RANDOM SUBSPACE METHOD IN
CLASSIFICATION AND MAPPING OF FMRI DATA PATTERNS

by

Tianwen Chen

A Dissertation
Submitted to the
Graduate Faculty
of
George Mason University
in Partial Fulfillment of
The Requirements for the Degree
of
Doctor of Philosophy
Computational Sciences and Informatics

Committee:

_____	Dr. Daniel E. Houser, Dissertation Director
_____	Dr. James E. Gentle, Committee Member
_____	Dr. Igor Griva, Committee Member
_____	Dr. M. Layne Kalbfleisch, Committee Member
_____	Dr. Dimitrios A. Papaconstantopoulos, Department Chair
_____	Dr. Richard Diecchio, Associate Dean for Academic and Student Affairs, College of Science
_____	Dr. Vikas Chandhoke, Dean, College of Science

Date: 05/12/2011

Summer Semester 2011
George Mason University
Fairfax, VA

Random Subspace Method in Classification and Mapping of fMRI Data Patterns

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at George Mason University

By

Tianwen Chen
Bachelor of Arts
Shanghai Jiaotong University, 2005

Director: Dr. Daniel E. Houser, Professor
Department of Economics

Summer Semester 2011
George Mason University
Fairfax, VA

Copyright © 2011 by Tianwen Chen
All Rights Reserved

Dedication

To my son, Ethan and my wife, Liang

To my Mom and Dad

Acknowledgments

I want to acknowledge the following people who have made this dissertation possible.

Dr. Daniel E. Houser, my advisor. I have been amazingly fortunate to have an advisor who sparked my interest in fMRI data analysis, trusted me, gave me the freedom to explore on my own, and at the same time provided the excellent guidance.

Dr. James E. Gentle, committee chair, who always keeps his door open. He has been always there to listen and give advice. I am deeply grateful to him for the long discussions that helped me improve the dissertation in many different ways.

Dr. Igor Griva and Dr. M. Layne Kalbfleisch, committee members. They have generously offered suggestions and comments on earlier drafts.

Most importantly, none of this would be possible without the love and constant support of my family. They always encouraged me and stood by me through the good and bad times.

Table of Contents

List of Tables

List of Figures

Abstract

RANDOM SUBSPACE METHOD IN CLASSIFICATION AND MAPPING OF FMRI DATA PATTERNS

Tianwen Chen, PhD

George Mason University, 2011

Dissertation Director: Dr. Daniel E. Houser

The functional magnetic resonance imaging (fMRI) technique is widely used in studying human brain functions. It measures brain activities both spatially and temporally. The past decade has witnessed a growing interest in the fMRI community in constructing accurate predictive models. Though achieving high prediction accuracy is crucial in building strong diagnostic models (e.g. for brain functional disorders), information mapping or model interpretability is critical in advancing a fundamental understanding of brain functions. Recently, two notable multivariate methods, recursive feature elimination using support vector machine (RFESVM) and logistic regression with an elastic net penalty (LREN), have been applied to meet the challenge of simultaneous classification and mapping of fMRI data patterns. However, both methods have limitations. First, they suffer from the curse of dimensionality by solving classification and feature selection tasks directly in the whole feature space. Second, feature selections in both methods critically depend on sampled values of tuning parameters and they lack of a control over false selections.

In this dissertation, I seek to address both limitations within a random subspace framework. The random subspace method is an effective approach to lessen the curse of dimensionality and explore data patterns from different local perspectives. It has been separately

applied in classifications and feature selections with success. But no previous studies have attempted to integrate them together, possibly because of the high computational cost involved in using a double-loop cross validation scheme to avoid the parameter selection bias. In chapter 2, I seek to solve the methodological issues. I first extend an efficient method, only using a single K-fold cross validation procedure, to alleviate the parameter selection bias of an ensemble classifier formed by the random subspace method. The extension allows independent tuning of base classifiers, making feature selection more adaptive to the local data structure. I then integrate a random probe method into the random subspace framework to control false selections. A threshold is derived based on the distribution of scores of permuted artificial variables.

In chapter 3 and 4, using extensive simulations, I empirically evaluate the developed random subspace framework with applications to LREN and RFESVM, respectively. I find that (1) the developed random subspace framework can boost performance of both LREN and RFESVM in classifications and feature selections; (2) the random probe method can effectively control false selection rates; (3) the proposed novel feature scoring method is capable of ranking informative features based on their individual discriminative capacities; (4) the random subspace framework is able to correctly determine informative features' discrimination directions.

Chapter 1: Introduction

The functional magnetic resonance imaging (fMRI) technique is widely used in studying human brain functions. It measures brain activities both spatially and temporally. Statistical methods analyzing fMRI data have long been focusing on understanding functions of certain brain regions under associated mental tasks. These methods include correlation analysis (Bandettini et al., 1993), general linear model (GLM) (Friston et al., 1995), region of interest (ROI) analysis, etc. However, Haxby et al. (2001), in their seminal paper, successfully demonstrated the possibility of building accurate predictive models from sub-maximally responding voxels discriminating experimental tasks. The past decade has witnessed a growing interest in the fMRI community in constructing accurate predictive models. Though achieving high prediction accuracy is crucial in building strong diagnostic models (e.g. for brain functional disorders), information mapping or model interpretability is critical in advancing a fundamental understanding of brain functions. Information mapping in fMRI data aims to select voxels or features with high discriminative ability or diagnostic power. Feature selection methods can be univariate or multivariate. Univariate methods examine features' discriminative capacity separately and are suboptimal for fMRI data where high correlations between discriminative voxels are prevalent. Multivariate methods, however, taking voxel dependency into account are well suited for analyzing fMRI data.

Recently, two notable multivariate methods, recursive feature elimination using support vector machine (RFESVM, Guyon et al., 2002; De Martino et al., 2008) and logistic regression with an elastic net penalty (LREN, Zou and Hastie, 2005; Ryali et al., 2010), have been applied to meet the challenge of simultaneous classification and mapping of fMRI data patterns. Both methods have data dependent tuning parameters that govern classification and feature selection. De Martino et al. (2008) compared RFESVM with several univariate

feature selection methods (e.g. t test, Wilcoxon rank test, GLM) on both simulated and real datasets. On simulated datasets, they showed that at higher contrast-to-noise ratios, RFESVM was superior both for classification accuracy and feature selection in terms of the area under the ROC curve. On real datasets, they reported that recursive feature elimination (RFE) improved classification accuracy especially after an initial dimension reduction in the feature space. The improvement using RFE was similarly observed in several other studies (Hanson and Halchenko, 2008; Staeren et al., 2009). Using extensive simulations, Ryali et al. (2010) reported that LREN tended to be superior to RFESVM in feature selection, especially at low prevalence rates. However, a fair comparison between RFESVM and LREN is difficult because each method has unique tuning parameters. The elastic net penalty has also been proposed for several other classifiers and consistently shown to be superior. Grosenick et al. (2008) added an elastic net penalty to a linear discriminant classifier and evaluated it on several brain regions of interest. They found that its classification accuracy significantly outperformed logistic regression, linear discriminant analysis and linear SVM. Similarly, Carroll et al. (2009) showed that linear regression with an elastic net penalty, when optimized, was more accurate in prediction than ordinary least squares with a fixed number of selected voxels, regardless of initial feature selection approaches.

However, both methods (LREN and RFESVM) have limitations. First, they suffer from the curse of dimensionality by solving classification and feature selection tasks directly in the whole feature space. This negative effect was observed in De Martino et. al. (2008), in which they showed that both classification accuracy and power in feature selection of RFESVM were greatly improved after an initial dimension reduction in the feature space. Thus, classification accuracy of both methods may degrade for problems with an even higher dimensionality, for example, classifying spatiotemporal patterns (Mourao-Miranda et al., 2007). Second, to map informative patterns in LREN and RFESVM, a feature subset that achieves the highest cross validation accuracy is selected. However, the selection quality critically depends on tuning parameters. Classifiers are usually optimized on values coarsely sampled in the tuning parameter space. One drawback is that the selected subset

may include too many irrelevant features or too few relevant features. The specificity of information mapping is unknown and out of the researcher's control.

The random subspace method (RSM) is a promising framework to overcome the limitations. RSM was first proposed as a multiple classifier system (an ensemble classifier) that integrates decisions from its base classifiers trained in much lower dimensional subspaces, each constructed with randomly sampled features (Ho, 1998). Since the sample-to-feature ratio in each subspace is greatly improved, RSM is able to alleviate the curse of dimensionality. RSM is flexible enough to be suitable for many base classifiers, and has been demonstrated to have higher classification accuracy than its base classifier (Ho, 1995, 1998; Skurichina and Duin, 2001, 2002). RSM is also competitive with other benchmark methods as well (linear SVM, Random Forest, Adaboost, etc., Kuncheva et al., 2010). Recently, RSM have also been used to construct powerful multivariate feature selection methods (Lai et al., 2006; Cai et al., 2007; Sona and Avesani, 2010). The success of RSM in feature selection mainly originates from two sources: (1) higher sample-to-feature ratios in subspaces boost selection accuracy; (2) weakly informative features have better chances to be dominant in some subspaces and be distinguished from irrelevant features, while in a direct whole feature space learning, they are more likely to be treated and discarded as irrelevant features, jeopardizing sensitivity in feature selection.

However, to my best knowledge, no previous studies have attempted to use RSM as an ensemble classifier and at the same time, use it to select relevant features. Two important issues need to be solved for its successful applications to LREN and RFESVM for simultaneous data classification and feature selection. First, using base classifiers with tuning parameters, when not handled properly, RSM is easily prone to the parameter selection bias. A general approach to avoid the bias is to use a double-loop cross validation procedure, in which a classifier is tuned in the inner loop and its classification accuracy is evaluated in the outer loop. But this scheme is computationally infeasible for fMRI data which typically has a very high feature dimension and limited sample size. Tibshirani and Tibshirani (2009) proposed an efficient bootstrap-like bias correction method. It allows both estimation of

true prediction accuracy and optimal feature selection to be completed within one cross validation process. A similarly efficient approach is desirable for RSM to allow separate tuning of base classifiers without involving severe selection bias. Second, for an accurate and effective mapping of informative patterns, a proper threshold needs to be set on the feature ranking list from RSM to control the rate of false selections. I first seek to address both issues and develop a novel RSM framework in chapter 2, and then in chapter 3 and 4, I apply the developed RSM framework to LREN and RFESVM, respectively. Using extensive simulations, I empirically examine whether RSM is able to boost performance of LREN and RFESVM in classification as well as feature selection.

In order to give a clear picture on the relationship of the developed RSM framework with other methods, I provide a general survey of analysis methods for fMRI/VBM data. Figure 1.1 shows three categories of analysis methods for fMRI and VBM data. The first category is the "Hypothesis Driven Methods", which carry out voxel-wise tests looking for activation information, for example, the widely used general linear model. The second category is the "Data Driven Methods". These methods are mainly Principle Component Analysis methods (PCA) and Independent Component Analysis methods (ICA). They can either extract interesting patterns or serve as dimension reduction methods. For the dimension reduction purpose, PCA and ICA are used to preprocess data and feed the dimension-reduced data into the third category that is "Multivariate Pattern Classification Methods". In pattern classification, various classifiers are used for different purposes. For a diagnostic model, classification is more likely to be the only goal such that there is no need to infer activation patterns underlying accurate predictions/classifications. Most of the time, classifiers are used for both classification and information mapping. The RSM framework I develop in the dissertation lies in the multivariate pattern classification category and strives to predict as well as detect activation patterns. RSM can either work on the whole brain data or on the lower-dimensional data from PCA or ICA.

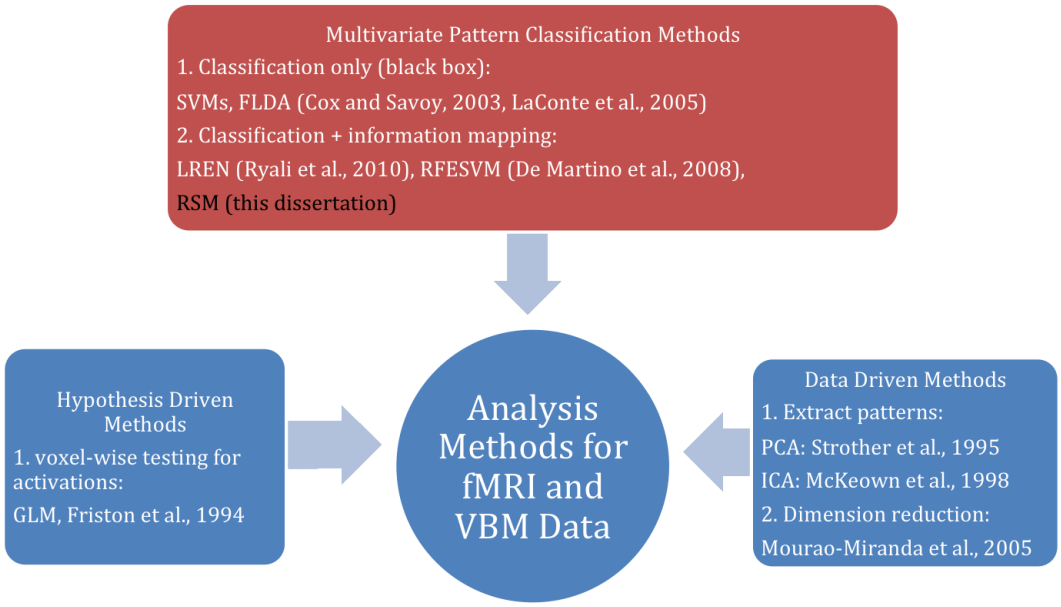

Figure 1.1: A general survey of fMRI/VBM analysis methods

Bias Correction and Feature Selection in Random Subspace Method

In this chapter, I develop an efficient and highly parallel random subspace framework to improve classification accuracy and assist feature selection. Specifically, I first extend the bias correction method in Tibshirani and Tibshirani (2009) to the ensemble classifier constructed by the random subspace method, which allows free parameter tuning in each subspace. The extended method is efficient and does not add much additional computation load. Second, I design a novel feature scoring method that is suitable for the two base classifiers to be examined (LREN and RFESVM). Third, I integrate a random probe method into RSM to help control false selections. A threshold is determined based on the empirical cumulative distribution of scores of artificial features. Forth, I propose an intuitive approach to determine informative features' discrimination directions. To empirically evaluate the developed RSM framework, I design three simulation studies with datasets whose characteristics are typically observed in real fMRI data. For a full evaluation, I vary informative patterns with respect to signal-to-noise ratios (SNR), prevalence rates (PR)

and discrimination directions of relevant features. Using the basic simulation settings in Ryali et al. (2010), the first simulation study aims to provide a fundamental understanding of LREN, RFESVM and their RSM versions. The second study evaluates whether the proposed feature scoring method is effective in ranking informative features. The final study examines whether the developed RSM framework is able to determine informative features' discrimination directions. Besides, to accurately compare generalization errors, I also simulate a large independent test dataset for each study.

Application of Random Subspace Method to Logistic Regression with an Elastic Net Penalty

In this chapter, using simulated datasets from chapter 2, I examine both the benchmark method (LREN) and its RSM version (RSMLREN). For evaluations on LREN, I first examine whether the bias correction method in Tibshirani and Tibshirani (2009) is effective. Second, I examine finite sample properties of LREN in feature selection with regards to false positive rates, selection sensitivity and similarity rates. For RSMLREN, I first evaluate the proposed error bias correction method for ensemble classifiers, which allows free parameter tuning on each subspace. Second, using test data, I examine whether RSMLREN has better classification accuracy than LREN. Third, I match false positive rates of RSMLREN to LREN on each dataset and evaluate whether RSMLREN performs better in feature selection sensitivity and similarity. Fourth, I empirically examine the preciseness of the proposed thresholding method evaluated on three target levels of false positive rates. I also discuss the effects of the subspace size and the number of subspaces on performance of classification, feature selection and false positive rate control. Finally, I examine whether the developed RSM can effectively rank informative features and determine their discrimination directions.

I obtain several interesting empirical results:

1. For both LREN and RSMLREN, higher SNR leads to smaller classification errors. Increasing PR initially decreases errors but finally increases errors. Without the bias

correction, error estimates are strongly under-biased. Bias of RSMLREN is more severe than LREN. Bias becomes smaller under higher SNRs. Both bias correction methods help shrink the bias.

2. RSMLREN has slightly lower test errors than LREN. As the subspace size increases, test errors of RSMLREN increases, approaching errors of LREN. Both minimal error curves and test error curves stabilize fast. Adjusted error curves are more variant, but the variance seems to be stable.

3. In feature selection, increasing SNR help decrease the false positive rate and increase both selection sensitivity and stability. Increasing PR consistently decreases sensitivity. RSMLREN has higher selection sensitivity rates than LREN in all scenarios and higher stability in most cases. Most drastic improvements occur at high PRs (PR = 10% and 20%).

4. Increasing the subspace size deteriorates feature selection performance but RSMLREN with three subspace sizes are all better than LREN. Higher SNR helps stabilize selection sensitivity as more iterations are used. Sensitivity rate curves converge very fast under median and high SNRs and may require more iterations under a low SNR.

5. In most cases, false positive rates in RSMLREN are well controlled. Empirical false positive rate curves are slightly stabler with more iterations.

6. RSMLREN is an effective feature ranking method and is able to correctly determine informative features' discrimination directions.

Application of Random Subspace Method to Recursive Feature Elimination using Support Vector Machine

In this chapter, using the datasets in chapter 2, I examine RFESVM and RSMRFESVM in a similar fashion as in Chapter 3. I first investigate whether the bias correction methods are effective for RFESVM and RSMRFESVM. Second, I examine effects of SNR and PR on finite sample properties of RFESVM in feature selection with criteria of false positive

rates, selection sensitivity and similarity rates. To compare RFESVM with RSMRFESVM, I use the same strategy in chapter 3 by first matching false positive rates. Third, since false positive rates are high at optimal elimination levels, I further investigate whether RSMRFESVM is still more effective in retrieving informative features at lower false positive rates. Fourth, I examine the proposed false positive rate controlling method for RSMRFESVM. I also discuss effects of both the subspace size and the number of subspaces on classification accuracy, feature selection and control of false positive rates. Moreover, I examine the feature ranking performance of RSMRFESVM as well as its ability in discerning discrimination directions of informative features. I also try to gain some insights into the different performance of RSMLREN and RSMRFESVM.

Several results are interesting:

1. For both RFESVM and RSMRFESVM, higher SNR leads to smaller classification errors. Increasing PR initially decreases errors but finally increases errors. The minimal cross validation errors of RFESVM have very small bias but the bias is very sensitive to tuning parameter values. In RSMRFESVM, the bias is much larger and the bias correction method is effective.

2. RSMRFESVM has slightly lower test errors than RFESVM for all cases except for PR = 1%. As the subspace size increases, test errors of RSMRFESVM increases, approaching errors of RFESVM. Both minimal error curves and test error curves converge fast as the number of subspaces increases, and the adjusted error curves have more variance.

3. In feature selection, increasing SNR decreases false positive rates and increases both sensitivity and stability rates. Increasing PR consistently decreases selection sensitivity. When matched to false positive rates of RFESVM, RSMRFESVM consistently has higher selection sensitivity and similarity rates in most cases. Most drastic improvements occur at PR = 10% and 20%.

4. When controlled at lower false positive rates, RSMRFESVM with the smallest subspace size is consistently the best at retrieving relevant features. RSMRFESVM with two larger subspace sizes are also better than RFESVM except for PR = 1% and 5% at median and high SNRs.

5. More iterations help stabilize selection sensitivity. RSMRFESVM with a large subspace size needs more iterations to be competitive with RFESVM.

6. In most cases, controlling false positive rates is successful. At high PRs (PR = 10% and 20%), empirical false positive rates tend to under-estimate target rates.

7. RSMRFESVM is also an effective feature ranking method and is able to correctly determine informative features' discrimination directions.

Contributions of the Dissertation

This dissertation makes several contributions to the literature on random subspace methods as well as on classification and mapping of fMRI data patterns.

- Methodologically, I develop an efficient RSM framework that enjoys both advantages of higher accuracy as an ensemble classifier and higher retrieval power as a multivariate feature selection method. Specifically,
  - I develop a bias correction method to alleviate the selection bias in ensemble classifiers formed by RSM using tunable base classifiers. It allows an efficient error estimation using only one loop of cross validation instead of a commonly used double-loop procedure, which is computationally infeasible for fMRI data.
  - I design a novel feature scoring method that can effectively rank features based on their individual discriminative capacities.
  - I develop a random probe method to determine an empirical threshold on the feature ranking list. The threshold is data dependent and is able to control false selections.

  - I design an intuitive approach to determine selected features' discrimination directions, which has not been addressed in the RSM literature but is important in discerning functions of different brain regions.

- Empirically, using extensive simulations resembling real fMRI data,

  - I fully examine finite sample properties of both LREN and RFESVM. I find that neither LREN or RFESVM is flexible enough for a whole brain as well as region of interest analysis. This calls for proper use of both methods in mapping fMRI data patterns.

  - I find that the developed RSM framework is more accurate in classification and more sensitive to informative patterns, indicating its usefulness in applications to real fMRI data.

  - I evaluate the effectiveness of the thresholding method in controlling false selections and find it quite adaptive to the underlying data patterns, which is more objective and does not require expert knowledge.

Organization of the Remainder of the Dissertation

The remainder of the dissertation is organized as follows. Chapter 2 presents the developed random subspace framework and is titled "*Bias Correction and Feature Selection in Random Subspace Method*". Chapter 3 and 4 report two empirical evaluations of the developed framework and are titled "*Application of Random Subspace Method to Logistic Regression with an Elastic Net Penalty*" and "*Application of Random Subspace Method to Recursive Feature Elimination using Support Vector Machine*". Chapter 5 concludes and discusses directions for future research.

Chapter 2: Bias Correction and Feature Selection in Random Subspace Method

2.1 Introduction

The random subspace method (RSM) was originally designed as a multiple classifier system to improve classification performance (Ho, 1995, 1998). In RSM, a lower dimensional subspace is generated by randomly sampling a subset of features from the original feature space, and samples are projected onto the subspaces. Base classifiers in subspaces are independently trained. RSM has several notable advantages compared to other ensemble classifiers. First, in each subspace, the sample-to-feature ratio is greatly improved and the overfitting problem is less severe. Second, RSM avoids the problem of class under-representation which may happen in other ensemble methods (e.g. bagging). Third, RSM also improves the diversity among base classifiers through the random sampling scheme in the feature space. The successful Random Forest algorithm (Breiman, 2001) utilizes a similar concept of RSM in node-splitting, in which the best node-splitting is searched among a random set of candidate features. RSM has been applied to many base classifiers and consistently been reported to have superior classification performance (decision trees, Ho, 1995, 1998; linear discriminant analysis, Skurichina et al., 2001, 2002; linear SVM and decision trees, Kuncheva et al., 2010). In addition to its success in classification, RSM is also highly capable in feature selection by providing a more accurate ranking list (Lai et al., 2006; Cai et al., 2007; Sona and Avesani, 2010). For a multivariate feature selection or ranking method focusing on the whole data pattern, the high feature-to-sample ratio severely degrades its selection accuracy because of the large data sampling variance in a high dimensional feature space. Weakly informative features are more likely to be treated and discarded as irrelevant features, jeopardizing sensitivity in feature selection. In contrast, RSM is able to alleviate

the curse of dimensionality. Feature selection or ranking in each subspace is more accurate and reliable because of the relatively higher sample-to-feature ratio. Weakly informative features have better chances to be dominant in some subspaces and be distinguished from irrelevant features. Additionally, integrating selections or rankings from all subspaces, RSM can effectively capture feature dependency.

However, current random subspace methods have several limitations. First, to avoid the parameter selection bias, a double-loop cross validation procedure is commonly used for RSM with tunable base classifiers (Lai et al., 2006). All training processes, including parameter tuning, occur only in the inner loop and estimation of the misclassification error is performed in the outer loop. But it becomes too computationally expensive and infeasible for the high dimensional fMRI data. Moreover, fMRI data typically has a limited sample size (in the order of tens). The training quality is severely impacted because even fewer samples are available in the inner loop. Second, current RSM frameworks in feature selection severely lack of an appropriate control over false selections. Features are selected either by a double-loop cross validation or by a researcher setting a threshold on the feature ranking list. The first approach is unable to control false selections. On one hand, if a union of the optimal selections from all inner loops is used, too many irrelevant features may be included and the mapping specificity is jeopardized. On the other hand, if an intersection of all optimal selections is used, too many relevant features may be excluded and the mapping power is lost. The second approach is very subjective, especially when there is no prior knowledge available. Ideally, the threshold should be data dependent. More top ranked features are to be selected if relevant features are abundant and less to be selected if relevant features are sparse. Thus, an objective threshold needs to be set on the feature ranking list such that irrelevant features can be well separated from relevant ones. Third, current studies of RSM focus on either classification or feature selection. No attempts have been made to use RSM as an ensemble classifier and at the same time, use it to select relevant features. Since RSM has been shown to have superior performance in both aspects, it is beneficial to integrate them into a unified framework.

In this study, I seek to fill in these gaps and develop a unified and efficient RSM framework for simultaneous classification and mapping of fRMI data patterns. Specifically, I first extend the efficient bias correction method in Tibshirani and Tibshirani (2009) to the case of an ensemble classifier formed by RSM. The extension allows base classifiers to be independently optimized in their subspaces, and at the same time it estimates the associated selection bias in the majority voting scheme using decisions made at the minimal CV error in each subspace. The framework is efficient because (1) it only requires a single loop cross validation instead of a double loop and (2) trainings in subspaces are independent and thus the framework can be highly parallelized. Second, I design a novel scoring method to rank features, which is especially suitable for base classifiers with an inherent feature selection ability (e.g. LREN and RFESVM). Third, using a similar strategy of adding artificial variables (Tuv et al. (2009)), I develop an objective method to control false selections by setting a threshold on the feature ranking list. The threshold is empirically determined based on the distribution of scores of permuted artificial variables. Finally, I propose an intuitive approach to determine the discrimination directions of selected features.

The remainder of this chapter is organized as follows: in section 2.2, I review issues in generating random subspaces as well as studies of RSM in classification and feature selection; in section 2.3, I develop the unified RSM framework; in section 2.4, I describe three simulation studies to evaluate the RSM framework applied to two base classifiers (LREN and RFESVM); in section 2.5, I discuss and conclude the study.

2.2 Review of the Random Subspace Method

RSM is an ensemble classifier consisting of multiple members or base classifiers. Usually, the same type of classifier is trained independently on each subspace. In the next three subsections, I sequentially review current literature on generation of random subspaces, RSM used in classification and RSM used in feature selection.

2.2.1 Generation of Random Subspaces

Suppose the data structure is (y, X), $y = [y_1, ..., y_i, ..., y_N]^T$ is a vector of class labels, in which

$$y_i = \begin{cases} 0 & \text{if } i^{th} \text{ sample belongs to class 1} \\ 1 & \text{if } i^{th} \text{ sample belongs to class 2} \end{cases},$$

$X = [x_1^T, ..., x_i^T, ..., x_N^T]^T$, where $x_i \in \mathbb{R}^p$ is the i^{th} sample, p is the number of features and N is the number of samples. In RSM, a subspace is generated by randomly sampling, without replacement, $l(\ll p)$ out of p features and the original data is projected onto the subspace. Let $(y, \tilde{X}^{(m)})$ be the projected data structure on the m^{th} subspace, where $\tilde{X}^{(m)} = [(\tilde{x}_1^{(m)})^T, ..., (\tilde{x}_i^{(m)})^T, ..., (\tilde{x}_N^{(m)})^T]^T$ and $\tilde{x}_i^{(m)} \in \mathbb{R}^l$. By construction, each subspace contains the same number of samples and but much fewer features.

The subspace generation is governed by two parameters: ensemble size (number of subspaces) and cardinality of a feature subset, assuming all subspaces have the same cardinality. Kuncheva et al. (2010) discussed the optimal selection of the two parameters with respect to three criteria: (1) usability of an ensemble classifier, defined as the probability that all base classifiers are usable. A usable classifier is the one that its feature space contains at least one relevant feature; (2) coverage of an ensemble classifier, defined as the probability that the whole feature space is covered by all subspaces; (3) diversity of an ensemble classifier, defined as the probability that usable classifiers is nonidentical in the sense that two feature subsets differ by at least one relevant feature. However, as shown in Kuncheva et al. (2010), no suitable pair can reconcile all three criteria. For example, small ensemble size and large cardinality improve criterion 1; large ensemble size and large cardinality are better for criterion 2; small ensemble size and large cardinality are favored by criterion 3. In their study, they chose small ensemble size and large cardinality, favoring usability and diversity. An alternative would assign a weight to each criteria as a combined optimization problem, though it requires expert knowledge and is data dependent.

2.2.2 Random Subspace Method In Classification

RSM is a framework to construct ensemble classifiers. Ensemble classifiers are claimed to benefit from "the wisdom of crowds". Hansen and Salamon (1990) stated that if ensemble members are accurate and diverse, then the ensemble classifier is more accurate than any ensemble member. A classifier is accurate if it has an error rate of better than a random guessing, on new unseen data points. Two classifiers are diverse if they make different errors on new data points. A simple illustration can be made through the binomial theorem. Suppose each ensemble member has the same error rate p on new data points, then the prediction error of the ensemble of M classifiers is

$$\mathrm{Err}_{\mathrm{ensemble}} = \sum_{k>\lfloor \frac{M}{2} \rfloor}^{M} \binom{M}{k} p^k (1-p)^{M-k},$$

where $\lfloor \cdot \rfloor$ rounds down a number to the largest integer that exceeds it. If $p < 0.5$, then $\mathrm{Err}_{\mathrm{ensemble}} \to 0$ when $M \to \infty$. However, classifiers being accurate and diverse is too strong to be true in real applications but it does shed some light on the success of ensemble classifiers. RSM shares some similarity with Random Forest algorithm proposed by Breiman (2001). He presented an upper bound on Random Forest's generalization error:

$$\mathrm{Err}_{\mathrm{RF}} \leq \bar{\rho} \frac{(1-s^2)}{s^2},$$

where ρ is the average pairwise correlation between predictions of base classifiers and s^2 is the average strength of base classifiers. Though the bound is loose in a general sense, it becomes tight when the average correlation between ensemble members decreases or the complexity of the ensemble classifier increases or both.

RSM has the advantage of improving both strength and diversity among classifiers. RSM lessens the impact from the curse of dimensionality by training a base classifier in a much lower dimensional subspace. Thus base classifiers tend to be more accurate than

being trained in the whole original feature space. In fact, RSM has the ability to shift the generalization error curve (a function of sample size) in the direction of the generalization error obtained with larger sample size (Skurichina et al., 2001, 2002). Another advantage of RSM is that the random sampling of features helps to increase diversity among base classifiers.

RSM have been applied to numerous base classifiers and shown to be superior than many well established classifiers. Ho (1995, 1998) originally proposed RSM and combined it with decision trees. Decision trees were grown separately in subspaces. The classification decision was made by a simple majority voting from all trees. She showed that RSM with decision trees outperformed decision trees constructed using bootstrapping and boosting. Skurichina and Duin (2001, 2002) further studied RSM with linear discriminant analysis. They showed that RSM was particularly useful for weak classifiers with a decreasing generalization error curve. For example, RSM improved FLD (Fisher linear discriminant function) while the bagging method was less effective. Kuncheva et al. (2010) was the first to apply RSM in classification of fMRI data. They combined RSM with decision trees and linear support vector machine and compared them with other 13 benchmark classifiers (e.g. linear support vector machine and its AdaBoost and Bagging versions, decision tree and its AdaBoost and Bagging versions, linear discriminant analysis, naive bayes, etc.). All methods were tested on three real fMRI datasets with different voxel selection methods. They found that RSM with linear SVM performed the best but RSM with decision trees was on average. In short, RSM is a powerful framework to form ensemble classifiers and is flexible enough for many well-known classifiers.

2.2.3 Random Subspace Method in Feature Selection

Guyon (2003) provided a nice introductory review over feature selection methods. Univariate selection methods (filters) consider features independently and do not take feature dependency into account. Filters do not incorporate learning and such methods include but not limited to, two sample t-test, correlation between each feature and the target variable,

etc. Filters are fast and usually are employed in the data preprocessing step to reduce the feature dimension, with a quite loose threshold on feature scores. This strategy has been used in a number of fMRI studies (Mitchell et al., 2004; Haynes and Rees, 2005; Mourao-Miranda et al., 2006). In contrast, multivariate selection methods consider the whole pattern instead of single feature. Therefore, multivariate methods are more powerful for fMRI data where neighboring voxels are usually highly correlated. Multivariate selection methods generally fall into two categories: wrappers and embedded methods (Guyon, 2003). In wrappers, features as a set are selected to maximize the generalization performance of a classifier. For example, the recursive feature elimination using support vector machines (RFESVM, Guyon et al., 2002; De Martino et al., 2008) recursively discard features with lowest rankings and select the subset of features that has a highest generalization accuracy (usually cross validation accuracy). Another category is called embedded methods, where feature selection and classification cannot separate. For example, in linear regression and logistic regression with an elastic net penalty (Zou and Hastie, 2005; Ryali et al., 2010), penalty parameters govern both feature selection and generalization performance.

Lai et al. (2006) first proposed the idea of using RSM as a multivariate feature selection method. They argued that in each lower dimensional feature space, a classifier can better handle the problem of dimensionality and thus can better retrieve informative features because of the increased sample-to-feature ratio. In their study, they combined RSM with two base classifiers: Liknon and RFESVM. Liknon is a linear SVM with an l_1 norm penalty on model parameters in the object function and thus can automatically select features by shrinking coefficients of some irrelevant features to exactly zeros. They fixed the cost parameter in Liknon and trained it in each subspace. Features were ranked by their scores, each of which was computed as an average of weights in subspaces that include the feature. For a base classifier of RFESVM, weights were replaced with the orders that a feature get discarded in subspaces. Specifically, in RFESVM, they eliminated features one by one. In order to select optimal features and avoid selection bias, they used a double-loop cross validation procedure (a 10-fold cross validation in both loops). The inner cross validation

loop was used for tuning parameters in RSM and selecting optimal features and the outer loop cross validation was used to assess the generalization error of the selected features from the inner loop. Note that in their implementation, the RFESVM was only used to compute feature scores in the inner loop while a Fisher classifier was employed to select an optimal list and assess generalization errors. Additionally, optimal features selected in all inner folds (100 folds) are merged in a final subset, a union of all features present. They found that RSM was able to create a better ranking list than RFE, Liknon and the forward sequential selection method, in terms of the power in retrieving informative features.

