter in penalized regression. We developed an algorithm to calculate p -values in high-dimensional data through data multi-splitting and p -value combination. By using simulation experiments and real data applications, we show that our approach is applicable to multiple penalized regression approaches. During optimal penalty searching, the construction of a proportion estimator plays a key role. As a result, we stated that compared with other approaches, our proposed method is robust for both

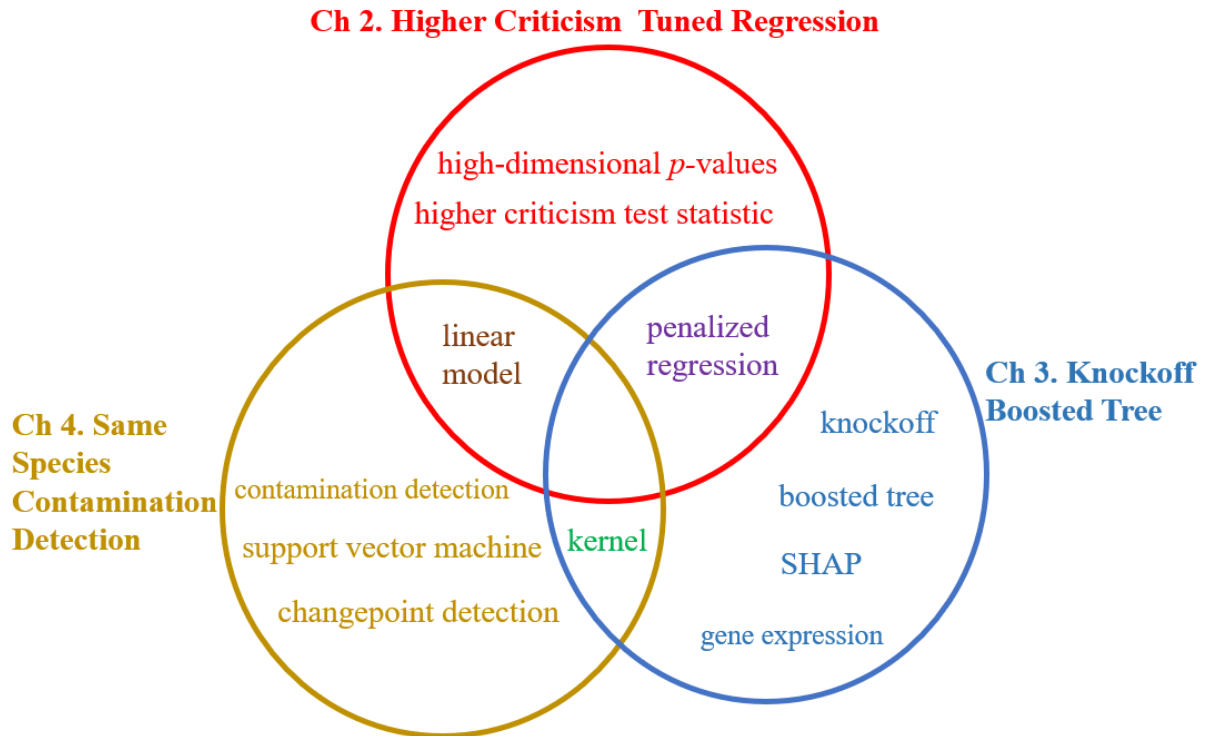


Figure 5.1: Contents of dissertation: Chapter 2. higher criticism tuned regression; Chapter 3. knockoff boosted tree; and Chapter 4. same species contamination detection.

weak and sparse signals. We applied our method on genome wide association study and DNA methylation data and find signals that match previous literature.

In Chapter 3, we introduced a new model-free variable selection method, called knockoff boosted trees. Since it is a tree-based model, the model does not require any prior model topology knowledge, such as whether an interaction term should be included or not. Unlike the current knockoff methods that have been only applied on linear models so far, we extend the models outside to nonlinear cases. Additionally, we introduced two new sampling methods to generate knockoffs, the principal component construction knockoff and the sparse Gaussian knockoff. Importantly, the principal component construction knockoff does not depend on Gaussian assumption of design matrix. Also, we proposed a new test statistic, *mean absolute angle of columns*, to evaluate the power of our created knockoffs. Our simulation experiments demonstrated the high power and nominal type I error using the method. We also tested different combinations of knockoffs and importance test statistics in tree models with main effect, interaction, exponential, and second order models. We demonstrated our method on TCGA data for gene selection in tumor purity estimation and

cancer type classification.

In Chapter 4, we introduced a strategy that detects same-species for within-species contamination from variant call information only using support vector machines. We created an R package, *vanquish*: Variant Quality Investigation Helper, to help users to perform analysis of VCF files in R. Currently, the supported variant callers are GATK, VarDict and strelka2. Our algorithm detects and filters most copy number variation regions to prevent calling tumor DNA contamination using our operational definition of contamination which does not include this type of situation. A radial kernel SVM model is trained using sample B-allele frequencies across the genome, and samples are classified as either contaminated or pure.

An important component of bioinformatics and statistical methods development is the dissemination of these methods with freely available implementation tools. For each of the methods we developed, corresponding R packages were produced so that other investigators can use these newest approaches.

5.2 Future Work

Bioinformatics benefits a lot from the recent advancements in mathematical algorithms, software development, and hardware supporting. As an interdisciplinary research field of statistics, biology, and computer science, bioinformatics focuses on method developing and data analysis for biological problems. Machine learning methods have become and will still be popular tools for bioinformaticians. For high-dimensional data, these machine learning methods perform well at variable selection, prediction and classification. Here we briefly discuss some future projects related to this dissertation.

1. **Tuning of multiple parameters:** Many concave penalties, such as the smoothly clipped absolute deviations (SCAD) penalty and the minimax concave penalty (MCP) penalty, involve a tuning parameter γ besides λ , which controls the concavity of the penalty. With the method proposed in Chapter 2, it is possible to tune multiple tuning parameters simultaneously. The potential application includes tuning complex penalty terms for association studies.
2. **Bayesian optimization for more appropriate penalization:** Current available grid search methods for tuning penalty terms decide a sequence of discrete values for generating penalized regression models. However, the spaces for those tuning parameters are continuous. Thus, grid searching is likely to ignore some values that

are better choices than all selected values. Bayesian optimization provide another solution to choose penalty parameters. The estimated tuning region introduced in Chapter 2 can be used as prior information. The application is to build a regression or classification model with better tuned penalty terms.

3. **Nonparametric knockoff generative adversarial network:** A generative adversarial network (GAN) (Goodfellow et al. 2014) learns to generate new data with the same statistics as the training set. These two neural networks then contest with each other. It is very interesting to apply nonparametric knockoff method introduced in Chapter 3 for generating new data and neural networks. Since our proposed method does not depend on any distribution assumption, it is very promising that the new nonparametric knockoff GAN can be robust for any data without distribution assumption. The application is conducting variable selection for extremely complex models with FDR control.

CHAPTER

6

R PACKAGES & SOFTWARE

Here is a summary of all developed R packages is documented in Table 6.1. Details of each R package are followed.

Table 6.1: A summary of R packages in the dissertation.

Name	Description	Chapter	Version	Availability
HCTR	Higher Criticism Tuned Regression	2	0.1.0	CRAN
KOBT	KnockOff Boosted Tree	3	0.1.0	CRAN
vanquish	Variant Quality Investigation Helper	4	1.0.0	CRAN

Package ‘HCTR’

October 7, 2019

Title Higher Criticism Tuned Regression

Version 0.1.0

Description A novel searching scheme for tuning parameter in high-dimensional penalized regression. We propose a new estimate of the regularization parameter based on an estimated lower bound of the proportion of false null hypotheses (Meinshausen and Rice (2006) <doi:10.1214/009053605000000741>). The bound is estimated by applying the empirical null distribution of the higher criticism statistic, a second-level significance testing, which is constructed by dependent p-values from a multi-split regression and aggregation method (Jeng, Zhang and Tzeng (2019) <doi:10.1080/01621459.2018.1518236>). An estimate of tuning parameter in penalized regression is decided corresponding to the lower bound of the proportion of false null hypotheses. Different penalized regression methods are provided in the multi-split algorithm.

Depends R (>= 3.4.0)

Imports glmnet (>= 2.0-18), harmonicmeanp (>= 3.0), MASS, ncvreg (>= 3.11-1), Rdpack (>= 0.11-0), stats

RdMacros Rdpack

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

NeedsCompilation no

Author Tao Jiang [aut, cre]

Maintainer Tao Jiang <tjiang8@ncsu.edu>

Repository CRAN

Date/Publication 2019-10-07 14:30:06 UTC

R topics documented:

bounding.seq	2
est.lambda	3

est.prop	3
final.selection	4
highdim.p	5
multi.adlasso	6
multi.lasso	6
multi.mcp	7
multi.scad	8
pmpv	8
Index	10

bounding.seq	<i>Bounding Sequence</i>
--------------	--------------------------

Description

Calculates bounding sequence of higher criticism for proportion estimator using p-values

Usage

bounding.seq(p.value, alpha)

Arguments

- p.value A matrix of p-values from permutation: row is from each permutation; column is from each variable.
- alpha Probability of Type I error for bounding sequence, the default value is $1 / \sqrt{\log(p)}$, where p is number of p-values in each permutation.

