

Department of Computer Sciences



**A Comparative Analysis of Feature Selection Techniques for a
Family of Nonlinear Target Functions and Breast Cancer
Diagnoses**

**by
Nicholas Sean Arellano Escanilla**

This thesis is submitted in partial fulfillment of the requirements for the degree of Master of
Science at The University of Wisconsin-Madison.

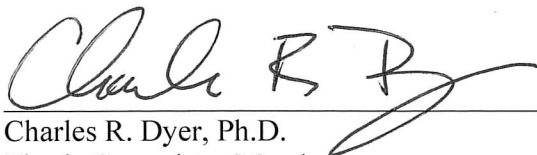
August 2017

We hereby certify that this thesis, submitted by Nicholas Sean Arellano Escanilla, conforms to acceptable standards and is fully adequate in scope and quality to partially fulfill the requirements for the degree of Master of Science.



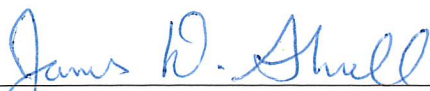
David C. Page, Ph.D.
Advisor and Head of Thesis Committee

8/9/17
Date



Charles R. Dyer, Ph.D.
Thesis Committee Member

8/11/2017
Date



James D. Shull, Ph.D.
Thesis Committee Member

8-11-17
Date

Department of Computer Sciences
University of Wisconsin-Madison

2017

Abstract

Due to advances in high-throughput technologies, data mining techniques for decision making processes have grown increasingly popular in the past decades. As more domains utilize these methods, we are seeing this surge in data – so-called big data – that requires preprocessing techniques to combat the curse of dimensionality. To handle such problems, dimensionality reduction techniques have been well-studied. Such an example is principal component analysis (PCA), a procedure that combines features to create new ones, thus reducing the dimensions in the dataset. This paper studies a different technique called feature selection. As opposed to using the original set of features in a dataset to build a predictive model, feature selection aims to find a subset of those features so that their combination is higher in predictive power, quality, and better interpretability of the data. In addition to an overview of feature selection techniques previously published in the literature, we present a novel feature selection algorithm and evaluate it using synthetic data labeled by a challenging family of nonlinear target functions. The method is also assessed on germline genomic data, from breast cancer patients and controls, comprised of single-nucleotide polymorphisms (SNPs) from a region of the human genome associated with breast cancer. Through these experiments, we show the advantages that our algorithm has over alternative methods when the task is to determine the subset of relevant features for a given input dataset.

Dedication

Every challenging work requires self-efforts as well as guidance from those who are very close to our heart. This thesis is dedicated to my family and my girlfriend Madison MacTavish, whose love and encouragement has and continues to support my growth and development both in and outside the academic realm.

Acknowledgements

I would like to extend my sincerest thanks and appreciation to all the individuals who have supported me in this endeavor. If it was not for their advice, encouragement and guidance, it is safe to say that I would not have finished.

First, I would like to thank my advisor, Dr. David Page. His patience and availability for me to pester him with a slew of questions has helped my understanding of machine learning grow exponentially. He has shown me that I can do great research in this domain. For that, I am truly grateful.

I would also like to thank my committee members, Dr. Chuck Dyer and Dr. Jim Shull, for their comments and suggestions on my thesis. From a mere idea to a final product, your feedback was invaluable to my research.

Table of Contents

Abstract.....	iii
List of Tables	vii
List of Figures.....	viii

Chapters

1. Introduction.....	1
1.1 Thesis Statement	2
2. Background	4
2.1 Learning models.....	4
2.1.1 Support Vector Machines	5
2.1.2 Artificial Neural Networks	8
2.2 Feature Selection.....	10
2.2.1 Filter Methods.....	12
2.2.2 Wrapper Methods.....	15
2.2.3 Embedded Methods	19
2.3 Applications	21
2.3.1 Correlation Immunity.....	21
2.3.2 Breast Cancer and Single-Nucleotide Polymorphisms	22
2.4 Motivation for the Proposed Work	23
3. Methodology	26
3.1 Recursive Feature Elimination Algorithm	26
3.2 Recursive Feature Elimination by Sensitivity Testing Algorithm	27
4. Results	32
4.1 Data	32
4.2 Synthetic Data Results	33
4.3 Germline Genomic Data for Breast Cancer Results	41
5. Discussion and Conclusion.....	45
Appendices.....	47
Bibliography	53

List of Tables

1. Truth table for *Drosophila* survival based on gender and Sxl gene activity1
2. Rank of 9 SNPs and all interaction pairs built with a linear SVM. A feature labeled with 'X' signifies an interaction pair. The weights from a linear SVM run determined the feature ranking43

List of Figures

1. A simple linear support vector machine	5
2. An example of a fully connected neural network	8
3. A typical process of feature selection split into two phases. Phase I determines the subset of relevant features. Phase II evaluates the performance of the chosen subset of features and learning model (Liu, Motoda, Rudy, & Zhao, 2010)	11
4. The Sequential Floating Forward Selection (SFFS) flow chart (Chandrashekar & Sahin, 2014)	16
5. Genetic Algorithm flow chart (Goldberg, 1989)	18
6. RFE and RFEST AUC for parity function. The total number of features increase from left to right graphs (20, 50, and 100 features, respectively)	34
7. RFE and RFEST number of features retained for parity function. The total number of features increase from left to right graphs (20, 50, and 100 features, respectively). The relevant features were set to the first two features in the dataset (i.e. X1 and X2) ...	35
8. RFE and RFEST AUC for ci-function number 87 (order four). The total number of features increase from left to right graphs (20, 50, and 100 features, respectively) ..	36
9. RFE and RFEST number of features retained for ci-function number 87 (order four). The total number of features increase from left to right graphs (20, 50, and 100 features, respectively). The relevant features were set to the first four features in the dataset (i.e. X1, X2, X3, and X4).....	37
10. RFE and RFEST AUC for ci-function number 584248 (order five). The total number of features increase from left to right graphs (20, 50, and 100 features, respectively)	38
11. RFE and RFEST number of features retained for function number 584248 (order five). The total number of features increase from left to right graphs (20, 50, and 100 features, respectively). The relevant features were set to the first five features in the	

dataset (i.e. X1, X2, X3, X4, and X5).....	39
12. RFE and RFEST AUC for ci-function number 2657 (order six). The total number of features increase from left to right graphs (20, 50, and 100 features, respectively) ..	40
13. RFE and RFEST number of features retained for ci-function number 2657 (order six). The total number of features increase from left to right graphs (20, 50, and 100 features, respectively). The relevant features were set to the first six features in the dataset (i.e. X1, X2, X3, X4, X5, and X6).....	41

Chapter 1

Introduction

As data continue to grow and become more accessible, many refer to our time as the Information Age; an era in which data are used to extract information and ultimately lead to better decision making. However, our ability to understand this amount of information does not keep pace with the exponential growth in data (Chandrashekar & Sahin, 2014).

The issue at hand is how we deal with big data. That is, data that contain many irrelevant or redundant variables. Machine learning is growing in popularity as a tool that researchers can use to automatically analyze data. To better analyze and interpret big data, a slew of dimensionality reduction methods is at a researcher's disposal. One that is considered fundamental to machine learning is feature selection. This technique allows a learning algorithm to concentrate on aspects of the data that are most useful for analysis and future prediction (Guyon & Elisseeff, An Introduction to Variable and Feature Selection, 2003).

Table 1. A truth table for Drosophila (fruit fly) survival based on gender and Sxl gene activity.

GENDER FEMALE	SXL ACTIVE	SURVIVAL
0	0	0
0	1	1
1	0	1
1	1	0

These issues arise in practice, for example in biology (Table 1) (Cline, 1979). In Table 1, the interpretation of this output is that flies that are either male and have an active *Sxl* gene or that are female and have an inactive *Sxl* gene will survive; the other fruit flies will not. While some machine learning methods such as deep neural networks or non-linear

support vector machines (SVMs) can approximate this function easily given only the relevant variables, the SVM's accuracy will degrade dramatically as irrelevant variables are added, unless the training set is quite large. We seek a method of feature selection that can remove these irrelevant variables and maintain SVM accuracy.

1.1 Thesis Statement

To introduce the thesis for this work, consider the generalized problem of feature selection describe by (Battiti, 1994):

“Given an initial set of n features, find the subset with $k < n$ features that is ‘maximally informative’ about the class.”

Applied to the biological realm, genome-disease association studies (genome-wide or limited) seek genetic features associated with disease, i.e., predictive of disease. In many cases, it is believed that such features may interact with one another in highly nonlinear ways to influence disease. Nevertheless, for practical reasons almost all association studies use linear models and hence can find only features that individually are correlated with disease (Balding, 2006). Consequently, key genetic features may be missed entirely.

The claim to this thesis is two-fold. The first is that with a novel feature selection algorithm, one can find a subset of features that is the most important modulator of the underlying function. Additionally, with the subset of features, it can be shown that their interactions are crucial to the underlying function.

To do this, we introduce a new algorithm that improves upon recursive feature elimination (RFE) (Guyon, Weston, Barnhill, & Vapnik, 2002) both theoretically and empirically. Our algorithm, at present, is limited to binary features but is extensible to various feature types; we leave this to future work.

The remainder of this thesis is organized as follows: Chapter 2 presents background information for several learning models, the importance of feature selection, applications that utilize feature selection, and motivation for a novel feature selection algorithm. Chapter 3 discusses the rationale and methodology for using the new algorithm. Chapter 4 shows the results of the experiments. Lastly, Chapter 5 discusses the results and possible extensions to the algorithm. Note that additional results can be found in the Appendix.

Chapter 2

Background

2.1 Learning Models

Machine learning can be thought of as the underlying foundation of data mining. It is used as a means of extracting information from raw data across a slew of applications. This allows researchers to accomplish tasks such as solving classification problems, learning the underlying structure of the data, and adding to domain knowledge, to name a few. Some of the major concepts that fall under machine learning are the following: rote learning, clustering, discovery, induction, and reinforcement learning. This paper falls under the inductive learning paradigm of machine learning. In inductive learning, the goal is to learn a general model with a given dataset so that it can accurately predict future examples. One can think of this task as approximating an unknown function f , such that:

$$f(x) = y \tag{1}$$

Where x is an input instance (or example), y is the class label, and f is approximated by the hypothesis h . We will review several popular machine learning methods for inductive learning in the next subsections. Among those are support vector machines and artificial neural networks.

2.1.1 Support Vector Machines

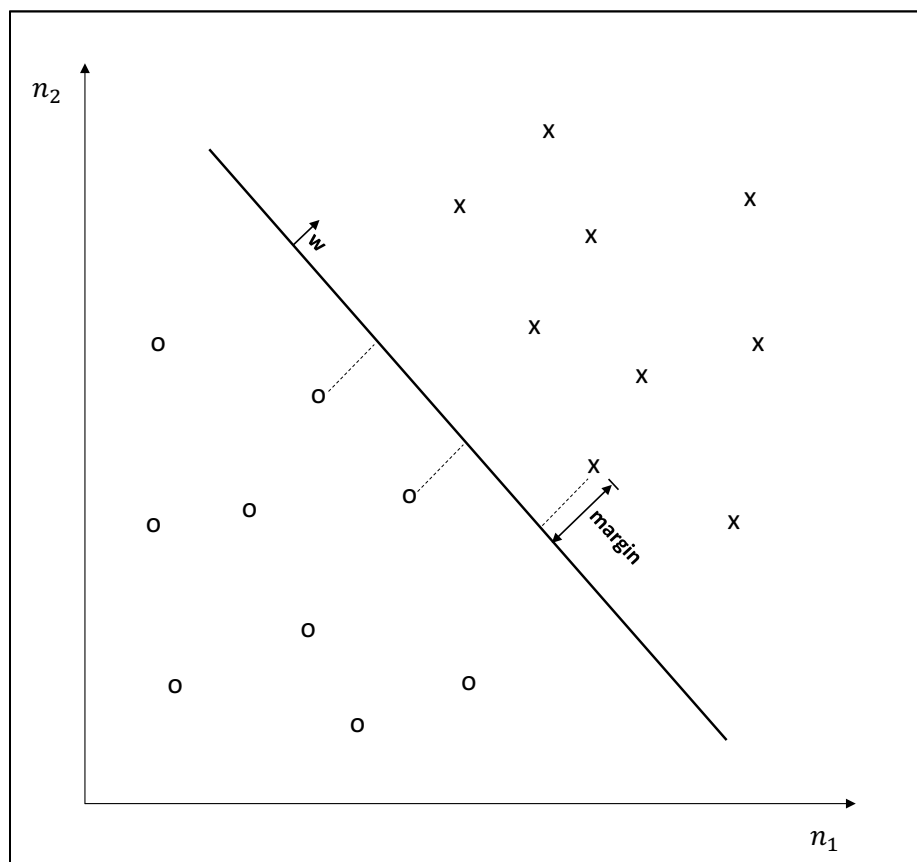


Figure 1. A simple linear support vector machine.

Linear support vector machines (SVMs) are based on an algorithm that finds a linear model known as the maximum-margin hyperplane. This hyperplane can be seen in Figure 1. In this case, we have a two-class dataset (denoted by o's and x's) whose classes are linearly separable. In other words, there exists a hyperplane that classifies all training instances correctly. Note that in this example, one can build an infinite number of hyperplanes that classify all training instances correctly (Ben-Hur & Weston, 2010). The *maximum*-margin hyperplane is the linear model that gives the greatest separation between the two classes. The examples that are closest to the maximum-margin hyperplane are the support vectors. This

subset of examples uniquely defines the hyperplane. Therefore, we can construct the equation of the linear model in terms of the support vectors.

Using Ben-Hur and Weston's notation (2010), given instances $x_1, \dots, x_n \in \mathbb{R}^d$ and corresponding class labels $y_1, \dots, y_n \in \{-1, 1\}$, the hard margin SVM primal problem is represented as:

$$\begin{aligned} & \text{minimize}_w \quad \frac{1}{2} \|\mathbf{w}\|^2 \\ & \text{s. t.} \quad y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1, \forall i = 1, \dots, n. \end{aligned} \tag{2}$$

In the case of non-linear problems, examples can be misclassified or in the margin. This adjusts the primal problem above and introduces slack variables, denoted by ξ_i . If example i falls in the margin, that example's slack variable is assigned the value $0 \leq \xi_i \leq 1$. If example i is misclassified, then $\xi_i > 1$. Therefore, the bound on the number of misclassified instances is the sum of all slack variables. A cost term, C , is applied to this sum to penalize margin and misclassification errors (Ben-Hur & Weston, 2010). The new primal problem is now:

$$\begin{aligned} & \text{minimize}_w \quad \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^n \xi_i \\ & \text{s. t.} \quad y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \forall i = 1, \dots, n. \end{aligned} \tag{3}$$

By solving this problem, we obtain the optimal w . However, we know nothing of the α_i , a parameter that represents the contribution of the i^{th} example to the final solution, w . To classify a query point x , one would have to explicitly compute the scalar product $w^T x$, which may be computationally expensive if the size of the dataset is large. The dual formulation allows us to solve for α_i and depends on the data solely through dot products:

$$\text{maximize}_{\alpha} \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n y_i y_j \alpha_i \alpha_j \mathbf{x}_i^T \mathbf{x}_j \quad (4)$$

$$s. t. \quad \sum_{i=1}^n y_i \alpha_i = 0, \quad 0 \leq \alpha_i \leq C.$$

Note that this formulation allows us to effectively learn non-linear separators through use of the kernel trick, a mapping to a higher dimensional feature space resulting in an ability to encode nonlinear separators in the original feature space (Maldonado, 2011).

A major strength of SVMs with nonlinear kernels is that they can find informative interactions between features without requiring explicit pre-computation of all possible *interaction terms*, or products of all subsets of two or more features (Stambaugh, Yang, & Breuer, 2013). For some practical applications, a weakness of nonlinear SVMs is that the resulting model is a black box, in the sense that the contributions of individual features cannot be extracted from the model in the form of feature weights. Hence in choosing an SVM kernel we sometimes face a tradeoff between accuracy and comprehensibility (Lantz, 2013).

Even though we cannot extract feature weights from a nonlinear SVM, if we had some other way to identify and remove irrelevant features, then in cases where many features are irrelevant we might be able to obtain a more comprehensible model. In this case, by removing irrelevant features we might reduce over-fitting and hence further improve model accuracy. A theoretical example to consider is correlation immune (CI) functions, which are non-linear Boolean functions (Section 2.3.1 covers these functions in detail). It is expected

that SVMs can efficiently learn this family of functions. Proving that this hypothesis holds is a major component of this paper.

2.1.2 Artificial Neural Networks

Neural networks can take one of two major forms: feedforward or recurrent. This section will describe feedforward networks in more detail and how this machine learning method can be used in feature selection.