Cai et al. (2007) investigated the feature selection performance of RSM combined with linear SVM. In their implementation, tuning parameters in RSM and linear SVM were fixed and did not require an optimization. The score of each feature was computed as the averaged squared weights from the trained linear SVMs weighted by their prediction accuracies. They did not attempt to select optimal features. Instead, they evaluated the feature raking quality of their method with comparisons to RFESVM and least squares bound sequential forward selection (LS-Bound SFS) method. They reported on a real gene dataset that top-ranked genes selected from their method was able to achieve the lowest generalization errors than those selected from the other two methods.

In fMRI applications, Sona et al. (2010) proposed to combine RSM with ridge regression and compared it to linear regression with an elastic net penalty trained using the whole feature space. For a fair comparison, they used the same parameter value of the l_2 norm in the ridge and elastic net penalty. Feature scores were computed as the averaged weights from all subspaces. Using simulated datasets, they showed that the ranking list from RSM-Ridge was more powerful in retrieving relevant features and more accurate in ranking informative features. On a real fMRI dataset, they showed that RSM-Ridge gave similar patterns across two fMRI runs under the same mental task.

2.3 A Unified Random Subspace Method for Classification and Feature Selection

2.3.1 Limitations of Current Random Subspace Methods

The above studies of RSM have several limitations. First, none of them provided an efficient framework to work with parameter tuning on base classifiers. When not handled correctly, parameter tuning is easily subject to the parameter selection bias. Lai et al. (2006) was the only study that attempted to tune parameters. However, their double-loop cross validation scheme is too computationally expensive and infeasible for the fMRI data. In their study, a simulated dataset had 300 features with 100 samples and the real dataset had 199 features with over 200 samples, while a typical fMRI data has tens of thousands of features and a very limited sample size, in the order of tens. Additionally, they did not take the potential advantage of RSM as an ensemble classifier. Instead, they used a separate Fisher classifier to perform classifications. Second, none of the studies is able to have a good control of false selections. In Lai et al. (2006), the final list of features is a union of optimally selected features from all inner loops and thus may include many irrelevant features. An alternative would use an intersection of feature lists but it may exclude too many relevant features. In Cai et al. (2007) and Sona et al. (2010), they only used RSM to generate a high-quality feature ranking list. They did not solve the feature selection problem and left it at the researcher's control. To control false selections, an intuitive approach is to start from the feature ranking list and set a threshold such that selected top-ranked features are mostly relevant.

In this section, I develop an RSM framework that (1) is efficient and suitable for base classifiers involving parameter tuning; (2) is an ensemble classifier; (3) is able to control false selections. In the first subsection, I extend the efficient bias correction method in Tibshirani and Tibshirani (2009) to an ensemble classifier formed by RSM. In the second subsection, I develop a thresholding method based on the random probe method in Tuv et al. (2009). I also design a novel feature scoring method and discuss its properties. Finally,

I describe and discuss the algorithm of the developed RSM framework.

2.3.2 Correction of the Parameter Selection Bias in the Random Subspace Method

Generally, to build a classifier, a dataset needs to be divided into three subsets: training set, validation set and test set. A classifier is trained with the training set and any parameter is tuned with the validation set. The true prediction error is estimated with the test set. Samples in the test set is not involved in any training or validation process. It is feasible to use this scheme if a dataset with many samples is available. However, for fMRI data which has very limited sample size, in order to use as much information as possible, a K-fold cross validation (CV) scheme is commonly used. In K-fold cross validation, a dataset is partitioned into K folds. One fold is left out for estimating generalization error and (K-1) folds are used to train a classifier. The procedure is repeated for K times and the average test error is taken as an estimate of the true prediction error of a classifier built on the whole dataset. Often, a classifier may have tuning parameters that need to be optimized for better classification and feature selection performance. For example, the elastic net penalty has two tuning parameters, one for l_1 norm and the other for l_2 norm. For such tunable classifiers, extra caution needs to be taken to use a K-fold cross validation scheme to estimate the generalization error. A common practice in a K-fold cross validation is to select optimal parameter values at the minimal cross validation error, and the minimal CV error is used as an estimate of the true expected prediction error. However, this scheme is subject to the parameter selection bias and the minimal CV error tends to have downward bias (Breiman, 1984; Varma and Simon, 2006; Efron, 2008) and Tibshirani and Tibshirani (2009)). To overcome the selection bias, a double-loop cross validation procedure is commonly used. However, this method is computationally impractical for fMRI data. To ease the computational burden, Tibshirani and Tibshirani (2009) developed an efficient bias correction method similar to a bootstrap estimate of the bias in Efron (1979). It directly uses results from each CV fold and does not require a significant amount of additional

computations and thus, is feasible for classifiers applied onto fMRI data. I first review this method and then extend it to ensemble classifiers built with RSM. For simplicity, I focus on a two-class classification problem and the loss function of interest is 0-1 loss.

The Bias Correction Method by Tibshirani and Tibshirani (2009)

Suppose we observe N independently and identically distributed data $(y_i, x_i), i = 1, ..., N$, where $x_i = (x_{i1}, ..., x_{ip})$ is a vector of predictors and y_i is a class label. A classifier trained on this data is denoted as $\hat{f}(x, \theta)$, where θ is the tuning parameter (a scalar or vector variable). For 0-1 loss function, $L(y, \hat{f}(x, \theta)) = \mathbf{1}(y \neq \hat{f}(x, \theta))$, where $\mathbf{1}(\cdot)$ is an indicator function. The expected predictor error defined on new independent test data (x_0, y_0) is $\mathrm{E}[L(y_0, \hat{f}(x_0, \theta))]$. In a K-fold cross validation, data is split into K equal subsets. For each $k = 1, ..., K$, the k^{th} subset is removed and a classifier $\hat{f}^{-k}(x, \theta)$ is fitted on the remaining data within (K-1) folds. Let C_k be the indices of data in the k^{th} fold. The cross validation estimate of the expected test error is

$$CV(\theta) = \frac{1}{N} \sum_{k=1}^{K} \sum_{i \in C_k} L(y_i, \hat{f}^{-k}(x_i, \theta)),$$

where $\hat{\theta}$ is chosen to minimize $CV(\theta)$. Generally, $CV(\theta)$ is a biased estimate. The bias is

$$\text{Bias} = \mathrm{E}[L(y_0, \hat{f}(x_0, \hat{\theta}))] - CV(\hat{\theta}).$$

Let N_k be the number of observations in the k^{th} fold. The error curve from predictions in the k^{th} fold is

$$e_k(\theta) = \frac{1}{N_k} \sum_{i \in C_k} L(y_i, \hat{f}^{-k}(x_i, \theta)).$$

The bias correction method uses the difference between e_k at θ and its minimum at $\hat{\theta}_k$ to mimic the bias in using the minimal CV error. The bias estimate is

$$\widehat{\text{Bias}} = \frac{1}{K}\sum_{k=1}^{K}[e_k(\hat{\theta}) - e_k(\hat{\theta}_k)].$$

The above bias correction method is similar to a bootstrap estimate of the bias, which is

$$\widehat{\text{Bias}}_{\text{boot}} = \frac{1}{B}\sum_{b=1}^{B}[CV(x^{*^b}, \hat{\theta}(x)) - CV(x^{*^b}, \hat{\theta}(x^{*^b}))],$$

where $\hat{\theta}(x)$ is the minimizer for the CV error curve on the original data and $\hat{\theta}(x^{*^b})$ is the minimizer for the CV error curve on bootstrapped sample x^{*^b}. The computation of bootstrap estimate of bias is very computationally intensive, requiring B K-fold cross validation, where B is typically in the order of 100. $\widehat{\text{Bias}}$ approximates $\widehat{\text{Bias}}_{\text{boot}}$ by using subsamples in the original cross validation folds instead of bootstrapped samples.

In Tibshirani and Tibshirani (2009), extensive simulations on different classifiers showed that the bias was relatively larger for $p \gg n$ compared with $p \ll n$. The bias adjusted estimate was more accurate than the minimal CV error and slightly overestimated the expected prediction error. Bias correction via a 5-fold cross validation was shown to be more accurate than a 10-fold cross validation because there are fewer samples in each fold for 10-fold cross validation.

Bias Correction for an Ensemble Classifier Formed by the Random Subspace Method

Suppose in RSM, M subspaces are generated. $\hat{f}_i(\tilde{x}^{(i)}, \theta_i)(i = 1, ..., M)$ is the classifier with its tuning parameter θ_i built in i^{th} subspace. The majority voting rule $\hat{g}(x, \theta)$ ($\theta =$

$[\theta_1, ..., \theta_M]$) is defined as:

$$\hat{g}(x,\theta) = \begin{cases} 0 & \text{if } \frac{1}{M}\sum_{i=1}^{M} \hat{f}_i(\tilde{x}^{(i)}, \theta_i) < 0.5 \\ 1 & \text{if } \frac{1}{M}\sum_{i=1}^{M} \hat{f}_i(\tilde{x}^{(i)}, \theta_i) > 0.5 \\ 0/1 & \text{if } \frac{1}{M}\sum_{i=1}^{M} \hat{f}_i(\tilde{x}^{(i)}, \theta_i) = 0.5 \end{cases} . \tag{2.1}$$

The 0-1 loss function is

$$L(y, \hat{g}(x,\theta)) = \mathbf{1}\{y \neq \hat{g}(x,\theta)\}.$$

In a K-fold cross validation, let C_k be the indices of observations in the k^{th} fold (test set), the cross validation error curve is

$$CV(\theta) = \frac{1}{N}\sum_{k=1}^{K}\sum_{j\in C_k} L(y_j, \hat{g}^{-k}(x_j,\theta)), \tag{2.2}$$

where $\hat{g}^{-k}(x,\theta)$ is the majority voting function for observations in k^{th} fold with base classifiers trained with data in the remaining (K-1) folds. It is defined as

$$\hat{g}^{-k}(x,\theta) = \begin{cases} 0 & \text{if } \frac{1}{M}\sum_{i=1}^{M} \hat{f}_i^{-k}(\tilde{x}^{(i)}, \theta_i) < 0.5 \\ 1 & \text{if } \frac{1}{M}\sum_{i=1}^{M} \hat{f}_i^{-k}(\tilde{x}^{(i)}, \theta_i) > 0.5 \\ 0/1 & \text{if } \frac{1}{M}\sum_{i=1}^{M} \hat{f}_i^{-k}(\tilde{x}^{(i)}, \theta_i) = 0.5 \end{cases} . \tag{2.3}$$

In equation (2.3), $\hat{f}_i^{-k}(\tilde{x}^{(i)}, \theta_i)$ is a function of θ_i and the cross validation error curve in the i^{th} subspace is

$$CV_i(\theta_i) = \frac{1}{N}\sum_{k=1}^{K}\sum_{j\in C_k} L(y_j, \hat{f}^{-k}(\tilde{x}_j^{(i)}, \theta_i)). \tag{2.4}$$

Let $\hat{\theta} = [\hat{\theta}_1, ..., \hat{\theta}_M]$ consists of $\hat{\theta}_i, (i = 1, ..., M)$ that minimizes equation (2.4). The expected test error with new and independent data (y_0, x_0) at $\hat{\theta}$ is

$$\text{Err} = \text{E}[L(y_0, \hat{g}(x_0, \hat{\theta}))]. \tag{2.5}$$

Using equation (2.2), the bias of $CV(\hat{\theta})$ as an estimate for Err is

$$\text{Bias} = \text{Err} - CV(\hat{\theta}) = \text{Err} - \frac{1}{N}\sum_{k=1}^{K}\sum_{j \in C_k} L(y_j, \hat{g}^{-k}(x_j, \hat{\theta})). \tag{2.6}$$

Let N_k be the number of observations in the k^{th} fold and the error curve calculated from the predictions in the k^{th} fold is

$$e_k(\theta) = \frac{1}{N_k}\sum_{j \in C_k} L(y_j, \hat{g}^{-k}(x_j, \theta)). \tag{2.7}$$

Similarly, let $\hat{\theta}^{(k)} = [\hat{\theta}_1^{(k)}, ..., \hat{\theta}_M^{(k)}]$, in which $\hat{\theta}_i^{(k)}, i = 1, ..., M$ minimizes

$$\frac{1}{N_k}\sum_{j \in C_k} L(y_j, \hat{f}_i^{-k}(\tilde{x}_j^{(i)}, \theta_i^{(k)})).$$

The proposed bias estimation is

$$\widehat{\text{Bias}} = \frac{1}{K}\sum_{k=1}^{K}[e_k(\hat{\theta}) - e_k(\hat{\theta}^{(k)})]. \tag{2.8}$$

If fold sizes are equal, then $CV(\hat{\theta}) = \frac{1}{K}\sum_{k=1}^{K} e_k(\hat{\theta})$ and the adjusted estimate of the

test error is

$$\mathrm{Err}_{\mathrm{Adj}} = CV(\hat{\theta}) + \widehat{\mathrm{Bias}} = 2CV(\hat{\theta}) - \frac{1}{K}\sum_{k=1}^{K} e_k(\hat{\theta}^{(k)}). \tag{2.9}$$

Note that there is one restriction that requires fold partitions to be the same across all random subspaces. This makes (2.3) well-defined and enables a calculation of equation (2.7). The bias correction also resembles a bootstrap estimate of the bias. Instead of drawing bootstrapped samples, K subsamples are used. Since the ensemble classifier in RSM is nonlinear, a 5-fold cross validation is used instead of 10-fold in order to increase the number of observations in each validation fold. The bias in the ensemble classifier formed by RSM (equation (2.6)) is also likely to be nonnegative because the minimal CV error in each subspace is an under-biased estimate. Moreover, test error of an ensemble classifier with optimized members is likely to have more bias than a single classifier (using an optimized single classifier).

2.3.3 Feature Selection in the Random Subspace Method

A Novel Feature Scoring Method

None of the previously developed scoring methods is applicable to RSM using LREN and RFESVM as base classifiers. First, for LREN, feature weights are shrunken towards zeros depending on the optimal parameter values in each subspace and are thus incomparable between subspaces. Second, for RFESVM in a high dimensional feature space, it is computationally infeasible to eliminate features one by one and rank them based on the orders of being discarded. Thus I design a novel scoring method as follows. For i^{th} feature (F_i), its score is defined as

$$FS_i = \frac{\sum_{m=1}^{M} \mathbf{1}\{F_i = 1 \wedge F_i \in \mathrm{RS}_{(m)}\}(1 - \mathrm{CVErr}_{(m)})}{\sum_{m=1}^{M} \mathbf{1}\{F_i \in RS_{(m)}\}}, \tag{2.10}$$

where $F_i = 1$ denotes that i^{th} feature is selected in a subspace, $\text{RS}_{(m)}$ is the m^{th} subspace, $\text{CVErr}_{(m)}$ is the minimal CV error in the m^{th} subspace.

The proposed feature scoring method has two ingredients. First, the scoring part without $(1 - \text{CVErr})$ is the probability that the i^{th} feature gets selected among all the subspaces that include it. Intuitively, a higher probability indicates a higher discriminative capacity. However, the probability does not take into account the discrimination quality of the selected subset. For example, if the percentage of informative features is very low, many subspaces contain only noises, and the selected features are mostly irrelevant. To alleviate this effect, I add a second ingredient that uses the best cross validation accuracy $(1-\text{CVErr})$ to represent the subset quality. A higher cross validation accuracy indicates a higher quality and is more likely to contain relevant features. However, an implicit assumption in using the minimal CV error is that a subset containing more relevant features leads to a lower minimal CV error. This assumption is empirically validated with simulated datasets.

Grouping Effect

A classifier f has a grouping effect for two highly correlated feature i and j if both features tend to be selected or discarded together. Grouping effect was first proposed in Zou and Hastie (2005) in studying the elastic net penalty. They proved that using an elastic net penalty in linear regression, the difference between coefficients of two features is bounded by $\sqrt{(1-\rho)}$ multiplied by a constant (a function of tuning parameters), where ρ is the correlation between two features. An extreme case is that when the two features are exactly the same ($\rho = 1$), their coefficients from using an elastic net penalty are exactly the same. The grouping effect of elastic net penalty turns out to be useful in fMRI data because discriminative features have a high correlation. This ensures that different brain regions engaged in discriminating two brain states are more likely to be discovered together. Sona et al. (2010) proved in the extreme case that if two features are the same and base classifier has the grouping effect, their feature scores from RSM are also the same. In the following, I provide a simple proof showing that in the extreme case considered in Sona et al. (2010),

the RSM framework using feature scores defined in equation (2.10) preserves the grouping effect as long as the deterministic base classifier has a grouping effect.

Proposition 1. Suppose all subspaces have the same dimension l and S^l is the set including all possible subspaces. If a base classifier f is a deterministic classifier with a grouping effect, then $\forall i, j$ features, such that $v_i = v_j$, it holds $FS_i = FS_j$, for feature score FS_i and FS_j defined in equation (2.10).

Proof. S^l can be partitioned into four different categories: $S_{ij} = \{s \in S^l | v_i \in s \text{ and } v_j \in s\}$; $S_{\bar{i}j} = \{s \in S^l | v_i \notin s \text{ and } v_j \in s\}$; $S_{i\bar{j}} = \{s \in S^l | v_i \in s \text{ and } v_j \notin s\}$; $S_{\bar{i}\bar{j}} = \{s \in S^l | v_i \notin s \text{ and } v_j \notin s\}$. On S_{ij}, since f has grouping effect and $v_i = v_j$, on S_{ij} partition, $FS_i^{(ij)} = FS_j^{(ij)}$. Since $v_i = v_j$, $S_{\bar{i}j} = S_{i\bar{j}}$ and any subset in $S_{\bar{i}j}$ has an exact replicate in $S_{i\bar{j}}$ and vice versa. Because the classifier f is a deterministic and has the same minimal CV error on both partitions, $FS_j^{(\bar{i}j)} = FS_i^{(i\bar{j})}$. On the subset of $S_{\bar{i}\bar{j}}$, neither feature i nor feature j is included and thus $FS_i^{(\bar{i}\bar{j})} = FS_j^{(\bar{i}\bar{j})} = 0$. Overall, combining feature scores from all categories, $FS_i = FS_j$. □

Remark 2.1. Proposition 1 establishes that if a deterministic classifier which exhibits a grouping effect, using the feature score defined in equation (2.10), RSM inherits the grouping effect. Though the above proof is for the perfect correlation condition, I conjecture that for correlated features, the difference between feature scores is bounded by some function of feature correlation. The intuition is as follows: on the four partitions used in the above proof, the partition that does not contain either feature i or feature j has no influence on feature scores. On the partition that contains both i^{th} and j^{th} features, because of the grouping effect of the base classifier, the difference between the two scores is bounded by a constant multiplying $\sqrt{(1-\rho)}$. On the two partitions containing either i^{th} or j^{th} feature, subspaces from two partitions can be exactly matched except for the two features of interest. On the matched subspaces, as feature correlation moves away from 1, minimal CV errors and coefficients of two features begin to diverge. Integrating scores from all matched subspaces,

averaged scores from the two partitions also diverge. Thus feature scores from all subspaces have more discrepancy as features are less correlated.

For the two base classifiers under investigation, it was proved that LREN has a grouping effect (Wang and Zhu, 2006). They had a proof under a very general condition: for the elastic net penalty, if the loss function is Lipschitz continuous then for any pair of features, their coefficient difference is bounded by a constant (a function of tuning parameters) multiplying $\sqrt{(1-\rho)}$. For LREN, the loss function is binomial deviance and is Lipschitz continuous, and thus LREN has the grouping effect. However, this is not the case for RFESVM. The loss function of a linear SVM is the hinge loss. As the grouping effect is from the ridge penalty (l_2 norm of coefficients), SVM has the grouping effect. In RFESVM, the number of elimination levels is another tuning parameter and its effect is the same as a hard thresholding instead of a soft thresholding in the elastic net penalty. Thus generally it does not have the grouping effect. However, if the two highly correlated features are both kept at the optimal elimination level, RFESVM has a grouping effect. Also, if the two features are exactly the same, their weights turn out to be exactly the same.

Discrimination Direction

The feature score defined in equation (2.10) does not differentiate a feature's discrimination direction. A feature selected in two subspaces may have opposite discrimination directions, favoring different class labels. Suppose a positive weight favors class 2 and a negative weight favors class 1. In order to infer a feature's discrimination direction in RSM framework, I propose an intuitive majority voting scheme to decide the sign: if a feature has a positive sign for more than half of the times for which a feature is selected out, the sign of that feature is positive; otherwise negative. This approach is empirically validated on simulated datasets.

Feature Selection Threshold

One of the major goals in mapping informative patterns of fMRI data is to extract as much true discriminative features as possible under an acceptable rate of false selections. Both

LREN and RFESVM determine the optimal feature selection by tuning parameter values to achieve the highest classification accuracy. However, a complete search over parameter space is just computationally infeasible and instead, in practice, multiple values are sampled from the whole parameter space. It turns out that feature selection critically depends on the sampled values. Some informative features may just be missed by not sampling enough parameter values. For example, in RFESVM, it is difficult to set the elimination rate at each elimination level because a higher rate may discard more informative features while a smaller rate increases computation time and may include more irrelevant features due to an early stopping. Another limitation is that even if a complete search is possible, it is still difficult to assess the quality of the selected features, that is, the specificity of the mapped patterns is unknown. Recently, random probe methods have been proposed to assist select relevant features (Stoppiglia et al., 2003; Tuv et al., 2009). The basic idea is to inject artificial variables/features into the original feature space. Since artificial variables are noises and are expected to behave similarly to the original noisy features, some control over false selections can be realized by utilizing the known information on artificial variables.

The random probe method was first developed by Stoppiglia et al. (2003). In their method, they injected a Gaussian random variable and computed the cumulative distribution of the rank of the artificial variable. The angle between a feature and the response variable was used to rank features. An iterative Gram-Schmidt orthogonalization procedure was used to account for the effect of explained variance by previously ranked features. The cumulative distribution of the rank of a random probe was theoretically computed. It was used to set a threshold on the ranks such that the probability that the rank of a random probe might explain the response variable better than one of the features above that threshold is under certain level (the risk that a researcher is willing to take to include irrelevant features). The random probe method provides an objective way to set the boundary between relevant and irrelevant features.

Tuv et al. (2009) applied the random probe method combined with the Random Forest to discard both irrelevant and redundant features. Feature redundancy is generally defined

as feature correlation. In their method, values of each variable $v_i, i = 1, ..., p$ were permuted to generate a realization of a random probe/artificial variable such that the permuted variable v_i' has no predictive power for the response variable. p artificial variables were added to the original data and the Random Forest is trained with the augmented data. Importance scores from the Random Forest were calculated for both real and artificial features. The procedure was repeated for R rounds, creating a importance matrix $\boldsymbol{IS}_{R\times 2p}$. The $(1-\alpha)$ percentile of the importance scores in round $r \in 1, ..., R$ was calculated from only random probe features. The vector of percentiles over the R replicates was $IP_{R\times 1}$. For each real feature v_i, a paired t test compared its importance scores to IP. v_i is identified as relevant variables if the test is significant. The significance level was chosen by a researcher using Bonferroni correction, false discovery rate control or other methods controlling false positives.

However, both methods are not applicable in RSM. The first method is limited to the Gram-Schmidt procedure, and usually a theoretical derivation of cumulative distribution is not available. The second method is too computationally expensive for RSM because it requires R replications. To balance computation load and accuracy in false selection control, I develop an efficient yet heuristically sound random probe method. The method is similar to the one used in Tuv et al. (2009) except that only one replication is used here. Specifically, p permutations of p real features are generated as realizations of the additional p artificial variables. After injection of artificial features, dimension of the feature space is augmented to $2p$. After applying the RSM to compute feature score using equation (2.10), scores of artificial features are sorted and the threshold is set at the $(1-\alpha)$ percentile. Though values of features are permuted, the marginal distribution of each artificial variable is kept the same as its corresponding real variable. Ideally, an informative original feature should have higher predictive power and a higher score than both its permuted variable and original noisy features. However, by chance, some artificial features might have more discriminative power than some weakly informative features. In order to increase detection power, $(1-\alpha)$ percentile is used instead of the highest score among artificial variables.

Intuitively, α measures to the risk of accepting irrelevant features as relevant ones.

2.3.4 Combining Classification and Feature Selection Using the Random Subspace Method

In this subsection, I describe the algorithm to combine classification and feature selection in a unified RSM framework. Instead of randomly sampling features, I randomly partition the feature space into multiple mutually exclusive subspaces. For example, if the subspace size is 10% of the original feature space size, 10 mutually exclusive random subspaces are generated for one partition.

A unified RSM Algorithm for Classification and Feature Selection

1. ***Generate random probes:*** For every real feature $v_i, i = 1, ..., p$, a corresponding random probe is generated by a permutation. p random probe features are added to the original feature space. The augmented data is $(\boldsymbol{y}_{N\times 1}, \tilde{\boldsymbol{X}}_{N\times 2p})$.

2. ***Generate random subspaces:*** Partition the augmented feature space into several mutually exclusively feature subsets and repeat the partition for multiple times. Data for classification is then projected onto each subspace.

3. ***Train a base classifier in each subspace:*** Train a base classifier in each subspace and compute the feature score defined in equation (2.10).

4. ***Compute an estimate of generalization error:*** Use the developed bias correction method to compute an adjusted estimate of generalization error for the ensemble classifier formed by RSM. Classification is performed using a majority voting scheme.

5. ***Eliminate irrelevant features:*** First, rank feature scores of random probe variables. Set the selection threshold at $(1 - \alpha)$ percentile on the ranking list. Second, rank feature scores of real features and discard those features whose scores are below the threshold. Keep remained features as relevant.

6. ***Determine discrimination direction for selected features:*** For each selected feature, determine its discrimination direction based on the percentage of a positive weight sign among all subspaces in which the feature is included and selected.

To classify new samples, the same random probe variables are added and the same number of subspaces are generated with each containing exactly the same features as in the K-fold cross validation. The classification is made by the majority voting from decisions in all subspaces.

Parallelism of the Developed Random Subspace Method

The developed random subspace method is very computationally intensive for fMRI data because more subspaces are required for a higher dimensional feature space in order to effectively explore the data structure. However, the high computational cost can be offset by the highly parallel nature of the RSM framework. Training of base classifiers is locally performed within each subspace and thus the whole training process can be easily parallelized. Moreover, the developed RSM framework is able to handle large datasets with an extremely high dimensional feature space (e.g. spatiotemporal fMRI data). However, for a single classifier facing the whole pattern, lots of computer memory is required for the training. In contrast, for the developed RSM, training is much easier because each subspace has a much lower feature dimension, demanding less computer memory.

2.4 Settings of Data Generation Process

I focus on binary classification problems which are more common in current fMRI data analysis. I carry out three simulation studies. The first large scale simulation study is similar to the one performed in Ryali et al. (2010), aiming to provide a detailed assessment of the developed RSM framework and a fair comparison with benchmark methods (LREN and RFESVM). The second simulation is to evaluate whether RSM is a good feature ranking method. The third simulation is to examine whether RSM can correctly determine features' discriminative directions. All simulated datasets are generated with a high feature-to-sample

ratio, representing real fMRI data. Specifically, the number of features is set at 1000 and the number of samples is 50, with 25 for each class.

First Simulation Study

For the first simulation study, I basically follow the settings in Ryali et al. (2010) to investigate effects of the signal-to-noise ratio (SNR) and prevalence rate (PR) of informative features. In Ryali et al. (2010), they set different pairwise voxel correlations in a brain region for different classes (0.5 for one and 0.7 for the other). Though varying spatial correlations is of course one way to introduce discriminative patterns, it may potentially confound effects of other factors. Therefore, I set a single region within which pairwise voxel correlation is fixed at 0.5. The signal-to-noise ratio is define as

$$\text{SNR} = \frac{|\mu_2 - \mu_1|}{\sigma},$$

where μ_i is the mean signals of informative features for class i. σ is the standard deviation of background noises. The prevalence rate is defined as the percentage of discriminate features compared to the total number of features. Patterns of interest are drawn from Gaussian distribution. Particularly, for class 1, (1000*PR) informative features are drawn from a multivariate Gaussian distribution with mean $\mu_1 = 1$, pairwise correlation $\rho = 0.5$ and standard deviation $\sigma = 1$. For class 2, the same (1000*PR) informative features are drawn from a multivariate Gaussian distribution with mean $\mu_2 = 1 + \text{SNR}$ and with the same variance-covariance structure as in class 1. The remaining 1000*(1-PR) irrelevant features are identically and independently drawn from a univariate Gaussian distribution with $\mu_0 = 0$ and $\sigma^2 = 1$. I vary SNR = 0.5, 1, 1.5, representing low, median and high discriminative power. I also vary PR = 1%, 5%, 10%, 20%. PR = 1% may represent a whole brain analysis since empirically informative brain regions are relatively small in size while 20% may represent an ROI analysis focusing on several candidate regions. Datasets are simulated in all combinations of parameter values, totally 12 scenarios.

For a full understanding of baseline methods (LREN and RFESVM), I generate 1000 training datasets for an accurate assessment. Since RSM based methods require much more computation power, to introduce some variation in data sampling, I simulate 10 training datasets for fair comparisons with their benchmark methods. An independent dataset is generated for estimating the expected prediction accuracy of both benchmark and RSM based methods under each combination of SNR and PR. The data generation process is the same but with 5000 samples, 2500 for each class.

Table 2.1 is an overview of parameters used in the first simulation study.

Table 2.1: Parameters in the first simulation study

Parameters	**Values**
Number of features	1000
Number of samples in a training dataset	50 (25 for each class)
Number of samples in a test dataset	5000 (2500 for each class)
Signal-to-noise ratio (SNR)	$\{0.5, 1, 1.5\}$
Prevalence rate (PR)	$\{1\%, 5\%, 10\%, 20\%\}$
Number of informative brain regions	1
Pairwise correlation between features within an informative region	0.5
Pairwise correlation between features outside informative regions	0
Variance of background noises	1

Second Simulation Study

In this simulation study, I examine whether RSM is an effective feature ranking method compared to the true rankings based on their SNRs. Instead of setting a single brain regions with a constant mean, I set 20 informative regions, each with 10 features. For class 1, means of 20 regions are set at 1 and for class 2, SNRs of 20 regions are in an increasing magnitude starting from 0.1 to 2.0 in the increment of 0.1. The pairwise correlation between informative features in each region under both classes is set at 0.5. Standard deviation of background noise is 1. 50 samples are set for the training dataset and 5000 samples for the test data. Samples are balanced between classes. 10 training datasets are simulated for evaluation and comparisons with benchmark methods. Table 2.2 summaries the settings for the second simulation study.

Table 2.2: Parameters in the second simulation study

Parameters	Values
Number of features	1000
Number of samples in a training dataset	50 (25 for each class)
Number of samples in a test dataset	5000 (2500 for each class)
Signal-to-noise ratio (SNR)	$\{0.1 : 0.1 : 2.0\}$
Number of informative brain regions	20
Number of features in each informative region	10
Pairwise correlation between features within an informative region	0.5
Pairwise correlation between features outside informative regions	0
Variance of background noises	1

Third Simulation Study

In this simulation study, I evaluate whether RSM is able to correctly determine informative features' discrimination direction. I set 10 informative regions. The first 5 regions favor class 2, with SNRs in an increasing order: from 0.5 to 1.5 in the increment of 0.25. The other 5 regions favor class 1, with SNRs in an increasing order: from -0.5 to -1.5 in an decrement of 0.25. Each region has 10 features and the pairwise correlation between features within each informative region is 0.5. Table 2.3 summaries the settings for the third simulation study.

Table 2.3: Parameters in the third simulation study

Parameters	Values
Number of features	1000
Number of samples in a training dataset	50 (25 for each class)
Number of samples in a test dataset	5000 (2500 for each class)
SNRs of the first 5 informative regions	$\{0.5 : 0.25 : 1.5\}$
SNRs of the second 5 informative regions	$\{-0.5 : -0.25 : -1.5\}$
Number of features in each informative region	10
Pairwise correlation between features within an informative region	0.5
Pairwise correlation between features outside informative regions	0
Variance of background noises	1

2.5 Summary and Conclusions

In this work, I develop a unified random subspace method that can simultaneously perform classification and feature selection and is suitable for the high dimensional fMRI data. I provide several contributions to the literature. Previous studies either use RSM as solely an ensemble classifier or a feature selection method. To my knowledge, this is the first attempt to take both advantages of RSM. Moreover, to avoid the selection bias, previous study of RSM involving tuning parameters either fix parameter values beforehand or resort to a double-loop cross validation procedure. Thus it requires either prior information about parameter values or excessive computations, which makes RSM infeasible for the fMRI data. I extend the bias correction method in Tibshirani and Tibshirani (2009) to an ensemble classifier formed by RSM with tunable classifiers. The extended method is much more efficient compared to a double-loop cross validation and eliminates the need of presetting tuning parameter values. Second, previous studies of RSM in feature selection problem only provided a feature ranking list, and it was hard to reliably select relevant features without expert knowledge. I propose to combine RSM with the random probe method to provide a data-dependent threshold in discerning relevant features from irrelevant ones. Unlike other computationally intensive nonparametric methods such as permutation test, the random probe method does not dramatically increase the computational load. I also discuss parallelizing the RSM framework to reduce the computation cost.

Chapter 3: Application to Logistic Regression with an Elastic Net Penalty

3.1 Introduction

Logistic regression is a popular method for solving pattern recognition problems. However, its application is limited in modern massive datasets because of the overfitting problem, where the number of variables or features is much larger than the number of samples. A number of regularization methods have been developed to alleviate the overfitting problem. Among them, least absolute angle shrinkage and selection operator (LASSO, Tibshirani, 1996) and elastic net (EN, Zou and Hastie, 2005) are becoming popular in fMRI application because they can simultaneously classify patterns and select important variables. Due to the l_1 norm regularization on model parameters, both LASSO and EN are potentially able to shrink coefficients of irrelevant variables to exactly zeros and thus are useful in mapping informative patterns. However, EN has some critical advantages over LASSO due to the additional l_2 norm regularization term (Zou and Hastie, 2005):

- LASSO can only select features up to the number of samples. EN does not have this limitation.
- When variables are highly correlated, which is common in fMRI data, LASSO tends to randomly select only one feature from each group. EN is able to select correlated features together as a group (a grouping effect).