Value

A bounding value of higher criticism with (1 - alpha) confidence.

References

Jeng XJ, Zhang T, Tzeng J (2019). "Efficient Signal Inclusion With Genomic Applications." *Journal of the American Statistical Association*, 1–23.

Examples

```
set.seed(10)
X <- matrix(runif(n = 10000, min = 0, max = 1), nrow = 100)
result <- bounding.seq(p.value = X)
```

est.lambda

est.lambda

Estimated Lambda

Description

Estimate upper and lower bound of new tuning region of regularization parameter Lambda.

Usage

```
est.lambda(cv.fit, pihat, p, cov.num = 0)
```

Arguments

<code>cv.fit</code>	An object of either class "cv.glmnet" from <code>glmnet::cv.glmnet()</code> or class "cv.ncvreg" from <code>ncvreg::cv.ncvreg()</code> , which is a list generated by a cross-validation fit.
<code>pihat</code>	estimated proportion from <code>HCTR::est.prop()</code> .
<code>p</code>	Total number of variables, except for covariates.
<code>cov.num</code>	Number of covariates in model, default is 0. Covariate matrix, <i>W</i> , is assumed on the left side of variable matrix, <i>X</i> . The column index of covariates are before those of variables.

Value

A list of (1) `lambda.max`, upper bound of new tuning region; (2) `lambda.min`, lower bound of new tuning region.

Examples

```
set.seed(10)
X <- matrix(rnorm(20000), nrow = 100)
beta <- rep(0, 200)
beta[1:100] <- 5
Y <- MASS::mvrnorm(n = 1, mu = X%*%beta, Sigma = diag(100))
fit <- glmnet::cv.glmnet(x = X, y = Y)
pihat <- 0.01
result <- est.lambda(cv.fit = fit, pihat = pihat, p = ncol(X))
```

est.prop

Proportion Estimation

Description

Estimates false null hypothesis Proportion from multiple p-values using higher criticism test estimator.

Usage

```
est.prop(p.value, cn, adj = TRUE)
```

Arguments

`p.value` A sequence of p-values from test data, not including p-values from covariates.
`cn` A value of bounding sequence generated by `HCTR::bounding.seq()`.
`adj` A boolean algebra to decide whether to use adjusted Higher Criticism test statistic, the default value is `TRUE`.

Value

An estimated proportion of false null hypothesis.

References

Meinshausen N, Rice J (2006). "Estimating the proportion of false null hypotheses among a large number of independently tested hypotheses." *The Annals of Statistics*, **34**(1), 373–393.

Examples

```
set.seed(10)
X <- matrix(runif(n = 10000, min = 0, max = 1), nrow = 100)
result <- bounding.seq(p.value = X)
Y <- matrix(runif(n = 100, min = 0, max = 1), nrow = 100)
test <- est.prop(p.value = Y, cn = result)
```

final.selection	<i>Final Selection</i>
-----------------	------------------------

Description

Returns the index of final selected variables in the final chosen model.

Usage

```
final.selection(cv.fit, pihat, p, cov.num = 0)
```

Arguments

`cv.fit` An object of either class "cv.glmnet" from `glmnet::cv.glmnet()` or class "cv.ncvreg" from `ncvreg::cv.ncvreg()`, which is a list generated by a cross-validation fit.
`pihat` estimated proportion from `HCTR::est.prop()`.
`p` Total number of variables, except for covariates.
`cov.num` Number of covariates in model, default is 0. Covariate matrix, W , is assumed on the left side of variable matrix, X . The column index of covariates are before those of variables.

highdim.p

Value

A sequence of index of final selected variables in the final chosen model.

Examples

```
set.seed(10)
X <- matrix(rnorm(20000), nrow = 100)
beta <- rep(0, 200)
beta[1:100] <- 5
Y <- MASS::mvrnorm(n = 1, mu = X%%beta, Sigma = diag(100))
fit <- glmnet::cv.glmnet(x = X, y = Y)
pihat <- 0.01
result <- est.lambda(cv.fit = fit, pihat = pihat, p = ncol(X))
lambda.seq <- seq(from = result$lambda.min, to = result$lambda.max, length.out = 100)
# Note: The lambda sequences in glmnet and ncvreg are different.
fit2 <- glmnet::cv.glmnet(x = X, y = Y, lambda = lambda.seq)
result2 <- final.selection(cv.fit = fit2, pihat = 0.01, p = ncol(X))
```

highdim.p

p-values in high-dimensional linear model

Description

Calculates p-values in high-dimensional linear models using multi-split method

Usage

```
highdim.p(Y, X, W = NULL, type, B = 100, fold.num)
```

Arguments

Y	A numeric response vector, containing nobs variables.
X	An input matrix, of dimension nobs x nvars.
W	A covariate matrix, of dimension nobs x nvars, default is NULL.
type	Penalized regression type, valid parameters include "Lasso", "AdaLasso", "SCAD", and "MCP".
B	Multi-split times, default is 100.
fold.num	The number of cross validation folds.

Value

A list of objects containing: (1) harmonic mean p-values; (2) original p-values; (3) index of selected samples; (4) index of selected variables

Examples

```

set.seed(10)
X <- matrix(rnorm(20000), nrow = 100)
beta <- rep(0, 200)
beta[1:100] <- 5
Y <- MASS::mvrnorm(n = 1, mu = X%%beta, Sigma = diag(100))
result <- highdim.p(Y=Y, X=X, type = "Lasso", B = 2, fold.num = 10)

```

multi.adlasso	<i>Multi-split Adaptive Lasso</i>
---------------	-----------------------------------

Description

Multi-splitted variable selection using Adaptive Lasso

Usage

```
multi.adlasso(X, Y, covar.num = NULL, fold.num)
```

Arguments

X	An input matrix, of dimension nobs x nvars.
Y	A numeric response vector, containing nobs variables.
covar.num	Number of covariates in model, default is NULL.
fold.num	The number of cross validation folds.

Value

A list of two numeric objects of index of (1) selected and (2) unselected variables.

multi.lasso	<i>Multi-split Lasso</i>
-------------	--------------------------

Description

Multi-splitted variable selection using Lasso

Usage

```
multi.lasso(X, Y, p.fac = NULL, fold.num)
```

multi.mcp

Arguments

- X An input matrix, of dimension nobs x nvars.
- Y A numeric response vector, containing nobs variables.
- p. fac A sequence of penalty factor applied on each variable.
- fold.num The number of cross validation folds.

Value

A list of two numeric objects of index of (1) selected and (2) unselected variables.

multi.mcp	<i>Multi-split MCP</i>
-----------	------------------------

Description

Multi-splitted variable selection using MCP

Usage

multi.mcp(X, Y, p.fac = NULL, fold.num)

Arguments

- X An input matrix, of dimension nobs x nvars.
- Y A numeric response vector, containing nobs variables.
- p. fac A sequence of penalty factor applied on each variable.
- fold.num The number of cross validation folds.

Value

A list of two numeric objects of index of (1) selected and (2) unselected variables.

multi.scad	<i>Multi-split SCAD</i>
------------	-------------------------

Description

Multi-splitted variable selection using SCAD

Usage

```
multi.scad(X, Y, p.fac = NULL, fold.num)
```

Arguments

X	An input matrix, of dimension nobs x nvars.
Y	A numeric response vector, containing nobs variables.
p.fac	A sequence of penalty factor applied on each variable.
fold.num	The number of cross validation folds.

Value

A list of two numeric objects of index of (1) selected and (2) unselected variables.

pmpv	<i>Permutation p-values</i>
------	-----------------------------

Description

Calculates

Usage

```
pmpv(Y, X, W = NULL, type, B = 100, fold.num = 10, perm.num = 1000)
```

Arguments

Y	A numeric response vector, containing nobs variables.
X	An input matrix, of dimension nobs x nvars.
W	A covariate matrix, of dimension nobs x ncors, default is NULL.
type	Penalized regression type, valid parameters include "Lasso", "AdaLasso", "SCAD", and "MCP".
B	Multi-split times, default is 100.
fold.num	The number of cross validation folds, default is 10.
perm.num	Permutation times, default is 1000.

pmpv

Value

A matrix containing harmonic mean p-values from permutation.