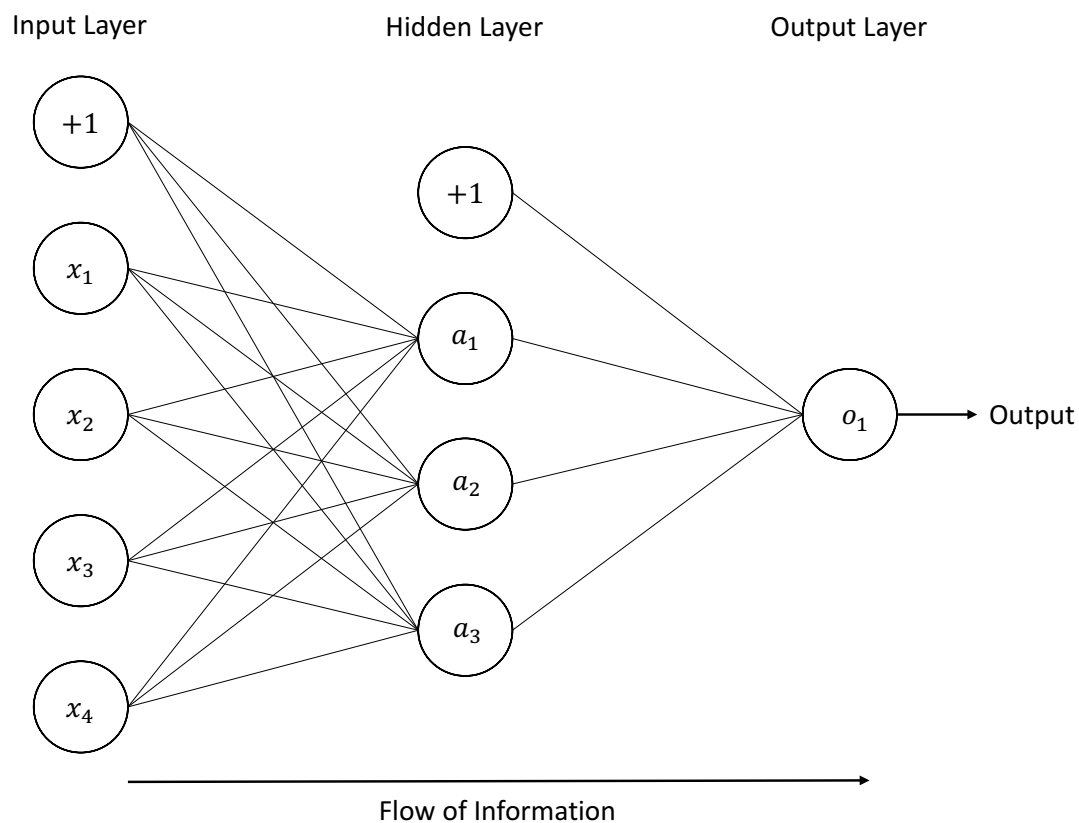


Figure 2. An example of a fully connected neural network.

In feedforward neural networks, information flows from input units to output units. To get from the input to output layer, information flows through hidden layer(s). The classic example of this being the multilayer perceptron (MLP). This network contains one input layer, one output layer, and one or more hidden layers. Refer to Figure 2 for an example of

an MLP. MLPs are fully connected, which means that a node in a layer, is connected by a certain weight w_{ij} , to every node in the following layer. To provide flexibility in the neural network, bias nodes (denoted by +1 in Figure 2) are used in the input and hidden layer(s). Training of the weights is done by backpropagation (Gardner & Dorling, 1998). That is, the weights are adjusted until the network's output is as close as possible to the desired output. It has been proven both theoretically and empirically that these networks are able to approximate nonlinear target concepts/functions (Jain, Mao, & Mohiuddin, 1996).

In a deep learning setting, an extension to the traditional multi-layer, feed forward network is the convolutional neural network (CNN). Unlike the traditional approach, where the network is fully connected, CNNs use local filters (i.e. weights) and local connections. Thus, eliminating the need for layers to be completely connected. As a result, we get translation-invariant and distortion-invariant local features (LeCun & Bengio, 1998). CNNs have become useful in applications that process array data. For example, they are used to process time series, acoustic, text, image, audio spectrogram, video, and volumetric data (LeCun & Bengio, 1998).

Deep learning allows for better features to be learned. In a sense, the problem at hand comes out naturally by the way a deep learning network is defined. In general, the characteristics of a neural network, therefore, allow a user to indirectly perform feature selection (Ledesma, Cerda, Aviña, Hernández, & Torres, 2008). Ledesma et al. (2008) provides an example of how to use neural networks and genetic algorithms for feature selection.

2.2 Feature Selection

Before exploring the concept of feature selection in detail, one must define what it means for a feature to be *relevant* or *redundant*. One of the principal definitions for a feature being relevant is presented by Blum and Langley:

- Definition 1 (*Relevant to the target*). A feature x_i is *relevant to a target concept c* if there exists a pair of examples A and B in the instance space such that A and B differ only in their assignment to x_i and $c(A) \neq c(B)$.

In other words, a feature is relevant if there exists an instance such that changing its values for this feature influences the classification by the target concept (Blum & Langley, 1997).

An alternative definition of relevance is the following: “A feature can be regarded as irrelevant if it is conditionally independent of the class labels” (Law, Figueiredo, & Jain, 2004). This means that a relevant feature cannot be independent from the class labels, but can be independent from the other features in the input data (Chandrashekar & Sahin, 2014).

Guyon and Elisseeff (2003) showed through an example of intra-class covariance that variables that are *perfectly* correlated are truly redundant. Note that variables that are almost perfectly correlated (i.e. have high variable correlation) may not be redundant. Therefore, given a set of features containing relevant, irrelevant, and/or redundant features, the goal of feature selection is to find a subset of those features that are most relevant to the target concept, that is, that influence the class values of the greatest number of instances.

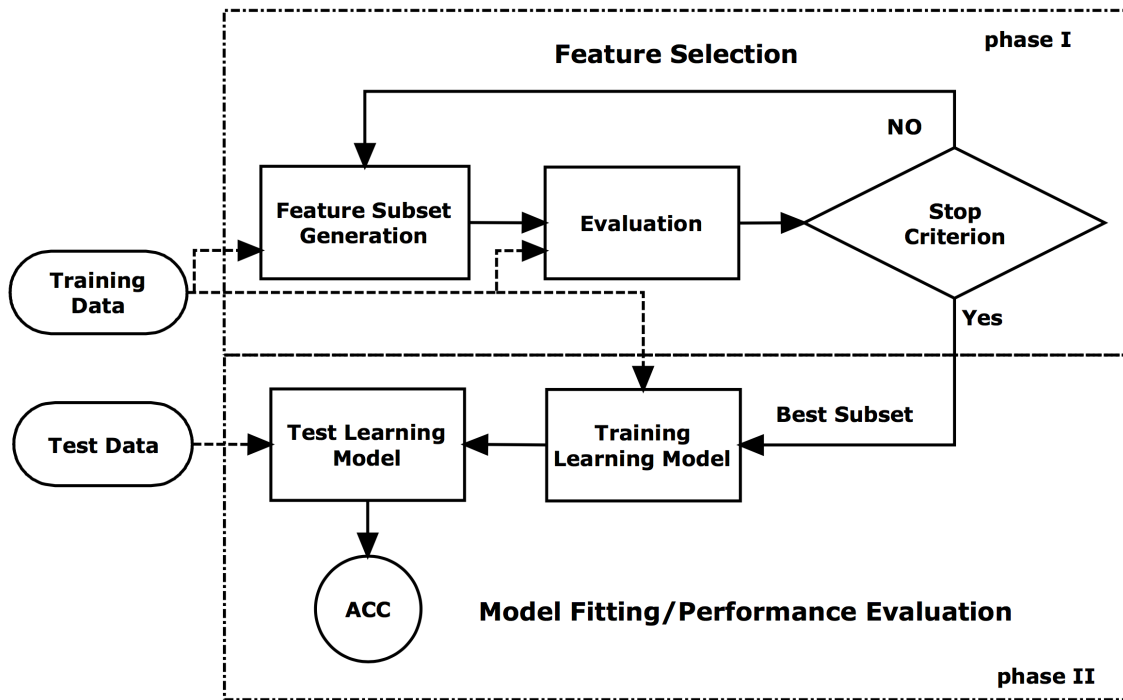


Figure 3. A typical process of feature selection split into two phases. Phase I determines the subset of relevant features. Phase II evaluates the performance of the chosen subset of features and learning model (Liu, Motoda, Rudy, & Zhao, 2010).

Almost all feature selection techniques require the following generalized components: an evaluation criterion used to compare different candidate subsets of features, a search method that is typically sequential (e.g. forward selection, backward elimination), and a stopping criterion. A more detailed explanation can best be described by Figure 3. (Liu, Motoda, Rudy, & Zhao, 2010) split feature selection into two phases. Phase I uses the training data to determine which features should be retained (or removed). First, by using specific search strategies (e.g., information gain), a subset of the original features is chosen and evaluated. This process is repeated until the stopping criteria is met. The output of Phase I is a subset of the features considered to be the most relevant to the task. Using this set of

features, a model is trained on the training data and an accuracy measure is returned based on how well the model performs on a separate test set.

A simple approach to feature selection would be to test all possible subsets of features. However, this is an NP-hard problem as the number of features increases (Chandrashekar & Sahin, 2014). In turn, feature selection techniques are used and can be divided into three categories: filter, wrapper, and embedded. The following subsections explain each type in detail and provide several examples of commonly used methods.

2.2.1 Filter Methods

Filter methods are considered a preprocessing technique because they select features based on characteristics of the data. In other words, they do not use any learning algorithms to determine feature relevance (Liu, Motoda, Rudy, & Zhao, 2010). Instead, a ranking criterion scores each feature and a threshold is used to remove features. Under certain assumptions, this process is considered optimal with respect to a predictor (Guyon & Elisseeff, An Introduction to Variable and Feature Selection, 2003).

This type of feature selection is a fast and scalable approach that does not rely on a classifier. It is a simple way of measuring a feature's ability to distinguish between the class labels. The following notation, adopted from (Chandrashekar & Sahin, 2014) will be used to provide several examples of filter methods: Input data will be represented by $[x_{ij}, y_k]$, where x_i ranges from $i = 1 \dots, N$, the j^{th} feature ranges from $j = 1 \dots, D$, and y_k is the class label $k = 1 \dots, Y$.

A well-known statistic used in filter methods is the Pearson correlation coefficient and is represented by the following equation:

$$R(i) = \frac{cov(x_i, Y)}{\sqrt{var(x_i) * var(Y)}} \quad (5)$$

Equation 5 measures the *linear* relationship between a variable x_i and its class label. $cov()$ and $var()$ represent the covariance and variance, respectively. The range of $R(i)$ is $[-1, 1]$, where -1 signifies a perfect negative linear relationship between the variable and the class label, and 1 means a perfect positive linear relationship between the variable and the class label. It is important to note that this ranking criterion only detects linear dependencies between each variable and the class. To extend to the non-linear case, one can use methods such as squaring, taking the log, or the square root, then using a correlation coefficient (Guyon & Elisseeff, An Introduction to Variable and Feature Selection, 2003).

Another example of a filter method is information gain (or mutual information).

Recall the definition for entropy, H :

$$H(Y) = \sum_{i=1}^k -\Pr(Y = y_i) \log \Pr(Y = y_i) \quad (6)$$

In this equation, H is a measurement of the information content associated with a set of examples. For a 2-class problem, 0 implies no information and 1 is maximum information. In other words, entropy is a measure of “disorder” on a set of examples.

To measure the total “disorder” of all variables in the input data, we introduce the notion of conditional entropy:

$$H(Y|X) = \sum_{v: \text{values of } X} \Pr(X = v) H(Y | X = v) \quad (7)$$

In conditional entropy, we take the weighted sum of the entropy of each subset of examples partitioned by all possible values of variable X . The interpretation behind conditional entropy is that by observing X , the uncertainty in Y is reduced (Chandrashekar & Sahin, 2014).

Information gain, I , takes the difference between entropy and conditional entropy for each candidate variable:

$$I(Y, X) = H(Y) - H(Y|X) \quad (8)$$

If $I(Y, X) = 0$, then X and Y are independent (i.e., $H(Y) = H(Y|X)$). They are dependent otherwise since information about X reduces the uncertainty in Y . The definition above is used for discrete variables. To extend to the continuous case, one may use the Kullback-Leibler divergence between two densities to measure information gain (Chandrashekar & Sahin, 2014).

A notable filter based approach is the RELIEF algorithm introduced by Kira and Rendell (1992). The feature relevance criterion is based on a statistical method and a threshold is set. This algorithm differs from the other methods mentioned before in that it can handle feature interactions (Kira & Rendell, 1992). However, the RELIEF algorithm's major downfall is the threshold parameter since it is chosen by the user.

Other examples of filter methods include χ^2 and the Markov blanket filter (Saeys, Inza, & Larrañaga, 2007). Although this method is computationally fast and avoids overfitting, a major issue is that ranking criteria such as R or I do not consider the interdependencies between variables. In other words, most filter methods neglect the fact that a feature by itself may not be informative, but may be informative when combined with other features in the input data. Another problem with this type of feature selection technique is

that it does not interact with the classifier. The next section describes methods that use the predictor to uncover a subset of relevant features.

2.2.2 Wrapper Methods

Since evaluating all possible subsets of features is an NP-hard problem, wrapper methods find suboptimal subsets. Unlike filter methods though, wrapper methods are coupled with the underlying learning algorithm. That is, they use the performance of the predictor in the evaluation step to determine relevant features. This type of feature selection can be divided into two categories: sequential selection algorithms and heuristic search algorithms (Chandrashekar & Sahin, 2014).

An example of a sequential selection algorithm is Sequential Feature Selection (SFS) (Chandrashekar & Sahin, 2014). This algorithm starts with an empty set of features, and then gradually adds one feature at a time to the current set of selected features. Consider the case where our current set of features is S , SFS will go over all features $i \notin S$, and apply the predictor to the set of features $S \cup \{i\}$. In turn, we get $|i|$ different predictors and the feature that produced the maximum classification accuracy (or minimum error) is added to S . SFS's counterpart, sequential backward selection (SBS), follows a similar format. Instead of starting with an empty set, SBS starts with all possible features and eliminates those that are considered irrelevant or redundant. Both SFS and SBS continue until a stopping criterion is met (e.g., until the algorithm finds the k best features) (Saeys, Inza, & Larrañaga, 2007).

Variations to the SFS algorithm have improved its performance. An example of such is the Sequential Floating Forward Selection (SFFS) algorithm. Chandrashekar and Sahin (2004) provided a flow chart for the algorithm which is given below:

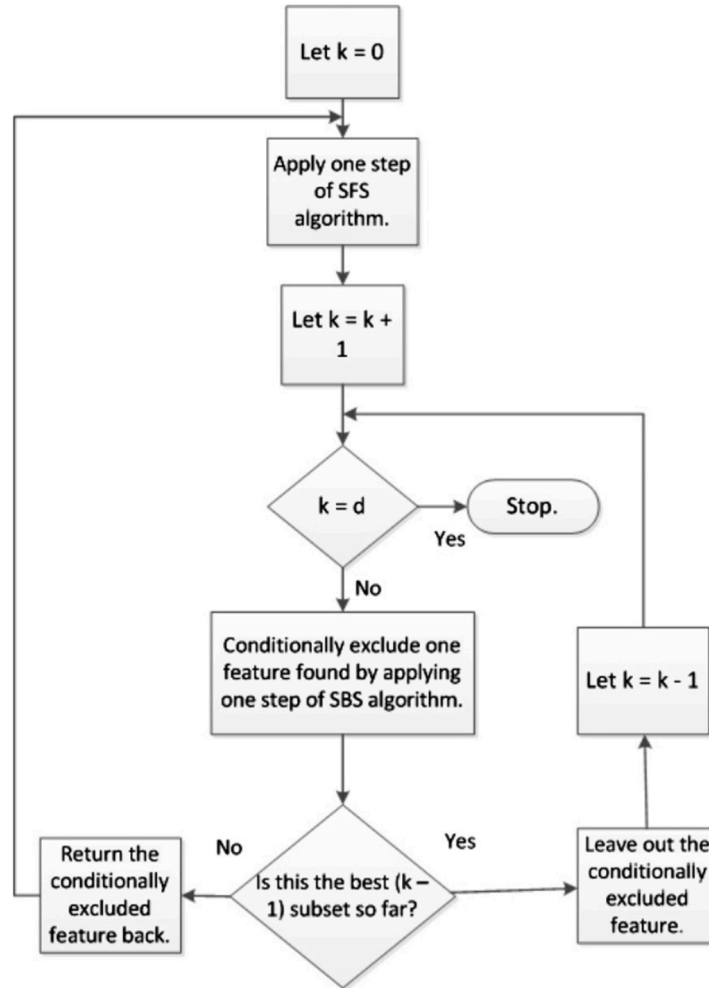


Figure 4. The Sequential Floating Forward Selection (SFFS) flow chart (Chandrashekar & Sahin, 2014).