Logistic regression with an EN penalty (LREN) has been successfully applied in classification and mapping of fMRI data patterns. Ryali et al. (2010) carried out extensive simulations comparing LREN with other competing methods including LASSO and RFESVM.

They found that LREN had better performance in feature selection in terms of sensitivity, specificity and accuracy. They also reported on a real fMRI data that LASSO picked focal patterns only in the left-hemisphere, while LREN was able to show bilaterally distributed patterns. For classifiers other than logistic regression, EN penalty has also been shown to have superior performance. Grosenick et al. (2008) added an EN penalty to a linear discriminant classifier and evaluated it on several regions of interest. They found that its classification accuracy significantly outperformed logistic regression, linear discriminant analysis and linear SVM. Carroll et al. (2009) also reported that linear regression with an optimized EN penalty was more accurate in prediction than ordinary least squares with fixed number of selected voxels, regardless of initial feature selection approaches.

However, LREN is usually trained directly on the whole feature space and suffers from the curse of dimensionality. Additionally, its feature selections critically depend on values of tuning parameters and lacks a control of false selections. In this chapter, built upon the developed RSM framework in chapter 2, I propose the RSMLREN method which uses LREN as a base classifier in each subspace generated by RSM. Using datasets generated with settings specified in chapter 2, I empirically evaluate performance of RSMLREN in both classification and feature selection with comparisons to the benchmark method - LREN trained with all features. I also examine whether RSMLREN is a successful feature ranking method and is able to determine each informative feature's discrimination direction. Finally, I evaluate its preciseness in control of false selections.

The remainder of the chapter is organized as follows. Section 3.2 provides an overview of the EN penalty in both linear and logistic regression. In section 3.3, I describe the proposed RSMLREN algorithm. In section 3.4 and 3.5, I present results from three simulation studies. Section 3.6 concludes the study.

3.2 Review of the Elastic Net Regularization

3.2.1 Linear Regression with an Elastic Net Penalty

In this chapter, I use v_i to represent the i^{th} variable which has a dimension of N and $V = [v_1, ..., v_p]$. I use x_i to represent the i^{th} sample with a dimension of p and $X = [x_1^T, ..., x_N^T]^T$. X and V represents the same data structure.

The elastic net regularization was first proposed by Zou and Hastie (2005) for linear regression problems. Let $y = [y_1, ..., y_N]^T$ be a vector of response variables. The "naive" elastic net estimator for fixed nonnegative λ_1 and λ_2 is $\hat{\beta} = \underset{\beta}{\operatorname{argmin}} L(\lambda_1, \lambda_2, \beta)$, where

$$L(\lambda_1, \lambda_2, \beta) = ||y - X\beta||_2^2 + \lambda_1|\beta|_1 + \lambda_2||\beta||_2^2, \tag{3.1}$$

$$|\beta|_1 = \sum_{j=1}^{p} |\beta_j|,$$

$$||\beta||_2^2 = \sum_{j=1}^{p} \beta_j^2,$$

where $\lambda_1|\beta|_1 + \lambda_2||\beta||_2^2$ is called the elastic net penalty, which is a combination of a LASSO penalty term $\lambda_1|\beta|_1$ and a ridge penalty term $\lambda_2||\beta||_2^2$.

Zou and Hastie (2005) showed that the minimization of equation (3.1) is equivalent to a LASSO type optimization and thus can be solved efficiently using LARS algorithm (Efron et al., 2004). Let the augmented data set be $(y^\star, X^\star)$ by,

$$X^\star_{(n+p)\times p} = (1 + \lambda_2)^{-\frac{1}{2}} \begin{pmatrix} X \\ \sqrt{\lambda_2}\mathbf{I} \end{pmatrix}, Y^\star_{(n+p)} = \begin{pmatrix} y \\ 0 \end{pmatrix}.$$

Let $\gamma = \frac{\lambda_1}{\sqrt{(1+\lambda_2)}}$ and $\beta^\star = \sqrt{1+\lambda_2}\beta$. Then the elastic net estimator is to minimize

$$L(\gamma, \beta) = L(\gamma, \beta^\star) = ||y^\star - X^\star\beta^\star||_2^2 + \gamma||\beta^\star||_1,$$

where

$$\hat{\beta^\star} = \underset{\beta^\star}{\operatorname{argmin}} L(\gamma, \beta^\star),$$

and

$$\hat{\beta} = \frac{1}{\sqrt{1+\lambda_2}}\hat{\beta^\star}.$$

In the augmented dataset, $y^\star$ has a dimension of $(n+p)$ and thus the elastic net is able to select the number of predictors up to $(n+p)$.

Grouping Effect

The elastic net penalty also exhibits a grouping effect such that the regression coefficients of highly correlated predictors tend to be equal. In fact, Zou and Hastie (2005) proved that coefficient paths of two correlated predictors are bounded by their correlation up to a constant. Suppose $\hat{\beta}_i(\lambda_1, \lambda_2)\hat{\beta}_j(\lambda_1, \lambda_2) > 0$ and define:

$$D_{\lambda_1,\lambda_2}(i,j) = \frac{1}{|y|_1}|\hat{\beta}_i(\lambda_1, \lambda_2) - \hat{\beta}_j(\lambda_1, \lambda_2)|,$$

then

$$D_{\lambda_1,\lambda_2}(i,j) \leq \frac{1}{\lambda_2}\sqrt{2(1-\rho)}.$$

For the extreme condition that two predictors are equal (or perfectly correlated), their regression coefficients are the same. LASSO, however, does not have this property. Tibshirani (1996) illustrated by a simple example that even if two variables are perfectly correlated, the difference between two coefficients does not shrink to zero. Grouping effect has very

important implications in analyzing fMRI data. From the perspective of neuroscience, brain regions engaged in a mental task tend to be correlated and should be selected together as a group for a more complete description of brain functions.

The grouping effect from elastic net penalty is not only restricted to the squared error loss function. In fact, it can be extended to any loss functions that is Lipschitz continuous in model parameters, i.e. $L(\beta_1) - L(\beta_2) \leq M||\beta_1 - \beta_2||_1$ for some positive finite constant M. Under Lipschitz continuous condition, Wang et al. (2006) proved that for any two standardized predictors v_i, v_j, for some constant M and sample size N, it has

$$|\hat{\beta}_i - \hat{\beta}_j| \leq \frac{\sqrt{N}M}{\lambda_2}\sqrt{2(1-\rho)}. \tag{3.2}$$

Since log-likelihood loss function is Lipschitz continuous, inequality (3.2) also holds for logistic regression. As an extension to the linear regression model, in the following subsection, I provide a brief review over the logistic regression with an elastic net penalty.

3.2.2 Logistic Regression with an Elastic Net Penalty

The above review is for the linear regression model with a squared error loss function. The elastic net penalty can also apply to logistic regression which uses a log-likelihood loss function. Logistic regression is a generalized linear model explicitly modeling the posterior probability of multiple classes via a linear function of predictors V. The review below focuses on a two class case. The logistic regression has the form defined as below

$$\log\frac{P(\mathbf{Y}=2|\mathbf{X}=X)}{P(\mathbf{Y}=1|\mathbf{X}=X)} = X\beta.$$

The posterior probability for each class is

$$P(\mathbf{Y}=1|\mathbf{X}=X) = \frac{1}{1+e^{-X\beta}},$$

$$P(\mathbf{Y} = 2|\mathbf{X} = X) = \frac{e^{-X\beta}}{1 + e^{-X\beta}}.$$

The loglikelihood loss function of logistic regression is

$$L(\beta) = \sum_{i}^{N} \{-y_i x_i \beta + \log(1 + e^{x_i \beta})\}.$$

With an elastic net penalty on regression coefficients, the estimator $\hat{\beta}$ is to minimize

$$\hat{\beta} = \underset{\beta}{\operatorname{argmin}} \frac{1}{N} \sum_{i}^{N} \{-y_i x_i \beta + \log(1 + e^{x_i \beta})\} + \lambda_1 |\beta|_1 + \lambda_2 ||\beta||_2^2. \tag{3.3}$$

3.3 Random Subspace Method Applied to Logistic Regression with an Elastic Net Penalty

In this section, I propose an algorithm to apply the developed RSM framework to logistic regression with an elastic net penalty. To solve the optimization problem in (3.3) efficiently, I use "Glmnet"[1] (a MATLAB toolbox developed by Department of Statistics at Stanford University) to fit LREN for every combination of specified values of λ_1 and λ_2.

3.3.1 The RSMLREN algorithm

- Specify sampled values of tuning parameters (λ_1 and λ_2) (I use default settings in the Glmnet toolbox)
- Generate random probe variables: form an augmented feature space by adding permuted variables to the original feature space
- Initialize the feature scores: $s = \mathbf{0}_{1 \times 2p}$
- Repeat until the specified number of iterations/partitions is met. For each iteration:

[1] Glmnet for Matlab, http://www-stat.stanford.edu/∼tibs/glmnet-matlab/

- Partition the augmented feature space into several mutually exclusive subspaces
- In each subspace:
  - Train LREN using a 5-fold cross validation scheme for each pair of tuning parameter values and determine the optimal tuning parameter values, which lead to the highest cross validation accuracy (BestCVA, if there is a tie, select a larger value for l_1 norm parameter, preferring a sparser solution).
  - Fit LREN with the optimal parameter values using all features in the subspace.
  - Update the feature score for each feature in the partition: $s_i = s_i$+BestCVA$\times$ $\mathbf{1}$(zero LREN weight or not) and $\mathbf{1}(.)$ is an indicator function that is 0 if i^{th} feature's weight in the above step is zero and 1 if nonzero.

- Compute the estimated prediction accuracy by the majority voting from all subspaces with the bias correction
- Update feature scores: $s = \frac{s}{\#Iterations}$
- Derive the threshold in controlling false positives based on the distribution of scores of random probe variables
- Determine discrimination directions of selected features

To classify future independent data point, the same random probe variables are added and the same number of subspaces are generated containing exactly the same features as in the K-fold cross validation. The classification is made by the majority voting from decisions in all subspaces. In the next subsection, I discuss choosing values for parameters involved in the proposed RSMLREN algorithm.

3.3.2 Parameters in RSMLREN

1. Tuning parameters in the elastic net penalty

   There are two tuning parameters involved in the elastic net penalty in LREN. λ_1 is

for l_1 norm regularization and λ_2 is for l_2 norm regularization. In the Glmnet toolbox, $\lambda_1|\beta|_1 + \lambda_2||\beta||_2^2$ is rewritten as $\lambda\{\frac{1}{2}(1-\alpha)||\beta||_2^2 + \alpha|\beta|_1\}, \alpha \in [0,1]$ (Friedman et al., 2010). The default sampled values of *alpha* is from 0.1 to 0.9 in the increment of 0.1. For each fixed α value, it turns out that there is a maximum λ value such that all feature weights are shrunken to zeros ($N\alpha\lambda_{max}^{(\alpha)} = \underset{l}{max}| < x_l, y > |$, Friedman et al., 2010). Default values for $\lambda^{(\alpha)}$ at each fixed α are sampled linearly in log-scale from $0.001\lambda_{max}^{(\alpha)}$ to $\lambda_{max}^{(\alpha)}$ (100 values in total)

2. Random subspaces

   The whole feature space is partitioned into mutually exclusive subspaces. Since each feature belongs to only one subspace in one partition, the number of subspaces generated in one partition depends on the subspace size. I set three subspace sizes: 10%, 25% and 50% of the whole feature space size. The other parameter is the number of partitions/iterations, for which I set 200. Therefore, for 10% subspace size, the total number of subspaces is 2000; for 25%, it is 800 and for 50%, it is 400.

3. Random probe variables

   I make permutations for all original features (1000 features) and add them to form the augmented feature space. Thus the total features are 2000.

Table 3.1 provides an overview of parameter values used in evaluations of RSMLREN algorithm.

Table 3.1: Values of parameters used in RSMLREN algorithm

Parameters	**Values**
α (in the EN penalty)	{0.1:0.1:0.9}
λ (in the EN penalty)	100 values, default in the toolbox
Subspace size	{10%, 25%, 50%}
Number of iterations	200
Number of random probe variables	1000

3.4 Simulation Results

3.4.1 First Simulation Study

The first simulation study resembles the study in Ryali et al. (2010). It is designed to examine effects of SNRs and PRs on classification and feature selection performance for both LREN and RSMLREN. Before I compare RSMLREN with LREN, I first document simulation results from LREN. Ryali et al. (2010) used a single simulated dataset to evaluate LREN, and thus their study was obscured from several interesting results, especially in the feature selection. Here, evaluations of LREN are based on 1000 simulated datasets. The detailed data generating process is described in Section 2.4.

Results of the Benchmark Method - LREN

Misclassification Errors

Table 3.2 shows various misclassification error rates at different combinations of PRs and SNRs. Adjustment to minimal CV errors uses the method from Tibshirani and Tibshirani (2009).

Higher SNR consistently results in lower misclassification errors (minimal CV errors, adjusted CV errors and test errors) because data distributions under two classes are more separable. Within each SNR, both minimal CV errors and adjusted CV errors decrease with an increasing PR. However, at median SNR=1 and high SNR=1.5, increasing PR does not necessarily decrease test errors. For example, test errors increase for PR = 20%. This increase does not happen at low SNR=0.5, where test errors monotonically decrease. However, the decrease is very small from PR=10% to PR=20%. These observations indicate that adding more redundant (correlated) features may eventually degrade classification accuracy.

Column 5 in table 3.2 tabulates the bias of minimal CV errors as estimates of test errors. It clearly shows that minimal CV errors underestimate test errors. The bias tends to be larger at low SNR=0.5 (around 5%) and smaller at median and high SNRs (SNR=1 and

SNR=1.5, around 3%). Tibshirani and Tibshirani (2009) reported similar results, where the bias in the "signal" case was smaller than "no signal" case. In most cases, adjusted CV errors are less biased than minimal CV errors (column 5 and 6 in table 3.2) and tend to be conservative estimates of test errors. The most significant improvement is at SNR=1.5 where adjusted CV errors are the least biased estimates. Overall, bias adjustments are reasonably good.

Table 3.2: Misclassification errors of LREN on 1000 simulated datasets, column 1 is the PR, column 2 is the minimal CV errors, column 3 is the errors after bias adjustment, column 4 is the test errors on an independent dataset, column 5 is the bias of the minimal CV errors and column 6 is the bias of adjusted CV errors

PR	**Min error**	**Adjusted error**	**Test error**	**Min-Test**	**Adjust-Test**
SNR = 0.5					
1%	0.4124 (0.0029)	0.4903 (0.0033)	0.4704 (0.0009)	-0.0580 (0.0028)	0.0199 (0.0032)
5%	0.3858 (0.0029)	0.4602 (0.0034)	0.4366 (0.0013)	-0.0508 (0.0027)	0.0236 (0.0031)
10%	0.3736 (0.0029)	0.4443 (0.0034)	0.4175 (0.0013)	-0.0439 (0.0027)	0.0268 (0.0032)
20%	0.3625 (0.0028)	0.4330 (0.0033)	0.4125 (0.0012)	-0.0500 (0.0026)	0.0205 (0.0030)
SNR = 1					
1%	0.2917 (0.0030)	0.3594 (0.0036)	0.3263 (0.0015)	-0.0345 (0.0027)	0.0332 (0.0032)
5%	0.2446 (0.0027)	0.2985 (0.0033)	0.2764 (0.0011)	-0.0318 (0.0024)	0.0221 (0.0030)
10%	0.2337 (0.0023)	0.2833 (0.0028)	0.2584 (0.0007)	-0.0246 (0.0022)	0.0250 (0.0027)
20%	0.2248 (0.0022)	0.2722 (0.0027)	0.2631 (0.0006)	-0.0383 (0.0022)	0.0091 (0.0026)
SNR = 1.5					
1%	0.1583 (0.0021)	0.2008 (0.0026)	0.1919 (0.0007)	-0.0336 (0.0020)	0.0089 (0.0025)
5%	0.1338 (0.0018)	0.1648 (0.0023)	0.1612 (0.0005)	-0.0274 (0.0018)	0.0036 (0.0022)
10%	0.1298 (0.0016)	0.1582 (0.0020)	0.1540 (0.0004)	-0.0242 (0.0016)	0.0042 (0.0020)
20%	0.1271 (0.0016)	0.1530 (0.0020)	0.1584 (0.0003)	-0.0313 (0.0016)	-0.0053 (0.0020)

Feature Selection Performance

Figure 3.1 shows the feature selection performance criteria at all combinations of SNRs and PRs (black line for SNR=0.5, blue line for SNR=1 and red line for SNR=1.5). I report results on false positive rates (FPRs), sensitivity rates and similarity rates.

The left column of figure 3.1 presents FPR curves. Not surprisingly, higher SNR leads to lower FPR because it is easier to distinguish informative features from noises. Generally, higher PR also decreases FPR because more discriminative information and less noisy features are present. The caveat is at low SNR=0.5, where the false positive rate curve

increases a little bit at median PRs (5% and 10%). It requires more datasets to confirm the upward trend. Overall, the FPR from LREN depends on both SNR and PR. On real fMRI data where there is no such information, it is difficult to assess the empirical FPRs.

The middle column of figure 3.1 shows the sensitivity rate curves. At each PR level, higher SNR helps retrieve more informative features. This is expected because informative features are more separated from noisy features under high SNRs. However, within each SNR, higher PR consistently leads to lower sensitivity rates and this effect is not due to lower FPRs. For example, at SNR=0.5 where FPR increases or at SNR=1 where FPR remains the same under PR=1% and 5%, the sensitivity rates still sharply drop. This illustrates the "bet on sparsity" principle (Hastie et al., 2009) in high dimensional learning problems, in which a sparser model has better generalization performance. Therefore, with an increasing PR, a sparse model outputs an decreasing selection percentage. The interesting effects of PR were not documented in Ryali et al. (2010). This observation strongly indicates that a good prediction model does not necessarily leads to good feature selections. It calls for cautions of using LREN for a region of interest analysis of fMRI data where correlated relevant features are even more prevalent.

The right column of figure 3.1 illustrates the similarity rate curves. Similarity rate measures feature selection stability. If a feature selection is stable, selections on any two i.i.d. datasets tend to be similar. Therefore, the similarity rate depends on both the FPR and the sensitivity rate. Here, similarity rate curves peak at PR=5%, reflecting a tradeoff between the false positive rate and sensitivity rate. For example, though PR=1% has the highest sensitivity, the number of false positives is much larger than the number of selected relevant features. At a higher PR, the decrease in FPR is insufficient to compensate the decrease in the sensitivity rate and the similarity rate is lower. High SNR=1.5 displays a much larger drop in similarity rates beyond PR=5% because with low FPRs, similarity rates are mainly determined by the sensitivity rates.

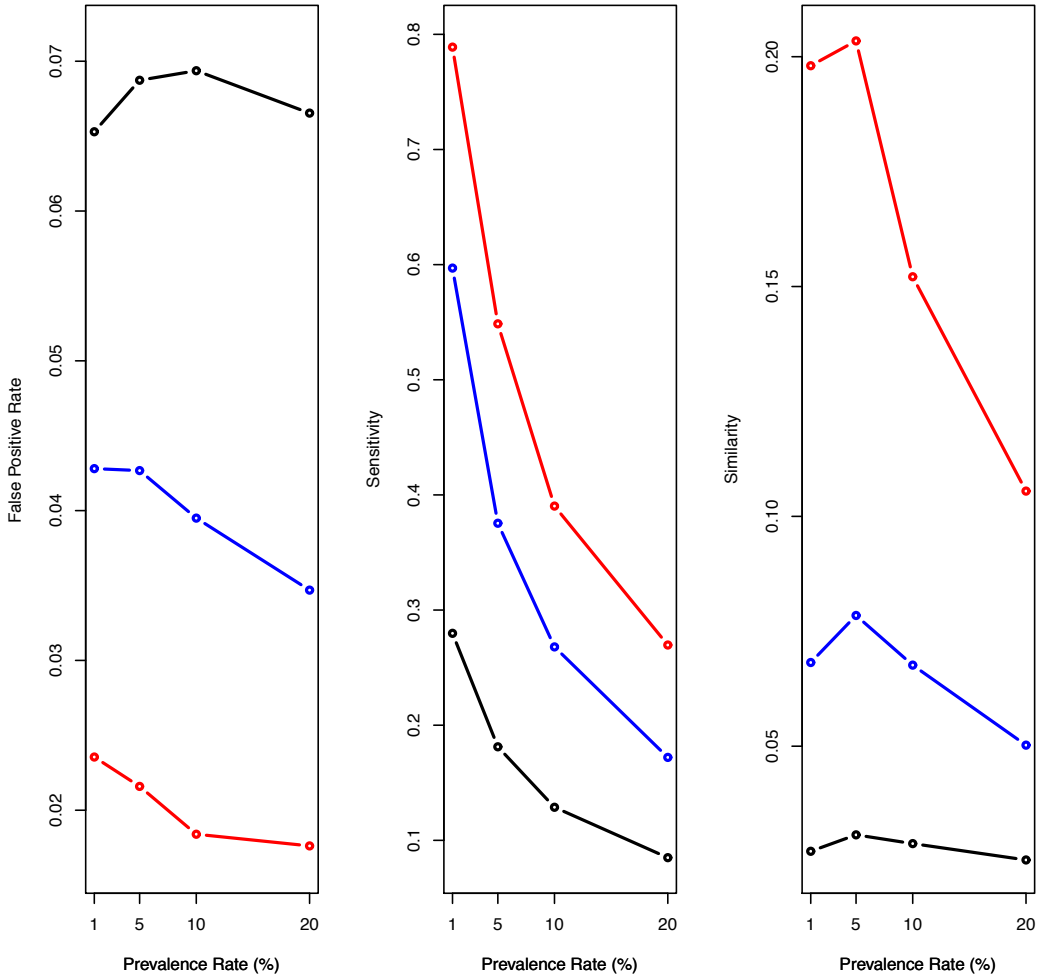

Figure 3.1: Feature selection performance of LREN on 1000 simulated datasets (left column: false positive rate; middle column: sensitivity rate; right column: similarity rate; black line: SNR=0.5, blue line: SNR=1, red line: SNR=1.5)

Results of RSMLREN and Comparisons with LREN

Because of the high computation load required by RSMLREN, I evaluate RSMLREN with 10 simulated datasets and compare it with LREN using the same datasets. I first present results in classification and feature selection and then results in FPR control. I also discuss the effect of the number of subspaces on all the above criteria.

Misclassification Errors

Table 3.3 shows misclassification errors at all combinations of PRs and SNRs for both LREN and RSMLREN on 10 simulated datasets. It is commonly observed that increasing SNR significantly decreases misclassification errors, for example, test errors decrease from around 45% at SNR=0.5 to around 15% at SNR=1.5. Because all methods are evaluated with 10 datasets, there are much more variance in errors estimates for LREN compared with its results on 1000 datasets. For example, here, test errors of LREN are larger than those on 1000 datasets while minimal CV errors tend to be smaller. Therefore, I focus mainly on comparing LREN with RSMLREN because they are all evaluated on the same datasets.

Column 3 of table 3.3 tabulates minimal CV errors in LREN and majority voting errors in RSMLREN. All estimates have downward bias (column 6) and the bias is much more severe for RSMLREN due to the amplifying effect of the majority voting. Comparing across subspace sizes, the amplifying effect is larger with a smaller subspace size. One possible explanation is that a smaller subspace size encourages more diversity among base classifiers, and the error from the majority voting is even smaller. Therefore, a bias adjustment is necessary for RSMLREN. Column 4 shows the adjusted errors and column 7 are their bias for test errors. Overall, adjusted errors are conservative and are pretty close to test errors except for some cases (for example, at SNR=1, PR=1% and 5% for RSMLREN10) possibly because of the data sampling variance. Increasing SNR similarly helps reduce bias in both minimal errors and adjusted errors. Comparing among three subspace sizes, bias adjustments perform roughly the same. Column 5 shows the test errors on an independent test dataset. Three RSMLREN methods are able to achieve slightly lower misclassification

errors than LREN in almost all cases due to the advantage of an ensemble classifier. Among three RSMLREN methods with different subspace sizes, RSMLREN10 has the lowest test errors probably because smaller subspaces share fewer common features, and the diversity among base classifiers is higher.

Table 3.3: Misclassification errors of LREN and RSMLREN on 10 simulated datasets

Method	PR	Min error	Adjusted error	Test error	Min-Test	Adjust-Test
SNR = 0.5						
LREN	1%	0.3800 (0.0223)	0.4600 (0.0281)	0.4740 (0.0100)	-0.0940 (0.0189)	-0.0140 (0.0230)
	5%	0.3660 (0.0298)	0.4340 (0.0340)	0.4435 (0.0136)	-0.0775 (0.0276)	-0.0095 (0.0312)
	10%	0.3340 (0.0341)	0.3860 (0.0339)	0.4069 (0.0162)	-0.0729 (0.0262)	-0.0209 (0.0262)
	20%	0.3540 (0.0246)	0.4260 (0.0268)	0.4395 (0.0131)	-0.0855 (0.0166)	-0.0135 (0.0191)
RSMLREN10	1%	0.3020 (0.0241)	0.4360 (0.0217)	0.4665 (0.0088)	-0.1645 (0.0257)	-0.0305 (0.0234)
	5%	0.3220 (0.0324)	0.4460 (0.0297)	0.4252 (0.0149)	-0.1032 (0.0357)	0.0208 (0.0332)
	10%	0.2940 (0.0370)	0.3900 (0.0438)	0.3851 (0.0146)	-0.0911 (0.0398)	0.0049 (0.0462)
	20%	0.2820 (0.0280)	0.3920 (0.0381)	0.3895 (0.0092)	-0.1075 (0.0295)	0.0025 (0.0392)
RSMLREN25	1%	0.3400 (0.0229)	0.4480 (0.0291)	0.4712 (0.0086)	-0.1312 (0.0245)	-0.0232 (0.0303)
	5%	0.3520 (0.0332)	0.4680 (0.0405)	0.4337 (0.0148)	-0.0817 (0.0364)	0.0343 (0.0431)
	10%	0.3100 (0.0364)	0.4100 (0.0367)	0.3942 (0.0171)	-0.0842 (0.0402)	0.0158 (0.0405)
	20%	0.3120 (0.0330)	0.4200 (0.0497)	0.4009 (0.0110)	-0.0889 (0.0347)	0.0191 (0.0509)
RSMLREN50	1%	0.3680 (0.0265)	0.4820 (0.0318)	0.4727 (0.0095)	-0.1047 (0.0282)	0.0093 (0.0331)
	5%	0.3640 (0.0372)	0.4640 (0.0459)	0.4448 (0.0144)	-0.0808 (0.0399)	0.0192 (0.0481)
	10%	0.3240 (0.0338)	0.4240 (0.0360)	0.4013 (0.0176)	-0.0773 (0.0381)	0.0227 (0.0401)
	20%	0.3580 (0.0299)	0.4380 (0.0334)	0.4176 (0.0135)	-0.0596 (0.0328)	0.0204 (0.0360)
SNR = 1						
LREN	1%	0.2720 (0.0283)	0.3400 (0.0350)	0.3206 (0.0148)	-0.0486 (0.0247)	0.0194 (0.0316)
	5%	0.2280 (0.0310)	0.2700 (0.0365)	0.2678 (0.0061)	-0.0398 (0.0274)	0.0022 (0.0339)
	10%	0.2180 (0.0319)	0.2600 (0.0397)	0.2582 (0.0067)	-0.0402 (0.0269)	0.0018 (0.0347)
	20%	0.2300 (0.0264)	0.2960 (0.0332)	0.2686 (0.0073)	-0.0386 (0.0248)	0.0274 (0.0319)
RSMLREN10	1%	0.2320 (0.0265)	0.3560 (0.0405)	0.3091 (0.0103)	-0.0771 (0.0285)	0.0469 (0.0418)
	5%	0.2200 (0.0256)	0.3000 (0.0308)	0.2549 (0.0042)	-0.0349 (0.0260)	0.0451 (0.0311)
	10%	0.1860 (0.0329)	0.2320 (0.0427)	0.2424 (0.0040)	-0.0564 (0.0332)	-0.0104 (0.0429)
	20%	0.1880 (0.0172)	0.2340 (0.0186)	0.2482 (0.0007)	-0.0602 (0.0172)	-0.0142 (0.0186)
RSMLREN25	1%	0.2420 (0.0214)	0.3300 (0.0283)	0.3109 (0.0128)	-0.0689 (0.0249)	0.0191 (0.0311)
	5%	0.2400 (0.0307)	0.3220 (0.0372)	0.2626 (0.0068)	-0.0226 (0.0314)	0.0594 (0.0378)
	10%	0.1940 (0.0303)	0.2440 (0.0345)	0.2455 (0.0050)	-0.0515 (0.0307)	-0.0015 (0.0349)
	20%	0.1940 (0.0200)	0.2340 (0.0251)	0.2508 (0.0012)	-0.0568 (0.0201)	-0.0168 (0.0252)
RSMLREN50	1%	0.2560 (0.0231)	0.3260 (0.0253)	0.3160 (0.0123)	-0.0600 (0.0261)	0.0100 (0.0281)
	5%	0.2560 (0.0383)	0.3300 (0.0478)	0.2697 (0.0081)	-0.0137 (0.0391)	0.0603 (0.0485)
	10%	0.1960 (0.0351)	0.2420 (0.0402)	0.2519 (0.0077)	-0.0559 (0.0360)	-0.0099 (0.0409)
	20%	0.2080 (0.0209)	0.2500 (0.0255)	0.2525 (0.0022)	-0.0445 (0.0210)	-0.0025 (0.0256)
SNR = 1.5						
LREN	1%	0.1480 (0.0177)	0.2040 (0.0210)	0.1878 (0.0048)	-0.0398 (0.0174)	0.0162 (0.0208)
	5%	0.1180 (0.0185)	0.1500 (0.0229)	0.1592 (0.0035)	-0.0412 (0.0180)	-0.0092 (0.0225)
	10%	0.1100 (0.0220)	0.1320 (0.0280)	0.1513 (0.0023)	-0.0413 (0.0210)	-0.0193 (0.0268)
	20%	0.1480 (0.0248)	0.1820 (0.0310)	0.1588 (0.0038)	-0.0108 (0.0240)	0.0232 (0.0303)
RSMLREN10	1%	0.1180 (0.0153)	0.1800 (0.0262)	0.1793 (0.0021)	-0.0613 (0.0155)	0.0007 (0.0262)
	5%	0.1260 (0.0263)	0.1680 (0.0380)	0.1501 (0.0005)	-0.0241 (0.0263)	0.0179 (0.0380)
	10%	0.1120 (0.0255)	0.1560 (0.0360)	0.1452 (0.0005)	-0.0332 (0.0255)	0.0108 (0.0360)
	20%	0.1220 (0.0159)	0.1360 (0.0178)	0.1509 (0.0005)	-0.0289 (0.0159)	-0.0149 (0.0178)
RSMLREN25	1%	0.1200 (0.0149)	0.1720 (0.0270)	0.1777 (0.0021)	-0.0577 (0.0150)	-0.0057 (0.0271)
	5%	0.1260 (0.0244)	0.1660 (0.0331)	0.1515 (0.0011)	-0.0255 (0.0244)	0.0145 (0.0331)
	10%	0.1040 (0.0253)	0.1340 (0.0333)	0.1463 (0.0008)	-0.0423 (0.0253)	-0.0123 (0.0333)
	20%	0.1260 (0.0158)	0.1400 (0.0176)	0.1512 (0.0006)	-0.0252 (0.0158)	-0.0112 (0.0176)
RSMLREN50	1%	0.1340 (0.0179)	0.1880 (0.0264)	0.1804 (0.0035)	-0.0464 (0.0182)	0.0076 (0.0266)
	5%	0.1360 (0.0275)	0.1780 (0.0363)	0.1538 (0.0020)	-0.0178 (0.0275)	0.0242 (0.0364)
	10%	0.0960 (0.0265)	0.1120 (0.0327)	0.1491 (0.0017)	-0.0531 (0.0265)	-0.0371 (0.0327)
	20%	0.1340 (0.0184)	0.1640 (0.0219)	0.1525 (0.0009)	-0.0185 (0.0184)	0.0115 (0.0219)

Since the majority voting scheme depends on the number of subspaces used, I further examine the effect of the number of subspaces on the majority voting error curve, adjusted error and test error curve, respectively. As three RSMLREN methods have similar error curves across SNRs, for illustrative purposes, I show curves from RSMLREN10 at SNR=1 in figure 3.2. Across four PRs, test error curves are the most stable because of the large

sample size. The majority voting error curves are less stable, but are much stabler than the adjusted error curves. Adjusted error curves are the most variant, and fluctuations of the adjusted error curves mimic the majority voting error curves with larger variations due to the additional variance in the bias estimation. The variance of adjusted errors does not seem to decrease when more subspaces are used. This is possibly due to the limited number of folds (5 folds) used to estimate the bias. However, its variance is quite stable beyond 500 subspaces. Both the majority voting error curve and test error curve converge fast and stay relatively constant beyond 500 subspaces. Another notable observation is that the majority voting curves seem to have dips at the first several subspaces. Recall that instead of randomly sampling features, I partition the whole feature space into mutually exclusive subspaces. At the first several partitions, the diversity among subspaces may be higher, but the number of subspaces is not large enough to significantly drive down the error. The dips strikes the balance between the diversity and the number of subspaces.