Examples

```
set.seed(10)
X <- matrix(rnorm(20000), nrow = 100)
beta <- rep(0, 200)
beta[1:100] <- 5
Y <- MASS::mvrnorm(n = 1, mu = X%%beta, Sigma = diag(100))
result <- pmpv(Y=Y, X=X, type = "Lasso", B = 2, fold.num = 10, perm.num = 10)
```

Package ‘KOBT’

February 20, 2020

Type Package

Title Knockoff Boosted Tree

Version 0.1.0

Description A novel strategy for conducting variable selection without prior model topology knowledge using the knockoff method (Barber and Candès (2015) <doi:10.1214/15-AOS1337>) with extreme boosted tree models (Chen and Guestrin (2016) <doi:10.1145/2939672.2939785>). This method is inspired by the original knockoff method, where the differences between original and knockoff variables are used for variable selection with false discovery rate control. In addition to the original knockoff generating methods, two new sampling methods are available to be implemented, namely the sparse covariance and principal component knockoff methods. As results, the indices of selected variables are returned.

Depends R (>= 3.4.0)

Imports glmnet (>= 2.0-18), knockoff, spcov, xgboost, Rdpack (>= 0.11-0), stats, MASS

RdMacros Rdpack

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 7.0.2

NeedsCompilation no

Author Tao Jiang [aut, cre]

Maintainer Tao Jiang <tjiang8@ncsu.edu>

Repository CRAN

Date/Publication 2020-02-20 14:00:10 UTC

R topics documented:

create.pc.knockoff	2
generate.knockoff	2
importance.score	3
kobt.select	4
reduce.dim	5

Index**6**

create.pc.knockoff *Create PC Knockoffs*

Description

Create non-parametric knockoffs based on principal component regression and residuals permutation.

Usage

```
create.pc.knockoff(X, pc.num)
```

Arguments

`X` An input original design matrix.
`pc.num` The number of principal components to be used for generating knockoff matrices.

Value

A principal component knockoff matrix.

Examples

```
set.seed(10)
X <- matrix(rnorm(100), nrow = 10)
tmp <- create.pc.knockoff(X = X, pc.num = 5)
```

generate.knockoff *Generate Knockoff Matrix*

Description

Generate different types of knockoff matrices given an original one.

Usage

```
generate.knockoff(X, type, num, num.comp = 10)
```

Arguments

`X` An input original design matrix.
`type` The knockoff type to be generated. There are three choices available: (1) "shrink" for the shrink Gaussian knockoff; (2) "sparse" for the sparse Gaussian knockoff; and (3) "pc" for the principal component knockoff.
`num` The number of knockoff matrices to be created.
`num.comp` The number of principal components to be used for generating knockoff matrices, the default is 10.

`importance.score`

Value

A list of created knockoff matrices.

References

Barber RF, Candès EJ, others (2015). “Controlling the false discovery rate via knockoffs.” *The Annals of Statistics*, **43**(5), 2055–2085. Candès E, Fan Y, Janson L, Lv J (2018). “Panning for gold: Knockoffs for high dimensional controlled variable selection.” *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **80**(3), 551–577. Bien J, Tibshirani RJ (2011). “Sparse estimation of a covariance matrix.” *Biometrika*, **98**(4), 807–820.

Examples

```
set.seed(10)
X <- matrix(rnorm(100), nrow = 10)
Z <- generate.knockoff(X = X, type = "shrink", num = 5)
```

<code>importance.score</code>	<i>Importance Score</i>
-------------------------------	-------------------------

Description

Generate SHAP (SHapley Additive exPlanations) and Saabas scores.

Usage

```
importance.score(fit, Y, X)
```

Arguments

<code>fit</code>	A fitted object of class <code>xgb.Booster</code> .
<code>Y</code>	A vector of responses.
<code>X</code>	An input design matrix.

Value

A list of (1) `shap`, a vector of Hapley Additive exPlanations for each feature; (2) `saabas`, a vector of an individualized heuristic feature attribution method, which can be considered as an approximation for `shap`.

References

Candès E, Fan Y, Janson L, Lv J (2018). “Panning for gold: Knockoffs for high dimensional controlled variable selection.” *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **80**(3), 551–577. Chen T, Guestrin C (2016). “Xgboost: A scalable tree boosting system.” In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 785–794. Lundberg SM, Lee S (2017). “A unified approach to interpreting model predictions.” In *Advances in neural information processing systems*, 4765–4774.

Examples

```

set.seed(10)
X <- matrix(rnorm(100), nrow = 10)
Y <- matrix(rnorm(10), nrow = 10)
dtrain <- xgboost::xgb.DMatrix(X, label = Y)
fit.model <- xgboost::xgb.train(data = dtrain, nrounds = 5)
tmp <- importance.score(fit = fit.model, Y = Y, X = X)

```

kobt.select

Knockoff Variable Selection

Description

Use knockoff to conduct variable selection with false discovery rate control.

Usage

```
kobt.select(score, fdr = 0.1, type = "modified")
```

Arguments

score	An n by $2p$ matrix of test statistics, which includes test statistics from n samples, p variables (first p columns), and p knockoff variables (last p columns).
fdr	The targeted false discovery rate (FDR), the default value is 0.1.
type	A character showing the type of calculated false discovery rate: (1) modified and (2) usual FDR, the default value is modified.

Value

Indices of selected columns/variables in the n by p original design matrix.

References

Candes E, Fan Y, Janson L, Lv J (2018). "Panning for gold: \tilde{X} -model-X knockoffs for high dimensional controlled variable selection." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **80**(3), 551–577.

Examples

```

set.seed(1010)
n <- 100
p <- 100
signal.num <- 20
W_left <- matrix(rnorm(n = n*signal.num, mean = 1, sd = 1), nrow = n)
W_right <- matrix(rnorm(n = n*(2*p-signal.num), mean = 0, sd = 1), nrow = n)
W <- cbind(W_left, W_right)
selected.index <- kobt.select(score = W)

```

reduce.dim

reduce.dim

Reduce Dimensionality

Description

Reduce the dimensionality (i.e., the column number) of a design matrix to a desired level using Lasso.

Usage

```
reduce.dim(fit, X, bound)
```

Arguments

fit	The fitted cross validation object generated by <code>glmnet::cv.glmnet</code> .
X	An input design matrix whose column number is the dimensionality to be reduced.
bound	The targeted number of dimensionality after reducing.

Value

A list of (1) `index.X`, indices of selected columns in the design matrix; (2) `sub.X`, indices of selected columns in the design matrix.

Examples

```
set.seed(10)
X <- matrix(rnorm(100), nrow = 10)
Y <- matrix(rnorm(10), nrow = 10)
set.seed(11)
cvob1 <- glmnet::cv.glmnet(X, Y)
tmp <- reduce.dim(fit = cvob1, X = X, bound = 3)
```

Package ‘vanquish’

September 5, 2018

Title Variant Quality Investigation Helper

Version 1.0.0

Description Imports Variant Calling Format file into R. It can detect whether a sample contains contaminant from the same species. In the first stage of the approach, a change-point detection method is used to identify copy number variations for filtering. Next, features are extracted from the data for a support vector machine model. For log-likelihood calculation, the deviation parameter is estimated by maximum likelihood method. Using a radial basis function kernel support vector machine, the contamination of a sample can be detected.

Depends R (>= 3.4.0)

Imports changepoint, e1071, ggplot2, stats, VGAM

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

NeedsCompilation no

Author Tao Jiang [aut, cre]

Maintainer Tao Jiang <tjiang8@ncsu.edu>

Repository CRAN

Date/Publication 2018-09-05 14:50:04 UTC

R topics documented:

config_df	2
defcon	3
generate_feature	4
getAlt2	4
getAnnoRate	5
getAvgLL	5

getLowDepth	6
getRatio	6
getSkewness	7
getSNVRate	7
getVar	8
locateFile	8
negll	9
readGATK	9
readStrelka	10
readVarDict	10
readVarPROWL	11
read_vcf	12
rho_est	13
rmChangePoint	13
rmCNVInVCF	14
summary_vcf	14
svm_class_model	15
svm_regression_model	15
train_ct	16
update_vcf	16
vcf_example	17

Index **18**

config_df	<i>Default parameters of config.</i>
-----------	--------------------------------------

Description

A dataframe containing default parameters.