Notice that SFFS contains an additional check by excluding one feature at a time (using SBS) in the current set and determining whether removal of that feature increases the value of the objective function. If so, that feature is left out and the process of excluding features continues. If not, then one step of the original SFS algorithm is applied and the process of removing features continues. The algorithm stops when it is in a d -dimensional feature space.

The simplicity of using a sequential selection algorithm certainly has its advantages. For example, this type of algorithm allows for interaction with the classifier to determine relevant features. In addition, we can model the interdependencies between features. However, there is a risk of overfitting to the predictor given the iterative nature of the algorithm. There is also a greater chance that sequential search algorithms will stop at a local optimum in comparison to heuristic search algorithms (Saeys, Inza, & Larrañaga, 2007).

Unlike sequential search algorithms, heuristic searches utilize domain knowledge to guide the selection of the best features. Many of the algorithms in this category are randomized methods. An example would be a search algorithm that uses simulated annealing. The following is a generalized summary of the algorithm:

Algorithm 1: Simulated Annealing
<ol style="list-style-type: none"> 1. Start with initial feature set S 2. Pick candidate feature s' to add to current set S, call this candidate set S' 3. if $f(S') - f(S) > 0$ then $S = S'$ <i>else</i> $S = S'$ for some small probability p 4. Go to step 2 until stopping criteria is reached

Note that a “bad move” is accepted with a small probability p . This probability is calculated from Boltzmann’s equation:

$$p = e^{\Delta E/T} \quad (9)$$

where $\Delta E = f(s') - f(s)$ represents the change in predictor performance and T is the *annealing temperature* that controls the frequency of moving to a bad state. The latter parameter decreases as the search continues, thus decreasing the probability of moving to a bad state. Although this method requires several parameters to be set, the chance of finding a global optimum (i.e. best subset of features) increases.

Another popular approach is to use a genetic algorithm. Inspired by natural evolution, this algorithm uses the concept of crossover, mutation, and natural selection to aid in finding the best set of features. The following is a flow chart for the basis of a genetic algorithm:

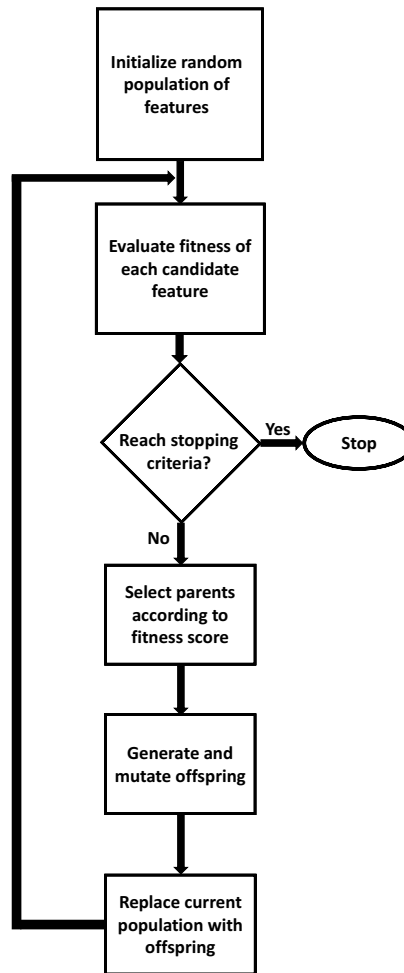


Figure 5. Genetic Algorithm flow chart (Goldberg, 1989).

This flow chart is similar to the SFFS algorithm's flow chart. However, using the mechanisms of evolutionary change (i.e. crossover, mutation, and natural selection) makes this algorithm less prone to stopping at a local optimum, thus finding the best subset of features (Chandrashekar & Sahin, 2014). Details on the algorithm are outside the scope of this thesis and can be found in (Goldberg, 1989).

In general, wrapper methods are tailored to the chosen classification model. That is, a search algorithm is essentially ‘wrapped’ around the learning algorithm. To investigate all possible subsets of features is intractable when the number of features grows exponentially. Therefore, a heuristic is applied in the search to help find the optimal subset. As previously mentioned, in both the sequential selection and heuristic search algorithms, there is a risk of overfitting since the subset of features is determined by the classifier. In addition, if the classifier has a high computational cost, then this algorithm may become computationally expensive (Saeys, Inza, & Larrañaga, 2007). To combat these issues, many use greedy searches such as forward selection (SFS) or backward elimination (SBS) (Guyon & Elisseeff, An Introduction to Variable and Feature Selection, 2003).

2.2.3 Embedded Methods

Like wrapper methods, embedded methods are specific to a given learning algorithm. The difference between the two being that the search for the optimal subset is built into the classifier. That is, feature selection is a part of the training process. The goal of embedded methods is to avoid retraining a predictor after every iteration (like in wrapper methods) that, in turn, avoids the computational costs that can come along with implementing a wrapper method.

Recall how mutual information can be used as a filter method. When coupled with a greedy search algorithm, it can be used to evaluate subsets of features. Battiti (1994) showed how to select an informative subset of features using a neural network classifier. For any iteration of the algorithm, let n be the number of features in the input data, C be the class, $f \in F$ be the candidate feature in set F , $s \in S$ be the current feature in optimal subset S , and

β a parameter set in the range of $[0.5, 1]$. Then the objective function used by Battiti (1994) is:

$$I(C, f) - \beta \sum_{s \in S} I(f; s) \quad (10)$$

That is, for all features $f \in F$, the equation above is calculated and the feature that maximizes the mutual information between the feature and class while minimizing the mutual information between feature f and the current set of selected features S is chosen. This process continues until $|S|$ is size k , for some $k < n$. The qualitative reason behind this calculation lies in the second term in Equation 10. Instead of measuring mutual information like in Equation 8, this considers the interdependencies between the candidate feature and the current set of features already chosen (Chandrashekar & Sahin, 2014).

A well-known approach in regression analysis that was introduced by (Tibshirani, 1996) is the Least Absolute Shrinkage and Selection Operator (LASSO). LASSO combines the strengths of subset selection and ridge regression and solves the problems associated with each technique. That is, subset selection is a discrete process in which minute changes in the data may result in diverse models, thus hurting prediction performance. Ridge regression offers a more stable approach but does not allow the coefficients in the model to be set to zero. Therefore, the model that is returned is still difficult to interpret (Kim & Kim, 2004). Following the notation in (Tang, Alelyani, & Liu, 2014), LASSO regularization is defined as:

$$penalty(\mathbf{w}) = \sum_{i=1}^n |\mathbf{w}_i| \quad (11)$$

Equation (11) allows for some weight(s) \mathbf{w}_i to be set to zero. Those that are set to zero correspond to features that can be eliminated in the learning process (Tibshirani, 1996). Therefore, LASSO provides an alternative method for feature selection.

A popular embedded-based algorithm called recursive feature elimination (RFE) will be thoroughly examined in Chapter 3. Other examples include decision trees such as CART, which has a built-in tool to perform feature selection, or a weighted naïve Bayes. An overview of these algorithms can be found in (Saeys, Inza, & Larrañaga, 2007).

2.3 Applications

Many feature selection techniques have been proposed throughout the past decades due to the slew of information that certain tasks contain. Areas that use feature selection include: bioinformatics, text mining, computer vision/image processing, and industrial applications. For example, in genomics research, where the number of features can drastically exceed the number of patients and the task is to separate those that are healthy from cancer patients, determining and removing the information that is irrelevant or redundant aids in better prediction/classification performance. This section covers both theoretical and real-world applications that, with feature selection methods, can overcome the issue of having irrelevant and/or redundant features affect the learning algorithm.

2.3.1 Correlation Immunity

A function is correlation immune (CI) if every single-feature marginal distribution is uninformative, i.e., no feature by itself is correlated with the function value, or class, even given the entire truth table or example space. An equivalent interpretation of CI functions are functions such that every variable has zero gain (with respect to any gain measure) when computed from the input data. Consider the Boolean function $f: B^k \rightarrow B$, where $B = \{-1, 1\}$

and $k \in \mathbb{Z}^+$. Let x_i represent the i th input variable. Then with respect to a uniform distribution on B^k , f is correlation immune if $\forall x_i$, the values of x_i and output $f(x_1, \dots, x_n)$ are independent (Hellerstein, Rosell, Bach, Ray, & Page, 2009).

As a result, these functions include some of the most challenging target concepts for most classification algorithms. The most famous non-linear separators in machine learning are exclusive-or (XOR) and exclusive-nor (XNOR), which are two-feature parity functions (e.g. *Sxl* gene example illustrated in Table 1, Chapter 1).

One would expect, for example, that SVMs, with a Gaussian kernel or polynomial kernel of degree at least two, would learn these functions with ease. Unfortunately, for the simplest case of XOR in the presence of even a modest number of irrelevant features, or variables, SVMs tend to have a difficult time learning and require a large sample size empirically. This problem is not specific to SVMs; it is known that no algorithm based on statistical queries can PAC-learn even parity functions of $\log(n)$ variables (Hellerstein & Servedio, On PAC learning algorithms for rich Boolean function classes, 2007). This thesis seeks a method of feature selection that can remove these irrelevant variables and restore SVM accuracy.

2.3.2 Breast Cancer and Single-Nucleotide Polymorphisms

Breast cancer is a heterogeneous disease that can be classified into subtypes based on pathobiological or molecular criteria. Examples include ductal carcinoma in situ (DCIS), where the cancer cells remain confined within a breast duct, and invasive ductal carcinoma, where cancer cells penetrate the duct wall and spread locally within the breast (Swain, 1992).

Breast cancer development is influenced by many genetic and environmental factors. An aim of this paper is to focus attention on the variations at single base pairs of the human

genome, which are known as single-nucleotide polymorphisms (SNPs). In cancer, both germline SNPs (the DNA sequence with which a person is born, and which is replicated in most of the cells in the body) and somatic mutations (variants that occur in select cells during replication and can lead to cancer) are important and are widely studied. To date, germline SNPs have received more attention as they can predict a person's future risk of breast cancer, and these are the focus in this paper (Tempfer, Hefler, Schneeberger, & Huber, 2006). Genome Wide Association Studies (GWAS) seek to find SNPs that are associated with risk for developing disease.

Currently, GWAS consider SNPs independently and do not take into account possible interactions between SNPs. The rationale behind this is that it is infeasible to consider all pairs of the $n = 1 \text{ million}$ SNPs that are typically measured. The main purpose of a thorough investigation of SNPs is to gain a better understanding of how these genetic variants act as biological markers. Given a set of SNPs, if we can help identify a subset of important SNPs that correlate with an effect in patients, then we will be able to investigate their interactions. In turn, this will help our decision processes about numerous aspects of medical care such as risk of developing a certain disease, effectiveness of various drugs, and adverse reactions to specific drugs.

2.4 Motivation for the Proposed Work

Thus far, we know that many feature selection algorithms utilize linear modeling approaches such as lasso-penalized logistic regression, linear SVMs, naïve Bayes or other weighted-voting schemes among features. An alternative is to utilize these same approaches after filtering features individually by information gain or by many single-variable logistic regression runs (Chandrashekar & Sahin, 2014). To account for interactions between

features, the standard approach would be to introduce interaction terms, but such terms typically are limited to pairs of features, and even then, they greatly increase both run-time and risk of over-fitting.

Nonlinear SVMs have the potential to more effectively find complex interactions among features, but insight into the important interactions is hard to extract from the learned model. This paper addresses that shortcoming by presenting a novel recursive feature elimination (RFE) algorithm and demonstrating that the algorithm makes it possible to identify the features that are relevant – i.e., play a role in the learned nonlinear model, even if individually they are completely uncorrelated with the class – while removing those features that are irrelevant, or do not influence the learned model.

Because we do not assume we are in an active learning setting – i.e., we do not have access to an oracle for membership queries that can label feature vectors with the value of any feature altered – our key insight is to use the trained non-linear SVM itself as such an oracle instead. While this trained model is not the target concept, we assume it is more accurate than random guessing and hence provides some information about feature relevance.

A widely-used approach to perform such a task is RFE, an embedded-based backward selection strategy (Stambaugh, Yang, & Breuer, 2013). RFE constructs an SVM, ranks the features according to the constructed SVM, removes the lowest-ranked feature or features (e.g., bottom ten percent), and repeats until further removal degrades the accuracy of the classifier, ideally on a tuning set rather than the training set. The ordinary RFE algorithm with a linear SVM simply ranks features by the absolute value coefficient the SVM gives each feature; this approach assumes features have been normalized to have comparable ranges.

Although SVM training produces coefficients on examples, for linear SVMs we can easily construct coefficients on features. Unfortunately for a non-linear SVM feature coefficients cannot be obtained. Guyon *et al.* (2002) presented a variant of RFE for use with non-linear SVMs. One weakness of this approach is that empirically the feature rankings tend to be less accurate, and consequently features need to be removed one-by-one in practice rather than in larger groups. In this paper, we propose an alternative RFE algorithm tailored to non-linear SVMs.

In homage to earlier work on membership queries to test the sensitivity of target concepts to individual features, we call our RFE algorithm “RFE by Sensitivity Testing,” or RFEST. The remainder of the paper presents the RFEST algorithm and theoretical and empirical evaluations of it, including novel and promising insights that it provides into genetic susceptibility to breast cancer.

Chapter 3

Methodology

3.1 Recursive Feature Elimination Algorithm

The approach used to compare the results of the RFEST algorithm will be the method proposed by Guyon et al. (2002). Due to the nature of the data sets used in their paper, Guyon utilized a linear SVM with RFE. However, they described how their method can be carried over to handle a nonlinear SVM implementation and this is the algorithm that we use as the baseline, which we describe next.

Recall that for SVMs, the cost function that is being minimized is the following:

$$J = \frac{1}{2} \alpha^T H \alpha - \alpha^T \mathbf{1} \quad (12)$$

where, $0 \leq \alpha_k \leq C$ and $\sum_k \alpha_k y_k = 0$

For training instances h and k , α is a numeric parameter determined by the SVM algorithm, y_h and y_k represent the class values for the training instances, \mathbf{x}_h and \mathbf{x}_k are the feature vectors for the training instances, C a regularization parameter, and $H = y_h y_k K(\mathbf{x}_h, \mathbf{x}_k)$, for K a kernel function (Guyon, Weston, Barnhill, & Vapnik, 2002).

To determine feature relevance, the change in cost function proposed by Guyon is the following ranking coefficient:

$$DJ(j) = \frac{1}{2} \alpha^T H \alpha - \frac{1}{2} \alpha^T H(-j) \alpha \quad (15)$$

$-j$ represents the j^{th} feature removed from the H matrix. In turn, the feature corresponding to the smallest $DJ(j)$ is removed.

Algorithm 2: RFE Algorithm

1. **Input:** data $d_{i,j}$, where $i \in \{1, \dots, n\}, j \in \{1, \dots, p\}$
2. **repeat**
3. Train SVM and output α and b parameters
4. Implement $DJ(j), \forall$ features j
5. Remove the feature(s) with the smallest $DJ(j)$
6. **until** k features remain ($k < p$)

Algorithm 2 describes the RFE algorithm for the nonlinear case in more detail (Lal, Chapelle, Weston, & Elisseff, 2006). The benefit of using RFE over a vanilla approach (e.g. train a new SVM for each candidate feature on every iteration) allows for each iteration of the algorithm to train only one SVM model. In other words, we assume that the vector α is fixed and consider the change in the kernel as a result of removing feature j . The qualitative justification behind this is that a feature's value to the learned model is measured by the change in the expected value of error when removing that candidate feature (Lal, Chapelle, Weston, & Elisseff, 2006).

3.2 Recursive Feature Elimination by Sensitivity Testing Algorithm

The preceding discussion motivates a revised RFE algorithm to learn the correlation immune functions for breast cancer datasets. While this work assumes that we are using binary features with a $\{0,1\}$ -encoding, it could be extended to handle continuous and/or categorical features as well. Standard RFE requires retraining a model for each feature removed and can become computational intractable with many features. In the case where there are thousands of features, (Guyon, Weston, Barnhill, & Vapnik, 2002) chose to remove half of the features at each iteration. Doing so allows for faster convergence to an idealized subset of

features, but key information may be lost. The algorithm below summarizes RFEST.

Algorithm 3: RFEST Algorithm
7. Input: data $d_{i,j}$, where $i \in \{1, \dots, n\}, j \in \{1, \dots, p\}$
8. repeat
9. Train SVM and output AUC
10. Implement $R(j), \forall$ features j
11. Remove the feature(s) with the largest ranking coefficient
12. until AUC is less than 95% of the max AUC achieved

There are two main differences between RFE and RFEST. The first being that we expect that with binary data flipping, the value of a feature for our test data will produce results equivalent to removing the feature. Note that a feature that is *flipped* means that if its current value is 0, then it now has a value of 1, and vice versa.