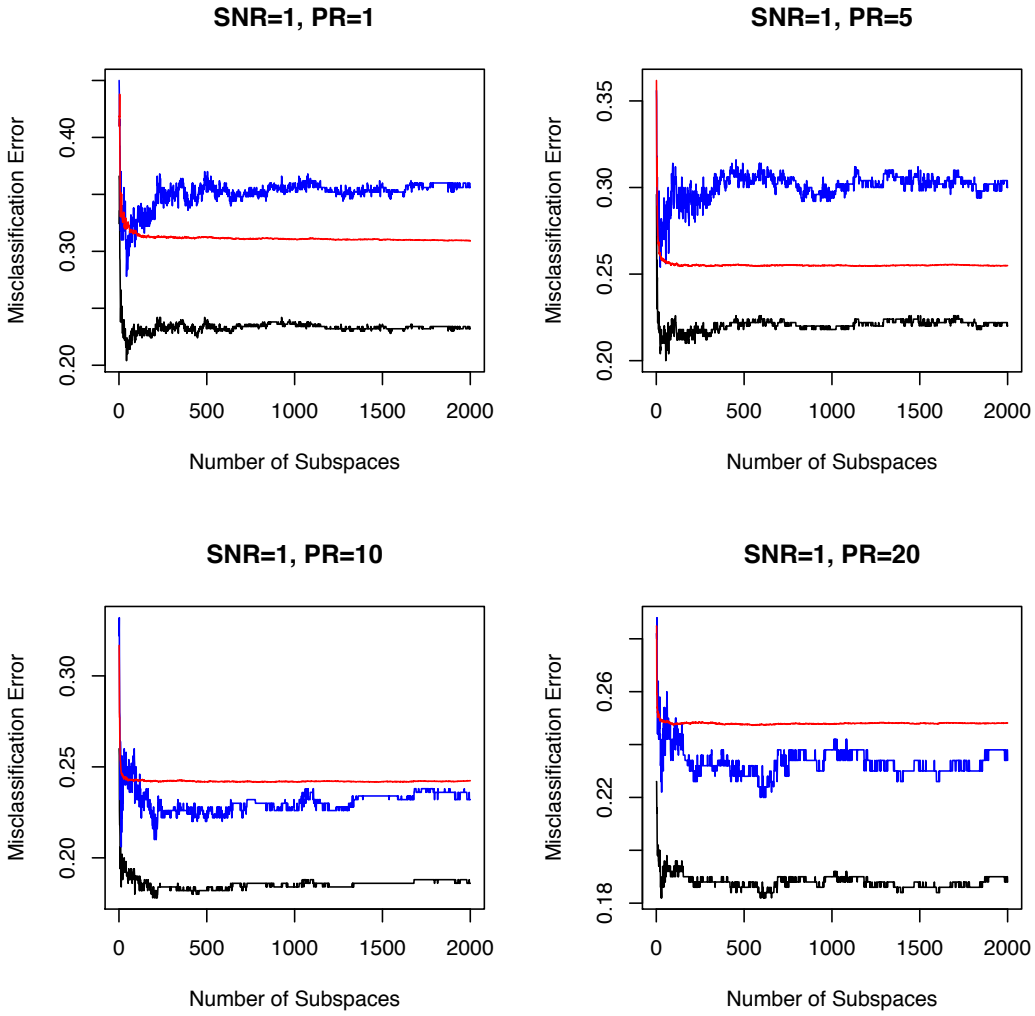

Figure 3.2: Effects of the number of subspaces on misclassification errors (For RSMLREN10 at SNR = 1, the majority voting error curve: black line; the adjusted error curve: blue line; the test error curve: red line

Feature Selection Performance

Using the same 10 datasets, I compare RSMLREN with LREN in terms of FPR, sensitivity rate and similarity rate. I also discuss the effect of the number of iterations on the sensitivity rate.

Figure 3.3 shows the FPR curves for four methods after 200 iterations. False positive rates of three RSMLREN methods are matched to LREN such that all methods select the same number of irrelevant features. Similar to the results of LREN on 1000 datasets, false positive rates decrease with increasing SNR. As less datasets are used, the effect of PR tends to be irregular. Figure 3.4 shows sensitivity rates for four methods by first fixing

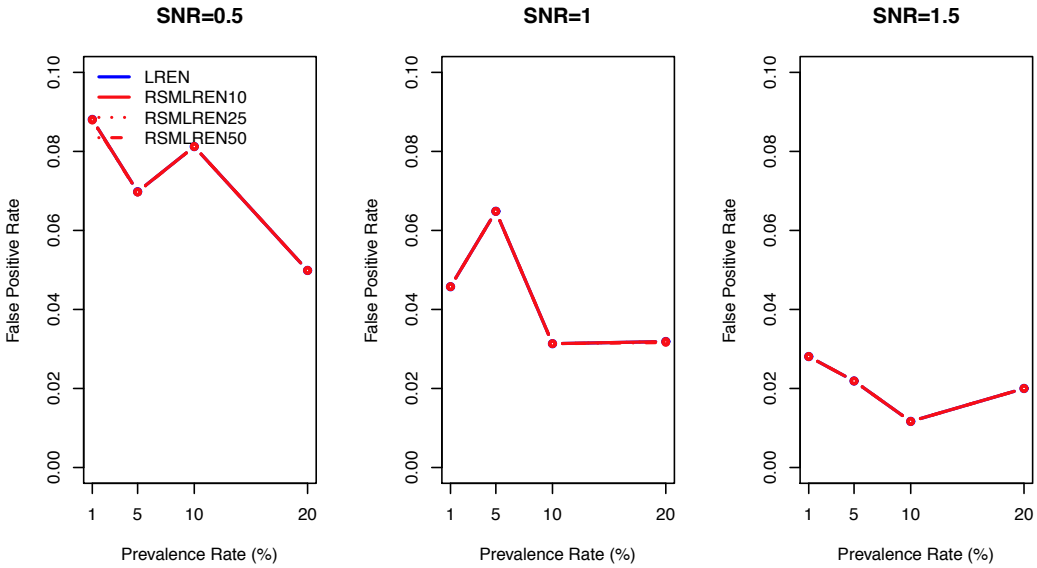

Figure 3.3: False positive rates in feature selection after 200 iterations (the four FPR curves are matched)

the number of false positives on the same level. Similar to observations in the benchmark method of LREN, higher SNR helps select more relevant features and higher PR deteriorates selection power. Three RSMLREN methods with different subspace sizes perform better than LREN. Improvements are smaller at lower PRs and larger at higher PRs. The success of RSMLREN can be explained in two ways. On one hand, the dimension of each subspace

is much lower and the sample-to-feature ratio is higher. Feature selection is thus more accurate. For example, when the number of features is less than the sample size in each subspace, it is more likely to select all relevant features in each subspace. On the other hand, those unselected informative features in LREN can have dominant discriminative abilities in some subspaces. Their importance get "boosted" in the RSM framework. The improvement is higher when SNR is higher because unselected features in LREN still carry "good" discriminative information and can be more distinguished from noisy features in RSMLREN. Among three different subspace sizes examined, RSMLREN10 performs the best. At each combination of SNRs and PRs, the sensitivity rate decreases as the subspace size increases.

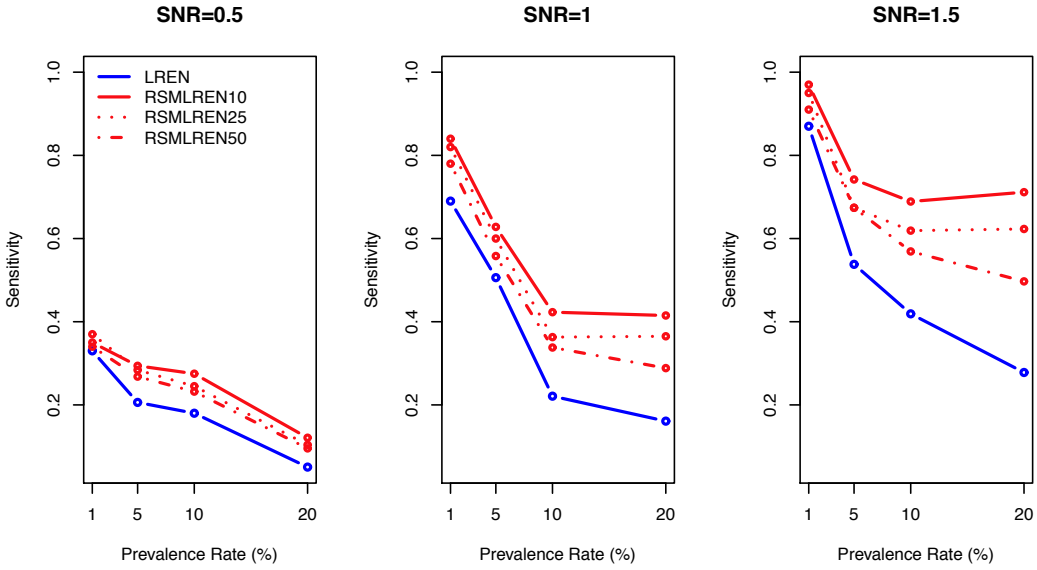

Figure 3.4: Feature selection sensitivity rates after 200 iterations

To investigate effects of the number of iterations on feature selection sensitivity, I rank the features and fix the false positive rates at FPR=0.01 for all three RSMLREN methods with different subspace sizes. Figure 3.5, 3.6, 3.7 present the results (black line is for RSMLREN10, blue line for RSMLREN25 and red line for RSMLREN50). Overall, more

iterations lead to stabler and better sensitivity rates. For the median SNR=1 and high SNR=1.5, sensitivity rate curves start to converge after 50 iterations. For the low SNR=0.5, curves are more variant especially for low PRs and thus more iterations are preferred in these cases. RSMLREN10 performs consistently better than the other two, most remarkably at SNR=1.5 and PR=20%. However, with regards to the curve stability, there is no much difference among three subspace sizes.

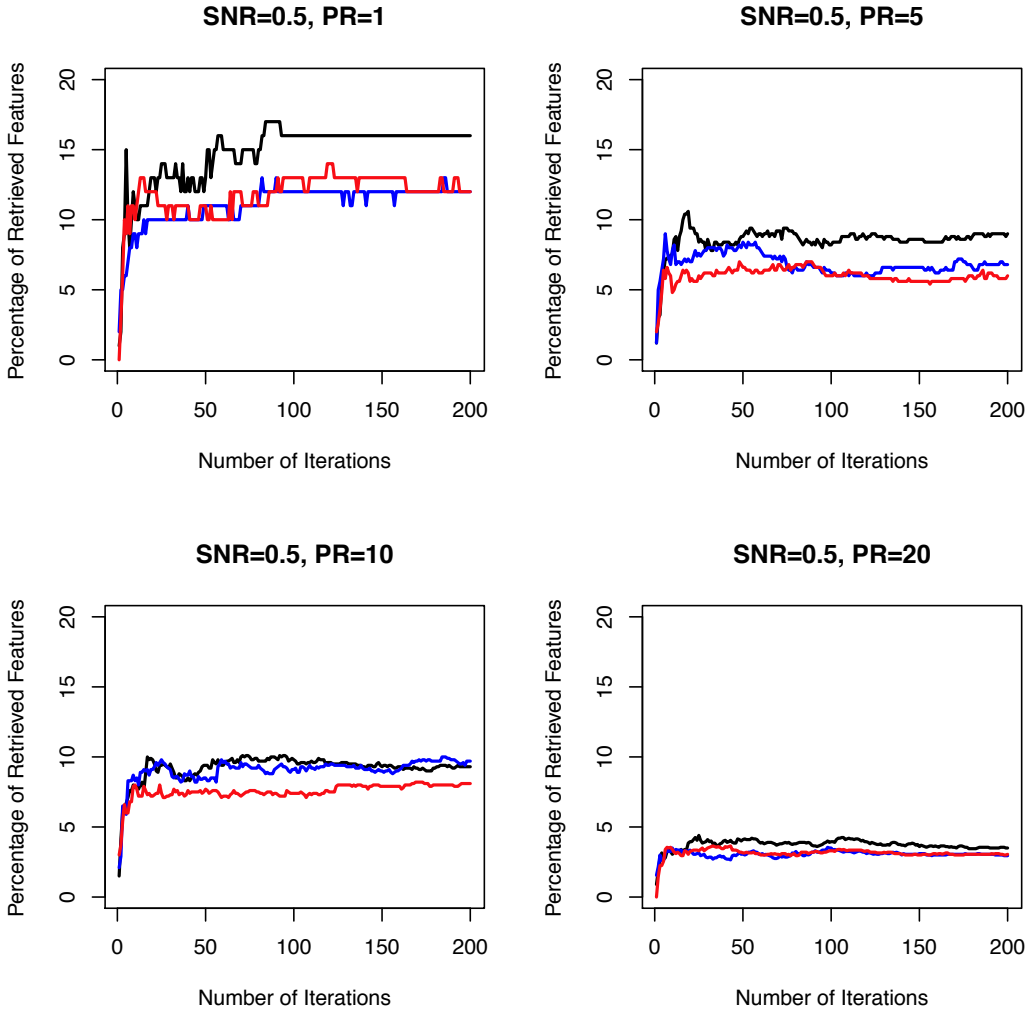

Figure 3.5: Effects of the number iterations on sensitivity rates at SNR=0.5 (black line is for RSMLREN10, blue line for RSMLREN25 and red line for RSMLREN50)

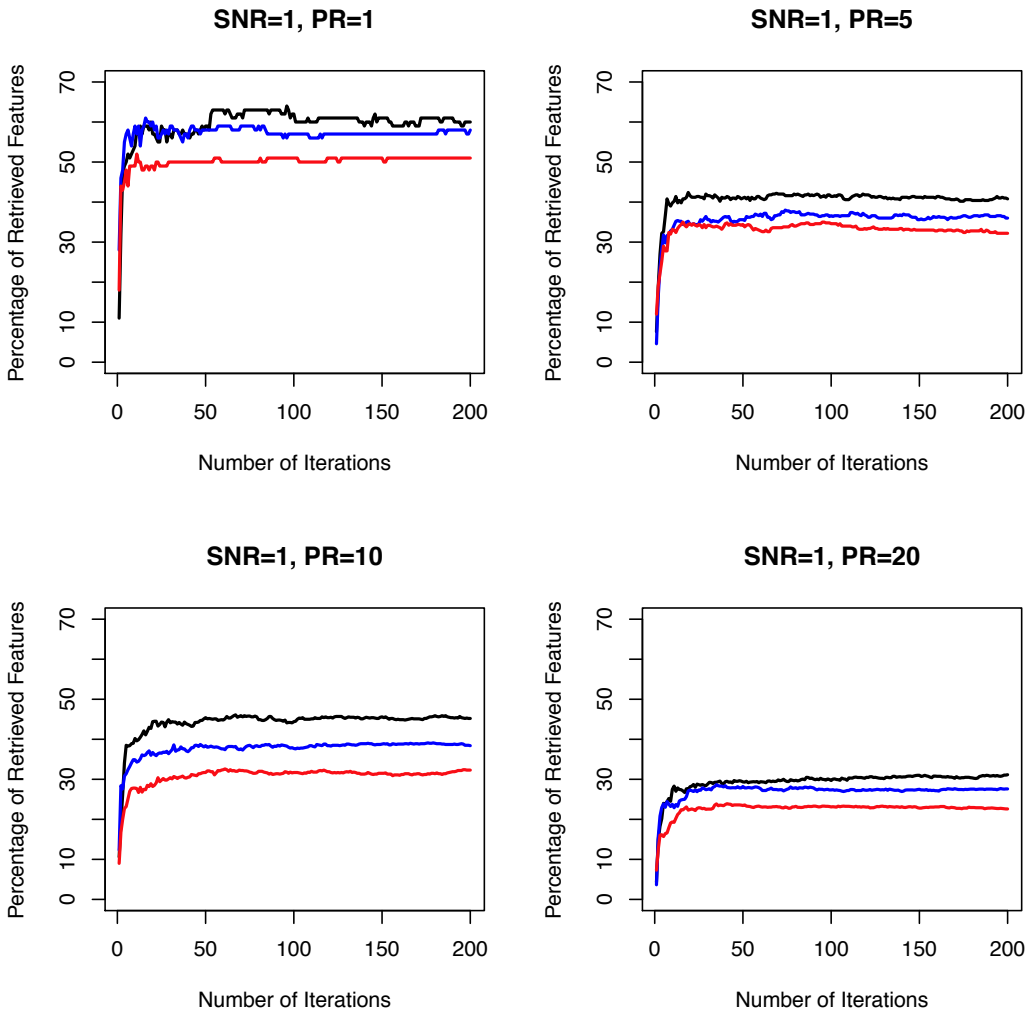

Figure 3.6: Effects of the number iterations on sensitivity rates at SNR=1 (black line is for RSMLREN10, blue line for RSMLREN25 and red line for RSMLREN50)

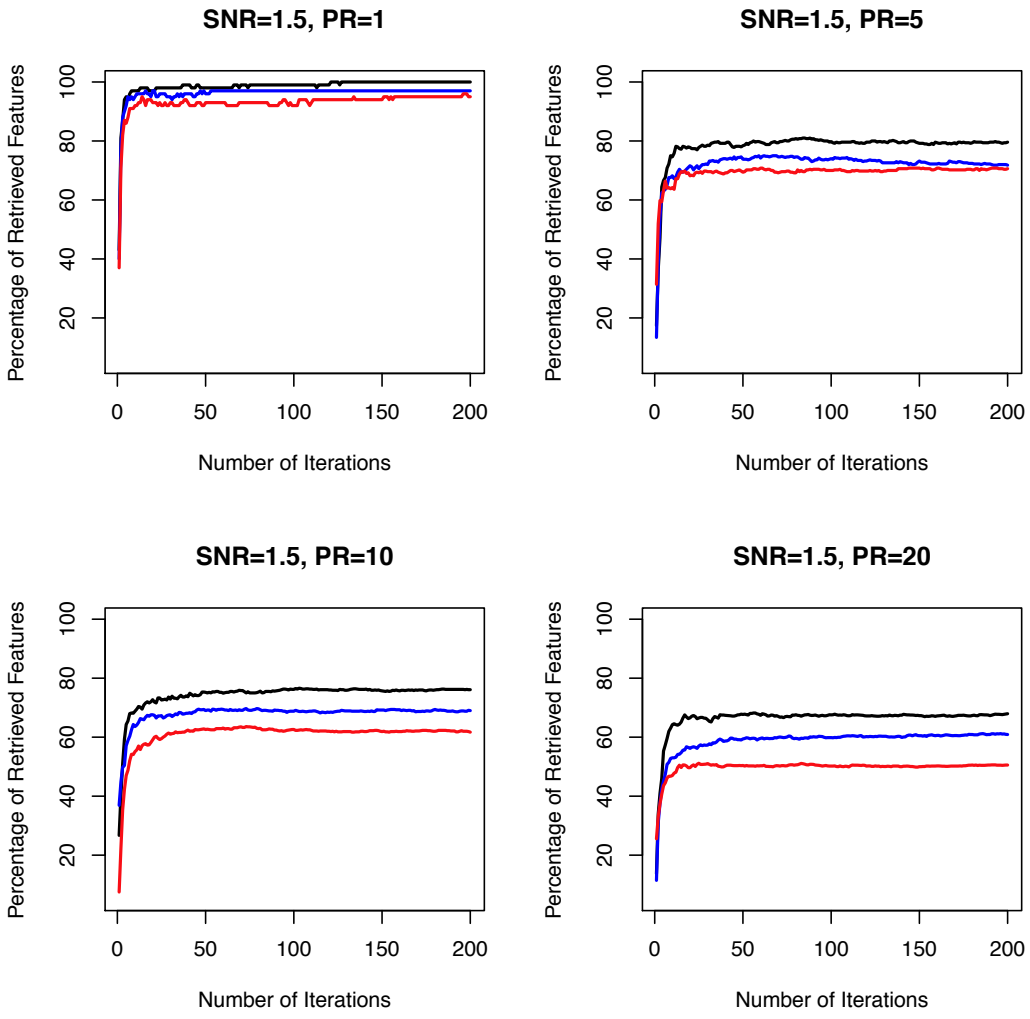

Figure 3.7: Effects of the number iterations on sensitivity rates at SNR=1.5 (black line is for RSMLREN10, blue line for RSMLREN25 and red line for RSMLREN50)

Figure 3.8 shows the averaged pairwise selection similarity rates on 10 simulated datasets. As FPRs are matched onto the same levels, difference in sensitivity rates critically determines the difference in similarity rates. Therefore, RSMLREN methods are stabler than LREN. Large improvements are observed at the median and high SNR, especially when PR is high. Among all methods, RSMLREN10 performs the best because it has the highest sensitivity rates.

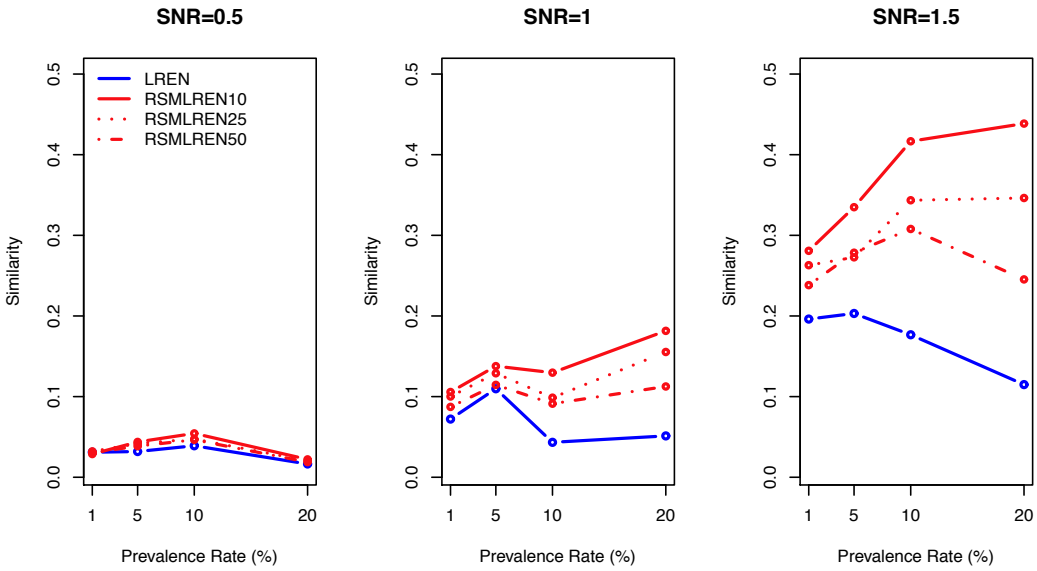

Figure 3.8: Similarity rates in feature selection (after 200 iterations)

Control of False Positive Rates

The threshold to control FPR is derived from scores of permuted real features. I set three target FPR levels to examine whether RSMLREN is able to provide a good control over FPRs (FPR=1%, 0.5% and 0.1%). Results after 200 iterations are shown in table 3.4 for three RSMLREN methods. For each target FPR level, empirical FPRs seem to stay relatively the same across SNRs. However, higher PRs seem to result in more conservative empirical FPRs. For example, empirical FPRs at PR=10% and 20% seem to be lower than PR=1% and 5%, especially at the lowest FPR=0.1%. Note that at higher PRs, the

number of permuted features is larger than the number of real noisy features. It could be possible that by chance, the threshold from permuted variable scores may be too stringent for real noisy feature scores. For example, at PR=20% and FPR=0.1%, the threshold is the highest score among artificial features and much higher than the highest score of real noisy features. Using more simulated datasets may eliminate the observed conservativeness of empirical FPRs at high PRs. In contrast to results on sensitivity rates, three RSMLREN methods performance roughly the same in controlling FPRs. However, this observation needs to be confirmed with more datasets. Overall, the random probe method does help control the false positive rates and the target rates generally fall within the one standard deviation bounds.

Similar to evaluations on the sensitivity rates, I also examine the effects of partition iterations on empirical FPR curves. For illustrative purpose, I present curves for RSMLREN10 at SNR=1 (all other curves at different SNRs with two other subspace sizes are similar). In figure 3.9, dotted lines are the target FPR levels (FPR=1%, 0.5%, 0.1%). Black lines are the empirical FPR curves for FPR=1%, blue lines for FPR=0.5% and red lines for FPR=0.1%. Gray lines are the one standard deviation bounds of the empirical FPRs. Generally, with more iterations, curves tend to be stabler, except for PR=5% where the FPR=1% curve arises. However, the curve begins to drop at the last several iterations and more iterations are required to confirm the trend. In most cases, after 100 iterations, the target FPRs fall either within one standard deviation bounds or above the upper bound.

Table 3.4: Empirical false positive rates (after 200 iterations)

Method	PR	FPR = 1%	FPR = 0.5%	FPR = 0.1%
SNR = 0.5				
RSMLREN10	1%	0.0109 (0.0009)	0.0051 (0.0009)	0.0014 (0.0006)
	5%	0.0120 (0.0009)	0.0055 (0.0007)	0.0014 (0.0005)
	10%	0.0116 (0.0016)	0.0056 (0.0008)	0.0010 (0.0007)
	20%	0.0098 (0.0017)	0.0056 (0.0012)	0.0010 (0.0005)
RSMLREN25	1%	0.0117 (0.0012)	0.0060 (0.0010)	0.0018 (0.0006)
	5%	0.0124 (0.0008)	0.0059 (0.0007)	0.0012 (0.0004)
	10%	0.0102 (0.0012)	0.0057 (0.0007)	0.0006 (0.0003)
	20%	0.0103 (0.0016)	0.0053 (0.0013)	0.0010 (0.0003)
RSMLREN50	1%	0.0119 (0.0011)	0.0054 (0.0009)	0.0015 (0.0007)
	5%	0.0111 (0.0007)	0.0059 (0.0005)	0.0014 (0.0005)
	10%	0.0106 (0.0012)	0.0057 (0.0010)	0.0006 (0.0003)
	20%	0.0099 (0.0012)	0.0051 (0.0011)	0.0010 (0.0004)
SNR = 1				
RSMLREN10	1%	0.0107 (0.0011)	0.0072 (0.0014)	0.0010 (0.0006)
	5%	0.0122 (0.0009)	0.0054 (0.0007)	0.0012 (0.0005)
	10%	0.0110 (0.0012)	0.0060 (0.0011)	0.0003 (0.0002)
	20%	0.0086 (0.0011)	0.0046 (0.0010)	0.0004 (0.0002)
RSMLREN25	1%	0.0111 (0.0012)	0.0068 (0.0011)	0.0016 (0.0007)
	5%	0.0114 (0.0007)	0.0058 (0.0012)	0.0009 (0.0004)
	10%	0.0100 (0.0012)	0.0054 (0.0010)	0.0002 (0.0001)
	20%	0.0099 (0.0012)	0.0043 (0.0006)	0.0009 (0.0004)
RSMLREN50	1%	0.0110 (0.0013)	0.0067 (0.0014)	0.0013 (0.0006)
	5%	0.0112 (0.0009)	0.0054 (0.0006)	0.0007 (0.0004)
	10%	0.0116 (0.0015)	0.0062 (0.0010)	0.0003 (0.0002)
	20%	0.0095 (0.0011)	0.0045 (0.0010)	0.0008 (0.0003)
SNR = 1.5				
RSMLREN10	1%	0.0104 (0.0015)	0.0068 (0.0016)	0.0013 (0.0006)
	5%	0.0111 (0.0010)	0.0066 (0.0008)	0.0018 (0.0008)
	10%	0.0106 (0.0010)	0.0050 (0.0011)	0.0008 (0.0004)
	20%	0.0080 (0.0010)	0.0049 (0.0010)	0.0005 (0.0003)
RSMLREN25	1%	0.0101 (0.0011)	0.0056 (0.0012)	0.0013 (0.0006)
	5%	0.0116 (0.0010)	0.0076 (0.0009)	0.0014 (0.0005)
	10%	0.0101 (0.0009)	0.0053 (0.0010)	0.0004 (0.0002)
	20%	0.0085 (0.0011)	0.0049 (0.00106)	0.0005 (0.0003)
RSMLREN50	1%	0.0103 (0.0015)	0.0063 (0.0013)	0.0011 (0.0005)
	5%	0.0112 (0.0010)	0.0057 (0.0007)	0.0008 (0.0004)
	10%	0.0104 (0.0010)	0.0048 (0.0008)	0.0009 (0.0003)
	20%	0.0085 (0.0017)	0.0044 (0.0008)	0.0008 (0.0003)

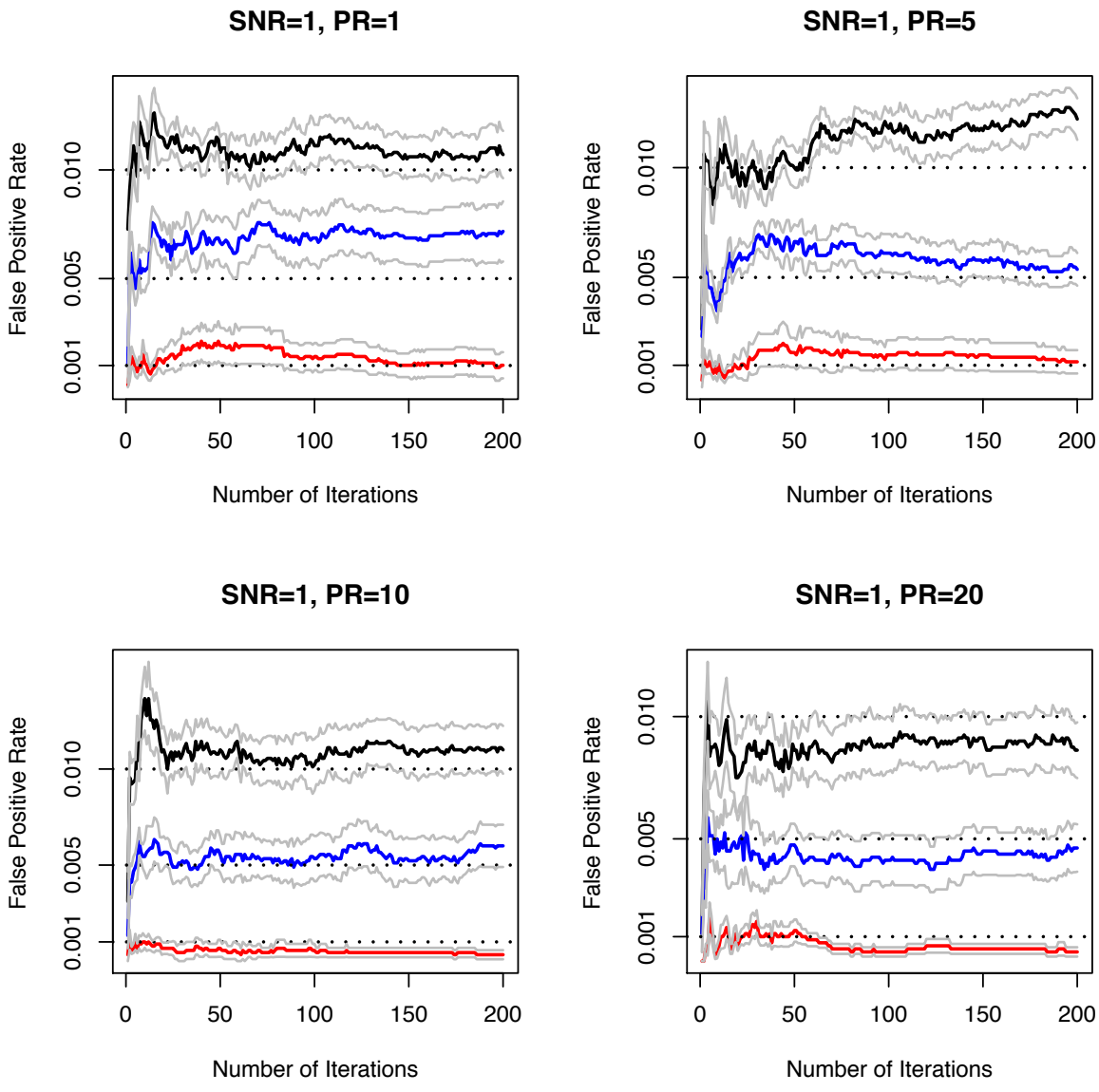

Figure 3.9: Effects of iterations on false positive rate curves at SNR = 1. Black line for FPR = 0.01, blue line for FPR = 0.005, red line for FPR = 0.001.

3.4.2 Second Simulation Study

The main goal of the second simulation study is to evaluate whether the feature scoring method defined in equation (2.10) can successfully rank informative features with varied SNRs. I also compare RSMLREN with LREN in feature selection sensitivity and similarity. Additionally, I evaluate the performance of RSMLREN in false positive rate control. For all evaluations, I use 200 iterations to compute feature scores. I do not present misclassification errors here because very high SNRs in this study drive error rates to exactly 0 or very close to 0.

Feature Ranking

For RSMLREN, I compute a score for each informative region (containing 10 features) by averaging feature scores within the region. A total 20 scores are obtained corresponding to 20 increasing SNR levels (from 0.1 to 2 with an increment of 0.1). For LREN, I compute feature scores in two different ways. The first approach is to average feature coefficients within each informative region. The second approach is to compute the percentage of selected features in each informative region. A Spearman rank correlation coefficient is computed for both RSMLREN and LREN, comparing region scores with their true ranks. Table 3.5 presents the average correlation coefficient on 10 simulated datasets. LREN uses the original feature space and LREN2000 uses the augmented feature space. Thus LREN2000 is comparable to RSMLRENs because they all operate on the same augmented feature space. For both LREN and LREN2000, the rank coefficients within the parenthesis are computed using regions scores from the second approach. The values from two approaches happen to be the same. The rank correlation coefficient of RSMLREN decreases with an increasing subspace size, especially beyond RSMLREN50. It is expected to converge to the value computed from the second approach for LREN2000. However, the coefficients of RSMLREN seem to be quite robust against subspace sizes. For example, when the subspace size increases from 10% to 50%, the coefficient stays roughly the same. It is an interesting topic for future research to investigate this effect. Overall, RSMLREN provides a good ranking

for informative features.