Usage

config_df

Format

A data frame with 12 variables:

threshold Threshold for allele frequency
 skew Skewness for allele frequency
 lower Lower bound for allele frequency region
 upper Upper bound for allele frequency region
 ldpthred Threshold to determine low depth
 hom_mle Hom MLE of p in Beta-Binomial model
 het_mle Het MLE of p in Beta-Binomial model

defcon

Hom_thred Threshold between hom and high
High_thred Threshold between high and het
Het_thred Threshold between het and low
hom_rho Hom MLE of rho in Beta-Binomial model
het_rho Het MLE of rho in Beta-Binomial model

Source

Created by Tao Jiang

defcon *DEtection of Frequency CONtamination*

Description

Usage

```
defcon(file, rmCNV = FALSE, cnvobj = NULL, config = NULL,  
       class_model = NULL, regression_model = NULL)
```

Arguments

<code>file</code>	VCF input object
<code>rmCNV</code>	Remove CNV regions, default is FALSE
<code>cnvobj</code>	CNV object, default is NULL
<code>config</code>	config information of parameters. A default set is generated as part of the model and is included in a model object, which contains
<code>class_model</code>	An SVM classification model
<code>regression_model</code>	An SVM regression model

Value

A list containing (1) `stat`: a data frame with all statistics for contamination estimation; (2) `result`: contamination estimation (Class = 0, pure; Class = 1, contaminated)

Examples

```
data(vcf_example)  
result <- defcon(file = vcf_example)
```

generate_feature	<i>Feature Generation for Contamination Detection Model</i>
------------------	---

Description

Generates features from each pair of input VCF objects for training contamination detection model.

Usage

```
generate_feature(file, hom_p = 0.999, het_p = 0.5, hom_rho = 0.005,
  het_rho = 0.1, mixture, homcut = 0.99, highcut = 0.7, hetcut = 0.3)
```

Arguments

file	VCF input object
hom_p	The initial value for p in Homozygous Beta-Binomial model, default is 0.999
het_p	The initial value for p in Heterozygous Beta-Binomial model, default is 0.5
hom_rho	The initial value for rho in Homozygous Beta-Binomial model, default is 0.005
het_rho	The initial value for rho in Heterozygous Beta-Binomial model, default is 0.1
mixture	A vector of whether the sample is contaminated: 0 for pure; 1 for contaminated
homcut	Cutoff allele frequency value between hom and high, default is 0.99
highcut	Cutoff allele frequency value between high and het, default is 0.7
hetcut	Cutoff allele frequency value between het and low, default is 0.3

Value

A data frame with all features for training model of contamination detection

getAlt2	<i>Second alternative allele percentage</i>
---------	---

Description

Second alternative allele percentage

Usage

```
getAlt2(f)
```

Arguments

f	Input raw file
---	----------------

Value

Percent of the second alternative allele

getAnnoRate

getAnnoRate	<i>Annotation rate</i>
-------------	------------------------

Description

Annotation rate

Usage

```
getAnnoRate(f)
```

Arguments

f Input raw file

Value

Percentage of annotation locus

getAvgLL	<i>Calculate average log-likelihood</i>
----------	---

Description

Calculate average log-likelihood

Usage

```
getAvgLL(df, hom_mle, het_mle, hom_rho, het_rho)
```

Arguments

df Input modified file
hom_mle Hom MLE of p in Beta-Binomial model, default is 0.9981416 from NA12878_1_L5
het_mle Het MLE of p in Beta-Binomial model, default is 0.4737897 from NA12878_1_L5
hom_rho Hom MLE of rho in Beta-Binomial model, default is 0.04570275 from NA12878_1_L5
het_rho Het MLE of rho in Beta-Binomial model, default is 0.02224098 from NA12878_1_L5

Value

meanLL

getLowDepth	<i>Low depth percentage</i>
-------------	-----------------------------

Description

Low depth percentage

Usage

```
getLowDepth(f, ldpthred)
```

Arguments

f	Input raw file
ldpthred	Threshold to determine low depth, default is 20

Value

Percentage of low depth

getRatio	<i>Get the ratio of allele frequencies with a region</i>
----------	--

Description

Get the ratio of allele frequencies with a region

Usage

```
getRatio(subdf, lower, upper)
```

Arguments

subdf	Dataframe with calculated statistics
lower	Lower bound for allele frequency region
upper	Upper bound for allele frequency region

Value

Ratio of allele frequencies with a region

getSkewness

<code>getSkewness</code>	<i>Get absolute value of skewness</i>
--------------------------	---------------------------------------

Description

Get absolute value of skewness

Usage

`getSkewness(subdf)`

Arguments

`subdf` Input dataframe

Value

Absolute value of skewness

<code>getSNVRate</code>	<i>SNV percentage</i>
-------------------------	-----------------------

Description

SNV percentage

Usage

`getSNVRate(df)`

Arguments

`df` Input raw file

Value

Percentage of SNV

getVar	<i>Calculate zygosity variable</i>
--------	------------------------------------

Description

Calculate zygosity variable

Usage

```
getVar(df, state, hom_mle, het_mle)
```

Arguments

df	Input modified file
state	Zygosity state
hom_mle	MLE in hom model
het_mle	MLE in het model

Value

Zygosity variable

locateFile	<i>Check input filename</i>
------------	-----------------------------

Description

Check input filename

Usage

```
locateFile(fn, extension)
```

Arguments

fn	Exact full file name of input file, including directory
extension	Expected input file extension: vcf & txt

Value

Valid directory

negll

negll	<i>Negative Log Likelihood</i>
-------	--------------------------------

Description

Calculates negative log likelihood for beta binomial distribution.

Usage

```
negll(x, size, prob, rho)
```

Arguments

x	Depth of alternative allele
size	Total depth
prob	Theoretical probability for heterozygous is 0.5, for homozygous is 0.999
rho	Rho parameter of Beta-Binomial distribution of alternative allele

readGATK	<i>Read in input vcf data in GATK format for Contamination detection</i>
----------	--

Description

Read in input vcf data in GATK format for Contamination detection

Usage

```
readGATK(dr, dbOnly, depCut, thred, content, extnum, keepall)
```

Arguments

dr	A valid input object
dbOnly	Use dbSNP as filter, default is FALSE, passed from read_vcf
depCut	Use a threshold for min depth , default is False
thred	Threshold for min depth, default is 20
content	Column names in VCF files
extnum	The column number or numbers to be extracted from vcf, default is 10; 0 for not extracting any columns
keepall	Keep unextracted column in output, default is TRUE, passed from read_vcf

Value

Dataframe from VCF file

readStrelka	<i>Read in input vcf data in strelka2 format for Contamination detection</i>
-------------	--

Description

Read in input vcf data in strelka2 format for Contamination detection

Usage

```
readStrelka(dr, dbOnly, depCut, thred, content, extnum, keepall)
```

Arguments

dr	A valid input object
dbOnly	Use dbSNP as filter, default is FALSE, passed from read_vcf
depCut	Use a threshold for min depth , default is False
thred	Threshold for min depth, default is 20
content	Column names in VCF files
extnum	The column number or numbers to be extracted from vcf, default is 10; 0 for not extracting any columns
keepall	Keep unextracted column in output, default is TRUE, passed from read_vcf

Value

Dataframe from VCF file

readVarDict	<i>Read in input vcf data in VarDict format for Contamination detection</i>
-------------	---

Description

Read in input vcf data in VarDict format for Contamination detection

Usage

```
readVarDict(dr, dbOnly, depCut, thred, content, extnum, keepall)
```

readVarPROWL

Arguments

<code>dr</code>	A valid input object
<code>dbOnly</code>	Use dbSNP as filter, default is FALSE, passed from <code>read_vcf</code>
<code>depCut</code>	Use a threshold for min depth , default is False
<code>thred</code>	Threshold for min depth, default is 20
<code>content</code>	Column names in VCF files
<code>extnum</code>	The column number to be extracted from vcf, default is 10; 0 for not extracting any column
<code>keepall</code>	Keep unextracted column in output, default is TRUE, passed from <code>read_vcf</code>

Value

Dataframe from VCF file

<code>readVarPROWL</code>	<i>Read in input vcf data in VarPROWL format</i>
---------------------------	--

Description

Read in input vcf data in VarPROWL format

Usage

```
readVarPROWL(dr, dbOnly, depCut, thred, content, extnum, keepall)
```

Arguments

<code>dr</code>	A valid input object
<code>dbOnly</code>	Use dbSNP as filter, default is FALSE, passed from <code>read_vcf</code>
<code>depCut</code>	Use a threshold for min depth , default is False
<code>thred</code>	Threshold for min depth, default is 20
<code>content</code>	Column names in VCF files
<code>extnum</code>	The column number or numbers to be extracted from vcf, default is 10; 0 for not extracting any columns
<code>keepall</code>	Keep unextracted column in output, default is TRUE, passed from <code>read_vcf</code>

Value

vcf Dataframe from VCF file

read_vcf

*VCF Data Input***Description**

Reads a file in vcf or vcf.gz file and creates a list containing Content, Meta, VCF and file_sample_name

Usage

```
read_vcf(fn, vcffor, dbOnly = FALSE, depCut = FALSE, thred = 20,
         metaline = 200, extnum = 10, keepall = TRUE, filter = FALSE)
```

Arguments

fn	Input vcf file name
vcffor	Input vcf data format: 1) GATK; 2) VarPROWL; 3) VarDict; 4) strelka2
dbOnly	Use dbSNP as filter, default is FALSE
depCut	Use a threshold for min depth , default is False
thred	Threshold for min depth, default is 20
metaline	Number of head lines to read in (better to be large enough), the lines will be checked if they contain meta information, default is 200
extnum	The column number to be extracted from vcf, default is 10; 0 for not extracting any column; extnum should be between 10 and total column number
keepall	Keep unextracted column in output, default is TRUE
filter	Whether to select "PASS" variants for analyses if they contain unfiltered variants, default is FALSE

Value

A list containing (1) Content: a vector showing what is contained; (2) Meta: a data frame containing meta-information of the file; (3) VCF: a data frame, the main part of VCF file; (4) file_sample_name: the file name and sample name, in case when multiple samples exist in one file, file and sample names might be different

Examples

```
file.name <- system.file("extdata", "example.vcf.gz", package = "vanquish")
example <- read_vcf(fn=file.name, vcffor="VarPROWL")
```

rho_est

rho_est *Estimate Rho for Alternative Allele Frequency*

Description

Estimates Rho parameter in beta binomial distribution for alternative allele frequency

Usage

```
rho_est(vl)
```

Arguments

vl A list of vcf objects from read_vcf function.