An SVM classifier can classify a dataset with an accuracy measurement called the Area Under the Curve (AUC). In addition, for each feature j , the same SVM classifier can be used to calculate the AUC when we flip j . Call this area under the curve $AUC_{flipped}$. In turn, the proposed ranking coefficient is:

$$R(j) = AUC_{flipped} - AUC \quad (16)$$

The interpretation of Equation (16) is as follows. For each j , if $AUC_{flipped} < AUC$, then the j^{th} feature is relevant because the model classified the instances at a lower AUC without j . In contrast, if $AUC_{flipped} > AUC$, then the j^{th} feature is irrelevant because the model classified the instances at a higher AUC without j (i.e., the classifier did better without feature j).

Therefore, the feature corresponding to the largest $R(j)$ will be removed. This is the second difference between RFE and RFEST. Like RFE, this process does not retrain a classifier for every candidate feature to be removed.

For our learning algorithm, a nonlinear SVM with an RBF kernel was used. The reason for doing so is because it is typically a good first choice in practice. Given the size and the nonlinear nature of our datasets (more detail in Chapter 4), it is evident that a linear kernel will suffer in predictive performance. One may then argue that a polynomial kernel would be a good choice for nonlinear features. However, it requires more hyper-parameters to be set as opposed to an RBF kernel (Hsu, Chang, & Lin, 2010).

To determine the final subset of features, RFEST stops when the *AUC* at any given iteration is less than 95% of the max *AUC* achieved thus far. Alternatively, other heuristics can be implemented with RFEST. A vanilla approach would be to stop when *AUC* decreases (i.e. a hill-climbing search). Another example would be to apply a simulated annealing method, where if the *AUC* decreases, we continue searching with small probability p . As the search continues, p decreases. Search methods such as simulated annealing or the approach RFEST currently uses are aimed towards avoiding a local optimum. We use our current search method since it avoids having to set additional parameters (such as in the case for a simulated annealing approach).

RFEST may suffer if the SVM model we use is overfitted to a given instance, since the model might then be sensitive to the value of every feature. For this reason, we used a ten-fold cross validation and allocated the dataset into separate training, tuning, and testing sets to produce an unbiased estimate of the efficacy of our approach. Additionally, nonlinear SVMs require setting several hyper-parameters (cost C and gamma γ). It has been shown that searching in exponentially growing sequences for C and γ is a good method for identifying their respective parameter values (Hsu, Chang, & Lin, 2010). Therefore, the best configuration for C and γ was chosen based off a grid search.

We next demonstrate the theoretical efficacy of RFEST in learning a classifier that labels examples according to the parity of a subset of input features. We show that, if a non-linear SVM can learn a sufficiently accurate approximation to such a classifier, then RFEST can accurately identify the relevant variables given only a polynomial sized sample. For simplicity and clarity of the proof, our theorem assumes an idealized version of RFEST that uses accuracy instead of AUC and draws a new sample for each decision it makes. The theorem we state here applies only to the parity function and uniformly distributed examples. Future work involves generalizing the theorem to a somewhat broader class of functions and product distributions.

As background to Theorem 3.1 and to the proof of the theorem, the expected number of successes according to a binomial distribution $b(m, p)$ is mp , where m is the number of independent trials of a Bernoulli experiment and p represents the probability of success. Following the notation presented in Kearns and Li (1993), Chernoff Bounds imply that for any $0 \leq \alpha \leq 1$, the probability of seeing at most $(1 - \alpha)mp$ successes is less than $e^{-\alpha^2 mp/2}$ (Kearns & Li, 1993). We will apply Chernoff Bounds to the observed error on m examples of a model whose true error is $0 < \gamma < 0.5$. Taking $\alpha = 1 - \frac{1}{2(1-\gamma)}$, we have that $0.5 = (1 - \alpha)(1 - \gamma)$. Therefore, if m is at least $\frac{(1-\gamma)}{(\gamma-\frac{1}{2})^2} \ln\left(\frac{1}{\delta^2}\right)$, then the probability of an observed error estimate (‘success’ rate) of at least 0.5 is δ . By inverting the notion of ‘success,’ by symmetry, if a model’s true error is $1 - \gamma$, then the probability that it achieves an error rate less than 0.5 on m examples is also less than δ .

Theorem 3.1. *Suppose machine learning algorithm is given as input a dataset S of examples defined on n Boolean variables drawn i.i.d. from the uniform distribution D , where examples are labeled according to a parity function computed over some subset of the n variables.*

Further, suppose the algorithm constructs a model M , with true error rate γ , where $\gamma < 0.5$. Then for any $0 < \delta < 1$, there is a quantity m that is polynomial in n , $\ln \frac{1}{\delta}$, and $\frac{1}{0.5-\gamma}$, such that if RFEST is given model M , together with a set of m new examples (generated in the same way as the examples in S), then RFEST will exactly identify the relevant variables with probability at least $1 - \delta$.

Proof. With respect to the uniform distribution D , assume model M has true accuracy $1 - \gamma$.

Flipping any bit in a parity function reverses the function output. Therefore, on a data set drawn from D with one relevant variable consistently mislabeled (flipped), the expected accuracy of M is γ . The expected accuracy on such a data set with any irrelevant variable flipped is $1 - \gamma$.

By Chernoff bounds, for $m \geq \frac{(1-\gamma)}{(\gamma-\frac{1}{2})^2} \ln \left(\frac{n^4}{\delta^2} \right)$ examples, with probability at least $1 - \frac{\delta}{n^2}$,

flipping an irrelevant variable will result in less than a $0.5 - \gamma$ drop in accuracy while flipping a relevant variable will result in greater than a $0.5 - \gamma$ drop in accuracy. On any iteration of RFE then, with probability $1 - \frac{\delta}{n^2}$, no irrelevant variable will result in as large a loss in accuracy when flipped as any relevant variable, and RFEST will therefore delete an irrelevant variable. It follows that with probability at least $1 - \frac{\delta}{n}$, RFEST will delete an irrelevant variable on a given round. Therefore, with probability $1 - \delta$, RFEST will finish with exactly the relevant variables. \square

One may wonder if the above theorem contradicts the known result that parity functions are not PAC-learnable from statistical queries. It does not, because it is preconditioned on a non-linear SVM learning a model that is significantly better than random guessing.

Chapter 4

Results

4.1 Data

We implemented RFEST and Guyon's RFE algorithm tailored to a non-linear SVM, and we evaluated it on two types of data with the programming language R (Karatzoglou, Smola, Hornik, & Zeileis, 2004; Kuhn, 2008; Wickham, 2011; Sing, Sander, Beerenwinkel, & Lengauer, 2005). The first consists of synthetic data that takes the form of a correlation immune function known as the parity function. CI functions of order four, five, and six were also evaluated. Note that because of the properties of CI functions, there are a great deal of functions that can be constructed. So, three randomly chosen functions for each order were chosen. The target concept for these datasets are fixed by the order of the CI function. That is, if the order of the function is n , then the first n features determine the class label. Feature values for all instances were chosen from a uniform distribution.

The second dataset used indicates that *Emca4*, a genetic determinant susceptibility to 17β -estradiol (E2)-induced mammary cancer in the rat that has been mapped to rat chromosome 7 (RNO7) (Colletti, et al., 2014; Schaffer, et al., 2006; Shull, 2007). Data presented herein indicate that *Emca4* harbors multiple genetic determinants of mammary cancer susceptibility and tumor aggressiveness that are orthologous to breast cancer loci mapped to chromosome *8q24.21* in genome wide association studies (GWAS) (Easton, et al., 2007; Turnbull, et al., 2010; Fletcher, et al., 2011; Michailidou, et al., 2013; Ahsan, et al., 2014).

The proceeding algorithm(s) used 76 of the SNPs in the designated region that are in the Hunter GWAS data set containing 1145 breast cancer cases and 1142 controls (Hunter, et

al., 2007). All patients that had incomplete SNP data (630 patients) were omitted from our analysis. The data was made available via dbGaP's Cancer Genetic Markers of Susceptibility (CGEMS) Breast Cancer GWAS.

4.2 Synthetic Data Results

As previously mentioned, the following were the different CI functions investigated: parity (order 2), order four, order five, and order six. For orders four, five, and six, three different functions per order were chosen for analysis. To best represent the results, a learning curve was created to show the AUC for n total features, where $n \in \{20, 50, 100\}$. For each n total features, we tested datasets that contained m examples, where $m \in \{100, 500, 1000, 2000\}$. In addition, a learning curve was designed to represent the number of features that were kept using the same number of features and examples as stated above. To make the comparison "fair," 10% percent of the total number of features remaining at a given iteration were removed. In (Guyon, Weston, Barnhill, & Vapnik, 2002), the authors removed half of the features. However, we believe removing 10% of features at each iteration gave more accurate results since the number of features to begin with was not as large as the number in Guyon's paper.

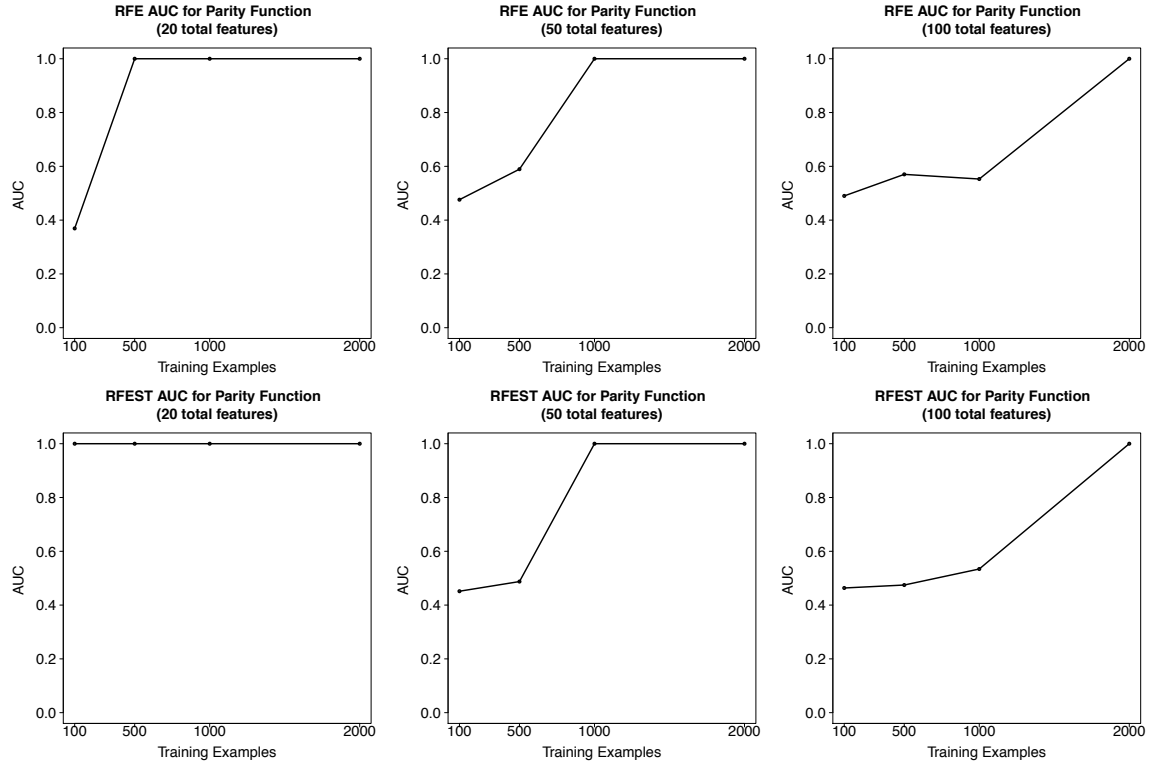


Figure 6. RFE and RFEST AUC for parity function. The total number of features increase from left to right graphs (20, 50, and 100 features, respectively).

Figure 6 shows the learning curves of AUCs for RFE and RFEST on the parity function. Specifically, the first row shows the results for RFE, and the second row is the results for RFEST. This follows suit for the succeeding figures. The performance of both algorithms seems to degrade as the number of total features increases, which is to be expected. One slight difference between the results for the parity function was that at 20 total features and 100 examples, RFEST's AUC was at 1.0. With the same dataset, RFE's AUC was 0.37, which is no better than random guessing. This is clearly a significant difference.

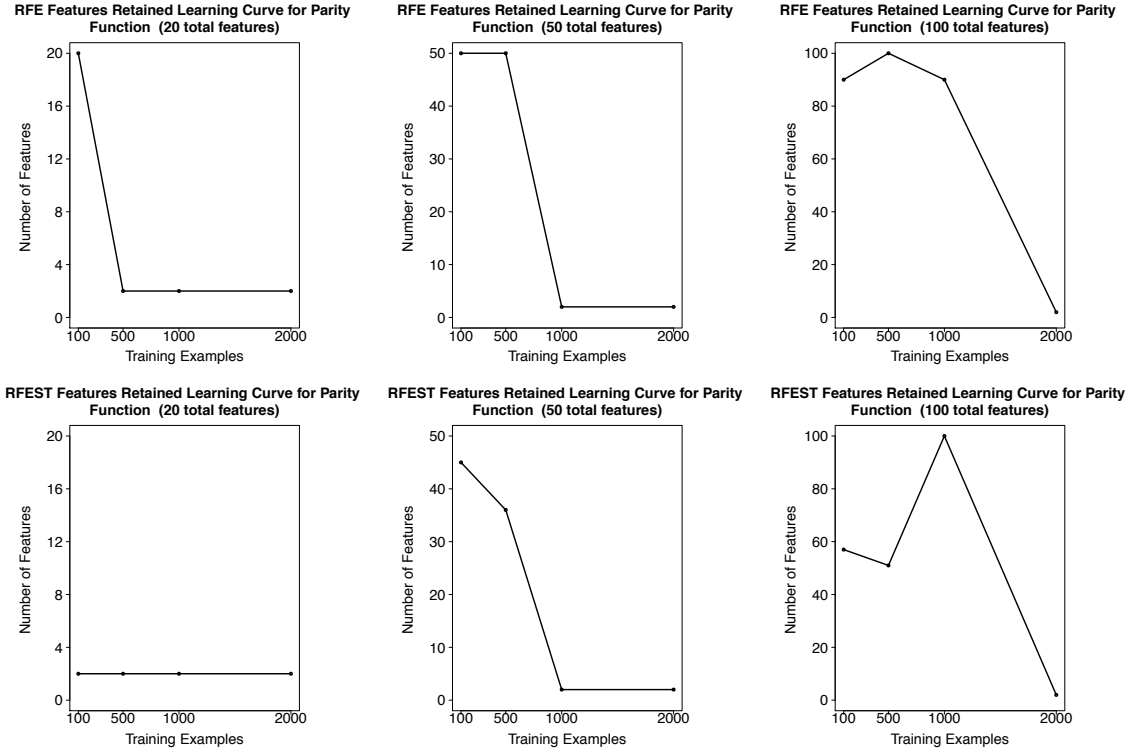


Figure 7. RFE and RFEST number of features retained for parity function. The total number of features increase from left to right graphs (20, 50, and 100 features, respectively). The relevant features were set to the first two features in the dataset (i.e. X_1 and X_2).

The optimal subset of features for the parity function was also calculated (Figure 7).

The goal is to keep only the relevant features. In the case of the parity function, we set the relevant features to be the first two features in our dataset (called X_1 and X_2 in the dataset). The remaining features are irrelevant. Both RFE and RFEST found and kept the relevant features. However, RFEST did so with 20 features and all varying instance sizes (i.e., 100, 500, 1000, and 2000 instances). RFE stops after the first iteration and returns the original feature set for 20 features and 100 instances. It is interesting to note that for 100 features and 1000 instances, RFEST stops after the first iteration and returns the original feature set. This may be due to the splitting of instances since at 2000 examples, they returned only X_1 and X_2 , respectively.