Table 3.5: Feature ranking performance of LREN and RSMLREN, LREN1000 is the LREN on the original feature space; LREN2000 is the LREN on the augmented feature space; 7 subspace sizes are listed (from 99% to 10%); values in the parenthesis for LREN and LREN2000 are computed using the second approach

Method	**Spearman Rank Correlation Coefficient**
LREN	0.849 (0.849)
LREN2000	0.858 (0.858)
RSMLREN99	0.875
RSMLREN90	0.891
RSMLREN80	0.916
RSMLREN70	0.921
RSMLREN50	0.938
RSMLREN25	0.935
RSMLREN10	0.937

Feature Selection

All comparisons are made by first matching the FPRs of RSMLREN to those of LREN (the average FPR=0.0005). Table 3.6 presents the average selection sensitivity and similarity rates of RSMLREN and LREN. Similar to the first simulation study, three RSMLREN methods have much higher sensitivity in selecting informative features. The sensitivity rate decreases as a larger subspace is used, conforming to the results from the first simulation study. Furthermore, I decompose the overall sensitivity into varied SNR levels (20 levels). The results are shown in figure 3.10. Numbers in the figure denotes SNR levels (1: SNR=0.1; 20: SNR=2). Obviously, RSMLREN methods are more powerful for most of the SNR levels, especially for median SNRs. For the very low SNR (<0.5), RSMLRENs perform roughly the same as LREN. Since FPR is tiny, similarity rates critically depends on the sensitivity rates and thus reflect the strength in detecting informative features.

False Positive Rate Control

I also set three target FPRs (FPR = 0.01, 0.005, 0.001) as in the first simulation study. Table 3.7 summaries the empirical FPRs. Overall, the FPR control provided in RSMLREN

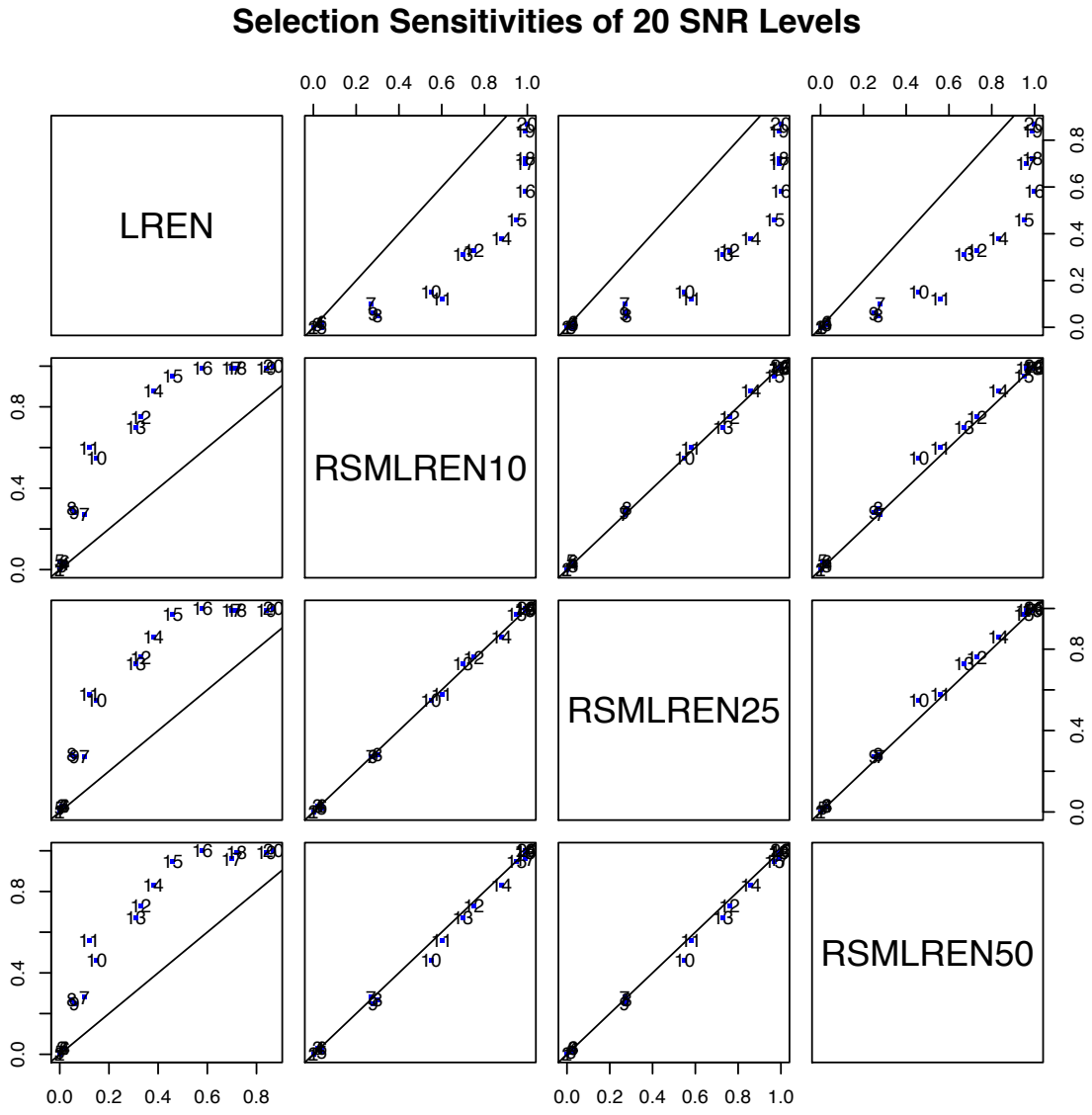

Figure 3.10: Selection sensitivity rates at 20 SNR levels, numbers correspond to SNR levels (1: SNR=0.1; 20: SNR=2)

Table 3.6: Selection sensitivity and similarity rate of LREN and RSMLREN with an matched average FPR=0.0005

Method	**Sensitivity Rate**	**Similarity Rate**
LREN	0.286	0.386
RSMLREN10	0.519	0.693
RSMLREN25	0.518	0.698
RSMLREN50	0.502	0.679

is quite successful with empirical FPRs very close to target levels. The conservativeness is also observed at the lowest FPR=0.1% since the PR in the second simulation is 20%.

Table 3.7: Empirical FPRs of RSMLREN

Method	**FPR=0.01**	**FPR=0.05**	**FPR=0.001**
RSMLREN10	0.0120 (0.0019)	0.0070 (0.0013)	0.0006 (0.0003)
RSMLREN25	0.0126 (0.0017)	0.0055 (0.0011)	0.0008 (0.0003)
RSMLREN50	0.0084 (0.0023)	0.0048 (0.0013)	0.0004 (0.0004)

3.4.3 Third Simulation Study

The main goal of the third simulation study is to examine whether RSMLREN can correctly infer features' discrimination directions. Here, the first 5 SNR levels are set from 0.5 to 1.5, with an increment of 0.25, representing an activation status. The second 5 SNRs are set from -0.5 to -1.5, with an decrement of 0.25, representing a deactivation status. I first evaluate the proposed approach in inferring discrimination directions. I then compare RSMLREN with LREN in terms of feature selection performance. I also examine the FPR control of RSMLREN. 200 iterations are used to compute feature scores.

Discrimination Direction

For each informative feature, I first compute the percentage of a positive weight sign compared to all cases in which the feature is selected. Then I average the percentages within a region (10 features with the same SNR). The first 5 SNRs are expected to have higher percentages. The next 5 SNRs are expected to have lower percentages. Table 3.8 presents the results for both RSMLRENs and LREN. Overall, all methods can correctly infer features' discrimination directions. In RSMLREN10, at the lowest SNR=0.5, there are some wrong signs due to sampling variance. With larger subspaces, there are less wrong signs because these features are not selected, being dominated by other stronger features.

Table 3.8: Feature's discrimination directions in LREN and RSMLREN

Method	**Percentage of a Positive Weight Sign**
LREN	{ 100.0 100.0 100.0 100.0 100.0 ; 0.0 0.0 0.0 0.0 0.0 }
RSMLREN10	{ 98.6 99.0 100.0 100.0 100.0 ; 2.4 0.0 0.0 0.0 0.0 }
RSMLREN25	{ 100.0 100.0 100.0 100.0 100.0 ; 0.0 0.0 0.0 0.0 0.0 }
RSMLREN50	{ 100.0 100.0 100.0 100.0 100.0 ; 0.0 0.0 0.0 0.0 0.0 }

Feature Selection

The FPRs of RSMLRENs are first matched to LREN (FPR=0.0159). Table 3.9 summaries the selection sensitivity and similarity rates for all methods. As expected, RSMLREN has a higher sensitivity rate than LREN. As the subspace size increases, the sensitivity rate

decreases slightly. I also examine the sensitivity rates at various SNR levels. Figure 3.11 presents the decomposed sensitivity rates (1: SNR=0.5, 5: SNR=1.5, -1: SNR=-0.5, -5: SNR=-1.5). Similar to figure 3.10 in the second simulation study, RSMLREN has higher rates at most SNR levels regardless of the discrimination direction. However, the improvement is not as high as in the second study possibly because SNR=2 in the second study has a stronger dominant role than SNR=1.5 in this study. It turns out that the FPR in this study is much higher (compared to FPR=0.0005 in the second study), leading to overall higher sensitivity rates of LREN.

Table 3.9: Selection sensitivity and similarity rates of LREN and RSMLREN

Method	**Selection Sensitivity Rate**	**Selection Similarity Rate**
LREN	0.503	0.355
RSMLREN10	0.604	0.457
RSMLREN25	0.580	0.425
RSMLREN50	0.564	0.389

False Positive Rate Control

Table 3.10 presents the averaged empirical FPRs at three targets. Overall, similar to previous studies, empirical FPRs are pretty close to the desired levels. However, no conservativeness is observed at the lowest FPR=0.1%. More datasets are needed to confirm this.

Table 3.10: Empirical control of FPRs in RSMLREN

Method	**FPR=0.01**	**FPR=0.05**	**FPR=0.001**
RSMLREN10	0.0101 (0.0013)	0.0061 (0.0010)	0.0010 (0.0004)
RSMLREN25	0.0112 (0.0013)	0.0061 (0.0011)	0.0013 (0.0005)
RSMLREN50	0.0106 (0.0016)	0.0063 (0.0012)	0.0012 (0.0006)

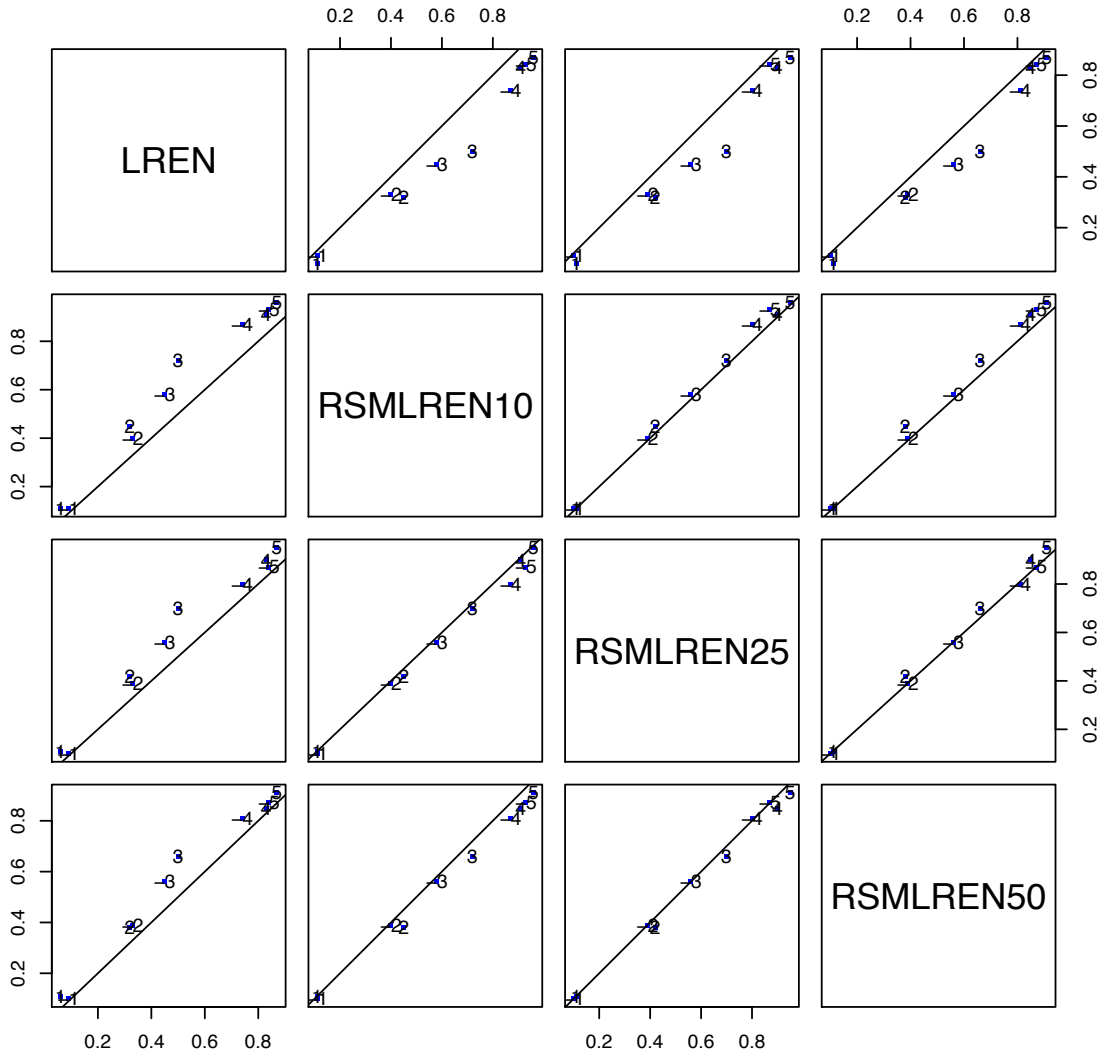

Figure 3.11: Selection sensitivity rates at 10 SNR levels with different discrimination directions, number 1 to number 5: SNR=0.5 to 1.5, number -1 to number -5: SNR=-0.5 to -1.5

3.5 Discussion of the Three Simulation Studies

The effects of SNR and PR on both classification and feature selection performance are clearly demonstrated in the first simulation. It is expected that high SNR leads to higher classification accuracy and sensitivity rates. However, PR's effect is more of interest for applications in real fMRI data. For LREN, it has a decreasing sensitivity rate curve with an increasing PR. This observation indicates that LREN may be suitable for a whole-brain analysis but not necessarily suitable for a region of interest analysis. RSMLREN is more robust than LREN against varied PRs, and thus it is more feasible to apply RSMLREN in both whole-brain and region of interest analysis.

RSMLREN has better performance in classification than LREN because of using the majority voting scheme and smaller subspaces in which base classifiers are built. As shown in the first study, RSMLREN has a slightly better test errors than LREN. In general, as argued in Breiman (2001), there are two factors that determine the classification accuracy of an ensemble classifier. The first element is the average classification strength of base classifiers, and the second element is the diversity among base classifiers. RSMLREN promotes the diversity by randomly sampling feature subsets. As more subspaces are sampled, the average diversity or correlation may converge to a fixed point. The value of the fixed point then depends on both the underlying data structure and the subspace size. For example, if a large percentage of features carry discriminative information, the diversity among base classifiers is expected to be low. For a larger subspace, two subspaces may contain more common features, and the average diversity is expected to be low. On the other hand, RSMLREN can also potentially improve the classification accuracy of base classifiers because of the lower feature dimension in each subspace. But if the subspace size is too small and the percentage of informative features is low, most of subspaces may contain only irrelevant features, and the average strength of base classifiers is lower. Therefore, a smaller subspace may induce higher diversity among base classifiers but may not have a higher average classification strength. In contrast, a larger subspace may induce a lower diversity but may have a higher average classification accuracy. The optimization of subspace size critically

depends on the underlying data structure. In this simulation study, the subspace size of 10% performs the best. However, as reported in Lai et al. (2006), the smallest subspace size they examined did not have the highest classification accuracy because of their different settings of informative features. They set pairs of informative features: one feature has a high SNR and the other has zero SNR but is highly correlated with the first one. A small subspace may not effectively capture features' dependency.

As for the feature selection, RSMLREN is more capable than LREN in selecting informative features because each informative feature can be a dominant feature in some subspaces. It is clearly illustrated in all three simulation studies which reflect different data structures. In the first study, informative features have the same expected SNR and are correlated. In the second and third studies, informative regions have varied SNRs. In all these cases, RSMLREN is able to "boost" those unselected features in LREN and thus informative features can be better distinguished from noisy features in RSMLREN. Similar to discussions in the classification, the optimal subspace size also depends on the underlying data structure. For example, a smaller subspace may not effectively capture features' dependency and loses power in feature selection, while the higher feature dimension of a larger subspace may make it harder to reveal the dependency.

RSMLREN provides a control over false positives using a threshold derived from scores of artificial variables. The FPR control in the three simulation studies is shown to be quite effective, with empirical FPRs pretty close to target levels. In fact, for the control to be accurate, the distribution of artificial variable scores should match the distribution of real variable scores. Permuted informative features generally do not have the same distribution as permuted noisy features. More informative features generally cause more mismatch between the two distributions. Though the conservativeness of empirical FPRs at higher PR is generally observed, it is unknown whether it is largely due to sampling variance or distribution mismatch. More datasets are needed for further analysis. There remains several interesting future research topics. First, it is unknown whether RSMLREN is still able to provide a good FPR control when noisy features are correlated. Second, it is interesting to

investigate how a non-symmetric distribution of feature values affects the FPR control of RSMLREN.

RSMLREN is also able to accurately rank informative features because the feature selection percentage in constructing the feature score reflects each feature's discriminative power. If a feature has a higher discriminative power, it has a higher percentage of being selected out of all subspaces that include it. In the second simulation study, RSMLREN has a relatively stable ranking performance with a subspace size below 50% and a slowly decreasing trend with size beyond 50%. Even at 99%, the rank coefficient is higher than LREN2000. Investigation of this robust behavior of RSMLREN is an interesting future research topic.

Though RSMLREN can correctly infer features' discrimination directions, it relies on the assumption that a feature's direction is the same in all subspaces that include it. For example, a feature may have totally different directions depending on the presence of another feature. The current approach need be modified in the future research to account for this effect.

Another interesting observation is that the computation cost of RSMLREN may be future reduced by using a less number of iterations, at which performance of RSMLREN is relatively stable. Though it is difficult to determine the minimal number without knowing the underlying data structure, it is feasible to monitor the stability by comparing the performance difference between two consecutive iterations. For future research, an adaptive RSM framework is promising. For example, partition of feature space is stopped if the difference between the l_2 norms of feature scores from two consecutive iterations is below a certain threshold.

3.6 Summary and Conclusions

In this chapter, I empirically evaluate the performance of the developed RSM framework using LREN as its base classifier. I obtain several interesting results. For classification, RSMLREN has lower test errors than LREN in most cases. Minimal errors from both

methods all have downward bias with respect to test errors. The adjusted errors have less bias and are more accurate. Compared to PR, SNR plays a more dominant role in reducing misclassification errors. For feature selection, PR has a negative effect such that higher PR results in lower selection sensitivity. In most cases, RSMLREN has a large improvement over LREN in terms of selection sensitivity and similarity. For three target false positive rate levels, RSMLREN is able to keep the empirical FPRs pretty close to the target rates. Additionally, RSMLREN is able to correctly rank informative features based on their individual discriminative capacities. Moreover, RSMLREN can also correctly determine features' discrimination directions. Finally, I find that as the subspace size increases, performance of RSMLREN approaches LREN. Among the three subspaces examined, the smallest subspace (10% of the original feature size) performs the best in most cases. Increasing the number of subspaces does help stabilize performance of RSMLREN. However, for the sake of less computation load, a smaller number of subspaces is feasible without much deterioration in overall performance in both classification and feature selection.

Chapter 4: Application to Recursive Feature Elimination Using Linear Support Vector Machine

4.1 Introduction

Support vector machines (SVMs) have been extensively investigated in various disciplines. SVMs find the maximum margin hyperplane that maximizes the minimum distance from the hyperplane to the closest training points of two classes. Though SVMs are able to handle nonlinear decision boundaries of arbitrary complexity, linear SVMs are popular in analyzing fMRI data because it is much easier to interpret feature weights. Recently, linear SVMs have been shown to be successful in classifying high dimensional fMRI data (Cox and Savoy, 2003; Mitchell et al., 2004; Mourao-Miranda et al., 2005; LaConte et al., 2005). A direct whole-brain classification is appealing since it explores the full pattern. However, it is usually sub-optimal because only a small portion of voxels/features carry discriminative information and most voxels are just noises. Classification methods suffer from overfitting when facing many irrelevant features (Kohavi and John, 1997; Guyon and Elisseef, 2003; Norman et al., 2006). In fact, dimension reduction (discarding irrelevant features) can dramatically improve the performance of SVM (Guyon et al., 2002).

Guyon et al. (2002) proposed a feature elimination algorithm to recursively discard genes with smallest absolute weights from a linear SVM (RFESVM). Multiple genes may be removed at a time. The best elimination level was determined as the one that achieved the highest cross validation accuracy. The method was shown to be more accurate at prediction than the linear SVM trained using all features. The superior generalization performance of RFESVM has also been demonstrated on fMRI data with different mental tasks (De Martino et al., 2008; Hanson and Halchenko, 2008). In addition to data classification, as

a multivariate feature selection method, RFESVM is also effective in mapping informative fMRI data patterns. De Martino et al. (2008) carried out extensive simulations showing that RFESVM was more sensitive to informative patterns than univariate feature selection methods such as general linear models, t-test and Wilcoxon rank test. RFESVM was especially more powerful if it was combined with an initial univariate feature screening. Ryali et al. (2010) carried out extensive simulations comparing LREN and RFESVM. Though LREN was reported to be superior in feature selection in terms of false positive rate, sensitivity and accuracy, it is difficult to perform fair comparisons because LREN and RFESVM involve different tuning parameters.

However, previous evaluations and applications of RFESVM have several limitations. First, both De Martino et al. (2008) and Ryali et al. (2010) evaluated RFESVM only on one simulated dataset. Sampling variance may obscure difference between methods. Second, they did not recognize the possible under-bias of using the minimal CV error as an estimate of generalization error. Third, starting with the whole feature space, RFESVM may render poor and unstable feature rankings at the first several elimination levels. Informative features previously removed can never be recovered at subsequent elimination levels. Moreover, because of its computationally intensive nature, features are removed in chunks and usually only a small number of elimination levels are implemented (10 elimination levels in De Martino et al. 2008; Ryali et al. 2010). Either too much noises are retained or too much informative features are discarded at the best level. As shown in simulations from Ryali et al. (2010), false positive rates of RFESVM were extremely high (above 40%) without an initial voxel screening, which greatly limits RFESVM as a useful method in the accurate mapping of fMRI data patterns. To solve this problem, De Martino et al. (2008) suggested using permutation tests on each feature to control false selections, but it was very computationally intensive.

In this chapter, I apply the developed RSM framework to RFESVM and seek to address the two important issues with RFESVM: deficiency in feature selection due to the high dimensionality and lacking of an appropriate control over false selections in mapping

informative patterns. Using Monte Carlo simulations, I first fully evaluate the RFESVM as a benchmark method in classification and feature selection. For classification, I focus on the bias using minimal CV errors of RFESVM and evaluate the bias correction method proposed by Tibshirani and Tibshirani (2009) on RFESVM. For feature selection, I document RFESVM's performance in false positive rates, sensitivity rates and similarity rates. Using a smaller scale simulations, I evaluate RSMRFESVM and compare it to RFESVM. Specifically, for classification, I examine the selection bias in RSMRFESVM and the effectiveness of the proposed bias correction method. I also compare RSMRFESVM with RFESVM in terms of selection sensitivity and similarity. I empirically examine the performance of RSMRFESVM in controlling false positive rates against several target levels. Effects of the subspace size and the number of subspaces are discussed. Finally, I investigate whether RSMRFESVM can (1) effectively rank informative features based on their individual discriminative abilities and (2) correctly determine features' discrimination directions.

The remainder of this chapter is organized as follows. In section 4.2, I review the RFESVM algorithms applied in Guyon et al. (2002), De Martino et al. (2008) and Ryali et al. (2010). In section 4.3, I describe the proposed RSMRFESVM. I report and discuss simulation results in section 4.4 and 4.5. Section 4.6 concludes the study.

4.2 Review of Recursive Feature Elimination using Support Vector Machine

If training data from two classes are linearly separable, lines SVM is a maximum margin classifier and its decision boundary is constructed to leave the largest possible margin on either side. In a common case of overlapping classes, a soft-margin SVM aims to maximize the margin but allows for some data points to be misclassified. In the following, I only

consider the soft-margin linear SVM and short it for SVM. The problem of SVM is to solve:

$$(\hat{\beta}, \hat{\beta}_0) = \min_{\beta,\beta_0} ||\beta||_2^2 + C \sum_{i=1}^{N} \xi_i,$$

$$\text{subject to: } \xi_i \geq 0, y_i(x_i^T \beta + \beta_0) \geq 1 - \xi_i. \tag{4.1}$$

Solving β is a quadratic programming problem. The solution for β has the form

$$\hat{\beta} = \sum_{i=1}^{N} \hat{\alpha_i} y_i x_i,$$

where $\hat{\alpha_i}$ is nonzero only for those samples that satisfy $y_i(x_i^T \beta + \beta_0) \geq 1 - \xi_i$. The samples corresponding to nonzero $\hat{\alpha_i}$ are support vectors and the decision boundaries are solely determined by support vectors. To classify new samples, the decision function is written as

$$y_i = \begin{cases} -1 & \text{if } x_i^T \hat{\beta} + \hat{\beta}_0 \leq 0 \\ 1 & \text{if } x_i^T \hat{\beta} + \hat{\beta}_0 > 0 \end{cases}.$$

A discriminative map can be constructed using weights $\hat{\beta}$ from a trained SVM. Voxels with higher absolute $|\hat{\beta}|$ contribute more to the discrimination of two classes. As a sensitivity measure, $|\hat{\beta}|$ or $\hat{\beta}^2$ can be used to rank features.

Guyon et al. (2002) proposed RFESVM as a recursive feature elimination method using weight magnitudes from SVM as feature ranking criterion. Their algorithm is as follows:

- Initialize surviving feature list: $s = [1, 2, ..., p]$ (p features in total)
- Initialize feature rank list: $r = \emptyset$
- Repeat until $s = \emptyset$

- Restrict training samples to survived feature indices: $X = X_0(:, s)$
- Train the SVM classifier and compute the weight vector: $w = \sum_i \alpha_i y_i x_i$
- Compute feature ranking score: $c_i = (w_i)^2, i \in s$
- Discard the feature with the smallest ranking score: $f = argmin(c)$
- Update the feature rank list: $r = [s(f), r]$
- Update the surviving feature list: $s = [1, ..., f-1, f+1, ..., \text{length}(s)]$

- Return the ranked list r

The above algorithm can be generalized to removing several features per step for fast computation. In Guyon et al. (2002), at each step, half of the surviving features were discarded.

De Martino et al. (2008) used simulated datasets and real fMRI data to examine the RFESVM. They had several different implementations in RFESVM from Guyon et al. (2002). First, De Martino et al. (2008) used a least squares SVM instead of the classical SVM. In the least square SVM, the constraints (inequality (4.1)) are equalities instead of inequalities. It is much easier to train a least squares SVM by solving a set of linear equations. Instead, in the classical SVM, it is a quadratic programming problem. Second, they further divided the training samples into several partitions and used the averaged absolute weights across all partitions as ranking scores. Third, they considered 10 elimination levels and the number of features to be discarded at each step was chosen so that the number of features after 10 steps is matched to a desired number. Fourth, the returned discrimination map was a union of K maps from K-fold cross validations.

In Ryali et al. (2010), a classical SVM was employed with 10 elimination levels. 10% of the smallest feature scores were discarded. They compared two approaches to compute weights (a single fit of SVM in Guyon et al. (2002) and average weights across partitions in De Martino et al. (2008)) and found little difference in feature selection performance. The best elimination level was also determined as the one that had the highest cross validation

accuracy. But to obtain the final discriminative map, they retrained the SVM starting from all features and recursively eliminated features up to the best level. They reported that compared to the union map used in De Martino et al. (2008), less false positives were included in the map.

4.3 Random Subspace Method Applied to Recursive Feature Elimination using Support Vector Machine

In this section, I apply the RSM framework to RFESVM and describe the algorithm used in simulation studies. Instead of a least squares SVM, I use a classical SVM because of its popularity. I also discuss several parameters involved in training RFESVM and RSMRFESVM. To train RFESVM, I use "LIBSVM"[1], a toolbox for MATLAB.

4.3.1 Algorithm of RSMRFESVM

- Generate random probe variables: form an augmented feature space by adding them to the original feature space
- Initialize the feature scores: $s = \mathbf{0}_{1\times 2p}$
- Repeat until the specified number of iterations is met. In each iteration:
  - Partition the augmented feature space into mutually exclusive subspaces
  - In each subspace:
    - Perform a recursive feature elimination process: use a 5-fold cross validation scheme
      - On data in training folds (4 folds), at each elimination level, features are ranked by the absolute weights from a trained linear SVM
      - Discard features with smallest scores and compute the prediction accuracy of the current level on data in the test fold (1 remaining fold)

[1]LIBSVM – A Library for Support Vector Machines, http://www.csie.ntu.edu.tw/∼cjlin/libsvm/

    - The best elimination level is chosen to achieve the highest cross validation accuracy (BestCVA)
    - Starting from all features in the subspace, recursively eliminate features up to the best level
    - Update the score for each feature in the partition: $s_i = s_i + \text{BestCVA} * I(\text{discarded or not})$ and $I(.)$ is an indicator function that is 0 if i^{th} feature is discarded after the best elimination level and 1 if it is retained

- Update feature scores: $s = \frac{s}{\#Iterations}$
- Compute the estimated prediction accuracy using the majority voting from all subspaces at their best cross validation accuracies and apply the error adjustment
- Derive a threshold to control the false positive rate based on the distribution of scores of random probe variables
- Determine discrimination directions of selected features

To classify future independent data point, the same random probe variables are added and the same number of subspaces are generated containing exactly the same features as in the K-fold cross validation. The classification is made by the majority voting from decisions in all subspaces. In the next subsection, I discuss choosing values for parameters involved in the proposed RSMRFESVM algorithm.

4.3.2 Parameters in RSMRFESVM

1. Linear Support Vector Machine

   The tuning parameter of linear SVM is the cost parameter C. In previous RFESVM studies, the value of C is chosen beforehand. However, as shown in Hastie et al. (2004), SVM's generalization performance varied with values of tuning parameters. The cost tuning parameter is more related in RSMRFESVM because of varied features

across subspaces. Therefore, I set the cost parameter value to vary among 5 values: $[0.01, 0.1, 1, 10, 100]$.

2. Recursive Feature Elimination

   Two parameters govern the recursive feature elimination: the number of elimination levels and the number of features to be discarded at each level. I follow the settings in Ryali et al. (2010) and set the number of elimination levels to 10. At each level, I discard 10% of the survived features with smallest absolute weights.

3. Random Subspaces

   I follow the same settings in RSMLREN. That is, I vary the subspace size as the percentage of the total feature space: [10%, 25%, 50%]. The number of iterations is set at 200. Therefore, the number of subspaces is 2000 for 10% subspace size, 800 for 25% and 400 for 50%.

4. Random Probe Variables

   I also follow the setting in RSMLREN and set the number of random probes to be 1000 which is the same size as the original feature space.

Table 4.1 presents an overview of parameter values for RSMRFESVM in this study.

Table 4.1: Values of tuning parameters in RSMRFESVM

Tuning Parameters	**Values**
Cost parameter (C) in linear SVM	$\{0.01, 0.1, 1, 10, 100\}$
Number of elimination levels	10
Elimination rate at each level	10%
Subspace size	$\{10\%, 25\%, 50\%\}$
Number of iterations	200
Number of random probe variables	1000

4.4 Simulation Results

4.4.1 First Simulation Study

This simulation study uses the same datasets as in the first simulation study of RSMLREN and also resembles the study in Ryali et al. (2010). Similar to the first simulation study in Chapter 3, I first evaluate RFESVM as a benchmark method and compare RFESVM to its RSM version (RSMRFESVM). Evaluations are performed with respect to misclassification errors, feature selection performance, false positive rate control and effects of the number of subspaces.

Results of the Benchmark Method - RFESVM

Misclassification Errors

Table 4.2 tabulates misclassification error rates at all combinations of SNRs and PRs. Higher SNR leads to lower misclassification errors. Within each SNR, misclassification errors generally decrease with an increasing PR. High PR=20%, however, may increases test errors a little bit, for example, at SNR=1.0 and 1.5. This results are similarly observed in LREN and further confirm that feature redundancy may eventually deteriorates a classifier's prediction accuracy.

A quite different observation from RFESVM is that minimal errors of RFESVM estimate test errors quite well with very small biases. Adjusted CV errors in most cases are conservative and the upward bias is much larger than the bias of minimal CV errors. However, the small bias of minimal errors does not mean that RFESVM is not subject to the parameter selection bias. I conduct another set of simulations with an increased elimination rate of 20%, with all other parameters kept the same. Table 4.3 presents the misclassification errors. The bias of RFESVM is now mostly around 2% and adjusted CV errors are still conservative in most cases but provide much better estimates. The additional simulations illustrate that biases of both the minimal CV error and adjusted CV error are quite sensitive to values of tuning parameters. Analyses afterwards are performed with the

Table 4.2: Misclassification errors of RFESVM on 1000 simulated datasets with a 10% reduction

PR	Min error	Adjusted error	Test error	Min-Test	Adjust-Test
SNR = 0.5					
1%	0.4613 (0.0028)	0.4945 (0.0029)	0.4663 (0.0005)	-0.0050 (0.0027)	0.0282 (0.0028)
5%	0.4153 (0.0030)	0.4424 (0.0032)	0.4208 (0.0008)	-0.0055 (0.0026)	0.0216 (0.0028)
10%	0.3988 (0.0028)	0.4226 (0.0031)	0.4003 (0.0008)	-0.0015 (0.0024)	0.0223 (0.0026)
20%	0.3921 (0.0029)	0.4126 (0.0031)	0.3995 (0.0008)	-0.0075 (0.0025)	0.0130 (0.0027)
SNR = 1					
1%	0.3778 (0.0028)	0.4077 (0.0030)	0.3783 (0.0006)	-0.0005 (0.0026)	0.0294 (0.0028)
5%	0.2789 (0.0026)	0.2962 (0.0028)	0.2847 (0.0006)	-0.0058 (0.0023)	0.0115 (0.0025)
10%	0.2603 (0.0023)	0.2735 (0.0024)	0.2632 (0.0004)	-0.0029 (0.0020)	0.0104 (0.0022)
20%	0.2541 (0.0022)	0.2637 (0.0024)	0.2648 (0.0003)	-0.0106 (0.0020)	-0.0011 (0.0021)
SNR = 1.5					
1%	0.2913 (0.0026)	0.3180 (0.0028)	0.2872 (0.0005)	0.0041 (0.0025)	0.0308 (0.0027)
5%	0.1706 (0.0020)	0.1804 (0.0021)	0.1765 (0.0003)	-0.0058 (0.0018)	0.0039 (0.0020)
10%	0.1548 (0.0018)	0.1614 (0.0019)	0.1601 (0.0002)	-0.0053 (0.0016)	0.0013 (0.0018)
20%	0.1518 (0.0017)	0.1565 (0.0018)	0.1611 (0.0002)	-0.0092 (0.0016)	-0.0046 (0.0017)

RFESVM with a 10% elimination rate instead of a 20% elimination rate, to be consistent with the setting in Ryali et al. (2010).