Value

A list containing (1) het_rho: Rho parameter of heterozygous location; (2) hom_rho: Rho parameter homozygous location;

Examples

```
data("vcf_example")
vcf_list <- list()
vcf_list[[1]] <- vcf_example$VCF
res <- rho_est(vl = vcf_list)
res$het_rho[[1]]$par
res$hom_rho[[1]]$par
```

rmChangePoint *Remove CNV regions within VCF files by change point method*

Description

Remove CNV regions within VCF files by change point method

Usage

```
rmChangePoint(vcf, threshold, skew, lower, upper)
```

Arguments

vcf Input VCF files
threshold Threshold for allele frequency
skew Skewness for allele frequency
lower Lower bound for allele frequency region
upper Upper bound for allele frequency region

Value

VCF object without change point region

rmCNVinVCF	<i>Remove CNV regions within VCF files given cnv file</i>
------------	---

Description

Remove CNV regions within VCF files given cnv file

Usage

```
rmCNVinVCF(vcf, cnvobj)
```

Arguments

vcf	Input VCF files
cnvobj	cnv object

Value

VCF object without change point region

summary_vcf	<i>VCF Data Summary</i>
-------------	-------------------------

Description

Summarizes allele frequency information in scatter and density plots

Usage

```
summary_vcf(vcf, ZG = NULL, CHR = NULL)
```

Arguments

vcf	VCF object from read_vcf function
ZG	zygosity: (1) null, for both het and hom, default; (2) het; (3) hom
CHR	chromosome number: (1) null, all chromosome, default; (2) any specific number

Value

A list containing (1) scatter: allele frequency scatter plot; (2) density: allele frequency density plot

svm_class_model

Examples

```
data("vcf_example")
tmp <- summary_vcf(vcf = vcf_example, ZG = 'het', CHR = c(1,2))
plot(tmp$scatter)
plot(tmp$density)
```

svm_class_model *Default svm classification model.*

Description

An svm object containing default svm classification model.

Usage

```
svm_class_model
```

Format

An svm object:

Source

Created by Tao Jiang

svm_regression_model *Default svm regression model.*

Description

An svm object containing default svm regression model.

Usage

```
svm_regression_model
```

Format

An svm object:

Source

Created by Tao Jiang

train_ct	<i>Train Contamination Detection Model</i>
----------	--

Description

Trains two SVM models (classification and regression) to detects whether a sample is contaminated another sample of its same species.

Usage

```
train_ct(feature)
```

Arguments

feature	Feature list objects from generate_feature()
---------	--

Value

A list contains two trained svm models: regression & classification

update_vcf	<i>Remove CNV regions within VCF files</i>
------------	--

Description

Remove CNV regions within VCF files

Usage

```
update_vcf(rmCNV = FALSE, vcf, cnvobj = NULL, threshold = 0.1,
           skew = 0.5, lower = 0.45, upper = 0.55)
```

Arguments

rmCNV	Remove CNV regions, default is FALSE
vcf	Input VCF files
cnvobj	cnv object, default is NULL
threshold	Threshold for allele frequency, default is 0.1
skew	Skewness for allele frequency, default is 0.5
lower	Lower bound for allele frequency region, default is 0.45
upper	Upper bound for allele frequency region, default is 0.55

Value

VCF file without CNV region

vcf_example

`vcf_example`

VCF example file.

Description

An example containing a list of 4 data frames.

Usage

`vcf_example`

Format

A list of 4 data frames:

Source

Created by Tao Jiang

REFERENCES

- Aran, D., Sirota, M., and Butte, A. J. (2015). Systematic pan-cancer analysis of tumour purity. *Nature communications*, 6:8971.
- Ayers, K. L. and Cordell, H. J. (2010). Snp selection in genome-wide and candidate gene studies via penalized logistic regression. *Genetic epidemiology*, 34(8):879–891.
- Barber, R. F., Candès, E. J., et al. (2015). Controlling the false discovery rate via knockoffs. *The Annals of Statistics*, 43(5):2055–2085.
- Barnett, I., Mukherjee, R., and Lin, X. (2017). The generalized higher criticism for testing snp-set effects in genetic association studies. *Journal of the American Statistical Association*, 112(517):64–76.
- Bergmann, E. A., Chen, B.-J., Arora, K., Vacic, V., and Zody, M. C. (2016). Conpair: concordance and contamination estimator for matched tumor–normal pairs. *Bioinformatics*, 32(20):3196–3198.
- Bhaskar, H., Hoyle, D. C., and Singh, S. (2006). Machine learning in bioinformatics: A brief survey and recommendations for practitioners. *Computers in Biology and Medicine*, 36(10):1104–1125.
- Bhatlekar, S., Fields, J. Z., and Boman, B. M. (2014). Hox genes and their role in the development of human cancers. *Journal of molecular medicine*, 92(8):811–823.
- Bien, J. and Tibshirani, R. J. (2011). Sparse estimation of a covariance matrix. *Biometrika*, 98(4):807–820.
- Breheny, P. and Breheny, M. P. (2019). Package 'ncvreg'. *R package version*.
- Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.
- Breiman, L. and Cutler, A. (2008). Random forests-classification manual. URL <http://www.math.usu.edu/~adele/forests>.
- Breiman, L., Friedman, J., Olshen, R., and Stone, C. (1984). Classification and regression trees. wadsworth int. *Group*, 37(15):237–251.
- Bühlmann, P. and Van De Geer, S. (2011). *Statistics for high-dimensional data: methods, theory and applications*. Springer Science & Business Media.
- Byrd, R. H., Lu, P., Nocedal, J., and Zhu, C. (1995). A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208.
- Candès, E., Fan, Y., Janson, L., and Lv, J. (2018). Panning for gold: 'model-x' knockoffs for high dimensional controlled variable selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 80(3):551–577.

- Carter, S. L., Cibulskis, K., Helman, E., McKenna, A., Shen, H., Zack, T., Laird, P. W., Onofrio, R. C., Winckler, W., Weir, B. A., et al. (2012). Absolute quantification of somatic dna alterations in human cancer. *Nature biotechnology*, 30(5):413.
- Chen, J., Rozowsky, J., Galeev, T. R., Harmanci, A., Kitchen, R., Bedford, J., Abyzov, A., Kong, Y., Regan, L., and Gerstein, M. (2016). A uniform survey of allele-specific binding and expression over 1000-genomes-project individuals. *Nature communications*, 7:11101.
- Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794. ACM.
- Chinchor, N. (1992). Muc-4 evaluation metrics. In *Proceedings of the 4th conference on Message understanding*, pages 22–29. Association for Computational Linguistics.
- Cho, S., Kim, K., Kim, Y. J., Lee, J.-K., Cho, Y. S., Lee, J.-Y., Han, B.-G., Kim, H., Ott, J., and Park, T. (2010). Joint identification of multiple genetic variants via elastic-net variable selection in a genome-wide association analysis. *Annals of human genetics*, 74(5):416–428.
- Chung, E., Romano, J. P., et al. (2013). Exact and asymptotically robust permutation tests. *The Annals of Statistics*, 41(2):484–507.
- Cibulskis, K., McKenna, A., Fennell, T., Banks, E., DePristo, M., and Getz, G. (2011). Contest: estimating cross-contamination of human samples in next-generation sequencing data. *Bioinformatics*, 27(18):2601–2602.
- Clarke, L., Zheng-Bradley, X., Smith, R., Kulesha, E., Xiao, C., Toneva, I., Vaughan, B., Preuss, D., Leinonen, R., Shumway, M., et al. (2012). The 1000 genomes project: data management and community access. *Nature methods*, 9(5):459.
- Cock, P. J., Fields, C. J., Goto, N., Heuer, M. L., and Rice, P. M. (2009). The sanger fastq file format for sequences with quality scores, and the solexa/illumina fastq variants. *Nucleic acids research*, 38(6):1767–1771.
- Consortium, C. T. et al. (2003). The nature and identification of quantitative trait loci: a community's view. *Nature Reviews Genetics*, 4(11):911.
- Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3):273–297.
- Danecek, P., Auton, A., Abecasis, G., Albers, C. A., Banks, E., DePristo, M. A., Handsaker, R. E., Lunter, G., Marth, G. T., Sherry, S. T., et al. (2011). 1000 genomes project analysis group (2011). *The variant call format and VCFtools*. *Bioinformatics*, 27(15):2156–2158.
- de Barros, J.-P. P., Boualam, A., Gautier, T., Dumont, L., Vergès, B., Masson, D., and Lagrost, L. (2009). Apolipoprotein ci is a physiological regulator of cholesteryl ester transfer protein activity in human plasma but not in rabbit plasma. *Journal of lipid research*, 50(9):1842–1851.