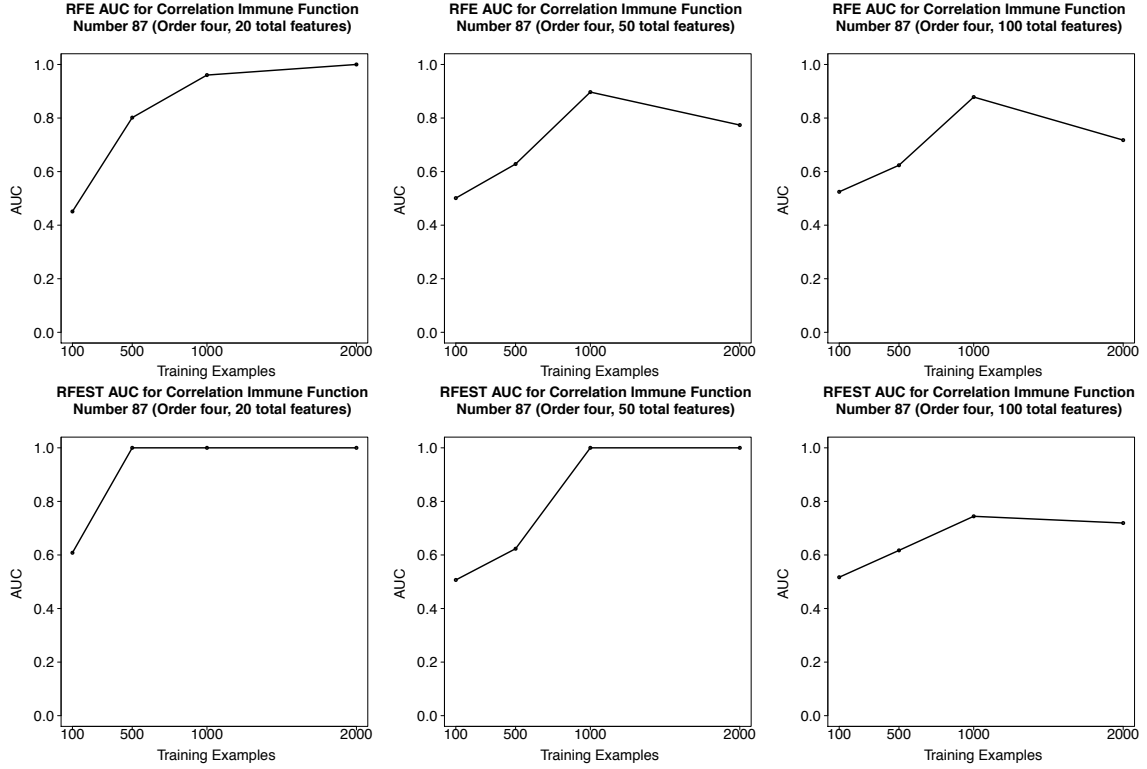


Figure 8. RFE and RFEST AUC for CI function number 87 (order four). The total number of features increase from left to right graphs (20, 50, and 100 features, respectively).

CI functions of order four were also analyzed (Figure 8). For both RFE and RFEST, the algorithms achieved an AUC of 1.0 for 20 features, respectively. However, note that for 50 total features the max AUC for RFE fell to 0.90, whereas the AUC for RFEST was 1.0. Note that RFE outperformed RFEST at 100 total features, achieving a max AUC of 0.88 at 1000 instances. The max AUC of RFEST for 100 features and 1000 instances was 0.74. This finding suggests that there are CI functions in which RFEST may not be the more robust method, in terms of its predictive abilities.

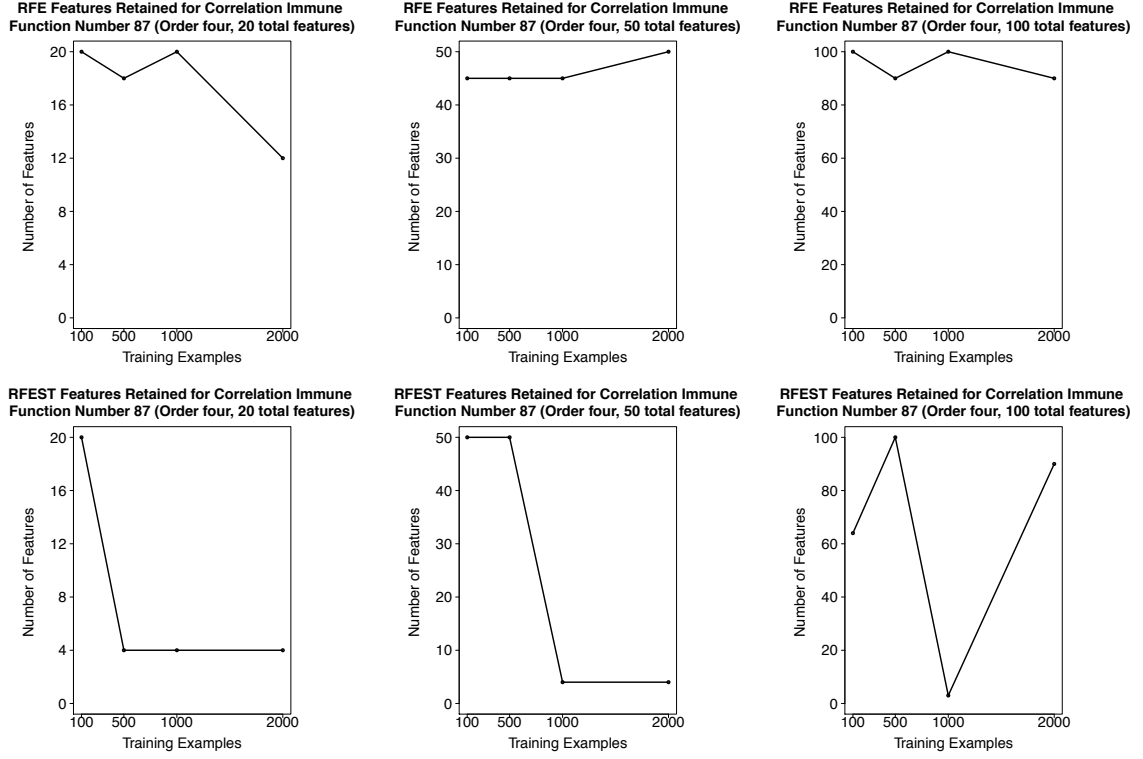


Figure 9. RFE and RFEST number of features retained for CI function number 87 (order four). The total number of features increase from left to right graphs (20, 50, and 100 features, respectively). The relevant features were set to the first four features in the dataset (i.e. X_1 , X_2 , X_3 , and X_4).

Figure 9 shows the number of features returned by each algorithm. The number of features that were returned for both RFEST decreased as the number of examples increased. Note that there is a significant difference in the number of features returned. For example, for 20 features, RFE returned all 20 features for 100 and 1000 instances, respectively. It returned 12 features given 2000 instances. RFEST returned 20 features for 100 instances, but then returned just the four relevant features (i.e. X_1 , X_2 , X_3 , and X_4) for 500, 1000, and 2000 instances. A similar trend is found when looking at 50 total features. Similar to the results of the parity function, at 100 features and 1000 examples to 100 features and 2000 examples, we see an increase in the number of features retained for RFEST.

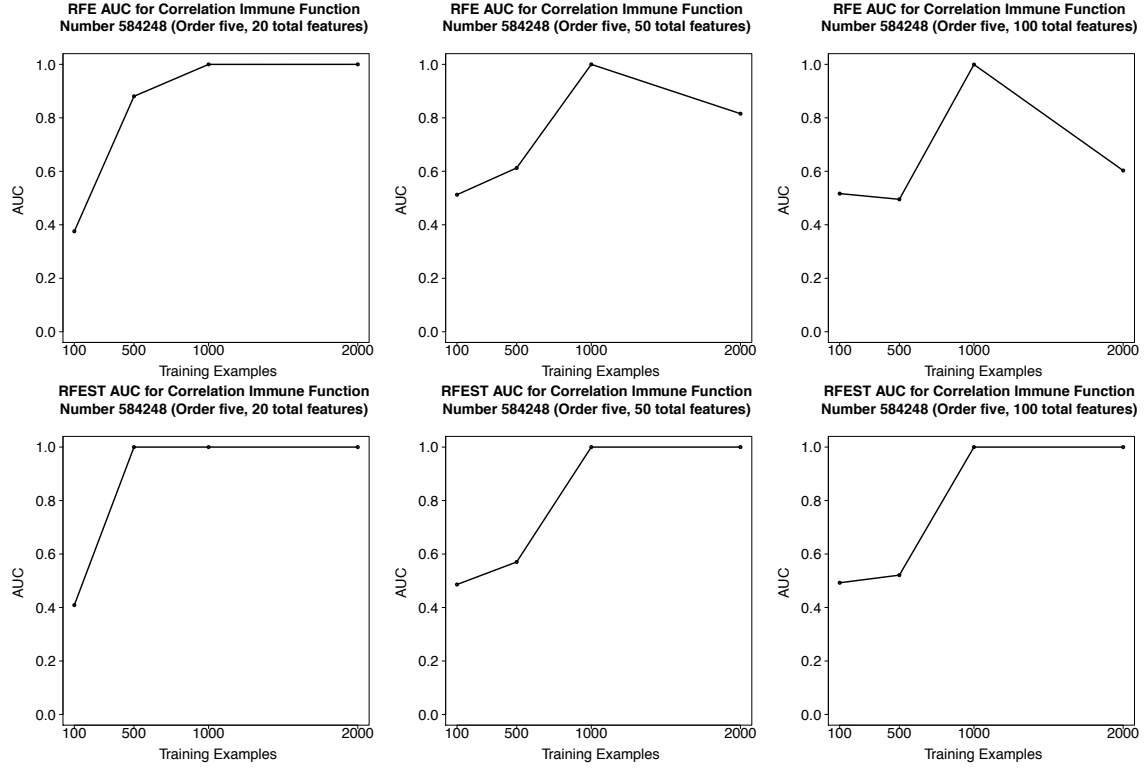


Figure 10. RFE and RFEST AUC for CI function number 584248 (order five). The total number of features increase from left to right graphs (20, 50, and 100 features, respectively).

A CI function of order five was analyzed and the AUCs were recorded in Figure 10.

For 20 features, RFE and RFEST performed very well as they achieved an AUC of 1.0 at 1000 and 500 instances, respectively. Note that at 50 and 100 features, RFE's AUC dropped when the number of instances grew to 2000. This is not the case for RFEST, where the AUC remained at 1.0 for those cases. With such high AUCs achieved, it would then be expected that the feature selection algorithms also recovered the relevant features.

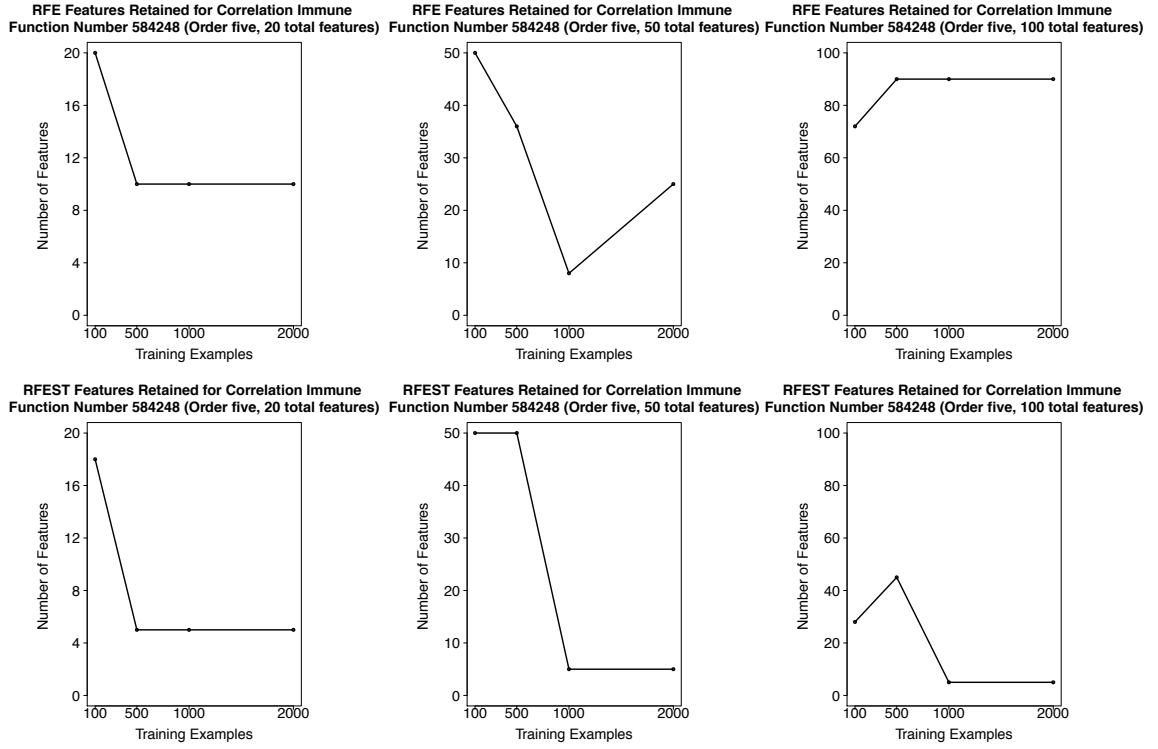


Figure 11. RFE and RFEST number of features retained for CI function number 584248 (order five). The total number of features increase from left to right graphs (20, 50, and 100 features, respectively). The relevant features were set to the first five features in the dataset (i.e. X_1 , X_2 , X_3 , X_4 , and X_5).

The relevant features for function number 584248 were set as the first five features in the dataset, namely, X_1 , X_2 , X_3 , X_4 , and X_5 . For 20 features, RFEST found the relevant features. However, RFE stopped prematurely, returning 10 features for 500, 1000, and 2000 examples, respectively. RFEST required 500 examples to return $X_1 \dots, X_5$. RFE also was not able to output just $X_1 \dots, X_5$ for 50 or 100 features. RFEST returned the relevant features given only 1000 examples. Therefore, for this CI function of order five, RFEST outperformed RFE both in *AUC* and the number of features output.

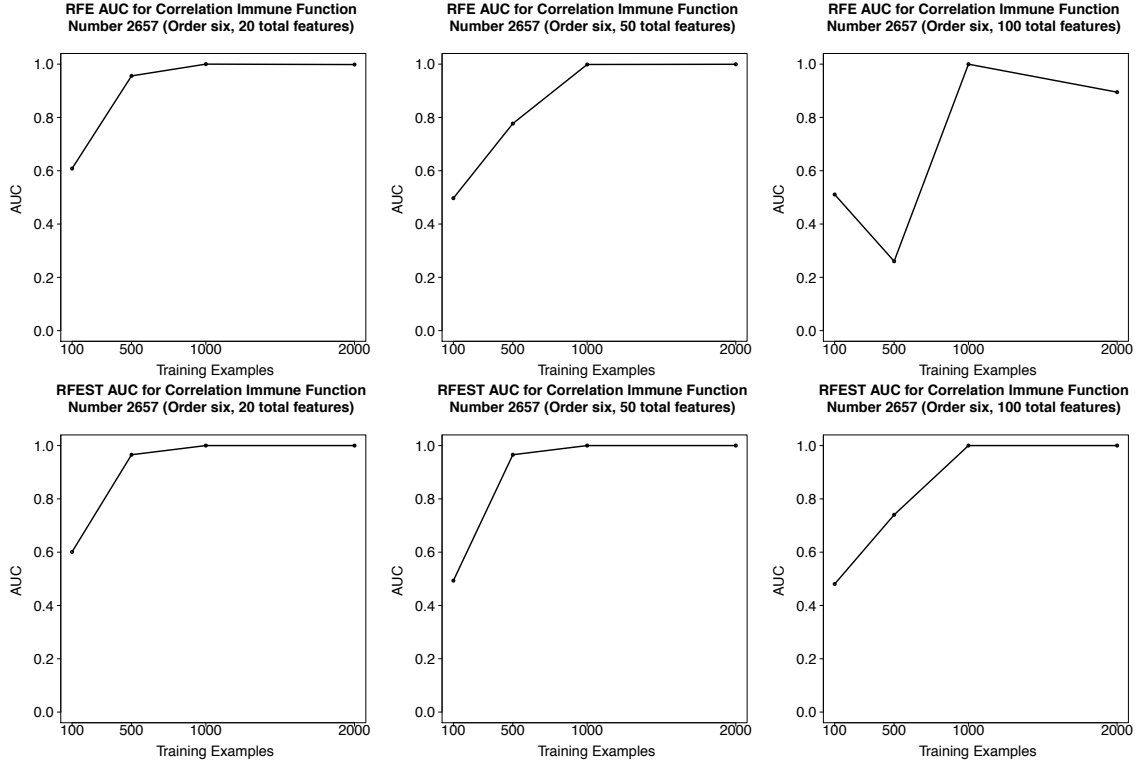


Figure 12. RFE and RFEST AUC for CI function number 2657 (order six). The total number of features increase from left to right graphs (20, 50, and 100 features, respectively).

Lastly, we investigated CI functions of order six. That is, the first six features were set to be the relevant features (i.e., $X_1 \dots, X_6$). The results were similar to the ones presented in Figure 10. RFE and RFEST performed well in AUC across all varying input data sizes. RFE achieved an AUC of 1.0 at 1000 examples for 20, 50, and 100 features. However, the AUC fell to 0.90 for 100 features with 2000 examples. The significant difference lies in what features the algorithm returned with the accompanied AUC.