Table 4.3: Misclassification errors of RFESVM on 1000 simulated datasets with a 20% reduction

PR	Min error	Adjusted error	Test error	Min-Test	Adjust-Test
SNR = 0.5					
1%	0.4364 (0.0027)	0.4975 (0.0030)	0.4688 (0.0006)	-0.0324 (0.0026)	0.0287 (0.0029)
5%	0.3990 (0.0028)	0.4512 (0.0031)	0.4293 (0.0009)	-0.0303 (0.0024)	0.0219 (0.0027)
10%	0.3854 (0.0027)	0.4342 (0.0030)	0.4094 (0.0009)	-0.0240 (0.0023)	0.0248 (0.0026)
20%	0.3760 (0.0028)	0.4193 (0.0031)	0.4075 (0.0009)	-0.0315 (0.0024)	0.0118 (0.0027)
SNR = 1					
1%	0.3467 (0.0028)	0.4013 (0.0031)	0.3747 (0.0008)	-0.0280 (0.0025)	0.0266 (0.0028)
5%	0.2663 (0.0025)	0.3022 (0.0029)	0.2924 (0.0007)	-0.0261 (0.0021)	0.0098 (0.0025)
10%	0.2504 (0.0022)	0.2819 (0.0026)	0.2713 (0.0006)	-0.0209 (0.0020)	0.0107 (0.0023)
20%	0.2439 (0.0021)	0.2705 (0.0025)	0.2725 (0.0005)	-0.0286 (0.0019)	-0.0020 (0.0022)
SNR = 1.5					
1%	0.2496 (0.0024)	0.2952 (0.0028)	0.2760 (0.0007)	-0.0264 (0.0022)	0.0192 (0.0026)
5%	0.1579 (0.0020)	0.1804 (0.0023)	0.1816 (0.0004)	-0.0237 (0.0017)	-0.0012 (0.0020)
10%	0.1460 (0.0017)	0.1622 (0.0020)	0.1652 (0.0003)	-0.0193 (0.0015)	-0.0030 (0.0018)
20%	0.1430 (0.0016)	0.1568 (0.0019)	0.1663 (0.0003)	-0.0233 (0.0015)	-0.0095 (0.0017)

Feature Selection Performance

Figure 4.1 presents RFESVM's feature selection performance in terms of false positive rates, sensitivity rates and similarity rates (black line is for SNR=0.5, blue line for SNR=1 and red line for SNR=1.5).

The left column shows the FPR curves at different PRs. Similar to results in LREN (figure 3.1), increasing SNR decreases FPR. However, at low and median SNRs, higher PR does not necessarily reduce FPR. It is particularly evident at PR=20%. The upward trend of FPR curves can be explained that including 1 false positive at a higher PR induces a higher FPR than at a lower PR. It requires an accurate feature selection to set off this effect. For example, at SNR=1.5 where patterns of interest are more separable, the FPR curve decreases with an increasing PR. LREN has a better FPR control because RFESVM has only 10 elimination levels and a 10% elimination rate, while LREN has the option to penalize feature weights very close to zeros. The middle column shows the sensitivity rate curves. More informative features are selected at a higher SNR but a higher PR results in lower sensitivity rate, similarly observed in LREN in Chapter 3. Compared to LREN, since RFESVM has much higher FPRs, sensitivity rates are higher in RFESVM. The right

column shows the similarity rate curves. Compared to results in LREN, SNR has a reversing effect. However, this is not unexpected considering the extremely high FPRs at SNR = 0.5 (over 50%), where a noisy feature has much greater probability to be selected in any two datasets.

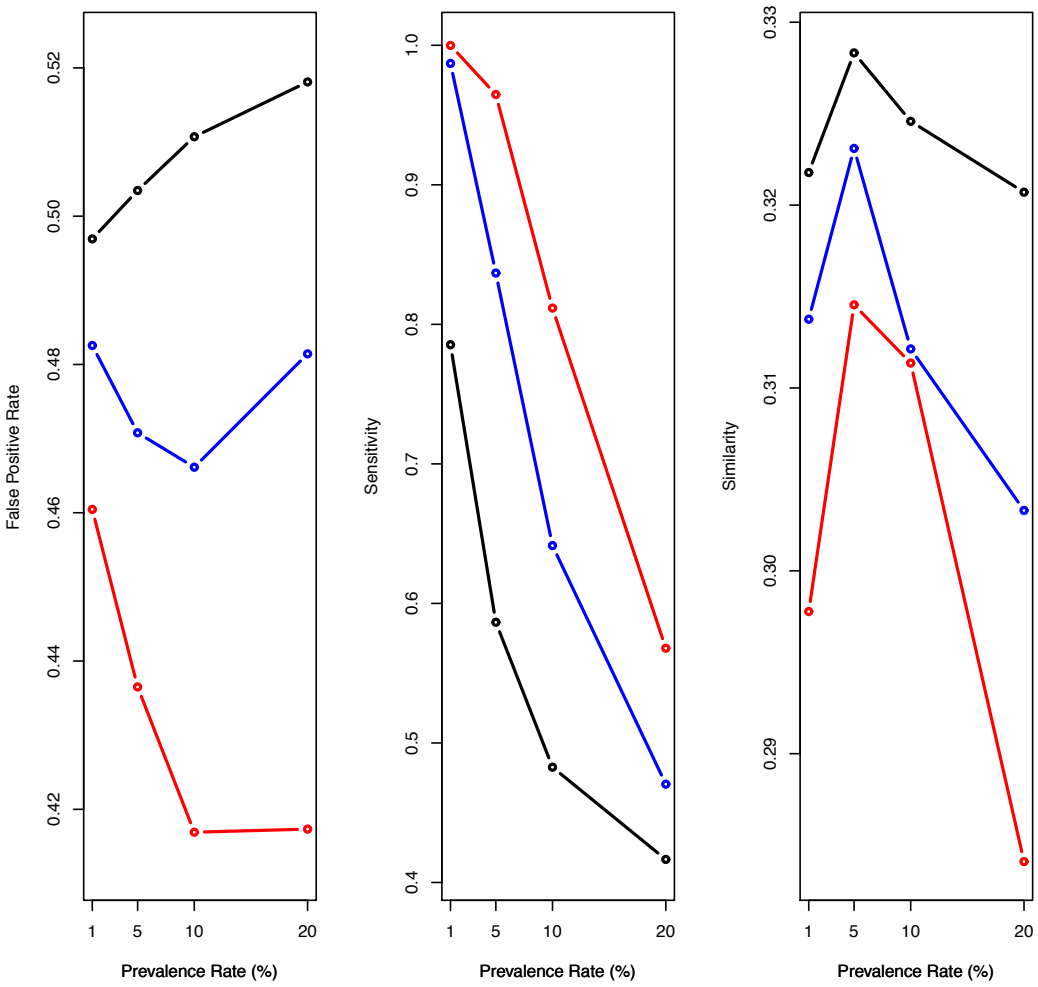

Figure 4.1: Feature selection performance for RFESVM with a 10 % feature reduction rate on 1000 simulated datasets (black: SNR=0.5, blue: SNR=1, red: SNR=1.5)

Results of RSMRFESVM and Comparisons with RFESVM

I evaluate both RSMRFESVM and RFESVM on 10 simulated datasets due to the high computation cost in RSMRFESVM. The same 10 datasets are also used in comparing RSMLREN and LREN in the first simulation study in chapter 3. I first examine both methods in misclassification errors and then in feature selection performance. FPR control in RSMRFESVM is evaluated with different FPR target levels. Finally, the effects of the number of subspaces are also discussed.

Misclassification Errors

Table 4.4 lists the misclassification errors for both RFESVM and RSMRFESVM. Similar to LREN, increasing SNR significantly decreases misclassification errors of both methods, and a high PR at a high SNR increases test errors.

Column 5 tabulates misclassification errors on a large independent test dataset. Three RSMRFESVM methods are able to achieve lower misclassification errors than RFESVM for most cases, though the improvements are quite small. This is also observed in comparing LREN with RSMLREN (see table 3.3), It is probably because like LREN, SVM itself is a stable classifier. Among three RSMRFESVM methods with different subspace sizes, RSMRFESVM with 10% of the original feature size (RSMRFESVM10) has the lowest test errors. As the subspace size increases, test errors increases.

Column 3 shows minimal errors and column 6 represents its bias. For RFESVM, due to the data sampling variance, minimal errors underestimate test errors, which is not observed using a larger amount of datasets (1000 datasets, table 4.2). Similar to results in RSMLREN, the downward bias is much more severe for RSMRFESVM and increases with a decreasing subspace size. Column 4 shows the adjusted errors using the developed bias correction method. Adjusted errors are more accurate than minimal errors (column 6 and column 7), but still underestimate test errors at high SNRs. However, this effect needs more datasets to confirm because of the high variance of the bias. Overall, the developed bias correction method is quite accurate compared to the sever bias of minimal errors.

Table 4.4: Misclassification errors for RFESVM and RSMRFESVM on 10 simulated datasets

Method	PR	Min error	Adjusted error	Test error	Min-Test	Adjust-Test
SNR = 0.5						
RFESVM	1%	0.4220 (0.0187)	0.4560 (0.0227)	0.4654 (0.0048)	-0.0434 (0.0195)	-0.0094 (0.0237)
	5%	0.3940 (0.0327)	0.4280 (0.0388)	0.4284 (0.0101)	-0.0344 (0.0283)	-0.0004 (0.0336)
	10%	0.3440 (0.0338)	0.3680 (0.0367)	0.3973 (0.0140)	-0.0533 (0.0267)	-0.0293 (0.0301)
	20%	0.3880 (0.0301)	0.4080 (0.0306)	0.3985 (0.0059)	-0.0105 (0.0256)	0.0095 (0.0261)
RSMRFESVM10	1%	0.3340 (0.0244)	0.4780 (0.0274)	0.4717 (0.0045)	-0.1377 (0.0248)	0.0063 (0.0278)
	5%	0.2940 (0.0359)	0.4020 (0.0473)	0.4225 (0.0106)	-0.1285 (0.0374)	-0.0205 (0.0485)
	10%	0.3000 (0.0379)	0.3860 (0.0510)	0.3919 (0.0141)	-0.0919 (0.0405)	-0.0059 (0.0529)
	20%	0.2920 (0.0250)	0.4040 (0.0384)	0.3844 (0.0046)	-0.0924 (0.0254)	0.0196 (0.0387)
RSMRFESVM25	1%	0.3960 (0.0251)	0.4560 (0.0270)	0.4735 (0.0038)	-0.0775 (0.0254)	-0.0175 (0.0272)
	5%	0.3980 (0.0367)	0.4740 (0.0400)	0.4266 (0.0097)	-0.0286 (0.0380)	0.0474 (0.0412)
	10%	0.3580 (0.0435)	0.4080 (0.0507)	0.3959 (0.0145)	-0.0379 (0.0458)	0.0121 (0.0527)
	20%	0.3580 (0.0290)	0.4020 (0.0346)	0.3938 (0.0056)	-0.0358 (0.0295)	0.0082 (0.0350)
RSMRFESVM50	1%	0.4340 (0.0229)	0.4940 (0.0229)	0.4722 (0.0039)	-0.0382 (0.0232)	0.0218 (0.0232)
	5%	0.4180 (0.0350)	0.4800 (0.0370)	0.4257 (0.0100)	-0.0077 (0.0364)	0.0543 (0.0383)
	10%	0.3740 (0.0422)	0.3980 (0.0397)	0.3927 (0.0126)	-0.0187 (0.0440)	0.0053 (0.0417)
	20%	0.3740 (0.0310)	0.4100 (0.0382)	0.3894 (0.0051)	-0.0154 (0.0314)	0.0206 (0.0385)
SNR = 1						
RFESVM	1%	0.3400 (0.0223)	0.3780 (0.0248)	0.3760 (0.0058)	-0.0360 (0.0226)	0.0020 (0.0250)
	5%	0.2580 (0.0324)	0.2700 (0.0331)	0.2881 (0.0068)	-0.0301 (0.0284)	-0.0181 (0.0293)
	10%	0.2120 (0.0270)	0.2220 (0.0290)	0.2612 (0.0065)	-0.0492 (0.0228)	-0.0392 (0.0247)
	20%	0.2560 (0.0171)	0.2660 (0.0198)	0.2637 (0.0028)	-0.0077 (0.0146)	0.0023 (0.0174)
RSMRFESVM10	1%	0.2700 (0.0270)	0.3940 (0.0348)	0.3893 (0.0045)	-0.1193 (0.0274)	0.0047 (0.0351)
	5%	0.1940 (0.0327)	0.2560 (0.0437)	0.2785 (0.0054)	-0.0845 (0.0331)	-0.0225 (0.0440)
	10%	0.1860 (0.0237)	0.2320 (0.0300)	0.2514 (0.0042)	-0.0654 (0.0240)	-0.0194 (0.0303)
	20%	0.1820 (0.0212)	0.2320 (0.0280)	0.2535 (0.0014)	-0.0715 (0.0212)	-0.0215 (0.0280)
RSMRFESVM25	1%	0.3360 (0.0234)	0.3920 (0.0194)	0.3952 (0.0044)	-0.0592 (0.0238)	-0.0032 (0.0199)
	5%	0.2580 (0.0348)	0.3100 (0.0432)	0.2862 (0.0060)	-0.0282 (0.0353)	0.0238 (0.0436)
	10%	0.2080 (0.0259)	0.2320 (0.0292)	0.2553 (0.0045)	-0.0473 (0.0262)	-0.0233 (0.0296)
	20%	0.1980 (0.0178)	0.2240 (0.0183)	0.2578 (0.0023)	-0.0598 (0.0179)	-0.0338 (0.0185)
RSMRFESVM50	1%	0.3700 (0.0220)	0.4240 (0.0221)	0.3944 (0.0047)	-0.0244 (0.0225)	0.0296 (0.0226)
	5%	0.2580 (0.0361)	0.2820 (0.0405)	0.2871 (0.0054)	-0.0291 (0.0365)	-0.0051 (0.0408)
	10%	0.2200 (0.0281)	0.2360 (0.0338)	0.2539 (0.0049)	-0.0339 (0.0285)	-0.0179 (0.0342)
	20%	0.2060 (0.0207)	0.2200 (0.0239)	0.2592 (0.0025)	-0.0532 (0.0208)	-0.0392 (0.0240)
SNR = 1.5						
RFESVM	1%	0.2740 (0.0246)	0.3120 (0.0257)	0.2875 (0.0051)	-0.0135 (0.0241)	0.0245 (0.0258)
	5%	0.1580 (0.0306)	0.1680 (0.0328)	0.1764 (0.0036)	-0.0184 (0.0278)	-0.0084 (0.0299)
	10%	0.1180 (0.0205)	0.1220 (0.0216)	0.1599 (0.0034)	-0.0419 (0.0175)	-0.0379 (0.0186)
	20%	0.1460 (0.0177)	0.1500 (0.0182)	0.1620 (0.0017)	-0.0160 (0.0169)	-0.0120 (0.0176)
RSMRFESVM10	1%	0.1980 (0.0239)	0.2940 (0.0328)	0.3038 (0.0032)	-0.1058 (0.0241)	-0.0098 (0.0330)
	5%	0.1240 (0.0229)	0.1580 (0.0316)	0.1698 (0.0026)	-0.0458 (0.0230)	-0.0118 (0.0317)
	10%	0.1060 (0.0211)	0.1320 (0.0275)	0.1521 (0.0015)	-0.0461 (0.0211)	-0.0201 (0.0276)
	20%	0.1000 (0.0140)	0.1120 (0.0164)	0.1548 (0.0009)	-0.0548 (0.0140)	-0.0428 (0.0164)
RSMRFESVM25	1%	0.2760 (0.0295)	0.3300 (0.0357)	0.3108 (0.0042)	-0.0348 (0.0298)	0.0192 (0.0359)
	5%	0.1520 (0.0250)	0.1700 (0.0282)	0.1761 (0.0036)	-0.0241 (0.0252)	-0.0061 (0.0284)
	10%	0.1140 (0.0211)	0.1240 (0.0240)	0.1541 (0.0022)	-0.0401 (0.0212)	-0.0301 (0.0241)
	20%	0.1040 (0.0151)	0.1100 (0.0167)	0.1571 (0.0006)	-0.0531 (0.0152)	-0.0471 (0.0167)
RSMRFESVM50	1%	0.3040 (0.0292)	0.3380 (0.0306)	0.3131 (0.0047)	-0.0091 (0.0296)	0.0249 (0.0310)
	5%	0.1520 (0.0283)	0.1580 (0.0312)	0.1757 (0.0031)	-0.0237 (0.0285)	-0.0177 (0.0313)
	10%	0.1220 (0.0201)	0.1280 (0.0213)	0.1558 (0.0020)	-0.0338 (0.0202)	-0.0278 (0.0214)
	20%	0.1100 (0.0161)	0.1140 (0.0177)	0.1574 (0.0013)	-0.0474 (0.0162)	-0.0434 (0.0177)

I also examine the effects of the number of subspaces on misclassification errors. To be consistent with RSMLREN in Chapter 3, I show misclassification error curves for RSMRFESVM10 at SNR = 1 (figure 4.2). Similar to results from RSMLREN, the minimal and test error curves start to converge beyond 500 subspaces and the adjusted error curve is most variant with a relatively stable variance. These effects are also observed in RSMLREN because both RSMLREN and RSMRFESVM use the majority voting scheme and the same bias adjustment method.

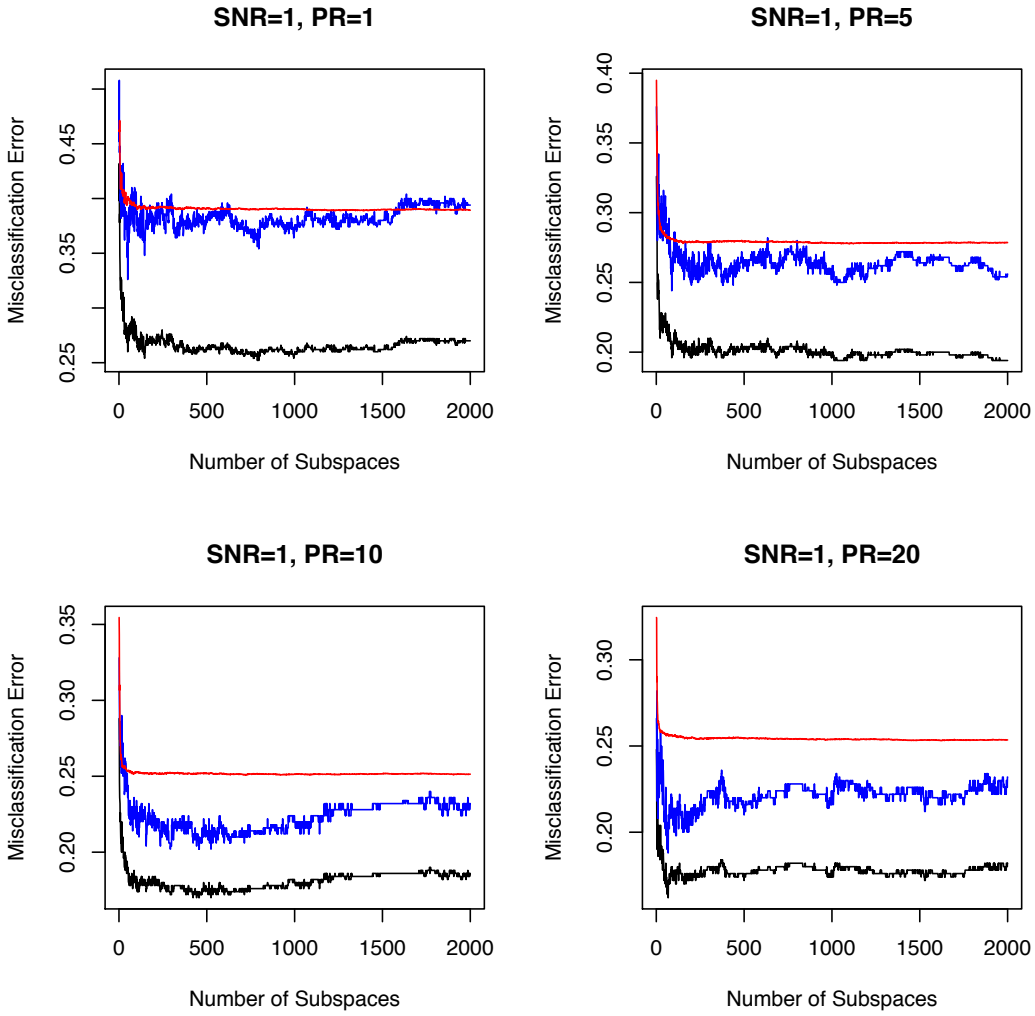

Figure 4.2: Effects of number of subspaces on misclassification errors (RSMRFESVM10 at SNR = 1, black line: minimal CV error, blue line: adjusted CV error and red line: test error

Feature Selection Performance

I evaluate the feature selection performance in terms of false positive rates, sensitivity rates and similarity rates. I also discuss the effects of the number of subspaces on sensitivity rates.

Figure 4.3 shows the false positive rate curves for four methods after 200 iterations. False positive rates of three RSMRFESVM methods are matched to RFESVM to ensure that the same number of irrelevant features are selected for a fair comparison. Because of the few datasets available, FPR curves tend to irregular.

Figure 4.4 shows feature selection sensitivity rate curves for four methods by first fixing the number of false positives on the same level. Similar to observations in the benchmark method RFESVM, a higher SNR helps select more relevant features and a higher PR has a detrimental effect. Three RSMRFESVM methods perform better than RFESVM and RSMRFESVM10 performs the best. RFESVM has comparable performance with RSMRFESVMs only at PR=1%, due to very high FPRs and a relatively small number of informative features. The explanation for the improvement in RSMRFESVM follows the same logic in RSMLREN in the third chapter, that is, feature ranking/feature selection in smaller subspaces is more accurate than in a higher dimensional feature space and lower ranked features in the original space can be "boosted" in some subspaces.

The sensitivity rates in figure 4.4 are evaluated where FPRs are very high (around 50%). It is unclear whether RSMRFESVM methods still perform better than RFESVM under lower FPRs. To investigate this, I first rank features using feature scores in RSMRFESVM and absolute SVM weights in RFESVM at the optimal elimination level. Then I select the number of highest ranked features from 1 to 300 which is roughly the number of features remained after a full 10-level elimination. At each number of selected features, I calculate the percentage of informative features retrieved compared to all informative features. Figure 4.5, 4.6 and 4.7 present the results under three SNRs comparing three RSMRFESVM methods with RFESVM. A general trend common to all methods is that a higher SNR leads to higher sensitivity rates and a high PR leads to lower sensitivity rates. At SNR = 0.5,

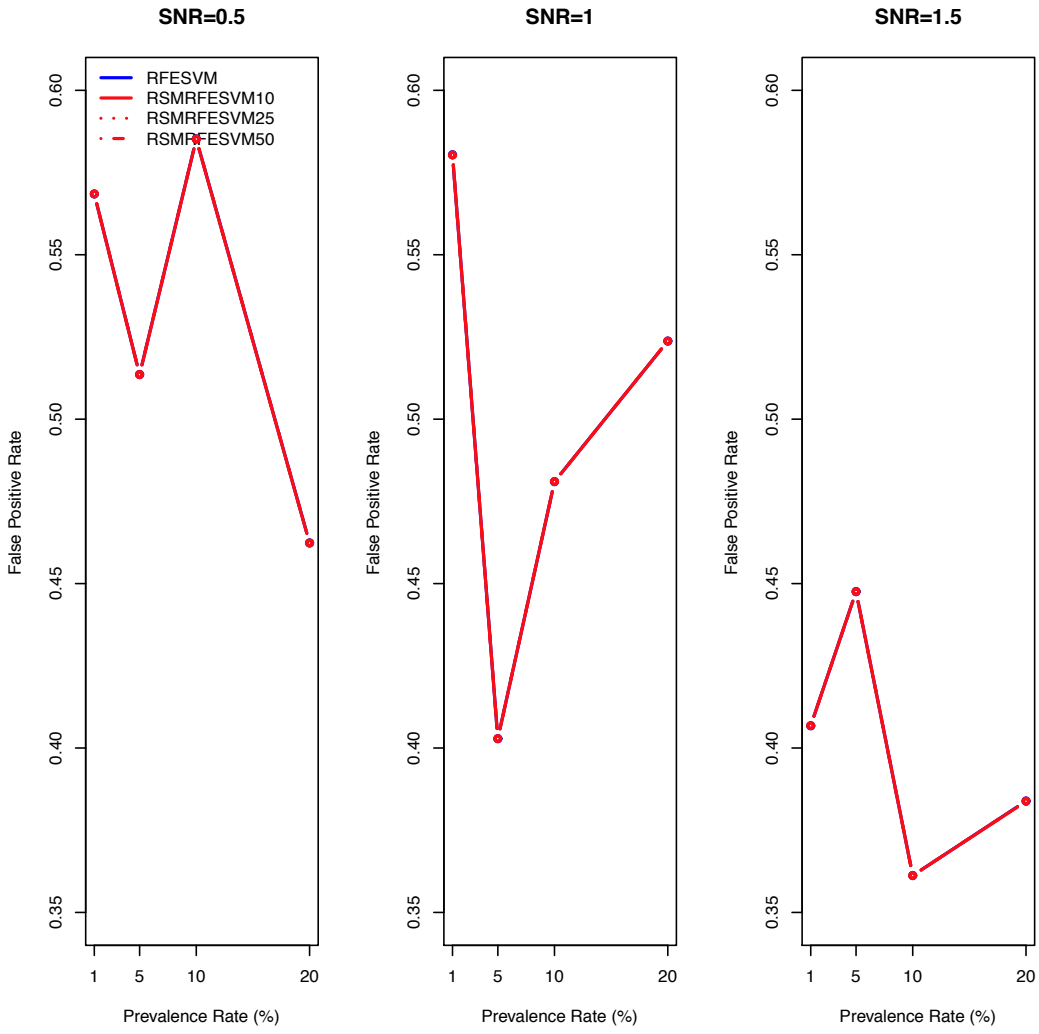

Figure 4.3: Feature selection: false positive rates after 200 iterations, FPRs of all methods are matched

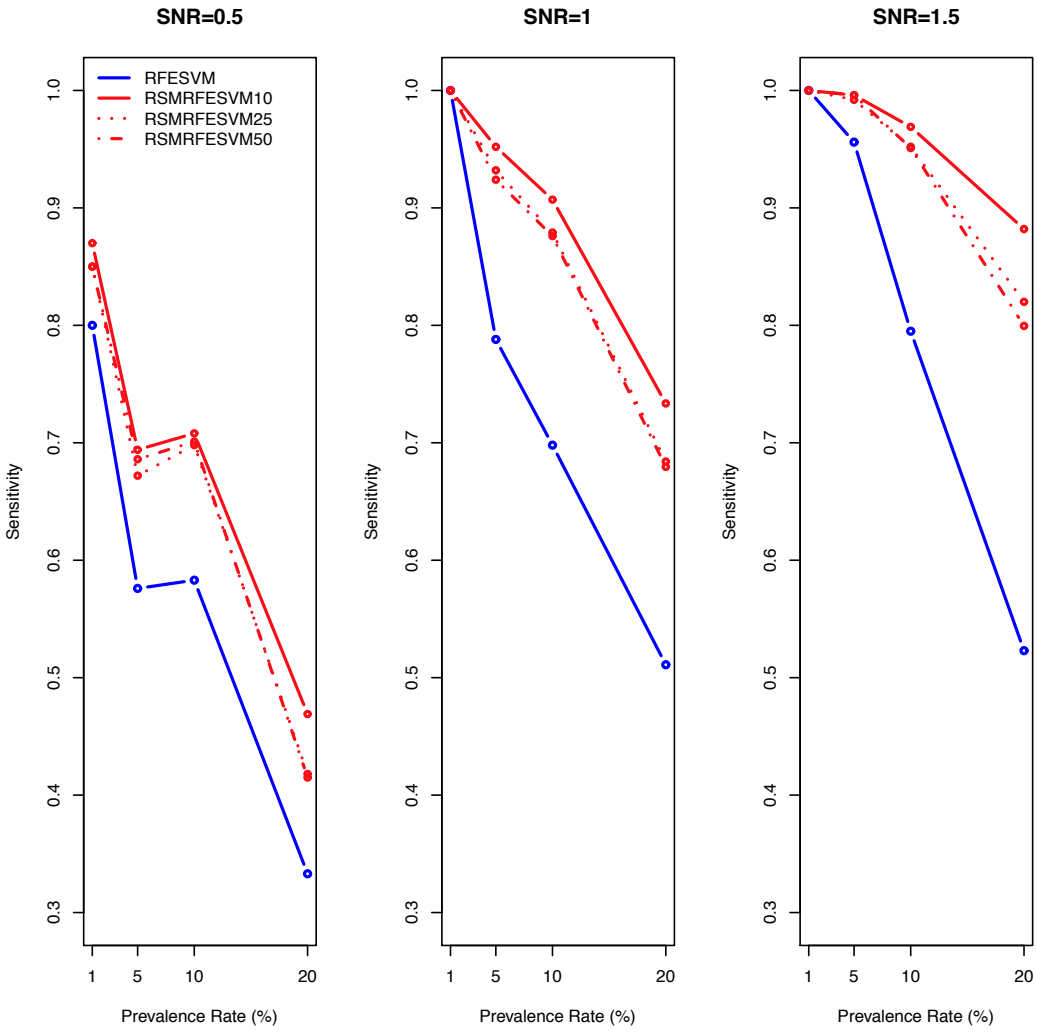

Figure 4.4: Feature selection sensitivity rates after 200 iterations

three RSMRFESVM methods with different subspace sizes perform better than RFESVM, especially at higher PRs. At SNR=1 and PR=1%, RSMRFESVM50 performs worse than RFESVM with 150 or less top ranked features but becomes better after that, and the other two RSMRFESVM methods are better than RFESVM. At SNR=1.5, RFESVM is still better than RSMRFESVM50 at PR=1% and better than RSMRFESVM50 at the first 150 features at PR=5%. At low PRs, RFESVM seems to have competitive performance with RSMRFESVM methods probably because of the different ranking methods. In RFESVM, features are ranked by their SVM weights. SVM may have a more accurate ranking for a small set of informative features than for a larger set, and informative features' rankings are higher than noisy features. However, in RSMRFESVM, the ranking of a feature is determined by the percentage that a feature is selected compared to all subspaces including it. In each subspace, small number of recursive eliminations (at most 10 levels) are not able to discard most of the noisy features. However, this negative effect can be offset by sampling more subspaces for different combinations of informative and uninformative features. For example, a noisy features selected in some subspaces may be discarded in other subspaces. Thus, more subspaces can better separate informative features from noisy features. Since RSMRFESVM10 has the highest number of subspaces, it is expected to perform the best. On the other hand, RSMRFESVM50 has the least number of subspaces and it performs worse than RFESVM at low PRs. In fact, from the results presented afterwards, it can be clearly observed that the sensitivity rate curve of RSMRFESVM50 at low PR is continuously rising as more subspaces are used, while the sensitivity rate is quite constant for RSMRFESVM10 after a certain number of iterations.