- Deshmukh, H. A., Colhoun, H. M., Johnson, T., McKeigue, P. M., Betteridge, D. J., Durrington, P. N., Fuller, J. H., Livingstone, S., Charlton-Menys, V., Neil, A., et al. (2012). Genome-wide association study of genetic determinants of ldl-c response to atorvastatin therapy: importance of lp (a). *Journal of lipid research*, 53(5):1000–1011.
- Dezeure, R., Bühlmann, P., Meier, L., and Meinshausen, N. (2015). High-dimensional inference: Confidence intervals, p-values and r-software hdi. *Statistical science*, pages 533–558.
- Dietterich, T. G. and Kong, E. B. (1995). Machine learning bias, statistical bias, and statistical variance of decision tree algorithms. Technical report, Technical report, Department of Computer Science, Oregon State University.
- Donoho, D., Jin, J., et al. (2004). Higher criticism for detecting sparse heterogeneous mixtures. *The Annals of Statistics*, 32(3):962–994.
- Dron, J. S. and Hegele, R. A. (2016). Genetics of lipid and lipoprotein disorders and traits. *Current genetic medicine reports*, 4(3):130–141.
- Esteve-Codina, A., Kofler, R., Palmieri, N., Bussotti, G., Notredame, C., and Pérez-Enciso, M. (2011). Exploring the gonad transcriptome of two extreme male pigs with rna-seq. *BMC genomics*, 12(1):552.
- Fan, J. and Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American statistical Association*, 96(456):1348–1360.
- Frank, E., Hall, M., Trigg, L., Holmes, G., and Witten, I. H. (2004). Data mining in bioinformatics using weka. *Bioinformatics*, 20(15):2479–2481.
- 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.
- Friedman, J., Hastie, T., and Tibshirani, R. (2009). glmnet: Lasso and elastic-net regularized generalized linear models. *R package version*, 1(4).
- Friedman, J., Hastie, T., and Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1.
- Friedman, J., Hastie, T., Tibshirani, R., et al. (2000). Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The annals of statistics*, 28(2):337–407.
- Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232.
- Friedman, J. H. (2002). Stochastic gradient boosting. *Computational statistics & data analysis*, 38(4):367–378.

- Friedman, J. H. et al. (1991). Multivariate adaptive regression splines. *The Annals of Statistics*, 19(1):1–67.
- Gao, C., Sun, H., Wang, T., Tang, M., Bohnen, N. I., Müller, M. L., Herman, T., Giladi, N., Kalinin, A., Spino, C., et al. (2018). Model-based and model-free machine learning techniques for diagnostic prediction and classification of clinical outcomes in parkinson’s disease. *Scientific reports*, 8(1):7129.
- Gerstein, H., Miller, M., Byington, R., Goff, D. J., Bigger, J., Buse, J., Cushman, W., Genuth, S., Ismail-Beigi, F., Grimm, R. J., Probstfield, J., Simons-Morton, D., and Friedewald, W. (2008). Effects of intensive glucose lowering in type 2 diabetes. *New England journal of medicine*, 358(24):2545–2559.
- Glas, A. S., Lijmer, J. G., Prins, M. H., Bonsel, G. J., and Bossuyt, P. M. (2003). The diagnostic odds ratio: a single indicator of test performance. *Journal of clinical epidemiology*, 56(11):1129–1135.
- 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., et al. (1999). Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *science*, 286(5439):531–537.
- Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680.
- Gretton, A., Borgwardt, K., Rasch, M., Schölkopf, B., and Smola, A. J. (2007). A kernel method for the two-sample-problem. In *Advances in neural information processing systems*, pages 513–520.
- Group, A. S. (2010). Effects of intensive blood-pressure control in type 2 diabetes mellitus. *New England Journal of Medicine*, 362(17):1575–1585.
- Hall, P., Jin, J., et al. (2010). Innovated higher criticism for detecting sparse signals in correlated noise. *The Annals of Statistics*, 38(3):1686–1732.
- Hastie, T., Tibshirani, R., Friedman, J., and Franklin, J. (2005). The elements of statistical learning: data mining, inference and prediction. *The Mathematical Intelligencer*, 27(2):83–85.
- Haynes, K., Eckley, I. A., and Fearnhead, P. (2014). Efficient penalty search for multiple changepoint problems. *arXiv preprint arXiv:1412.3617*.
- Hesper, B. and Hogeweg, P. (1970). Bioinformatica: een werkconcept. *Kameleon*, 1(6):28–29.
- Ho, T. K. (1995). Proceedings of the 3rd international conference on document analysis and recognition. *Random Decision Forests*, pages 278–282.

- Hoadley, K. A., Yau, C., Hinoue, T., Wolf, D. M., Lazar, A. J., Drill, E., Shen, R., Taylor, A. M., Cherniack, A. D., Thorsson, V., et al. (2018). Cell-of-origin patterns dominate the molecular classification of 10,000 tumors from 33 types of cancer. *Cell*, 173(2):291–304.
- Hoerl, A. E. and Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67.
- Houseman, E. A., Accomando, W. P., Koestler, D. C., Christensen, B. C., Marsit, C. J., Nelson, H. H., Wiencke, J. K., and Kelsey, K. T. (2012). Dna methylation arrays as surrogate measures of cell mixture distribution. *BMC bioinformatics*, 13(1):86.
- Jeng, X. J., Zhang, T., and Tzeng, J.-Y. (2019). Efficient signal inclusion with genomic applications. *Journal of the American Statistical Association*, pages 1–23.
- Joehanes, R., Just, A. C., Marioni, R. E., Pilling, L. C., Reynolds, L. M., Mandaviya, P. R., Guan, W., Xu, T., Elks, C. E., Aslibekyan, S., et al. (2016). Epigenetic signatures of cigarette smoking. *Circulation: Cardiovascular Genetics*, 9(5):436–447.
- Jong, M. C., Hofker, M. H., and Havekes, L. M. (1999). Role of apocs in lipoprotein metabolism: functional differences between apoc1, apoc2, and apoc3. *Arteriosclerosis, thrombosis, and vascular biology*, 19(3):472–484.
- Jun, G., Flickinger, M., Hetrick, K. N., Romm, J. M., Doheny, K. F., Abecasis, G. R., Boehnke, M., and Kang, H. M. (2012). Detecting and estimating contamination of human dna samples in sequencing and array-based genotype data. *The American Journal of Human Genetics*, 91(5):839–848.
- Killick, R. and Eckley, I. (2014). changepoint: An r package for changepoint analysis. *Journal of statistical software*, 58(3):1–19.
- Killick, R., Fearnhead, P., and Eckley, I. A. (2012). Optimal detection of changepoints with a linear computational cost. *Journal of the American Statistical Association*, 107(500):1590–1598.
- Kim, B.-M., Buchner, G., Miletich, I., Sharpe, P. T., and Shivdasani, R. A. (2005). The stomach mesenchymal transcription factor barx1 specifies gastric epithelial identity through inhibition of transient wnt signaling. *Developmental cell*, 8(4):611–622.
- Kim, B.-M., Woo, J., Kanellopoulou, C., and Shivdasani, R. A. (2011). Regulation of mouse stomach development and barx1 expression by specific micrnas. *Development*, 138(6):1081–1086.
- Kim, H. and Loh, W.-Y. (2001). Classification trees with unbiased multiway splits. *Journal of the American Statistical Association*, 96(454):589–604.
- Korneliusson, T. S., Albrechtsen, A., and Nielsen, R. (2014). Angsd: analysis of next generation sequencing data. *BMC bioinformatics*, 15(1):356.