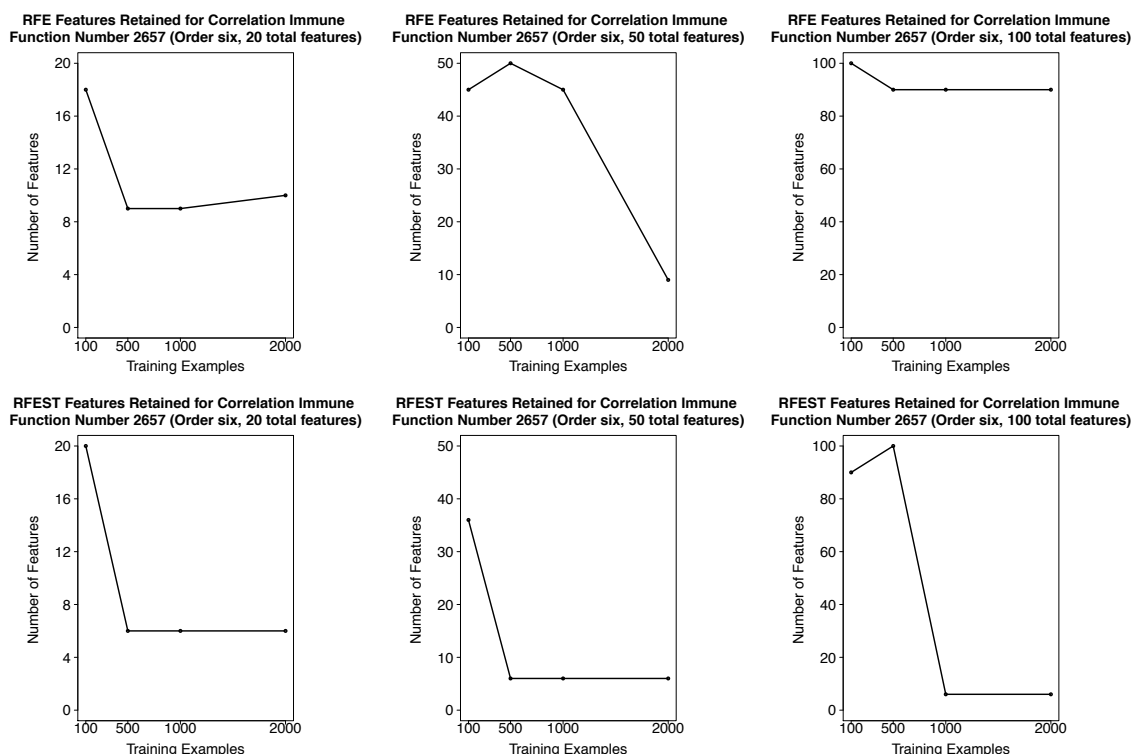


Figure 13. RFE and RFEST number of features retained for CI function number 2657 (order six). The total number of features increase from left to right graphs (20, 50, and 100 features, respectively). The relevant features were set to the first six features in the dataset (i.e. $X1$, $X2$, $X3$, $X4$, $X5$, and $X6$).

For this function, RFE was not able to return solely the relevant variables across all datasets investigated. For 100 features, this algorithm returned a subset of features much larger than solely the relevant features. RFEST, on the other hand, found and returned the relevant features (i.e., $X1 \dots, X6$) given enough examples. For 20 and 50 features, the number of examples needed was 500. For 100 features, 1000 examples were needed. Figure 13 shows these results in more detail. Across the CI functions, we showed that RFEST outperforms RFE in both *AUC* and number of features retained.

4.3 Germline Genomic Data for Breast Cancer Results

There is great interest in associating variations in the human genome with disease risk. Much of this work focuses on associating with any given disease the variations in “SNPs,” which are single-nucleotide polymorphisms or single positions in the genome where individuals may differ from one another. Such work assumes the SNPs, and the variations in disease risk that they cause, are independent of one another; in general, this assumption is wrong and results in lost accuracy.

We may examine variations in the germline DNA with which a person is born or variations that arise from somatic mutations in individual cells, such as in the developments of cancers and which may vary widely even within the same tumor. One particular disease for which both types of variations have been studied is breast cancer. For predicting disease risk, germline genomic data is the more natural choice to use.

The data we investigated contains 76 SNPs, translating to 152 binary features, in a particular region of the human genome that is orthologous to a region of the rat genome known to modulate breast cancer risk. We used the CGEMS data set of SNP genotypes for 1145 breast cancer cases and 1142 healthy age and gender-matched controls (Hunter, et al., 2007). Applying to this data to *RFEST*, as run in the previous section, produces a cross-validated *AUC* of 0.56, which outperforms linear SVM and non-linear SVM cross-validated runs (0.53 and 0.54, respectively). Likewise, *RFEST* outperformed *RFE* as *RFE* returned an *AUC* of 0.53, no better than either a linear or non-linear SVM run with the same input data.

While all runs were performed by eliminating 10% of features at a time, our novel algorithm is also effective (when compared to *RFE*) when removing 20% or even 30% of the remaining features at a time. Removing 10% of the features at a time not only resulted in an *AUC* of 0.56 but most notably retained only nine features. With such a small set of features,

one can then exhaustively generate all pairs (and even more) of interaction terms. In turn, the top 13 features are all pairs of SNPs rather than individual SNPs (refer to *Table 2* for the complete ranking). This suggests that interactions play a major role in the effect of SNP variations in this region on breast cancer risk, as has been suspected. Studies are under way to further evaluate these nine selected SNPs.

Table 2. Rank of 9 SNPs and all interaction pairs built with a linear SVM. A feature labeled with 'X' signifies an interaction pair. The weights from a linear SVM run determined the feature ranking.

Rank	Features	Absolute Value of Weights
1	X.rs3870371.A.rs1562430.G.	0.451720104
2	X.rs3870371.A.rs6470588.A.	0.42713695
3	X.rs1562430.G.rs6470588.A.	0.423986855
4	X.rs2935776.T.rs1264202.T.	0.365437269
5	X.rs2935776.T.rs3870371.A.	0.354584473
6	X.rs2935776.T.rs1562430.G.	0.329954375
7	X.rs3870371.A.rs9792269.A.	0.328540808
8	X.rs1456315.A.rs1264202.T.	0.300546816
9	X.rs1456315.A.rs6470588.A.	0.299881282
10	X.rs3870371.A.rs1456315.A.	0.299641976
11	X.rs1456315.A.rs7014346.A.	0.298781744
12	X.rs3870371.A.rs284489.A.	0.297515487
13	X.rs7014346.A.rs284489.A.	0.292526626
14	rs1562430.G	0.267090006
15	X.rs1562430.G.rs1264202.T.	0.253908246
16	X.rs3870371.A.rs1264202.T.	0.250721852
17	X.rs2935776.T.rs284489.A.	0.18627039
18	X.rs1562430.G.rs9792269.A.	0.17608929
19	X.rs1456315.A.rs9792269.A.	0.168792895
20	X.rs1562430.G.rs284489.A.	0.167114611
21	X.rs2935776.T.rs6470588.A.	0.146167862
22	X.rs2935776.T.rs7014346.A.	0.130404389
23	rs284489.A	0.129134758
24	X.rs6470588.A.rs7014346.A.	0.126196353
25	X.rs1264202.T.rs7014346.A.	0.125589857
26	X.rs7014346.A.rs9792269.A.	0.123551947
27	rs7014346.A	0.113268757
28	rs3870371.A	0.108279896
29	X.rs6470588.A.rs284489.A.	0.098166989
30	rs2935776.T	0.097135632
31	X.rs6470588.A.rs9792269.A.	0.089032231
32	rs6470588.A	0.089032231
33	X.rs2935776.T.rs1456315.A.	0.085613599
34	X.rs3870371.A.rs7014346.A.	0.063277325
35	X.rs1562430.G.rs7014346.A.	0.063277325
36	X.rs2935776.T.rs9792269.A.	0.060314928
37	rs1456315.A	0.045613599
38	X.rs1264202.T.rs9792269.A.	0.031480933
39	rs1264202.T	0.028301637
40	X.rs9792269.A.rs284489.A.	0.012314054
41	rs9792269.A	0.003179296
42	X.rs1264202.T.rs284489.A.	0.002563605
43	X.rs6470588.A.rs1264202.T.	0.001043659
44	X.rs1562430.G.rs1456315.A.	0.000431674
45	X.rs1456315.A.rs284489.A.	0.000181784

It has been shown that incorporating as risk factors germline SNPs associated with breast cancer can significantly improve prediction and even mammography-based diagnosis even though breast cancer is estimated to be only 30% heritable (Liu, et al., 2014). In this section, we showed that avoiding the independence assumption regarding SNPs, by using a non-linear SVM with our novel RFE algorithm, makes it possible to associate with breast cancer new SNPs and their interactions, and that this association can enable more accurate breast cancer risk prediction than could be made from these SNPs without taking interactions into account. A post-hoc analysis of these nine SNPs confirmed our collaborating biologist's suspicion that interactions (rather than specific SNP values) were the most important modulator of breast cancer risk in this genomic region, and revealed which interactions were important.

Chapter 5

Discussion and Conclusion

This thesis claims that not only can one find a subset of relevant features but can also show that interactions are crucial to the underlying task (e.g. breast cancer diagnoses).

Evaluation of this claim was accomplished by creating an alternative recursive feature elimination (RFE) algorithm called RFEST that is tailored to a non-linear SVM. RFEST was theoretically and empirically tested to support this claim.

The original RFE algorithm presented by (Guyon, Weston, Barnhill, & Vapnik, 2002) is an embedded-based approach but RFEST behaves like a wrapper-based approach. It uses a non-linear SVM as a black box to determine feature relevance. In principle, with this approach one can then use *any* learning algorithm to remove irrelevant or redundant features. RFEST differs from RFE in two important ways: it perturbs rather than eliminates each feature to test sensitivity and measure loss in accuracy (using *AUC*) instead of the loss in weighted sum of distances from the margin. These differences result in improvements across correlation-immune functions and a real-world breast cancer genomics problem. It also results in substantial speed-up. Each time a variable is tested, RFE must recompute the entire modified kernel matrix and make predictions on all examples. RFEST simply makes predictions on all examples, resulting in an $O(n)$ speed-up for n examples.

Unfortunately, there is no panacea for feature selection, let alone machine learning. That is, no learning algorithm is superior to all others. Research in this domain allows us to provide insight into the strengths and limitations of a set of algorithms. Given background knowledge and domain-specific insight to an application, practitioners can choose the

algorithms to apply. The use of RFEST can enhance the performance of feature selection algorithms and return a subset of relevant features. The beauty behind RFEST is that the learning algorithm is treated as a black box, which also allows the practitioner to decide on the learner.

Extending the feature types used by RFEST is left for future work. Additionally, one may want to implement different search heuristics. An example would be to use a greedy hill climbing or a simulated annealing approach (see **Algorithm 1**). Perhaps if one knew that the input data contained many correlated features, then applying a filter algorithm before RFEST will aid in removing redundant features. Lastly, it would be interesting to compare RFEST with more feature selection algorithms and to more real-world datasets.

Appendix A

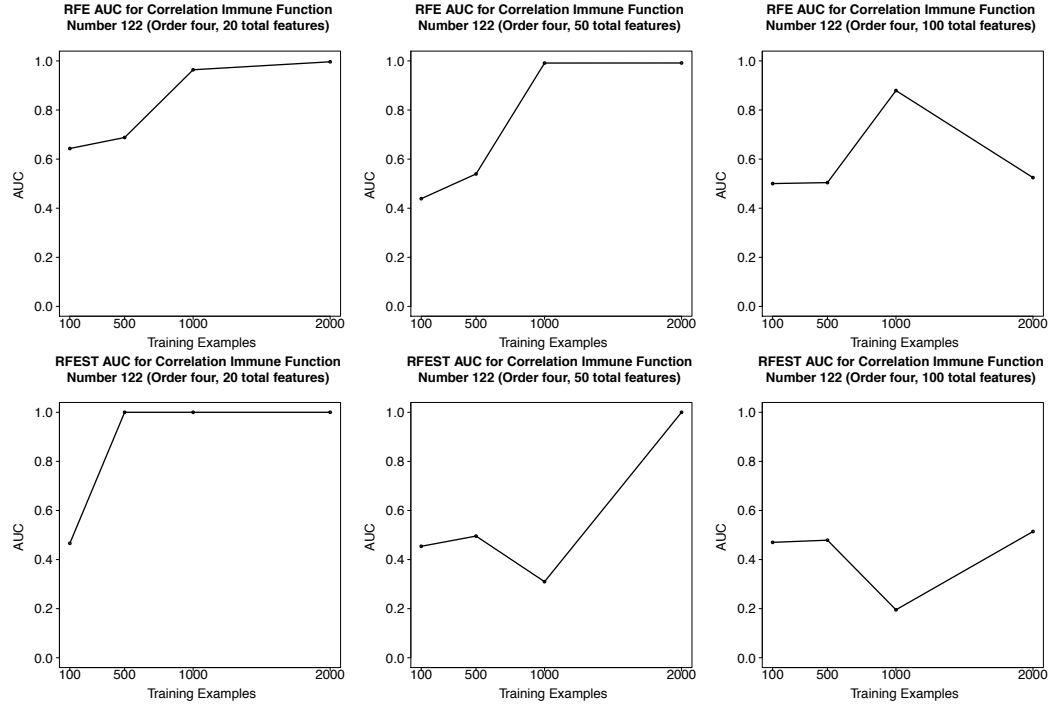


Figure A.1. RFE and RFEST AUC for CI function number 122 (order four). The total number of features increase from left to right graphs (20, 50, and 100 features, respectively).

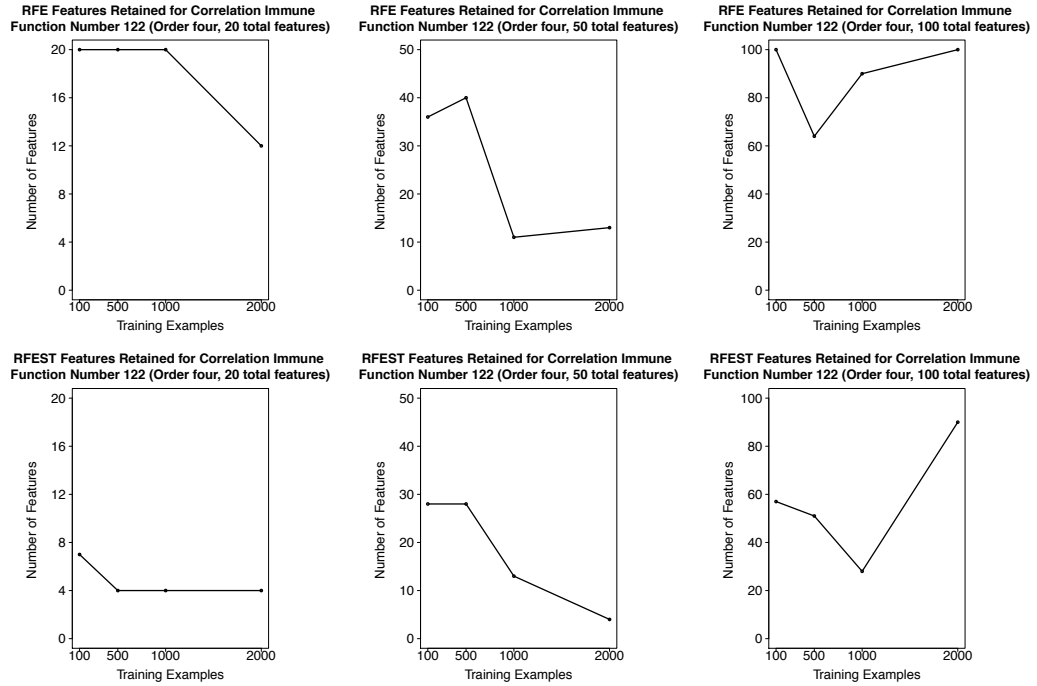


Figure A.2. RFE and RFEST number of features retained for CI function number 122 (order 122). The total number of features increase from left to right graphs (20, 50, and 100 features, respectively). The relevant features were set to the first four features in the dataset (i.e. X_1 , X_2 , X_3 and X_4).

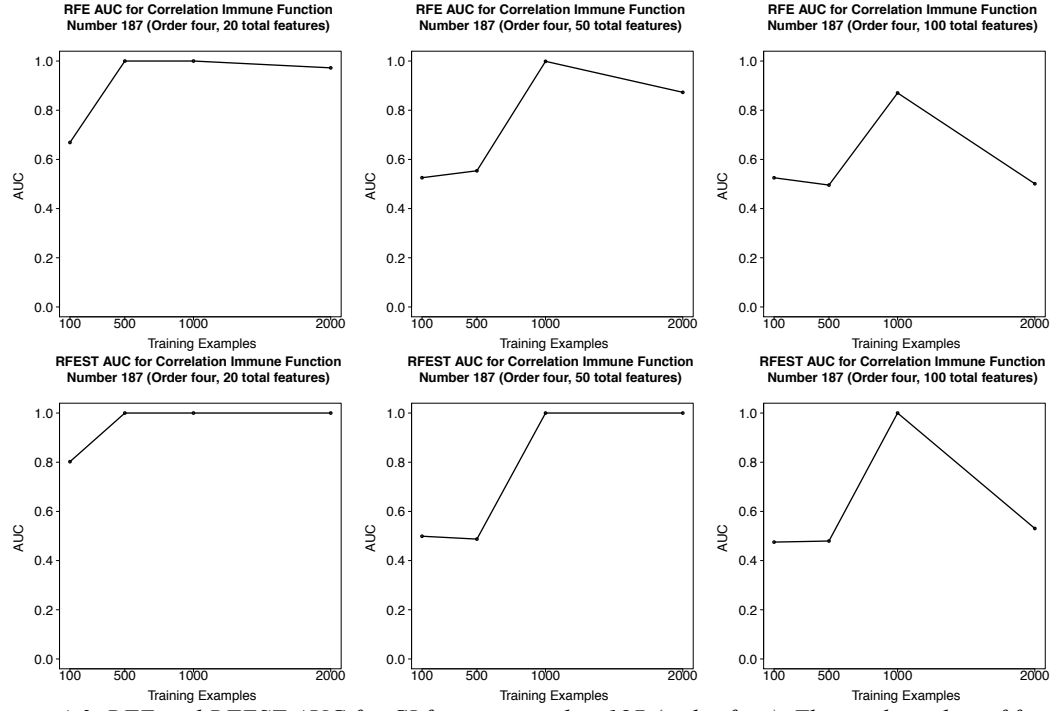


Figure A.3. RFE and RFEST AUC for CI function number 187 (order four). The total number of features increase from left to right graphs (20, 50, and 100 features, respectively).