To examine the effects of the number of subspaces on sensitivity rates, I rank the features according to their scores and fix the false positive rate at FPR = 0.1 for all three RSMRFESVM methods. Figure 4.8, 4.9, 4.10 show the results (black line is for RSMRFESVM10, blue line for RSMRFESVM25 and red line for RSMRFESVM50). RSMRFESVM10 is the most stable and its sensitivity rate curve begins to converge after 100 iterations. Curves

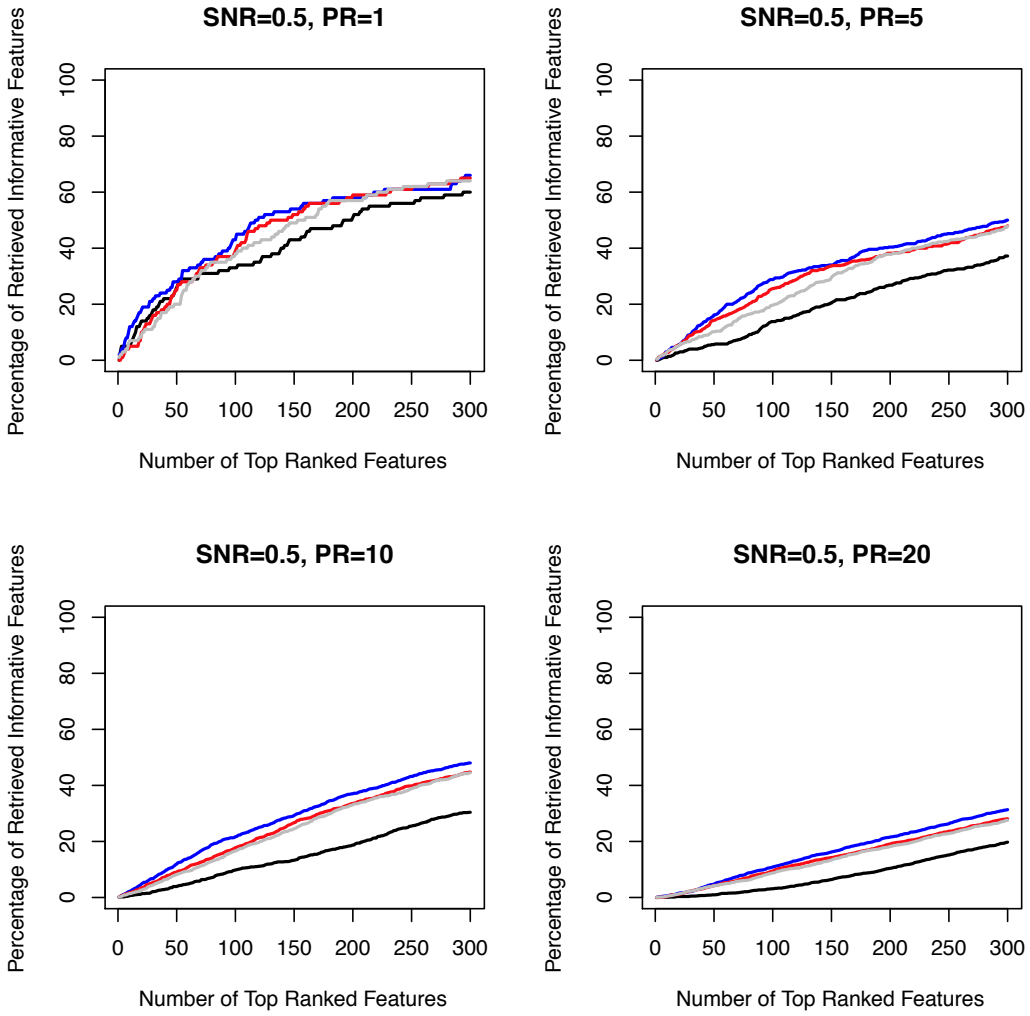

Figure 4.5: Informative feature retrieval efficiency at SNR = 0.5. Black line is for RFESVM, blue line for RSMRFESVM10, red line for RSMRFESVM25 and grey line for RSMRFESVM50

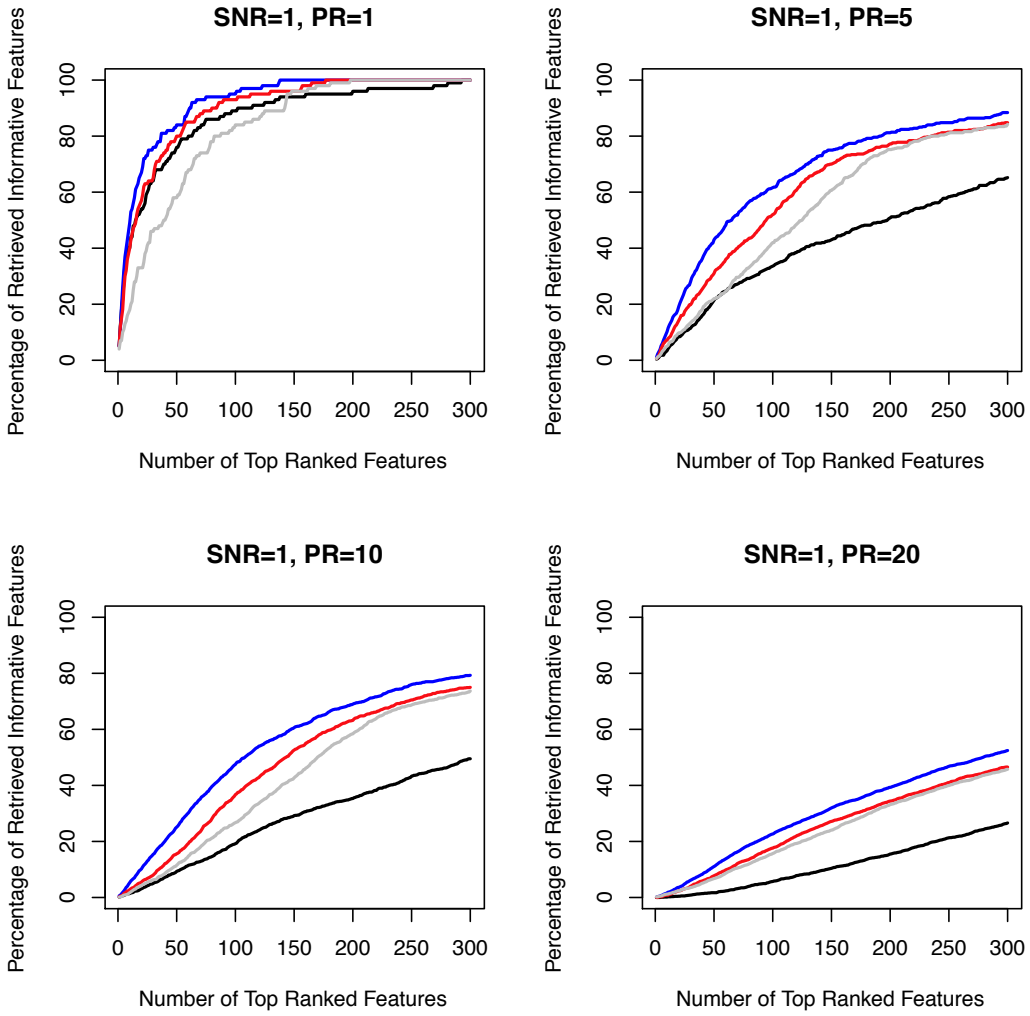

Figure 4.6: Informative feature retrieval efficiency at SNR = 1. Black line is for RFESVM, blue line for RSMRFESVM10, red line for RSMRFESVM25 and grey line for RSMRFESVM50

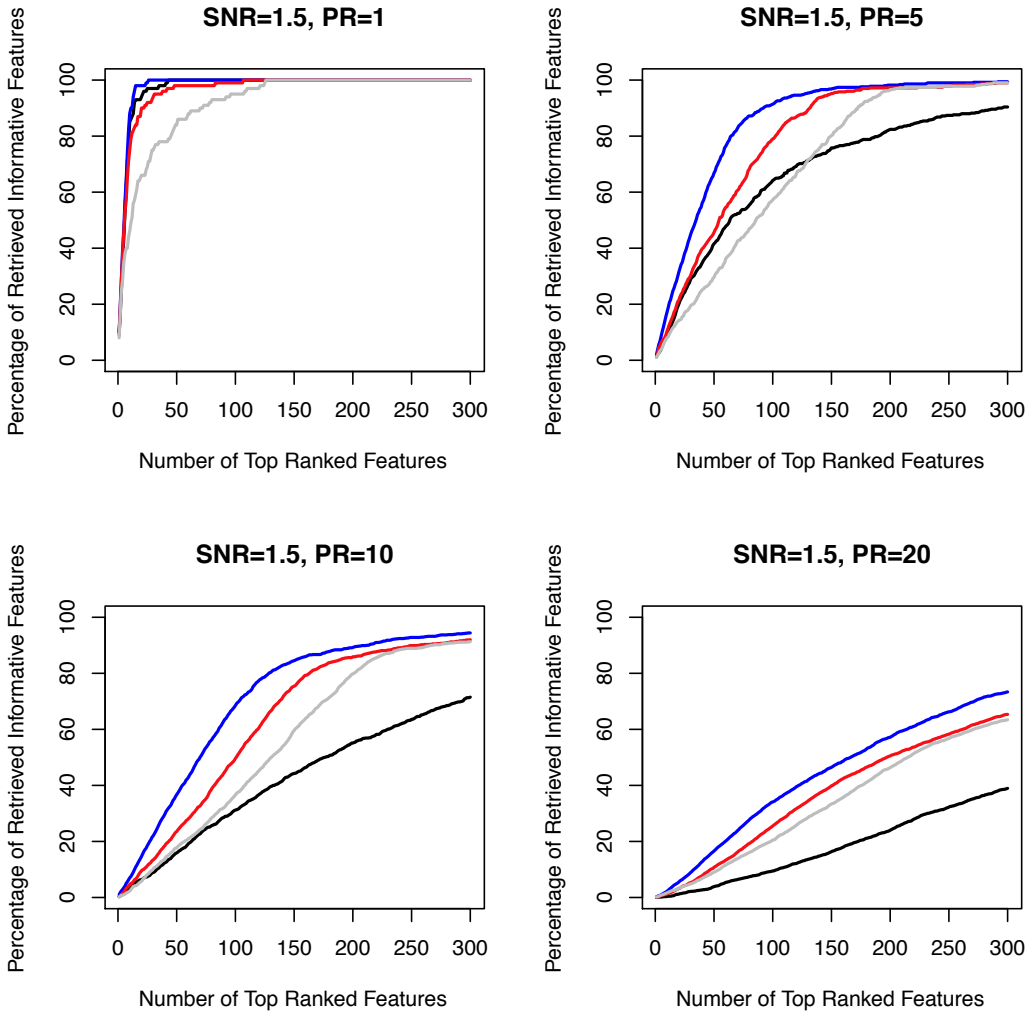

Figure 4.7: Informative feature retrieval efficiency at SNR = 1.5. Black line is for RFESVM, blue line for RSMRFESVM10, red line for RSMRFESVM25 and grey line for RSMRFESVM50

of RSMRFESVM25 and RSMRFESVM50 are stabler at higher SNRs. One interesting observation that is quite different from RSMLREN is that curves of both RSMRFESVM25 and RSMRFESVM50 are still improving beyond 200 iterations at SNR=1 and 1.5. The effect is more evident for RSMRFESVM50. A possible explanation is that more variation among subspaces is required to construct more effective feature scores to distinguish relevant features from irrelevant ones. As RSMRFESVM50 has the largest subspace size, more iterations are needed to achieve good variations, while for RSMRFESVM10 with the smallest subspace size, less iterations are needed. In fact, RSMRFESVM10 does not clearly display an upward trend beyond 100 or even 50 iterations. This also explains why RSMRFESVM50 is inferior to RFESVM in the retrieval power under high SNRs with low PRs (figure 4.6 and 4.7).

Figure 4.11 shows the selection similarity rate curves. Biggest improvements of RSMRFESVMs over RFESVM are at high PRs (10% and 20%) under SNR = 1 and SNR = 1.5. Other than those cases, RSMRFESVM is slightly better than RFESVM because at low PRs, high FPRs offset improvements in sensitivity rates. Since the FPRs are very high, a detailed analysis of the selection similarity in RFESVM and RSMRFESVM is less interesting.

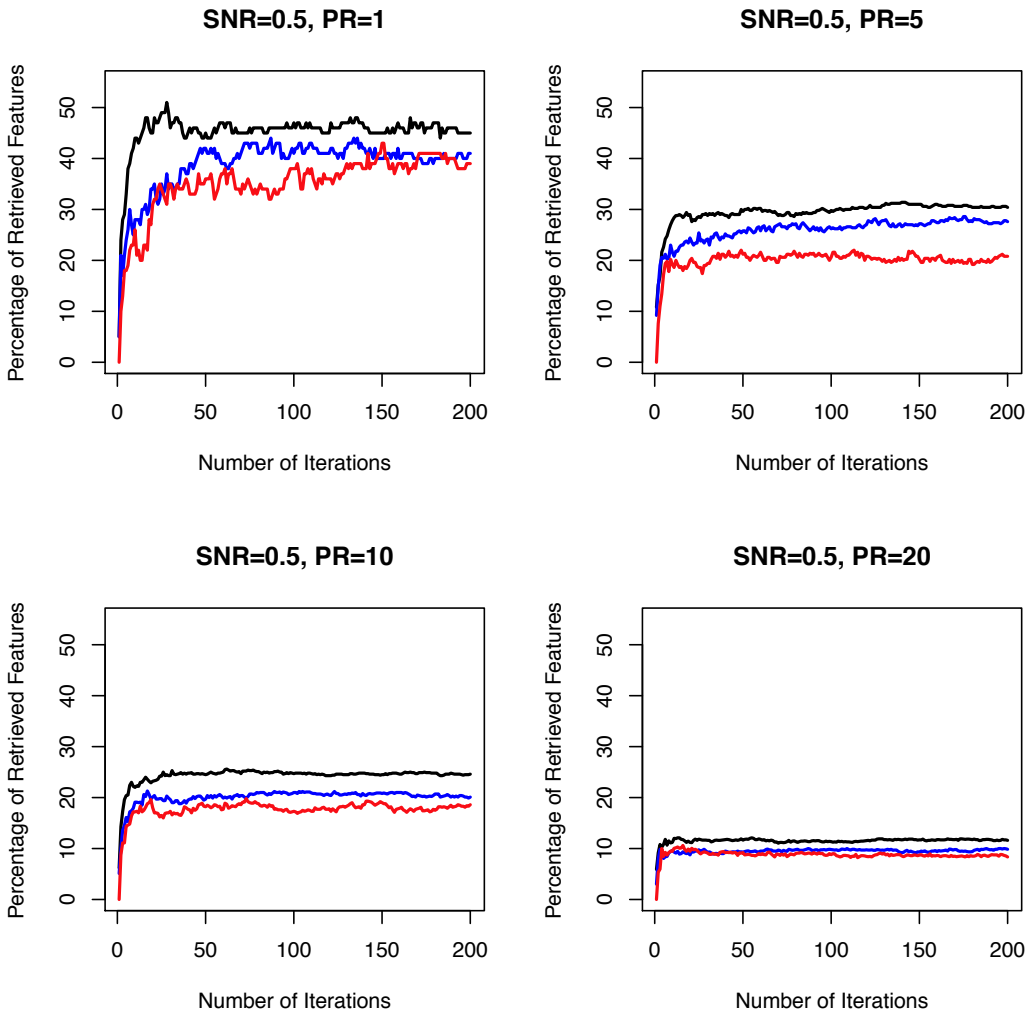

Figure 4.8: Effects of the number iterations on feature selection sensitivity rates, SNR=0.5. Black line is for RSMRFESVM10, blue line for RSMRFESVM25 and red line for RSMRFESVM50

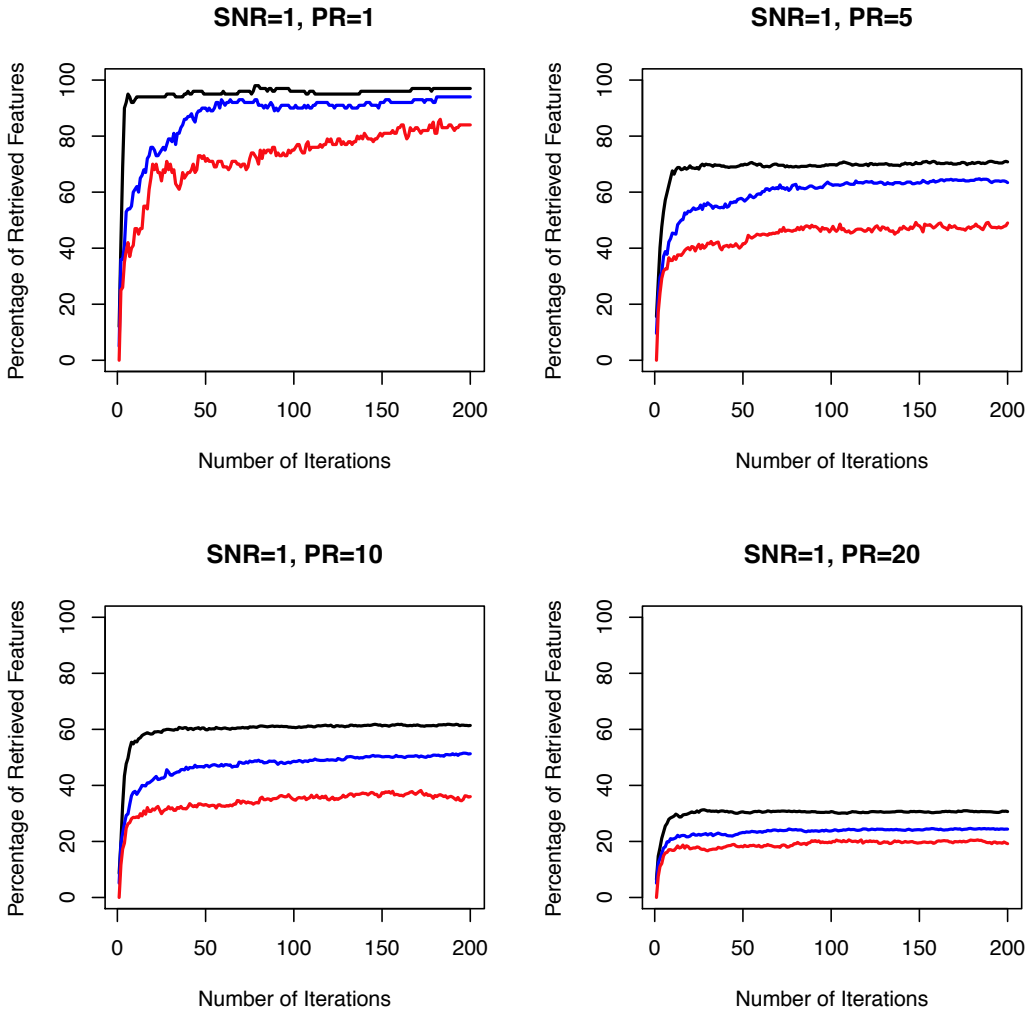

Figure 4.9: Effects of the number iterations on feature selection sensitivity rates, SNR=1. Black line is for RSMRFESVM10, blue line for RSMRFESVM25 and red line for RSMRFESVM50

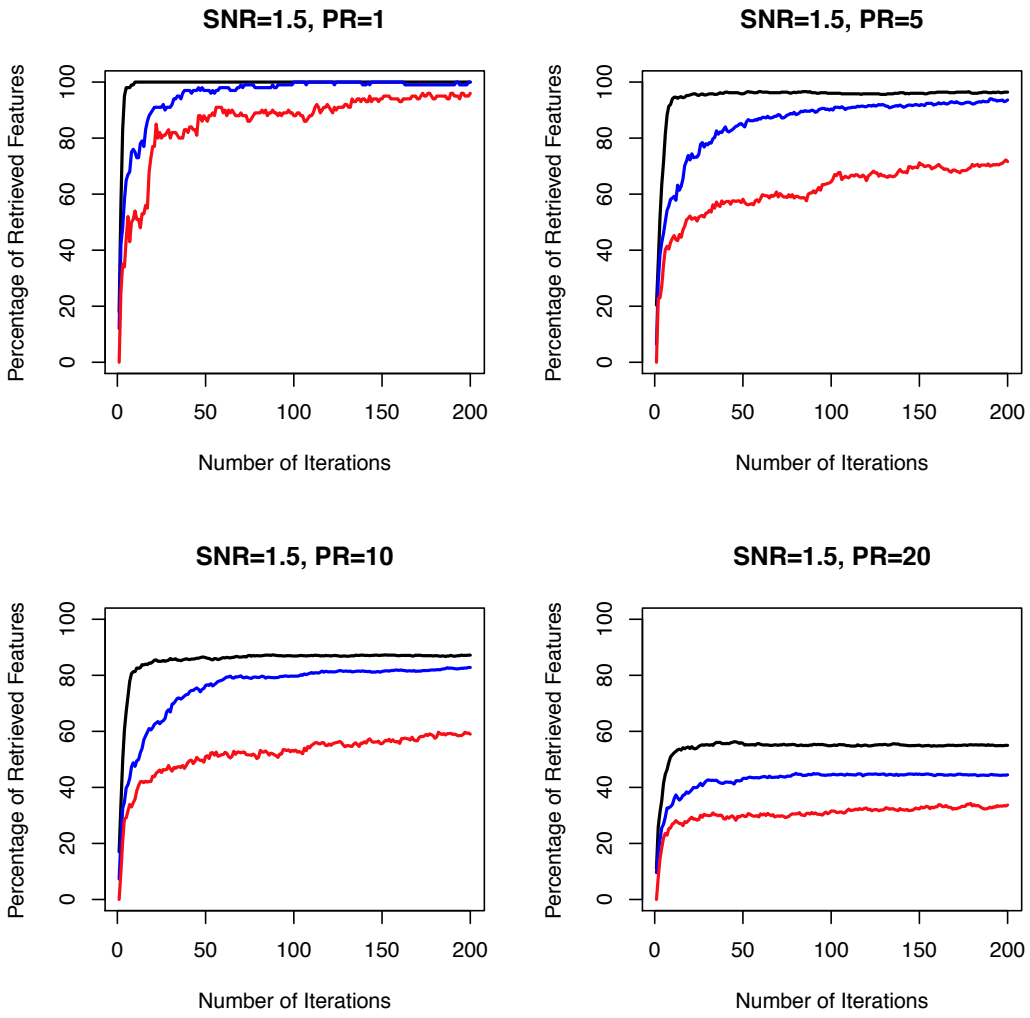

Figure 4.10: Effects of the number iterations on feature selection sensitivity rates, SNR=1.5. Black line is for RSMRFESVM10, blue line for RSMRFESVM25 and red line for RSMRFESVM50

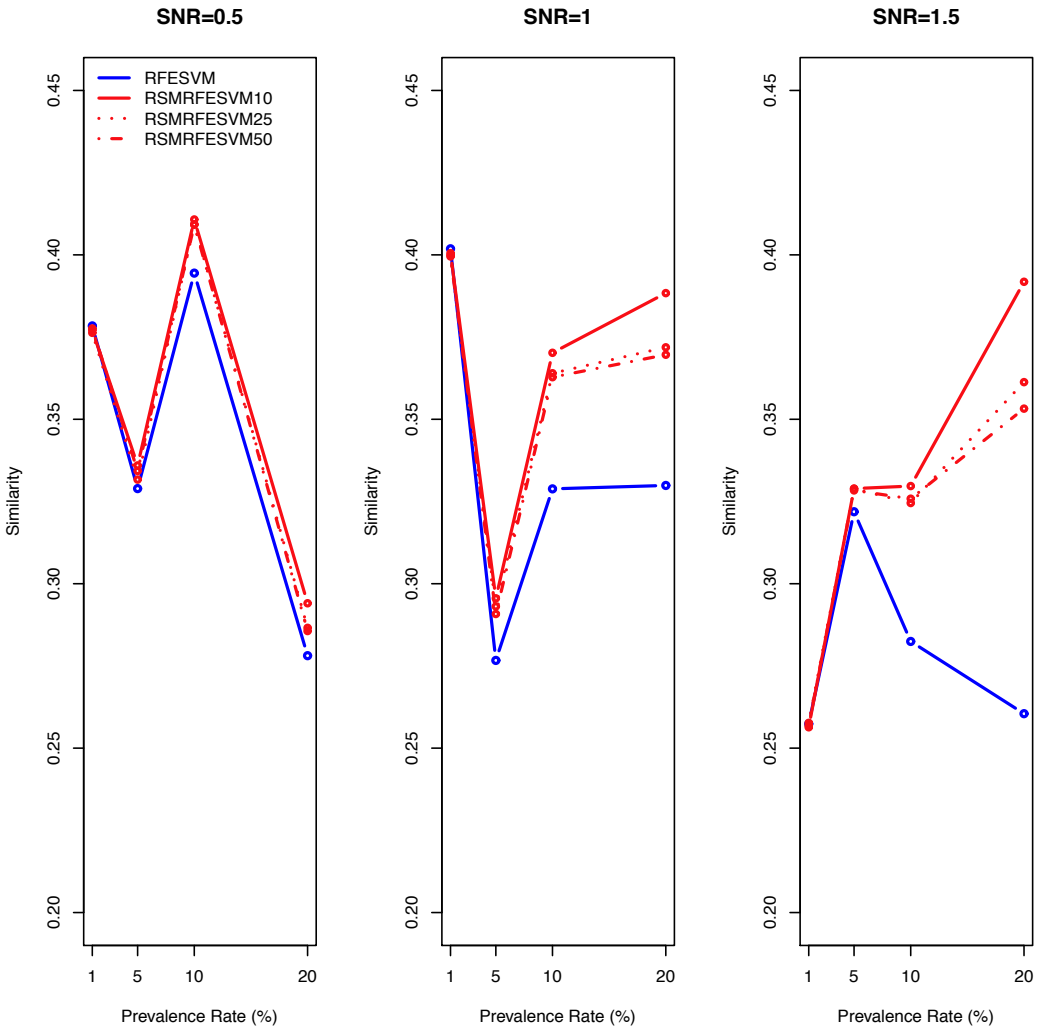

Figure 4.11: Feature selection: similarity rates after 200 iterations

Control of False Positive Rates

Table 4.5 tabulates false positive rates on the original noisy features using the threshold derived from scores of artificial variables after 200 iterations. I set three FPR target levels: FPR=1% (column 3), FPR=0.5% (column 4) and FPR=0.1% (column 5). Overall, the random probe method servers its purpose and controls the false positive rates close to the target levels, mostly within the one standard deviation bounds. However, for high PRs (PR = 10% and 20%), empirical FPRs tend to be conservative. As the PR increases, there is more mismatch between the number of original noisy features and the number of random probes which is kept as constant at 1000. In other words, it is likely to have a noisy random probe with relatively high discriminative power which dominates original noisy features and thus drives down the FPR estimate on original feature space. On the other hand, for low PRs, the gap is small and FPR estimates are better. More datasets are needed to mitigate this effect and drive false positive rates closer to target FPR levels.

I also examine the effects of the number of iterations on false positive rates. To be consistent across studies, figure 4.12 shows false positive rates as a function of the number of iterations for RSMRFESVM10 at SNR = 1. One notable observation is that as the PR increases, all three curves have an increasing downward trend below the corresponding FPR target levels. This is similarly observed in empirical FPRs after 200 iterations in the results above.

Optimization of the Number of Iterations and the Subspace Size

Subspace size has a similar effect observed in RSMLREN. A smaller subspace size leads to lower misclassification errors and higher sensitivity rates. However, as argued in the Chapter 3, the optimization of subspace size critically depends on the underlying data structure, which may require expert knowledge. With regards to the number of iterations, it is obvious that more iterations lead to stable minimal error and test error curves but it does not necessarily decrease the variance of adjusted error curves possibly because of the additional variance from bias estimation. However, the variance of adjusted error curves seems to be

Table 4.5: False positive rate control in RSMRFESVM

Method	PR	FPR = 1%	FPR = 0.5%	FPR = 0.1%
SNR = 0.5				
RSMRFESVM10	1%	0.0123 (0.0016)	0.0058 (0.0007)	0.0015 (0.0005)
	5%	0.0116 (0.0009)	0.0062 (0.0012)	0.0011 (0.0005)
	10%	0.0094 (0.0008)	0.0044 (0.0004)	0.0003 (0.0002)
	20%	0.0096 (0.0014)	0.0049 (0.0009)	0.0009 (0.0004)
RSMRFESVM25	1%	0.0095 (0.0013)	0.0044 (0.0006)	0.0010 (0.0004)
	5%	0.0099 (0.0017)	0.0051 (0.0014)	0.0004 (0.0002)
	10%	0.0109 (0.0013)	0.0059 (0.0011)	0.0006 (0.0003)
	20%	0.0108 (0.0022)	0.0049 (0.0011)	0.0008 (0.0004)
RSMRFESVM50	1%	0.0106 (0.0019)	0.0048 (0.0010)	0.0007 (0.0003)
	5%	0.0088 (0.0014)	0.0044 (0.0008)	0.0009 (0.0004)
	10%	0.0093 (0.0012)	0.0047 (0.0011)	0.0008 (0.0003)
	20%	0.0099 (0.0010)	0.0056 (0.0010)	0.0011 (0.0003)
SNR = 1				
RSMRFESVM10	1%	0.0109 (0.0012)	0.0048 (0.0006)	0.0010 (0.0002)
	5%	0.0112 (0.0015)	0.0058 (0.0010)	0.0008 (0.0003)
	10%	0.0087 (0.0010)	0.0040 (0.0005)	0.0001 (0.0001)
	20%	0.0081 (0.0012)	0.0031 (0.0006)	0.0004 (0.0002)
RSMRFESVM25	1%	0.0118 (0.0017)	0.0062 (0.0011)	0.0013 (0.0004)
	5%	0.0108 (0.0016)	0.0052 (0.0009)	0.0014 (0.0005)
	10%	0.0090 (0.0009)	0.0040 (0.0006)	0.0009 (0.0004)
	20%	0.0098 (0.0015)	0.0038 (0.0006)	0.0006 (0.0002)
RSMRFESVM50	1%	0.0115 (0.0020)	0.0062 (0.0016)	0.0006 (0.0003)
	5%	0.0091 (0.0015)	0.0052 (0.0008)	0.0014 (0.0004)
	10%	0.0101 (0.0013)	0.0058 (0.0008)	0.0010 (0.0003)
	20%	0.0086 (0.0013)	0.0046 (0.0010)	0.0004 (0.0003)
SNR = 1.5				
RSMRFESVM10	1%	0.0118 (0.0014)	0.0053 (0.0009)	0.0006 (0.0002)
	5%	0.0107 (0.0012)	0.0060 (0.0012)	0.0012 (0.0006)
	10%	0.0077 (0.0010)	0.0033 (0.0004)	0.0002 (0.0001)
	20%	0.0074 (0.0012)	0.0039 (0.0008)	0.0005 (0.0003)
RSMRFESVM25	1%	0.0113 (0.0013)	0.0071 (0.0010)	0.0019 (0.0006)
	5%	0.0101 (0.0013)	0.0061 (0.0011)	0.0009 (0.0003)
	10%	0.0097 (0.0014)	0.0041 (0.0011)	0.0006 (0.0004)
	20%	0.0089 (0.0007)	0.0040 (0.0009)	0.0004 (0.0004)
RSMRFESVM50	1%	0.0120 (0.0022)	0.0065 (0.0018)	0.0010 (0.0006)
	5%	0.0085 (0.0009)	0.0037 (0.0007)	0.0009 (0.0003)
	10%	0.0106 (0.0014)	0.0063 (0.0011)	0.0009 (0.0004)
	20%	0.0104 (0.0017)	0.0048 (0.0011)	0.0009 (0.0004)

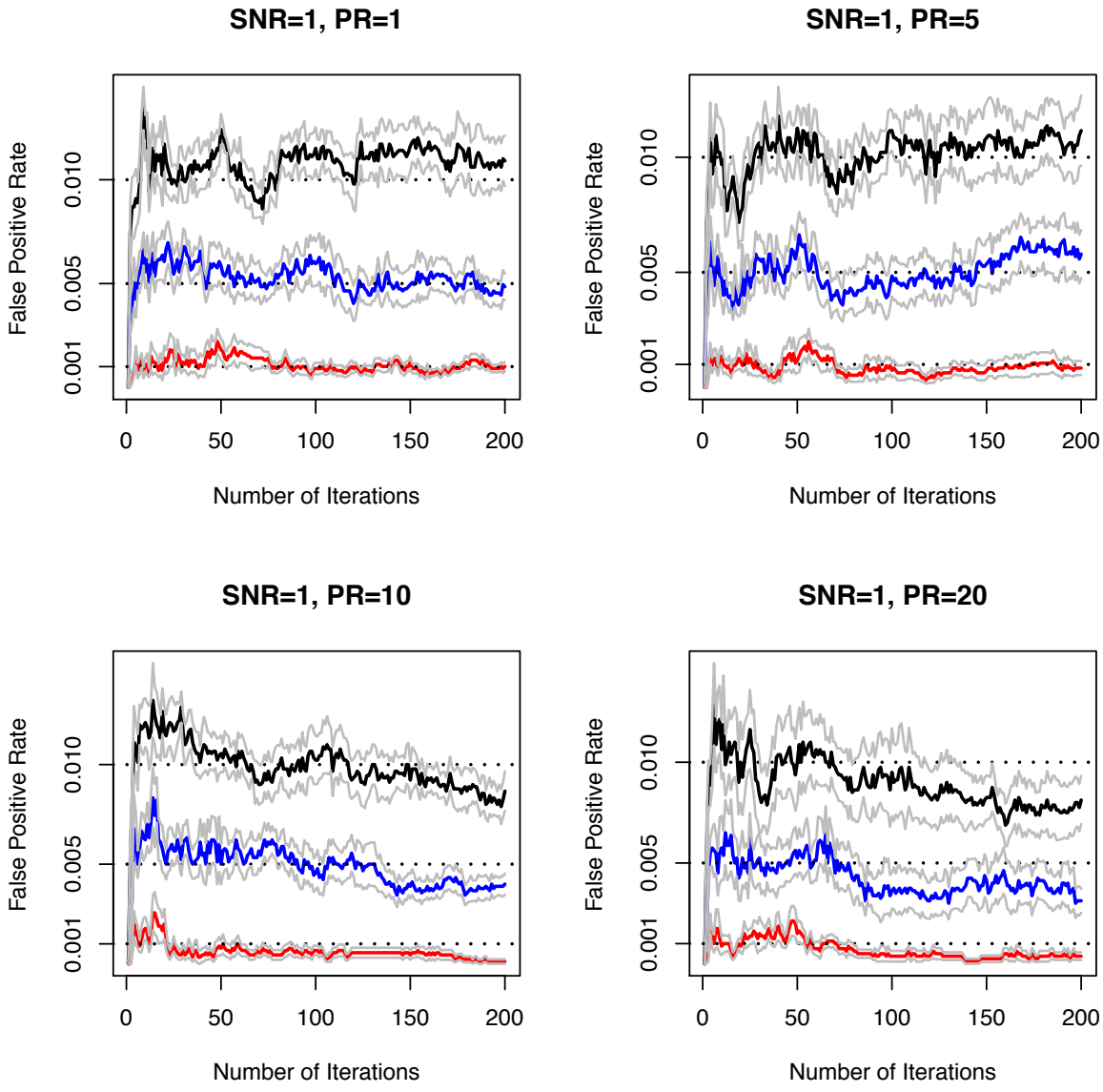

Figure 4.12: Effects of the number of iterations on false positive rates (RSMRFESVM10 at SNR = 1, FPR = 0.01: black line, FPR = 0.005: blue line, FPR = 0.001: red line

stable, and it is feasible to use less iterations to lower computation cost without sacrificing much classification performance. As for sensitivity rates, performance of RSMRFESVM10 is still stable with a less number of iterations but this is not the case for a higher subspace size (50%, RSMRFESVM50) because sensitivity rates are still improving even beyond 200 iterations. Overall, using less iterations is possible with RSMRFESVM10 but not with RSMRFESVM25 or RSMRFESVM50.