- Ku, C., Polychronakos, C., Tan, E., Naidoo, N., Pawitan, Y., Roukos, D., Mort, M., and Cooper, D. N. (2013). A new paradigm emerges from the study of de novo mutations in the context of neurodevelopmental disease. *Molecular psychiatry*, 18(2):141.
- Langdon, W. B. (2014). Mycoplasma contamination in the 1000 genomes project. *BioData Mining*, 7(1):3.
- Laurence, M., Hatzis, C., and Brash, D. E. (2014). Common contaminants in next-generation sequencing that hinder discovery of low-abundance microbes. *PLoS one*, 9(5):e97876.
- LeCun, Y., Bengio, Y., et al. (1995). Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995.
- Li, H. (2012). seqtk toolkit for processing sequences in fasta/q formats.
- Li, L., Dennis Cook, R., and Nachtsheim, C. J. (2005). Model-free variable selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):285–299.
- Li, Y., Kang, K., Krahn, J. M., Croutwater, N., Lee, K., Umbach, D. M., and Li, L. (2017). A comprehensive genomic pan-cancer classification using the cancer genome atlas gene expression data. *BMC genomics*, 18(1):508.
- Li, Y., Umbach, D. M., Bingham, A., Li, Q.-J., Zhuang, Y., and Li, L. (2019). Putative biomarkers for predicting tumor sample purity based on gene expression data. *BMC genomics*, 20(1):1–12.
- Liu, C., Min, L., Kuang, J., Zhu, C.-s., Qiu, X.-y., and Zhu, L. (2019). Bioinformatic identification of mir-622 key target genes and experimental validation of the mir-622-rnf8 axis in breast cancer. *Frontiers in oncology*, 9:1114.
- Lockhart, R., Taylor, J., Tibshirani, R. J., and Tibshirani, R. (2014). A significance test for the lasso. *Annals of statistics*, 42(2):413.
- Loh, W.-Y. (2002). Regression tress with unbiased variable selection and interaction detection. *Statistica Sinica*, pages 361–386.
- Lundberg, S. M., Erion, G. G., and Lee, S.-I. (2018). Consistent individualized feature attribution for tree ensembles. *arXiv preprint arXiv:1802.03888*.
- Lundberg, S. M. and Lee, S.-I. (2017a). Consistent feature attribution for tree ensembles. *arXiv preprint arXiv:1706.06060*.
- Lundberg, S. M. and Lee, S.-I. (2017b). A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, pages 4765–4774.
- Marvel, S. W., Rotroff, D. M., Wagner, M. J., Buse, J. B., Havener, T. M., McLeod, H. L., and Motsinger-Reif, A. A. (2017). Common and rare genetic markers of lipid variation in subjects with type 2 diabetes from the accord clinical trial. *PeerJ*, 5:e3187.

- Mayba, O., Gilbert, H. N., Liu, J., Haverty, P. M., Jhunjhunwala, S., Jiang, Z., Watanabe, C., and Zhang, Z. (2014). Mbased: allele-specific expression detection in cancer tissues and cell lines. *Genome biology*, 15(8):405.
- McKenna, A., Hanna, M., Banks, E., Sivachenko, A., Cibulskis, K., Kernytsky, A., Garimella, K., Altshuler, D., Gabriel, S., Daly, M., et al. (2010). The genome analysis toolkit: a mapreduce framework for analyzing next-generation dna sequencing data. *Genome research*, 20(9):1297–1303.
- Meinshausen, N., Meier, L., and Bühlmann, P. (2009). P-values for high-dimensional regression. *Journal of the American Statistical Association*, 104(488):1671–1681.
- Meinshausen, N., Rice, J., et al. (2006). Estimating the proportion of false null hypotheses among a large number of independently tested hypotheses. *The Annals of Statistics*, 34(1):373–393.
- Merchant, S., Wood, D. E., and Salzberg, S. L. (2014). Unexpected cross-species contamination in genome sequencing projects. *PeerJ*, 2:e675.
- Meyer, D., Dimitriadou, E., Hornik, K., Weingessel, A., Leisch, F., Chang, C.-C., Lin, C.-C., and Meyer, M. D. (2019). Package ‘r1071’. *The R Journal*.
- Miller, P. J., Lubke, G. H., McArtor, D. B., and Bergeman, C. (2016). Finding structure in data using multivariate tree boosting. *Psychological methods*, 21(4):583.
- Monick, M. M., Beach, S. R., Plume, J., Sears, R., Gerrard, M., Brody, G. H., and Philibert, R. A. (2012). Coordinated changes in ahrr methylation in lymphoblasts and pulmonary macrophages from smokers. *American journal of medical genetics part B: neuropsychiatric genetics*, 159(2):141–151.
- Morgan, J. E., Carr, I. M., Sheridan, E., Chu, C. E., Hayward, B., Camm, N., Lindsay, H. A., Mattocks, C. J., Markham, A. F., Bonthron, D. T., et al. (2010). Genetic diagnosis of familial breast cancer using clonal sequencing. *Human mutation*, 31(4):484–491.
- Morieri, M. L., Gao, H., Pigeyre, M., Shah, H. S., Sjaarda, J., Mendonca, C., Hastings, T., Buranasupkajorn, P., Motsinger-Reif, A. A., Rotroff, D. M., et al. (2018). Genetic tools for coronary risk assessment in type 2 diabetes: a cohort study from the accord clinical trial. *Diabetes care*, 41(11):2404–2413.
- Nielsen, D. (2016). Tree boosting with xgboost. *NTNU Norwegian University of Science and Technology*.
- O’Leary, N. A., Wright, M. W., Brister, J. R., Ciufu, S., Haddad, D., McVeigh, R., Rajput, B., Robbertse, B., Smith-White, B., Ako-Adjei, D., et al. (2015). Reference sequence (refseq) database at ncbi: current status, taxonomic expansion, and functional annotation. *Nucleic acids research*, 44(D1):D733–D745.

- Olson, R. S., La Cava, W., Mustahsan, Z., Varik, A., and Moore, J. H. (2017). Data-driven advice for applying machine learning to bioinformatics problems. *arXiv preprint arXiv:1708.05070*.
- Opgen-Rhein, R. and Strimmer, K. (2007). Accurate ranking of differentially expressed genes by a distribution-free shrinkage approach. *Statistical applications in genetics and molecular biology*, 6(1).
- Patel, R. K. and Jain, M. (2012). Ngs qc toolkit: a toolkit for quality control of next generation sequencing data. *PloS one*, 7(2):e30619.
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830.
- Perneger, T. V. (1998). What's wrong with bonferroni adjustments. *Bmj*, 316(7139):1236–1238.
- Philibert, R. A., Beach, S. R., and Brody, G. H. (2012). Demethylation of the aryl hydrocarbon receptor repressor as a biomarker for nascent smokers. *Epigenetics*, 7(11):1331–1338.
- Pickrell, J. K., Marioni, J. C., Pai, A. A., Degner, J. F., Engelhardt, B. E., Nkadori, E., Veyrieras, J.-B., Stephens, M., Gilad, Y., and Pritchard, J. K. (2010). Understanding mechanisms underlying human gene expression variation with rna sequencing. *Nature*, 464(7289):768.
- Price, A. L., Patterson, N. J., Plenge, R. M., Weinblatt, M. E., Shadick, N. A., and Reich, D. (2006). Principal components analysis corrects for stratification in genome-wide association studies. *Nature genetics*, 38(8):904.
- Rashmi, K. V. and Gilad-Bachrach, R. (2015). Dart: Dropouts meet multiple additive regression trees. In *AISTATS*, pages 489–497.
- Rätsch, G., Onoda, T., and Müller, K.-R. (2001). Soft margins for adaboost. *Machine learning*, 42(3):287–320.
- Reinius, L. E., Acevedo, N., Joerink, M., Pershagen, G., Dahlén, S.-E., Greco, D., Söderhäll, C., Scheynius, A., and Kere, J. (2012). Differential dna methylation in purified human blood cells: implications for cell lineage and studies on disease susceptibility. *PloS one*, 7(7).
- Rotroff, D. M., Pijut, S. S., Marvel, S. W., Jack, J. R., Havener, T. M., Pujol, A., Schluter, A., Graf, G. A., Ginsberg, H. N., Shah, H. S., et al. (2018a). Genetic variants in *hsd17b3*, *smad3*, and *ipo11* impact circulating lipids in response to fenofibrate in individuals with type 2 diabetes. *Clinical Pharmacology & Therapeutics*, 103(4):712–721.
- Rotroff, D. M., Yee, S. W., Zhou, K., Marvel, S. W., Shah, H. S., Jack, J. R., Havener, T. M., Hedderson, M. M., Kubo, M., Herman, M. A., et al. (2018b). Genetic variants in *cpa6* and *prpf31* are associated with variation in response to metformin in individuals with type 2 diabetes. *Diabetes*, 67(7):1428–1440.