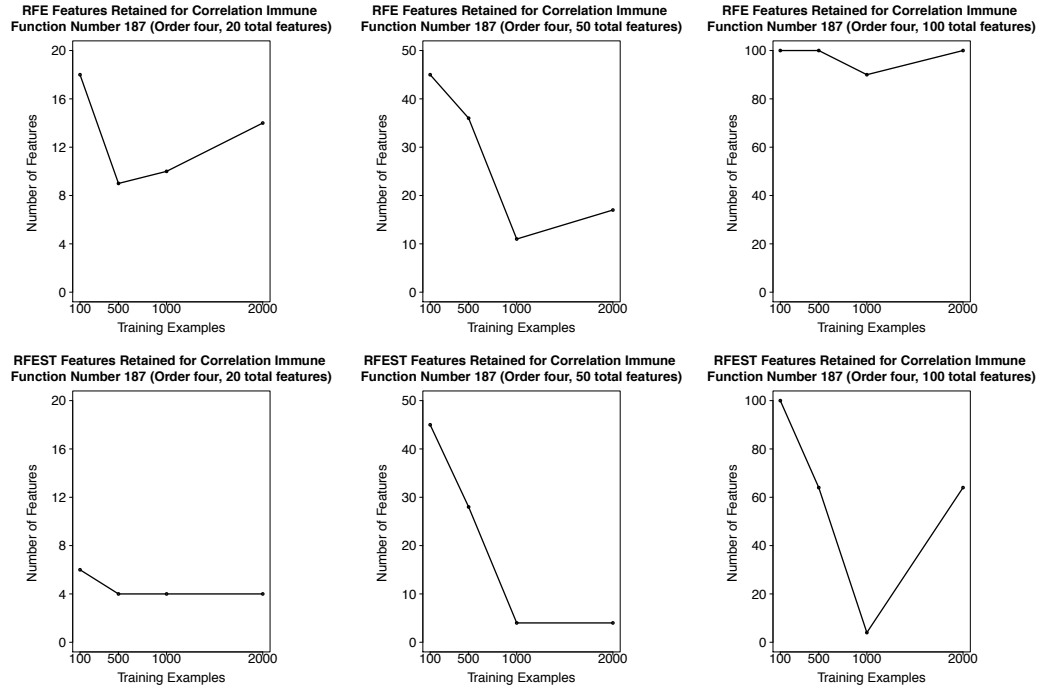


Figure A.4. RFE and RFEST number of features retained for CI function number 187 (order four). The total number of features increase from left to right graphs (20, 50, and 100 features, respectively). The relevant features were set to the first four features in the dataset (i.e. X_1 , X_2 , X_3 and X_4).

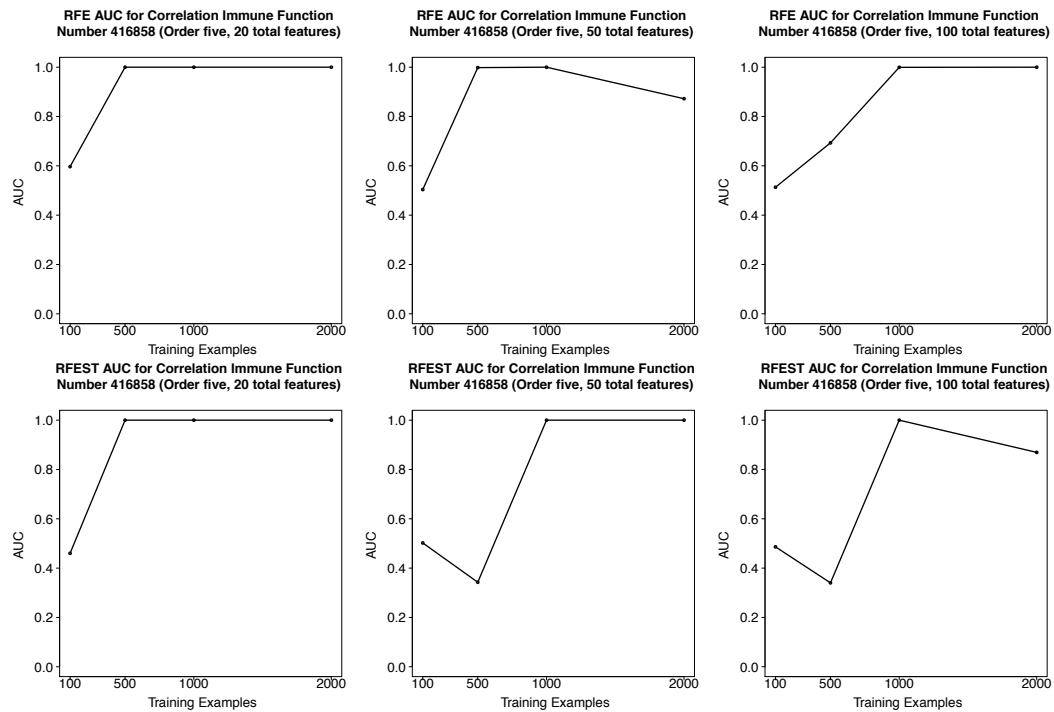


Figure A.5. RFE and RFEST AUC for CI function number 416858 (order five). The total number of features increase from left to right graphs (20, 50, and 100 features, respectively).

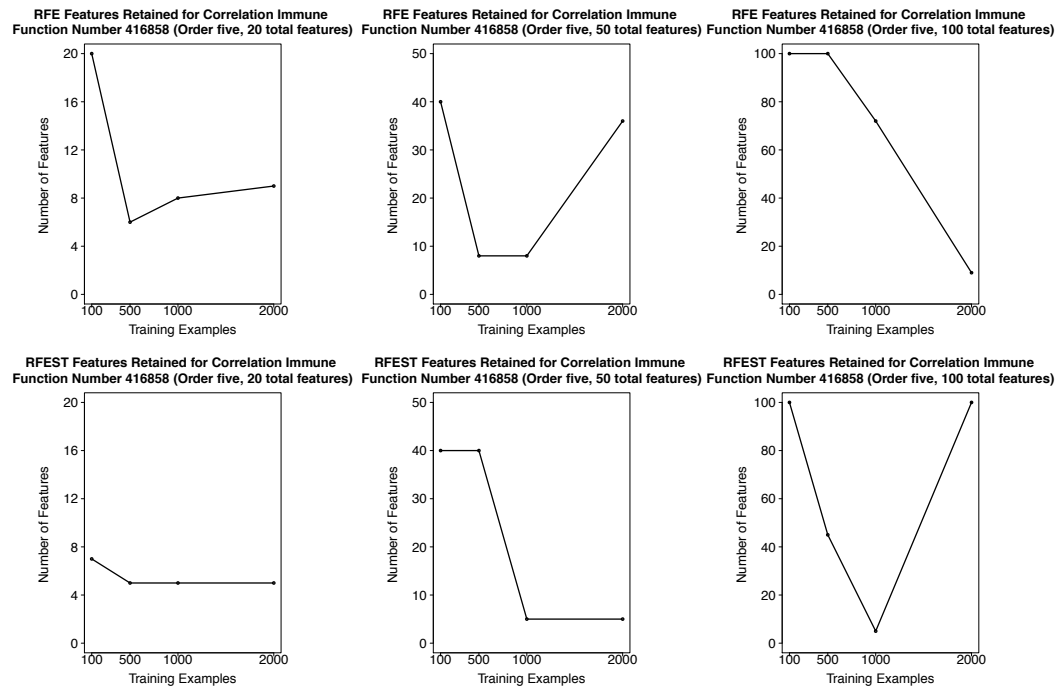


Figure A.6. RFE and RFEST number of features retained for CI function number 416858 (order five). The total number of features increase from left to right graphs (20, 50, and 100 features, respectively). The relevant features were set to the first five features in the dataset (i.e. X_1 , X_2 , X_3 , X_4 , and X_5).

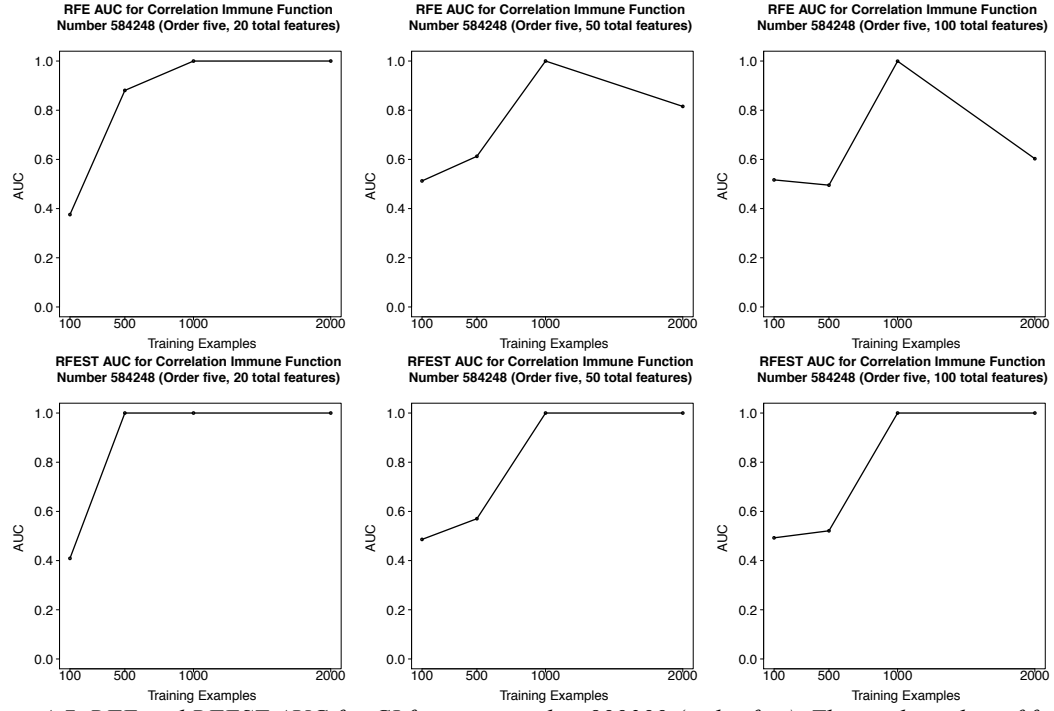


Figure A.7. RFE and RFEST AUC for CI function number 899399 (order five). The total number of features increase from left to right graphs (20, 50, and 100 features, respectively).

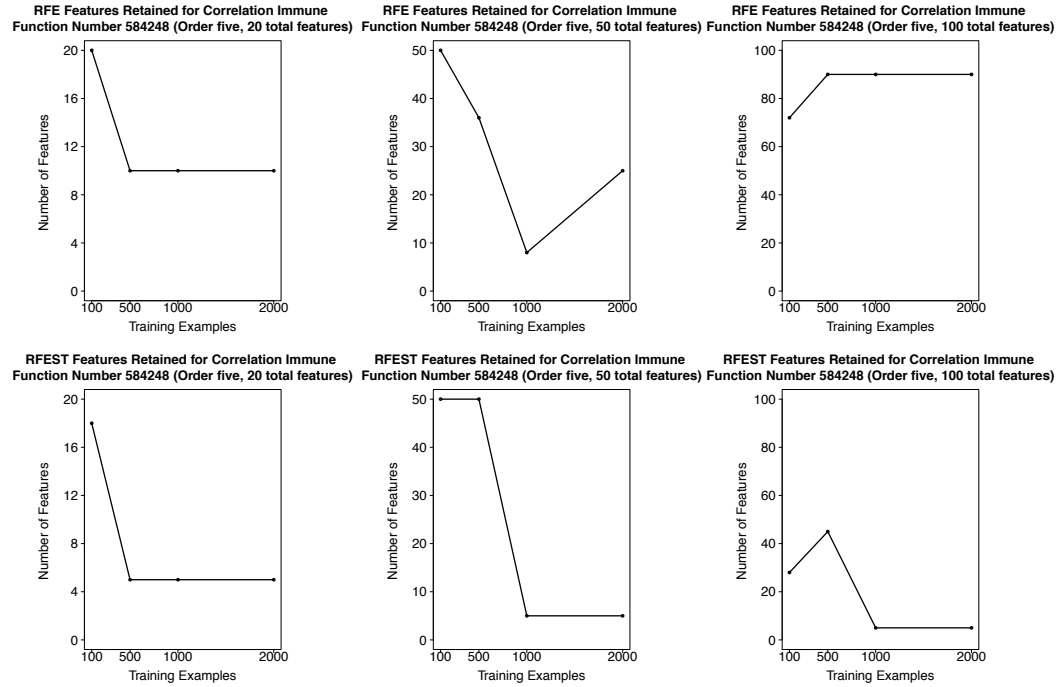


Figure A.8. RFE and RFEST number of features retained for CI function number 899399 (order five). The total number of features increase from left to right graphs (20, 50, and 100 features, respectively). The relevant features were set to the first five features in the dataset (i.e. X_1 , X_2 , X_3 , X_4 , and X_5).

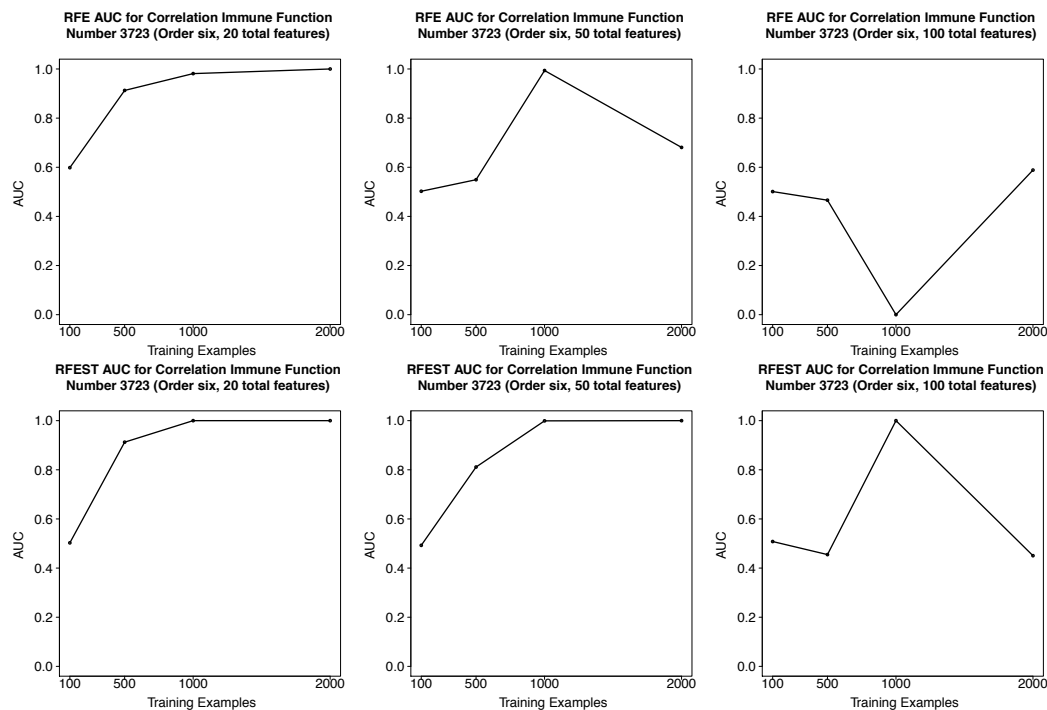


Figure A.9. RFE and RFEST AUC for CI function number 3723 (order six). The total number of features increase from left to right graphs (20, 50, and 100 features, respectively).