4.4.2 Second Simulation Study

The second simulation study aims to examine whether the feature scoring method defined in equation (2.10) can successfully rank informative features with varied SNRs for RSMRFESVM. To be complete, I also compare the feature selection sensitivity and similarity of RSMRFESVM to RFESVM in this study. Additionally, I investigate the performance of RSMRFESVM in false positive rate control. I use 200 iterations to compute feature scores. Misclassification errors are not reported here because all are driven very close to zeros due to the high SNRs.

Feature Ranking

For RSMRFESVM, I compute the average feature score for each informative region (containing 10 correlated features) and a total of 20 scores are obtained corresponding to 20 increasing SNR levels (from 0.1 to 2 with an increment of 0.1). For RFESVM, two approaches are used to compute the score. In the first approach, an average of SVM weights is used and in the second approach, the percentage of features being selected within each informative region is used. A Spearman rank correlation coefficient is computed for both RSMRFESVM and RFESVM. Table 4.6 presents the average correlation coefficient across 10 simulated datasets. RFESVM uses the original 1000 features and RFESVM2000 uses an augmented feature space with 2000 features. The rank coefficients within the parenthesis are computed using the second approach. Rankings from both RFESVM and RSMRFESVM are highly correlated with true rankings. There are two notable observations with respect to the rank coefficient computed with the first approach. First, LREN has a lower ranking quality than RFESVM. Second, unlike rankings in RSMLREN, the ranking accuracy in RSMRFESVM decreases with an increasing subspace size. These results can be explained as follows. Note that LREN has a much lower FPR and sensitivity rate (FPR=0.0005, sensitivity rate=0.286) than RFESVM (FPR=0.2377, sensitivity rate=0.784, see below in feature selection performance). In LREN, most informative features have zero coefficients. But in RFESVM, most informative features have nonzero coefficients and their rankings can better

reflect features' discriminative powers in terms of SNR magnitudes. Therefore, RFESVM is expected to have a better ranking performance than LREN. For RSMRFESVM, in a subspace, selected features are treated to be equal (same feature scores) and score difference only originates from the difference between subspaces. As more subspaces are sampled, difference between informative features with varied SNRs are more distinguished. Since 2000 subspaces are sampled in RSMRFESVM10 compared to 800 in RSMRFESVM25 and 400 subspaces in RSMRFESVM50, the feature ranking in RSMRFESVM10 is expected to better than RSMRFESVM25 and RSMRFESVM50. For the second rank coefficients of RFESVM and RFESVM2000 (values in the parenthesis), the rank coefficient of RSMRFESVM converges to RFESVM2000 when the subspace size increases to 100% because both RSMRFESVM and RFESVM2000 operate with the same feature space and use a similar approach to calculate the region score (percentage of features of being selected within an informative region)

Table 4.6: Feature ranking in RFESVM and RSMRFESVM. RFESVM uses 1000 original features and RFESVM2000 uses an augmented feature space with 2000 features

Method	**Spearman Rank Correlation Coefficient**
RFESVM	0.918 (0.842)
RFESVM2000	0.939 (0.787)
RSMRFESVM10	0.929
RSMRFESVM25	0.874
RSMRFESVM50	0.830

Feature Selection Performance

Comparisons are made by first matching FPRs of RSMRFESVM to RFESVM. The average FPR is 0.2377. Table 4.7 presents the averaged selection sensitivity and similarity rates across all SNR levels. RSMRFESVMs have higher power in selecting informative features than RFESVM, and the power decreases with an increasing subspace size. RSMRFESVMs also have slightly higher feature selection similarity rates than RFESVM. Moreover, I decompose the average sensitivity rate into multiple SNR levels (20 levels). The results are

shown in figure 4.13. Three RSMRFESVM methods have higher power than RFESVM at low to median SNRs (0.5-1.0). Since the FPR is very high, sensitivity rates at high SNRs (> 1.0) are very close, and the improvements from RSMRFESVM at those SNRs are small. On the other hand, it is very difficult to detect features with very low SNRs (< 0.5) and improvements are also very small. Among three subspaces, there is not much difference in sensitivity rates.

Table 4.7: Selection sensitivity and similarity rate of RFESVM and RSMRFESVM: second simulation study

Method	**Selection Sensitivity Rate**	**Selection Similarity Rate**
RFESVM	0.784	0.355
RSMRFESVM10	0.833	0.380
RSMRFESVM25	0.828	0.378
RSMRFESVM50	0.827	0.376

False Positive Rate Control

I set three target FPR levels (FPR = 0.01, 0.005, 0.001) as in the first simulation study. Table 4.8 summaries the empirical FPRs. Overall, RSMRFESVM has a good FPR control under all three subspace sizes. The empirical FPRs are very close to target levels. However, the conservativeness of empirical FPRs is not clearly observed in this study. It is interesting to further evaluate this with more datasets.

Table 4.8: Empirical FPRs of RSMRFESVM: second simulation study

Method	**FPR=0.01**	**FPR=0.05**	**FPR=0.001**
RSMRFESVM10	0.0115 (0.0015)	0.0062 (0.0011)	0.0011 (0.0004)
RSMRFESVM25	0.0098 (0.0014)	0.0036 (0.0004)	0.0016 (0.0006)
RSMRFESVM50	0.0089 (0.0018)	0.0042 (0.0009)	0.0005 (0.0002)

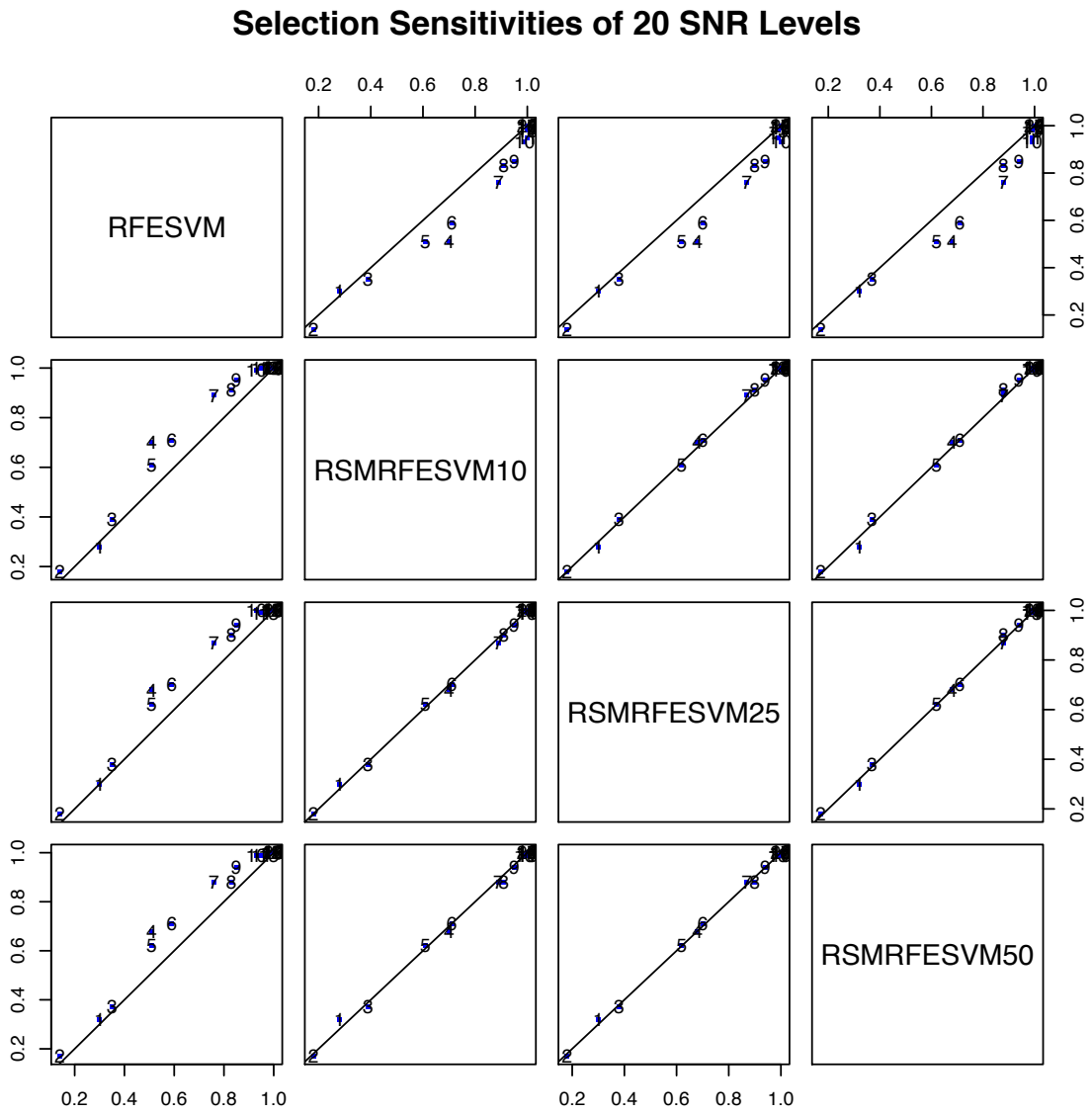

Figure 4.13: Selection sensitivity rates at 20 SNR levels. The numbers reflect SNR levels. 1: SNR=0.1, 20: SNR=2

4.4.3 Third Simulation Study

In this simulation study, the first 5 SNR levels are varied from 0.5 to 1.5, with an increment of 0.25 and the next 5 levels from -0.5 to -1.5, with an decrement of 0.25. I first examine whether the approach to determine feature's discrimination directions is successful. I also examine selection sensitivity and similarity in RSMRFESVM with comparisons to RFESVM. FPR control in RSMRFESVM is also evaluated. I use 200 iterations to compute feature scores.

Discrimination Direction

For each feature, I first compute the percentage of a positive weight sign compared to all cases in which the feature is selected. Then I average the percentages within a region (10 features of the same SNR). The first 5 SNRs favor class 2 and are expected to have higher percentages (SNRs are in an increasing order: from 0.5 to 1.5). The next 5 SNRs favor class 1 and are expected to lower percentages (SNRs are in a decreasing order: from -0.5 to -1.5). Table 4.9 presents the results for both RSMRFESVM and RFESVM. Overall, all four methods can reliably determine a feature's discrimination direction. As the SNR increases, there are less wrong signs. There is not much difference between RSMRFESVM with different subspace sizes.

Table 4.9: Feature's discrimination directions in RFESVM and RSMRFESVM. The first 5 regions have SNRs from 0.5 to 1.5 and the next 5 regions have SNRs from -0.5 to -1.5.

Method	Percentage of a Positive Weight Sign
RFESVM	{ 93.3 100.0 100.0 100.0 100.0; 0.0 0.0 0.0 0.0 0.0 }
RSMRFESVM10	{ 94.1 96.7 100.0 100.0 100.0; 8.2 0.2 0.3 0.0 0.0 }
RSMRFESVM25	{ 94.2 96.9 100.0 100.0 100.0; 7.8 0.0 0.1 0.0 0.0}
RSMRFESVM50	{ 95.2 98.0 100.0 100.0 100.0; 8.5 0.0 0.1 0.0 0.0 }

Feature Selection Sensitivity

FPRs of RSMRFESVM methods are matched to RFESVM and the average FPR is

0.3109. Table 4.10 summaries the selection sensitivity and similarity rates for both methods. RSMRFESVM methods have higher sensitivity rates than RFESVM, though their similarity rates are just slightly higher. I also examine the sensitivity rates at different SNR levels. Figure 4.14 presents the decomposed sensitivity rates. Compared to RFESVM, RSMRFESVM has higher rates at low and median SNRs but relatively the same rates at high SNRs. The results are similarly observed in the second simulation study.

Table 4.10: Selection sensitivity and similarity rate of RFESVM and RSMRFESVM

Method	**Selection Sensitivity Rate**	**Selection Similarity Rate**
RFESVM	0.856	0.285
RSMRFESVM10	0.894	0.294
RSMRFESVM25	0.892	0.295
RSMRFESVM50	0.889	0.294

False Positive Rate Control

Table 4.11 presents the average empirical FPRs at three target levels (FPR = 0.01, 0.005, 0.001). Overall, similar to previous simulation studies, empirical FPRs are pretty close to the desired levels. In this study, similar to the second simulation study, the empirical FPRs do not seem to be conservative estimates of target FPRs.

Table 4.11: Empirical FPRs of RSMRFESVM

Method	**FPR=0.01**	**FPR=0.05**	**FPR=0.001**
RSMRFESVM10	0.0099 (0.0012)	0.0044 (0.0010)	0.0011 (0.0005)
RSMRFESVM25	0.0090 (0.0015)	0.0043 (0.0010)	0.0009 (0.0003)
RSMRFESVM50	0.0088 (0.0015)	0.0044 (0.0010)	0.0012 (0.0005)

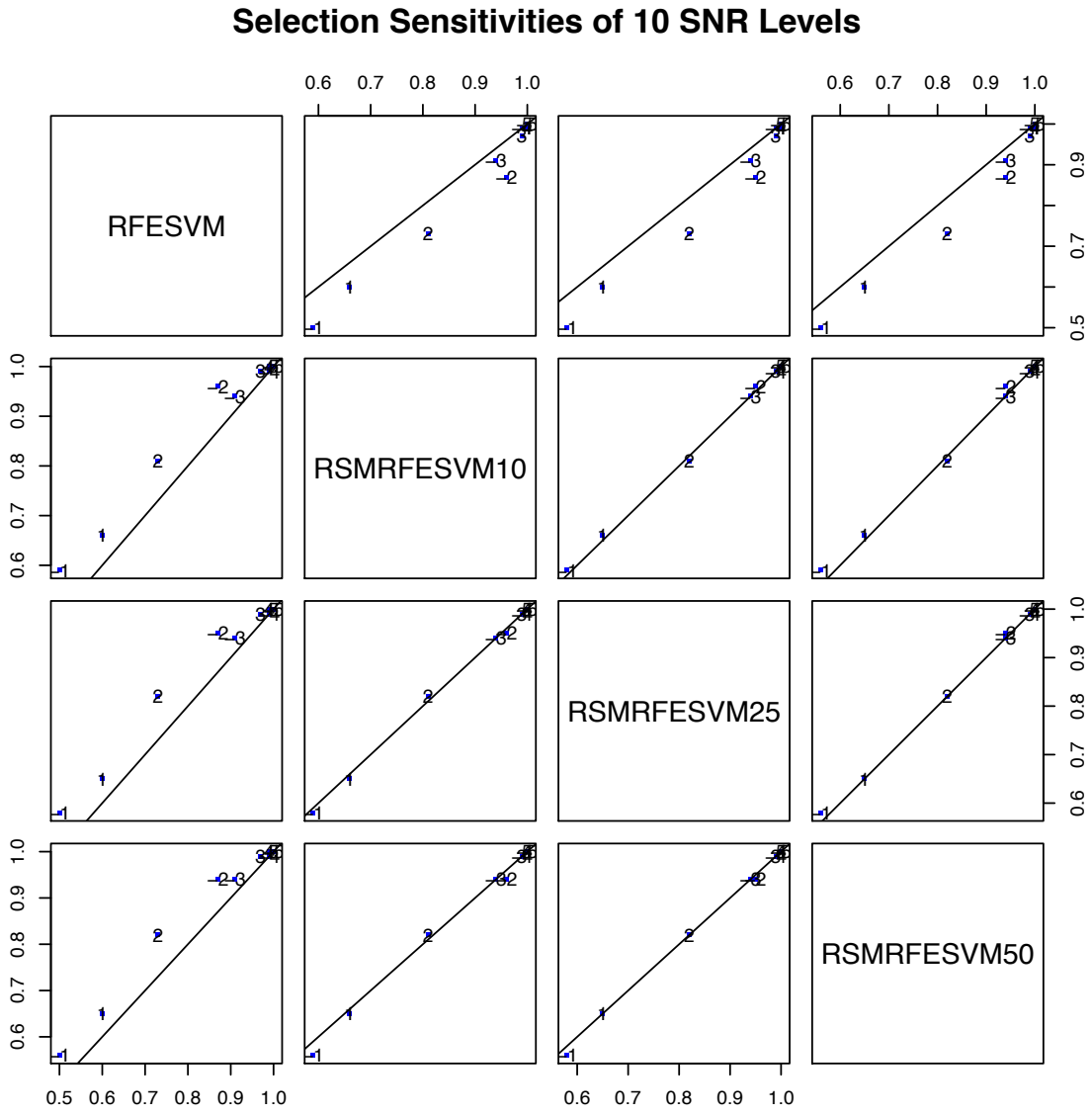

Figure 4.14: Selection sensitivity rates at 10 SNR levels. The number reflect SNR levels. 1: SNR=0.5, 5: SNR=1.5, -1: SNR=-0.5, -5: SNR=-1.5

4.5 Discussion of the Three Simulation Studies

In general, RSMRFESVM improves RFESVM in both classification and feature selection, due to the similar reasons in RSMLREN. In the rest of discussions, I focus on the comparisons between RSMLREN and RSMRFESVM.

Compared to LREN, RFESVM lacks of the ability to effectively discard noisy features due to the low elimination levels (10 levels) and a small elimination rate (10%). Thus, FPRs of RFESVM are much higher than FPRs of LREN. Though it is possible to increase elimination levels and further decrease the elimination rate, the scheme is computationally infeasible. RSMRFESVM is able to overcome this drawback by providing a control over the FPR. However, RSM methods does depend on the feature selection performance in each subspace. For example, in RSMRFESVM, since RFESVM induces a high FPR in each subspace, noisy features are treated to be equal to informative features in calculating feature scores. Therefore, more subspaces are required to distinguish informative features from irrelevant ones. This effect is clearly demonstrated in the first simulation study of RSMRFESVM (figure 4.6 and 4.7). Sensitivity rate curves of larger subspace sizes (25%, 50%) are still improving after 200 iterations. In contrast, RSMLREN does not seem to have this problem probably due to the higher specificity in feature selection. For example, the sensitivity rate curves of all three subspace sizes are relatively stable even after 100 iterations (figure 3.5, 3.6, 3.7). It still remains unclear whether RSMLREN performs better than RSMRFESVM in feature selection. A future research topic is to fully compare the ROC curves of the two methods.

This also relates to the issue of optimizing the subspace size and the number of iterations. Since it is difficult to choose an optimal subspace size, it is much safer to use more iterations in real fMRI data, probably beyond 200 when using RSMRFESVM. The parallel nature of the RSM framework can ease the computation burden. Of course, an adaptive scheme described in section 3.5 can be employed.

4.6 Summary and Conclusions

In this chapter, I empirically evaluate the performance of the developed RSM framework using RFESVM as its base classifier. I obtain several interesting results. For classification, RSMRFESVM has a lower test error than RFESVM in most cases. Minimal errors from both methods all have downward bias for test errors, though RFESVM has a much smaller bias. Bias of RFESVM is sensitive to tuning parameter values. The adjusted errors have less bias than minimal errors for RSMRFESVM. SNR plays a more dominant role in reducing errors than PR. For feature selection, PR has a negative effect such that higher PR results in lower selection sensitivity. In most cases, RSMRFESVM has a large improvement on RFESVM in terms of selection sensitivity and similarity, especially under higher PRs. For three target levels on the false positive rate, RSMRFESVM is able to keep the empirical FPRs pretty close to targets. Additionally, RSMRFESVM is able to correctly rank informative features based on their individual discriminative capacity, though RFESVM performs equally well. Moreover, RSMRFESVM can also correctly determine features' discrimination directions. Finally, I find that the number of iterations/subspaces is critical for RSMRFESVM50's feature selection performance. To encourage subspace diversity, more iterations are needed for RSMRFESVM50.

Compared to LREN, RFESVM has a much smaller search space for tuning parameters. That means RSMLREN is more capable to adapt to local data structure. Thus, for RSMLREN, a reduced number of iterations is feasible without much decrease in classification and feature selection performance. However, for RSMRFESVM, to ensure a good performance, more iterations are needed, especially for a large subspace size.

Chapter 5: Summary and Future Work

This dissertation contributes to the literature in two aspects. Methodologically, I develop a unified framework in which the random subspace method (RSM) operates as both an ensemble classifier and a feature selection method. Empirically, I evaluate the developed RSM framework combined with two state-of-the-art classifiers in fMRI literature (logistic regression with elastic net (LREN) and recursive feature elimination using support vector machine(RFESVM)).

Bias Correction and Feature Selection in Random Subspace Method

In this chapter, I extend the bias correction method in Tibshirani and Tibshirani (2009) to the case of ensemble classifiers formed by RSM. For ensemble classifiers with tunable base classifiers, the extension evades the need of a double loop cross validation that is commonly employed to avoid the parameter selection bias (Varma and Simon, 2006). It is computationally efficient by using information all available in cross validation folds. To discern relevant features from irrelevant ones, I develop a random probe method by adding permuted artificial variables. Using the cumulative distribution of scores of artificial variables, I empirically determine a threshold on the ranking list of real features. The threshold is data-dependent and does not require any expert knowledge. Finally, I integrate the bias correction and feature selection method into a unified RSM framework. I also discuss some nice properties of the proposed novel feature scoring method and the high parallelism of the RSM framework.

Application to Logistic Regression with an Elastic Net Penalty

In this chapter, using Monte Carlo simulations, I first evaluate several small sample properties of logistic regression with an elastic net penalty (LREN, Zou and Hasite, 2005;

Ryali et al., 2010). I then use LREN as a base classifier in the RSM framework (RSMLREN) and compare it to LREN in classification and feature selection performance. The false positive rate control method is evaluated against several target rates. The effects of the number of subspaces and subspace size are discussed. Finally, I examine the feature ranking performance of RSMLREN and also whether RSMLREN is able to correctly differentiate features's discrimination directions.

I obtain several important empirical results. Signal-to-noise ratio (SNR) plays a critical role in improving classification accuracy, while increasing prevalence rate (PR) of informative features has marginally decreasing benefits. RSMLREN has a slightly higher classification accuracy than LREN, which conforms to the study in Kuncheva et al. (2010). Both bias correction methods (for LREN and RSMLREN) perform well and in most cases provide relatively conservative estimates. In feature selection, for LREN, sensitivity rate decreases dramatically with increasing PR, which has not been documented in related literature. This point to the limitation of LREN as a classifier in a region of interest (ROI) analysis where a higher percentage of features is likely to be informative. Though sensitivity of RSMLREN also decreases with increasing PR, the rate is much slower than LREN. Largest improvements of RSMLREN are observed at higher PRs. Thus RSMLREN is flexible for a whole brain analysis as well an ROI analysis. In addition, RSMLREN is an accurate feature ranking method and scores of informative features from RSMLREN are highly correlated with their true ranks. Moreover, I find that RSMLREN is able to correctly determine each informative feature's discrimination direction, which is not discussed in RSM related literature. This property has important implications in fMRI study because brain regions may take totally different roles although they are all together engaged in a mental task. In controlling false positives, the threshold derived from artificial variables is quite accurate with target rates falling into the one standard deviation error bounds in most cases. Comparing among three subspace sizes, 10% of the original feature size performs best in most cases. Increasing the number of subspaces does not significantly decrease variance of adjusted classification errors but it stabilizes feature selection performance. Less number of subspaces may be

used for less computation time without scarifying much classification and feature selection accuracy.

Application to Recursive Feature Elimination Using Linear Support Vector Machine

In this chapter, I follow the same evaluation procedures in comparing RSMLREN with LREN. In RSMRFESVM, similar to RSMLREN, I find SNR has a much more important role in determining classification accuracy than PR. Contrary to the prevalent improvements of RSMLREN over LREN in classification accuracy, I find in the lowest SNR and PR, RFESVM has a slightly higher classification accuracy than RSMRFESVM. With a small feature elimination rate, bias of minimal CV error of RFESVM is tiny but with a larger rate, large bias shows up, indicating the sensitivity of bias to tuning parameter values. The bias correction method for RSMRFESVM shrinks the bias and provides conservative estimates especially at low SNRs. In feature selection, due to a small number of elimination levels and low elimination rate, RFESVM has very high false positive rates. Decreasing sensitivity curves are also observed in RFESVM and RSMRFESVM. But RSMRFESVM is more robust to PR variations. I also find that the superior performance of RSMRFESVM in feature selection is similarly observed at lower false positive rates. Similar to RSMLREN, RSMRFESVM has a good feature ranking performance and can differentiate feature's discrimination direction. 10% subspace size performs much better than the other two subspace sizes. Likewise, less subspaces may be used for the sake of less computation load for small subspace sizes. However, for a large subspace size (50%), more iterations may be required for better and stable performance.

Future Research Directions

There are several possible future research directions that I have not pursued in this dissertation. Methodologically, besides the random sampling in the feature space, random sampling in sample space is also feasible. Recently, Meinshausen and Buhlmann (2010) developed a stability selection method using repeated subsamples. They showed that the

stability selection method has finite sample control over false positive rates (feature-wise and family-wise) and is insensitive to the proper amount of regularization for structure estimation. However, it uses the whole feature space and likewise suffers from high dimensionality. As shown in discussions on the paper, stability selection may suffer from power lose compared to other methods (e.g. sure independence screening, Fan and Lv, 2008). It is interesting to explore whether RSM can improve power of the stability selection method while retaining its advantages in control of false positives. The double randomness resembles the Random Forest algorithm (Breiman, 2001) but its effect on classifiers other than tree based methods are unknown. The double randomness scheme inevitably incurs much more computation time and one fruitful direction is to use GPU (graphic processing unit) programing technique, which is highly parallel and suitable for operations on random subspaces. Second, in the majority voting, all subspaces are employed. However, some subspaces may only contain irrelevant features. Those subspaces may degrade the overall classification performance. This issue is more evident for a large-scale dataset with a high dimensional feature space containing a very small number of informative features (e.g. spatiotemporal fMRI data). It is beneficial to further identify and discard or de-weight those useless subspaces. A possible solution would apply a similar idea of the boosting algorithm into the feature space.

Empirically, in this work, I mainly focus on the spatial data patterns in fMRI data. A further evaluation of the developed RSM in spatiotemporal fMRI data is interesting and should be fruitful because brain functions operates both temporally and spatially. As the feature dimension of spatiotemporal data is even larger, classical methods (LREN and RFESVM) will be further suffered. Second, in this dissertation, I look at the discriminative ability of voxels in fMRI data. Recently, there is an increasing interest in brain networks (correlation between voxels) and using voxel connections to discriminate brain states (Richiardi et al., 2010). Classifying correlations faces a even higher feature dimension and the RSM framework may find very interesting applications. However, a more

challenging research direction is to explore whether RSM is useful in recovering a true underlying partial correlation structure. It may require a new scoring method to integrate results from subspaces.

Bibliography

Bibliography

[1] P. Bandettini, A. Jesmanowicz, E. Wong, and J. Hyde, "Processing strategies for timecourse data sets in functional mri of the human brain," *Magnetic Resonance in Medicine*, vol. 30, no. 2, pp. 161–173, 1993.

[2] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[3] R. Cai, Z. Hao, and W. Wen, "A novel gene ranking algorithm based on random subspace method," *Neural Networks, 2007. IJCNN 2007. International Joint Conference on*, pp. 219–223, 2007.

[4] M. Carroll, G. Cecchi, I. Rish, R. Garg, and A. Rao, "Prediction and interpretation of distributed neural activity with sparse models," *NeuroImage*, vol. 44, no. 1, pp. 112–122, 2009.

[5] D. D. Cox and R. L. Savoy, "Functional magnetic resonance imaging (fmri) "brain reading": detecting and classifying distributed patterns of fmri activity in human visual cortex," *NeuroImage*, vol. 19, no. 2, pp. 261 – 270, 2003.

[6] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Mach. Learn. Res.*, vol. 3, pp. 1157–1182, March 2003.

[7] K. Friston, A. Holmes, K. Worsley, J. Poline, C. Frith, and R. Frackowiak, "Statistical parametric maps in functional imaging: a general linear approach," *Human Brain Mapping*, vol. 2, no. 4, pp. 189–210, 1994.

[8] L. Grosenick, S. Greer, and B. Knutson, "Interpretable classifiers for fmri improve prediction of purchases," *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 16, no. 6, pp. 539 –548, 2008.

[9] L. K. Hansen and P. Salamon, "Neural Network Ensembles," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, pp. 993–1001, 1990.

[10] S. J. Hanson and Y. O. Halchenko, "Brain reading using full brain support vector machines for object recognition: There is no face identification area," *Neural Computation*, vol. 20, no. 2, pp. 486–503, 2008.

[11] T. Hastie, S. Rosset, R. Tibshirani, and J. Zhu, "The entire regularization path for the support vector machine," *J. Mach. Learn. Res.*, vol. 5, pp. 1391–1415, December 2004.

[12] J.-D. Haynes and G. Rees, "Predicting the orientation of invisible stimuli from activity in human primary visual cortex," *Nature Neuroscience*, vol. 8, no. 5, pp. 686–691, 2005.

[13] T. K. Ho, "Random decision forests," in *Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on*, vol. 1, Aug. 1995, pp. 278 –282 vol.1.

[14] ——, "The random subspace method for constructing decision forests," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, no. 8, pp. 832–844, 1998.

[15] R. Kohavi, "Wrappers for feature subset selection," *Artificial intelligence*, 1997.

[16] L. Kuncheva and J. Rodríguez, "Random subspace ensembles for fMRI classification," *Medical Imaging*, 2010.

[17] S. LaConte, S. Strother, V. Cherkassky, J. Anderson, and X. Hu, "Support vector machines for temporal classification of block design fmri data," *NeuroImage*, vol. 26, no. 2, pp. 317 – 329, 2005.

[18] C. Lai, M. Reinders, and L. Wessels, "Random subspace method for multivariate feature selection," *Pattern Recognition Letters*, vol. 27, no. 10, pp. 1067–1076, 2006.

[19] F. de Martino, G. Valente, N. Staeren, J. Ashburner, R. Goebel, and E. Formisano, "Combining multivariate voxel selection and support vector machines for mapping and classification of fMRI spatial patterns," *NeuroImage*, vol. 43, no. 1, pp. 44–58, 2008.

[20] M. J. McKeown, S. Makeig, G. G. Brown, T. P. Jung, S. S. Kindermann, A. J. Bell, and T. J. Sejnowski, "Analysis of fMRI data by blind separation into independent spatial components," *Human Brain Mapping*, vol. 6, pp. 160–188, 1998.

[21] T. M. Mitchell, R. Hutchinson, R. S. Niculescu, F. Pereira, X. Wang, M. Just, and S. Newman, "Learning to decode cognitive states from brain images," *Mach. Learn.*, vol. 57, pp. 145–175, 2004.

[22] J. Mourao-Miranda, A. L. W. Bokde, C. Born, H. Hampel, and M. Stetter, "Classifying brain states and determining the discriminating activation patterns: Support Vector Machine on functional MRI data," *NeuroImage*, vol. 28, no. 4, pp. 980–995, 2005.

[23] J. Mourao-Miranda, E. Reynaud, F. McGlone, G. Calvert, and M. Brammer, "The impact of temporal compression and space selection on svm analysis of single-subject and multi-subject fmri data," *NeuroImage*, vol. 33, no. 4, pp. 1055 – 1065, 2006.

[24] K. Norman, S. Polyn, G. Detre, and J. Haxby, "Beyond mind-reading: multi-voxel pattern analysis of fMRI data," *Trends in Cognitive Sciences*, vol. 10, no. 9, pp. 424–430, 2006.

[25] S. Ryali, K. Supekar, D. A. Abrams, and V. Menon, "Sparse logistic regression for whole-brain classification of fMRI data," *NeuroImage*, vol. 51, no. 2, pp. 752–764, 2010.

[26] M. Skurichina and R. Duin, "Bagging and the random subspace method for redundant feature spaces," *Multiple Classifier Systems*, pp. 1–10, 2001.

[27] ——, "Bagging, boosting and the random subspace method for linear classifiers," *Pattern Analysis and Applications*, vol. 5, no. 2, pp. 121–135, 2002.

[28] D. Sona and P. Avesani, "Feature Rating by Random Subspaces for Functional Brain Mapping," *Brain Informatics*, pp. 112–123, 2010.

[29] N. Staeren, H. Renvall, F. De Martino, R. Goebel, and E. Formisano, "Sound Categories Are Represented as Distributed Patterns in the Human Auditory Cortex," *Current Biology*, vol. 19, no. 6, pp. 498–502, 2009.

[30] H. Stoppiglia, G. Dreyfus, R. Dubois, and Y. Oussar, "Ranking a random feature for variable and feature selection," *The Journal of Machine Learning Research*, vol. 3, pp. 1399–1414, 2003.

[31] S. C. Strother, I. Kanno, and D. A. Rottenberg, "I. Principal Component Analysis, Variance Partitioning, and "Functional Connectivity"," *J Cereb Blood Flow Metab*, vol. 15, pp. 353–360, 1995.

[32] R. J. Tibshirani and R. Tibshirani, "A bias correction for the minimum error rate in cross-validation," *The Annals of Applied Statistics*, vol. 3, no. 2, pp. 822–829, 2009.

[33] E. Tuv, A. Borisov, G. Runger, and K. Torkkola, "Feature selection with ensembles, artificial variables, and redundancy elimination," *The Journal of Machine Learning Research*, vol. 10, pp. 1341–1366, 2009.

[34] S. Varma and R. Simon, "Bias in error estimation when using cross-validation for model selection," *BMC Bioinformatics*, vol. 7, no. 1, p. 91, 2006.

[35] L. Wang and J. Zhu, "The doubly regularized support vector machine," *Statistica Sinica*, no. 589–615, 2006.

[36] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 67, no. 2, pp. 301–320, 2005.

Curriculum Vitae

Tianwen Chen received a Bachelor of Arts in Economics and a Bachelor of Science in Mathematics from Shanghai Jiaotong University, Shanghai, China, in July 2005. He spent the summers in 2009 and 2010 as a Research Assistant in the Stanford Cognitive and Systems Neuroscience Laboratory at Stanford University, Stanford, CA, developing statistical methods for fMRI data analysis. He has worked as a Research Assistant at the Interdisciplinary Center for Economic Science since 2005.