- Saabas, A. (2014). Interpreting random forests. *Diving into data*, 24.
- Schiltz, F., Masci, C., Agasisti, T., and Horn, D. (2018). Using regression tree ensembles to model interaction effects: a graphical approach. *Applied Economics*, 50(58):6341–6354.
- Schmidt, T., Hummel, S., and Herrmann, B. (1995). Evidence of contamination in pcr laboratory disposables. *Naturwissenschaften*, 82(9):423–431.
- Schmieder, R. and Edwards, R. (2011). Fast identification and removal of sequence contamination from genomic and metagenomic datasets. *PloS one*, 6(3):e17288.
- Sehn, J. K., Spencer, D. H., Pfeifer, J. D., Bredemeyer, A. J., Cottrell, C. E., Abel, H. J., and Duncavage, E. J. (2015). Occult specimen contamination in routine clinical next-generation sequencing testing. *American journal of clinical pathology*, 144(4):667–674.
- Shah, H. S., Gao, H., Morieri, M. L., Skupien, J., Marvel, S., Paré, G., Mannino, G. C., Buranasupkajorn, P., Mendonca, C., Hastings, T., et al. (2016). Genetic predictors of cardiovascular mortality during intensive glycemic control in type 2 diabetes: findings from the accord clinical trial. *Diabetes Care*, 39(11):1915–1924.
- Shah, H. S., Morieri, M. L., Marcovina, S. M., Sigal, R. J., Gerstein, H. C., Wagner, M. J., Motesinger-Reif, A. A., Buse, J. B., Kraft, P., Mychaleckyj, J. C., et al. (2018). Modulation of glp-1 levels by a genetic variant that regulates the cardiovascular effects of intensive glycemic control in accord. *Diabetes care*, 41(2):348–355.
- Shatwan, I. M., Winther, K. H., Ellahi, B., Elwood, P., Ben-Shlomo, Y., Givens, I., Rayman, M. P., Lovegrove, J. A., and Vimalaswaran, K. S. (2018). Association of apolipoprotein e gene polymorphisms with blood lipids and their interaction with dietary factors. *Lipids in health and disease*, 17(1):98.
- Shen, A., Fu, H., He, K., and Jiang, H. (2019). False discovery rate control in cancer biomarker selection using knockoffs. *Cancers*, 11(6):744.
- Sikdar, S., Joehanes, R., Joubert, B. R., Xu, C.-J., Vives-Usano, M., Rezwani, F. I., Felix, J. F., Ward, J. M., Guan, W., Richmond, R. C., et al. (2019). Comparison of smoking-related dna methylation between newborns from prenatal exposure and adults from personal smoking. *Epigenomics*, 11(13):1487–1500.
- Simion, P., Belkhir, K., François, C., Veyssier, J., Rink, J. C., Manuel, M., Philippe, H., and Telford, M. J. (2018). A software tool 'croco' detects pervasive cross-species contamination in next generation sequencing data. *BMC biology*, 16(1):28.
- Skelly, D. A., Johansson, M., Madeoy, J., Wakefield, J., and Akey, J. M. (2011). A powerful and flexible statistical framework for testing hypotheses of allele-specific gene expression from rna-seq data. *Genome research*, 21(10):1728–1737.

- Snoek, J., Larochelle, H., and Adams, R. P. (2012). Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, pages 2951–2959.
- Strong, M. J., Xu, G., Morici, L., Bon-Durant, S. S., Baddoo, M., Lin, Z., Fewell, C., Taylor, C. M., and Flemington, E. K. (2014). Microbial contamination in next generation sequencing: implications for sequence-based analysis of clinical samples. *PLoS pathogens*, 10(11):e1004437.
- Su, X., Meneses, K., McNeese, P., and Johnson, W. O. (2011). Interaction trees: exploring the differential effects of an intervention programme for breast cancer survivors. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 60(3):457–474.
- Tang, C. S., Zhang, H., Cheung, C. Y., Xu, M., Ho, J. C., Zhou, W., Cherny, S. S., Zhang, Y., Holmen, O., Au, K.-W., et al. (2015). Exome-wide association analysis reveals novel coding sequence variants associated with lipid traits in chinese. *Nature communications*, 6:10206.
- Tang, Y., Lenzini, P. A., Busui, R. P., Ray, P. R., Campbell, H., Perkins, B. A., Callaghan, B., Wagner, M. J., Motsinger-Reif, A. A., Buse, J. B., et al. (2019). A genetic locus on chromosome 2q24 predicting peripheral neuropathy risk in type 2 diabetes: Results from the accord and bari 2d studies. *Diabetes*, page db190109.
- Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288.
- Tibshirani, R. J., Taylor, J., Lockhart, R., and Tibshirani, R. (2016). Exact post-selection inference for sequential regression procedures. *Journal of the American Statistical Association*, 111(514):600–620.
- Trompet, S., de Craen, A. J., Postmus, I., Ford, I., Sattar, N., Caslake, M., Stott, D. J., Buckley, B. M., Sacks, F., Devlin, J. J., et al. (2011). Replication of ldl gwas hits in prosper/phase as validation for future (pharmaco) genetic analyses. *BMC medical genetics*, 12(1):131.
- Tsubokawa, F., Nishisaka, T., Takeshima, Y., and Inai, K. (2002). Heterogeneity of expression of cytokeratin subtypes in squamous cell carcinoma of the lung: with special reference to ck14 overexpression in cancer of high-proliferative and lymphogenous metastatic potential. *Pathology international*, 52(4):286–293.
- Turley, S. J., Cremasco, V., and Astarita, J. L. (2015). Immunological hallmarks of stromal cells in the tumour microenvironment. *Nature reviews immunology*, 15(11):669.
- van de Geer, S., Bühlmann, P., Zhou, S., et al. (2011). The adaptive and the thresholded lasso for potentially misspecified models (and a lower bound for the lasso). *Electronic Journal of Statistics*, 5:688–749.

- Van Dijk, E. L., Auger, H., Jaszczyszyn, Y., and Thermes, C. (2014). Ten years of next-generation sequencing technology. *Trends in genetics*, 30(9):418–426.
- Waldmann, P., Mészáros, G., Gredler, B., Fuerst, C., and Sölkner, J. (2013). Evaluation of the lasso and the elastic net in genome-wide association studies. *Frontiers in genetics*, 4:270.
- Wasserman, L. and Roeder, K. (2009). High dimensional variable selection. *Annals of statistics*, 37(5A):2178.
- Westfall, P. H., Young, S. S., et al. (1993). *Resampling-based multiple testing: Examples and methods for p-value adjustment*, volume 279. John Wiley & Sons.
- Wilson, D. J. (2019). The harmonic mean p-value for combining dependent tests. *Proceedings of the National Academy of Sciences*, 116(4):1195–1200.
- Yoshihara, K., Shahmoradgoli, M., Martínez, E., Vegesna, R., Kim, H., Torres-Garcia, W., Treviño, V., Shen, H., Laird, P. W., Levine, D. A., et al. (2013). Inferring tumour purity and stromal and immune cell admixture from expression data. *Nature communications*, 4:2612.
- Yu, J., Pressoir, G., Briggs, W. H., Bi, I. V., Yamasaki, M., Doebley, J. F., McMullen, M. D., Gaut, B. S., Nielsen, D. M., Holland, J. B., et al. (2006). A unified mixed-model method for association mapping that accounts for multiple levels of relatedness. *Nature genetics*, 38(2):203.
- Yuan, M. and Lin, Y. (2006). Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67.
- Zhang, C.-H. et al. (2010a). Nearly unbiased variable selection under minimax concave penalty. *The Annals of statistics*, 38(2):894–942.
- Zhang, S., Wang, F., Wang, H., Zhang, F., Xu, B., Li, X., and Wang, Y. (2014). Genome-wide identification of allele-specific effects on gene expression for single and multiple individuals. *Gene*, 533(1):366–373.
- Zhang, T., Yu, B., et al. (2005). Boosting with early stopping: Convergence and consistency. *The Annals of Statistics*, 33(4):1538–1579.
- Zhang, Z., Ersoz, E., Lai, C.-Q., Todhunter, R. J., Tiwari, H. K., Gore, M. A., Bradbury, P. J., Yu, J., Arnett, D. K., Ordovas, J. M., et al. (2010b). Mixed linear model approach adapted for genome-wide association studies. *Nature genetics*, 42(4):355.
- Zhao, P. and Yu, B. (2006). On model selection consistency of lasso. *Journal of Machine learning research*, 7(Nov):2541–2563.
- Zhou, X. and Stephens, M. (2012). Genome-wide efficient mixed-model analysis for association studies. *Nature genetics*, 44(7):821.

Zou, H. (2006). The adaptive lasso and its oracle properties. *Journal of the American statistical association*, 101(476):1418–1429.

Zou, H. and Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the royal statistical society: series B (statistical methodology)*, 67(2):301–320.