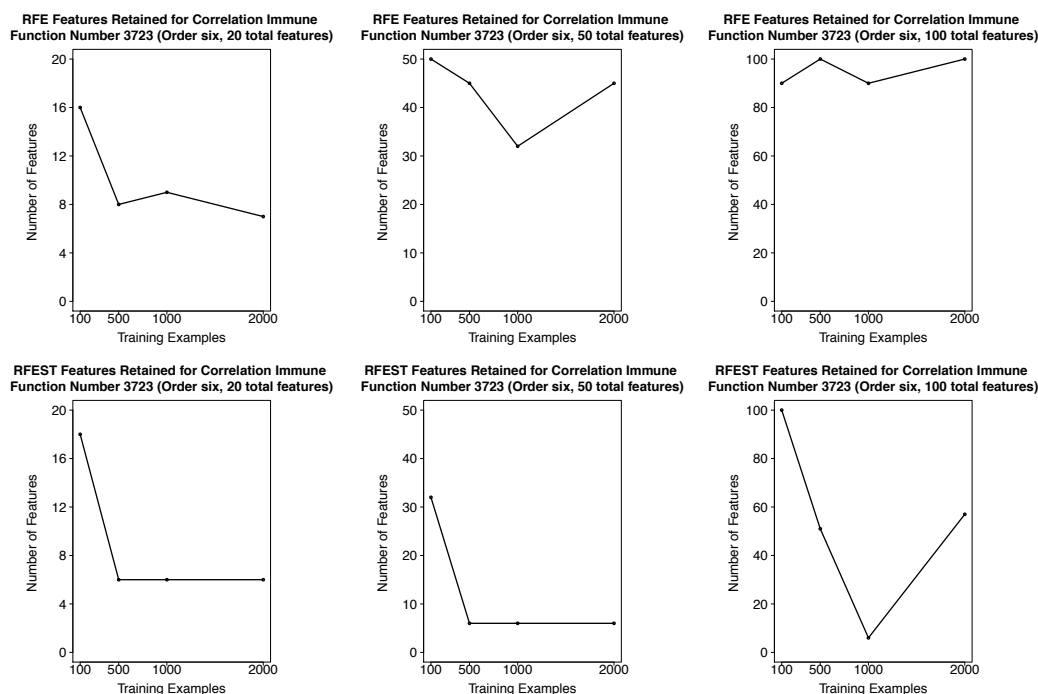


Figure A.10. RFE and RFEST number of features retained for CI function number 3723 (order six). The total number of features increase from left to right graphs (20, 50, and 100 features, respectively). The relevant features were set to the first six features in the dataset (i.e. X_1 , X_2 , X_3 , X_4 , X_5 , and X_6).

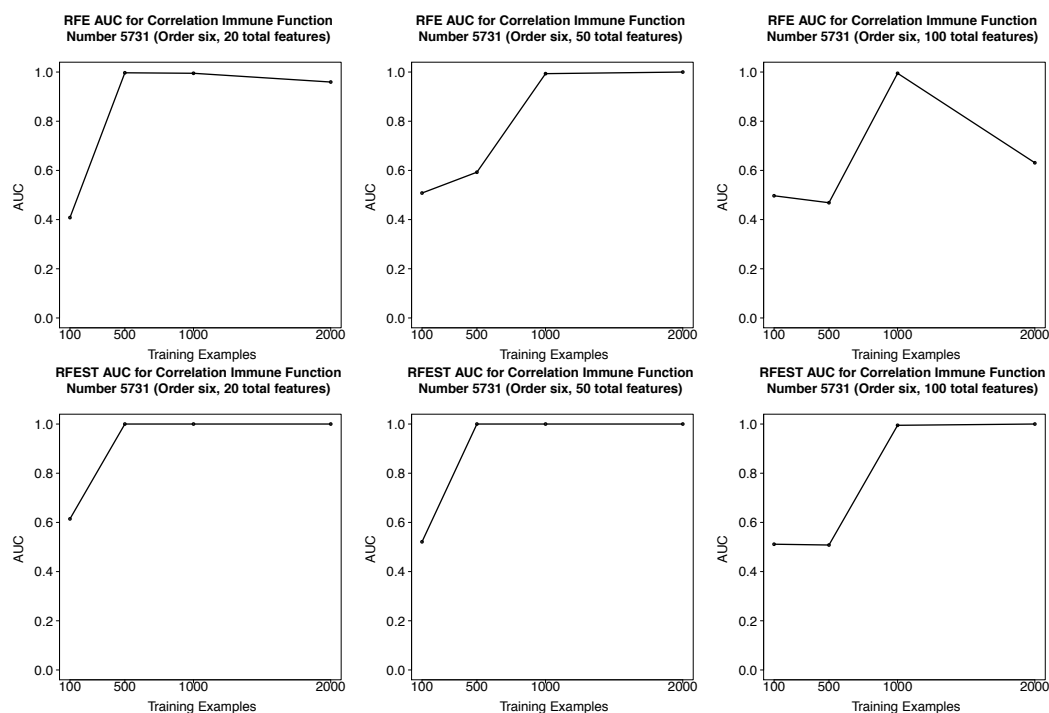


Figure A.11. RFE and RFEST AUC for CI function number 5731 (order six). The total number of features increase from left to right graphs (20, 50, and 100 features, respectively).

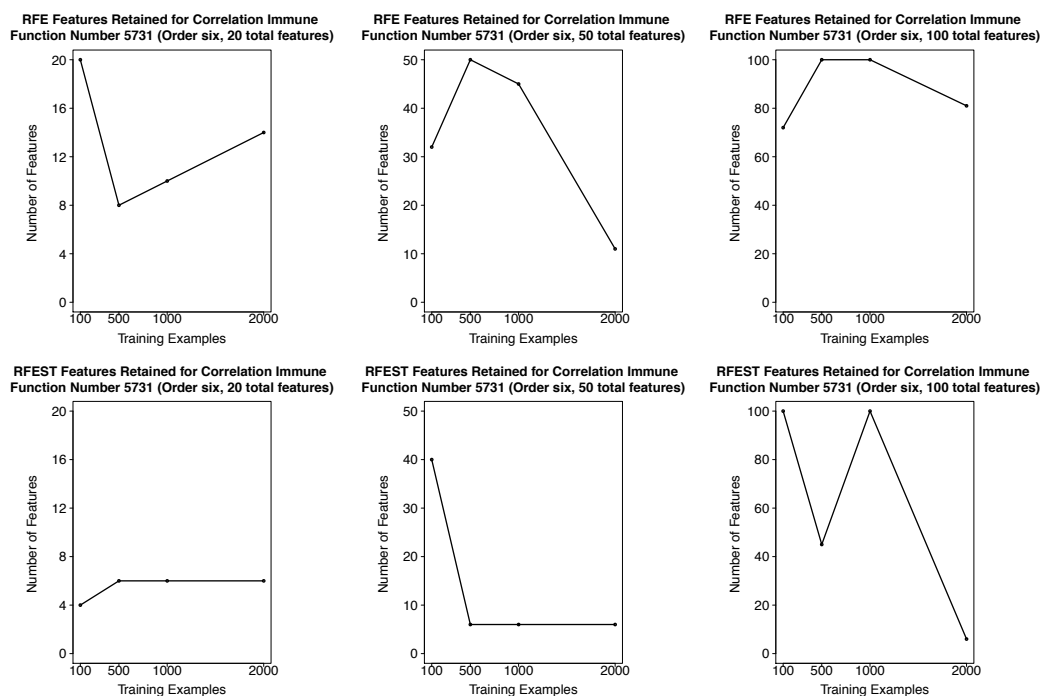


Figure A.12. RFE and RFEST number of features retained for CI function number 5731 (order six). The total number of features increase from left to right graphs (20, 50, and 100 features, respectively). The relevant features were set to the first six features in the dataset (i.e. X_1 , X_2 , X_3 , X_4 , X_5 , and X_6).

Bibliography

- Ahsan , H., Halpem, J., Kibriya, M., Pierce , B., Tong , L., Gamazon, E., . . . Whittmore, A. (2014). A genome-wide association study of early-onset breast cancer identifies PFKM as a novel breast cancer gene and supports a common genetic spectrum for breast cancer at any age. *Cancer Epidemiology and Prevention Biomarkers*, 4, 658-669.
- Balding, D. J. (2006, 10). A tutorial on statistical methods for population association studies. *Nat Rev Genet* , 7(10), 781-791 .
- Battiti, R. (1994, 07). Using mutual information for selecting features in supervised neural net learning. *IEEE Transactions on Neural Networks*, 5(4), 537-550.
- Ben-Hur, A., & Weston, J. (2010). A User's Guide to Support Vector Machines. In O. a. Carugo, *Data Mining Techniques for the Life Sciences* (pp. 223--239). Totowa, NJ: Humana Press.
- Blum, A., & Langley, P. (1997, 12 01). Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97(1), 245-271.
- Caruana, R., & Niculescu-Mizil, A. (2006). An Empirical Comparison of Supervised Learning Algorithms. *ICML '06* (pp. 161-168). Pittsburgh: Proceedings of the 23rd International Conference on Machine Learning.
- Chandrashekar, G., & Sahin, F. (2014). A survey on feature selection methods. *Computers & Electrical Engineering*, 40(1), 16-28.
- Cline, T. W. (1979, 06 16). A male-specific lethal mutation in *Drosophila melanogaster* that transforms sex . *Developmental Biology* , 72(2), 266-275.
- Colletti, J. A., Leland-Wavrin, K. M., Kurz, S. G., Hickman, P. M., Seiler, N. L., Samanas, N. B., . . . Shull, J. D. (2014). Validation of Six Genetic Determinants of Susceptibility to Estrogen-Induced Mammary Cancer in the Rat and Assessment of Their Relevance to Breast Cancer Risk in Humans. *G3: Genes, Genomes, Genetics*, 4(8), 1385--1394.
- Easton, D. F., Pooley, K. A., Dunning, A. M., Pharoah, P. D., Thompspon, D., Ballinger, D. G., . . . Ponder, B. A. (2007, 06). Genome-wide association study identifies novel breast cancer susceptibility loci. *Nature*, 447(7148), 1087-1093.
- Fletcher, O., Johnson, N., Orr, N., Hosking, F. J., Gibson, L. J., Walker , K., . . . Peto, J. (2011, 03 02). Novel Breast Cancer Susceptibility Locus at 9q31.2: Results of a Genome-Wide Association Study. *JNCI: Journal of the National Cancer Institute*, 103(5), 425-435.
- Gardner, M. W., & Dorling, S. R. (1998). Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. *Atmospheric Environment*, 32(14), 2627-2636.
- Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning* (1st ed.). Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.
- Guyon, I., & Elisseeff, A. (2003, 3 3). An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research*, 3, 1157-1182.
- Guyon, I., Weston, J., Barnhill, S., & Vapnik, V. (2002). Gene Selection for Cancer Classification using Support Vector Machines. *Machine Learning*, 46(1), 389--422.

- Hellerstein, L., & Servedio, R. A. (2007, 06). On PAC learning algorithms for rich Boolean function classes. *Theoretical Computer Science*, 384(1), 66-76.
- Hellerstein, L., Rosell, B., Bach, E., Ray, S., & Page, D. (2009, 12 1). Exploiting Product Distributions to Identify Relevant Variables of Correlation Immune Functions. *J. Mach. Learn. Res.*, 10, 2374-2411.
- Hsu, C.-w., Chang, C.-c., & Lin, C.-j. (2010). A practical guide to support vector classification.
- Hunter, D. J., Kraft, P., Jacobs, K. B., Cox, D. G., Yeager, M., Hankinson, S. E., . . . Chanock, S. J. (2007, 07). A genome-wide association study identifies alleles in FGFR2 associated with risk of sporadic postmenopausal breast cancer. *Nat Genet*, 39(7), 870-874.
- Jain, A., Mao, J., & Mohiuddin, K. (1996). Artificial neural networks: a tutorial. *Computer*, 29(3), 31-44.
- Karatzoglou, A., Smola, A., Hornik, K., & Zeileis, A. (2004). kernlab - An S4 Package for Kernel Methods in R. *Journal of Statistical Software, Articles*, 11(9), 1-20.
- Kearns, M., & Li, M. (1993). Learning in the Presence of Malicious Errors. *SIAM Journal on Computing*, 22(4), 807-837.
- Kim, Y., & Kim, J. (2004). Gradient LASSO for Feature Selection. *ICML '04* (pp. 60--). New York: ACM.
- Kira, K., & Rendell, L. A. (1992). *The Feature Selection Problem: Traditional Methods and a New Algorithm*. San Jose, California, USA: AAAI Press.
- Kuhn, M. (2008). Building Predictive Models in R Using the caret Package. *Journal of Statistical Software, Articles*, 28(5), 1-26.
- Lal, T. N., Chapelle, O., Weston, J., & Elisseeff, A. (2006). Embedded Methods. In I. Guyon, M. Nikravesh, S. Gunn, & L. A. Zadeh (Eds.), *Feature Extraction: Foundations and Applications* (pp. 137-165). Berlin, Heidelberg: Springer Berlin Heidelberg.
- Lantz, B. (2013). *Machine Learning with R*. Packt Publishing.
- Law, M., Figueiredo, M., & Jain, A. (2004, 09). Simultaneous feature selection and clustering using mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26, 1154-1166.
- LeCun, Y., & Bengio, Y. (1998). The Handbook of Brain Theory and Neural Networks. In M. A. Arbib (Ed.). Cambridge, MA, USA: MIT Press.
- Ledesma, S., Cerda, G., Aviña, G., Hernández, D., & Torres, M. (2008). Feature Selection Using Artificial Neural Networks. In A. Gelbukh, & E. F. Morales (Eds.), *MICAI 2008: Advances in Artificial Intelligence: 7th Mexican International Conference on Artificial Intelligence* (pp. 351--359). Atizapán de Zaragoza, Mexico: Springer Berlin Heidelberg.
- Liu, H., Motoda, H., Rudy, S., & Zhao, Z. (2010). Feature Selection: An Ever Evolving Frontier in Data Mining. In H. Liu, H. Motoda, S. Rudy, & Z. Zhao (Ed.), *Proceedings of the Fourth International Workshop on Feature Selection in Data Mining*, 10, pp. 4-13. Hyderabad, India: PMLR.
- Liu, J., Page, D., Peissig, P., McCarty, C., Onitilo, A. A., Trentham-Dietz, A., & Burnside, E. (2014). New Genetic Variants Improve Personalized Breast Cancer Diagnosis. *AMIA Summits on Translational Science Proceedings, 2014*, 83--89.
- Maldonado, S. W. (2011). *Embedded Feature Selection for Support Vector Machines: State-of-the-Art and Future Challenges*. (C. a.-W. San Martin, Ed.) Heidelberg, Berlin: Springer Berlin Heidelberg.

- Michailidou, K., Hall, P., Gonzalez-Neira, A., Ghoussaini, M., Dennis, J., Milne, R. L., . . . Easton, D. F. (2013, 04). Large-scale genotyping identifies 41 new loci associated with breast cancer risk. *Nat Genet*, 45(4), 353-361.
- Saeys, Y., Inza, I., & Larrañaga, P. (2007). A review of feature selection techniques in bioinformatics. *Bioinformatics*, 23(19), 2507-2517.
- Schaffer, B. S., Lachel, C. M., Pennington, K. L., Murrin, C. R., Strecker, T. E., Tochacek, M., . . . Shull, J. D. (2006). Genetic Bases of Estrogen-Induced Tumorigenesis in the Rat: Mapping of Loci Controlling Susceptibility to Mammary Cancer in a Brown Norway x ACI Intercross. *Cancer Research*, 66(15), 7793-7800.
- Shull, J. (2007). The rat oncogenome: comparative genetics and genomics of rat models of mammary carcinogenesis. *Breast Disease*, 28, 69-86.
- Sing, T., Sander, O., Beerenwinkel, N., & Lengauer, T. (2005, 10). ROCR: visualizing classifier performance in R. *Bioinformatics*, 21(20), 3940-3941.
- Stambaugh, C., Yang, H., & Breuer, F. (2013). Analytic Feature Selection for Support Vector Machines. *CoRR*, abs/1304.5678.
- Swain, S. M. (1992). Ductal Carcinoma in Situ. *Cancer Investigation*, 10(5), 443-454.
- Tang, J., Alelyani, S., & Liu, H. (2014). *Feature Selection for Classification: A Review*. Data .
- Tempfer, C., Hefler, L., Schneeberger, C., & Huber, J. (2006). How valid is single nucleotide polymorphism (SNP) diagnosis for the individual risk assessment of breast cancer? *Gynecological Endocrinology*, 22(3), 155-159.
- Tibshirani, R. (1996). Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1), 267-288.
- Turnbull, C., Ahmed, S., Morrison, J., Pernet, D., Renwick, A., Maranian, M., . . . Easton, D. F. (2010, 06). Genome-wide association study identifies five new breast cancer susceptibility loci. *Nat Genet*, 42(6), 504-507.
- Wickham, H. (2011). The Split-Apply-Combine Strategy for Data Analysis. *Journal of Statistical Software, Articles*, 40(1), 1-29.