UNIVERSITY OF CALIFORNIA, SAN DIEGO

Prediction and variable selection in political science

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Political Science

by

Adeline Yunchen Lo

Committee in charge:

Professor James Fowler, Chair
Professor Claire Adida
Professor David Lake
Professor Margaret Roberts
Professor Yixiao Sun
Professor Tian Zheng

2016

The dissertation of Adeline Yunchen Lo is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

_____

_____

_____

_____

_____

_____

Chair

University of California, San Diego

2016

DEDICATION

To Vicky, Alexander, Shaw-hwa and Héctor.

EPIGRAPH

Share your knowledge.
It is a way to achieve immortality.
—Dalai Lama XIV

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to acknowledge Professor James Fowler for his help and advice as my primary advisor. His support throughout my time at UCSD has been invaluable, particularly his willingness and enthusiasm in backing my, sometimes unorthodox, ideas and methods of acquiring new tools and knowledge. Thanks for reminding me to be excited about my ideas, always.

I would also like to acknowledge the help throughout of Professors Tian Zheng and Shaw-Hwa Lo for mentorship and guidance in the technical and statistical aspects of my dissertation work. My visit to Columbia Statistics meant leaps and bounds in the growth of my statistics education. Professors Claire Adida and David Lake helped me ground myself as a political scientist and challenge myself to consider methodology from the standpoint of an applied comparative political scientist. Thanks particularly to Professor Adida for sending me to Benin on fieldwork almost immediately in the program. My idea book and my personal approach to research benefited enormously from the firsthand experience of running a survey experiment so early on. Professor Yixiao Sun taught me nearly everything I know about panel data — and to remember that statistical methods should serve a purpose to the applied scientist. I'd also like to thank Professor Herman Chernoff for inspiring me throughout our collaborations as a researcher; I can only hope to have a fraction of your energy and passion for intellectual pursuits throughout my life.

My research group has been a source of knowledge and immeasurable support throughout these five years. I thank Veena Blume, Maya Duru, Erin Giffin, Shanthi Manian, Héctor Pifarré i Arolas, and Erin Wolcott for the countless feedback on my papers, brainstorming sessions, and practice talks. Thanks in particular to Héctor, who has seen nearly every single failed, semi-successful and actually successful idea I've thought about in the last five years and patiently given positive feedback.

I am grateful to my coauthors for permission to include some of our work in my dissertation. Chapter 1, in full, is a reprint of the material as it appears in "Why significant variables aren't automatically good predictors" 2015, Lo, Adeline; Chernoff, Herman; Lo, Shaw-hwa; Zheng, Tian., Proceedings of the National Academy of Sciences. This paper was coauthored amongst Adeline Lo, Herman Chernoff, Shaw-Hwa

Lo, and Tian Zheng. The material in Chapter 2 is currently being prepared for submission for publication of the material. This material was coauthored amongst Adeline Lo, Herman Chernoff, Shaw-Hwa Lo, and Tian Zheng. All remaining errors are my own.

VITA

2008	B. A. in Economics & Political Science, Columbia College, Columbia University, New York
2010	M. A. in Politics, New York University, New York
2011-2016	Graduate Teaching Assistant, University of California, San Diego
2016	Ph. D. in Political Science, University of California, San Diego

PUBLICATIONS

Lo, A. and Fowler, J.H., 2013. Social science: The mathematics of murder. *Nature*, 501(7466), pp.170-171.

Adida, C., Combes, N., Lo, A. and Verink, A., 2015. The Political Implications of Cross-Ethnic Marriage in Africa. *Comparative Political Studies*. 49(5), pp. 635-661.

Lo, A., Chernoff, H., Zheng, T. and Lo, S.H., 2015. Why significant variables aren't automatically good predictors. *Proceedings of the National Academy of Sciences*, 112(45), pp.13892-13897.

ABSTRACT OF THE DISSERTATION

Prediction and variable selection in political science

by

Adeline Yunchen Lo

Doctor of Philosophy in Political Science

University of California, San Diego, 2016

Professor James Fowler, Chair

This dissertation considers the topics of prediction and variable selection for the applied political scientist, particularly in the context of high dimensional data.

In Chapter 1, we consider the puzzle of why highly significant variables aren't automatically good predictors. This problem occurs in both simple and complex data. We offer explanations and statistical insights into the puzzle. We suggest shifting the research agenda toward searching for a criterion to locate highly predictive variables rather than highly significant variables. We offer an alternative approach, the Partition Retention (PR) method, which was effective in reducing prediction error from 30% to 8% on a long studied breast cancer data set.

In Chapter 2, we propose approaching prediction from a framework grounded in finding the correct prediction rates of variables. While intuitively obvious, not nearly

enough attention has been paid to creating a clear theoretical framework for prediction. We present an objective function for prediction rates and consider but ultimately reject an estimator based on the sample analog of the solution due to its inability to distinguish predictive variables from noisy ones, which leads to an inability to estimate it. We offer an alternative solution and demonstrate that the PR's I-score asymptotically approaches this alternative solution. The I-score for a variable set can be written as an asymptotic lower bound for the correct prediction rate. We offer simulations and applications of the I-score on real data.

In Chapter 3, I propose a new approach to predicting civil war onsets that emphasizes variable selection. A good variable selection approach should search for variables based on a criterion of predictivity and find variable interactions. I suggest the PR method to conduct variable selection and illustrate with simulations and an application to civil wars data, comparing results with alternative approaches. The PR identifies variable sets, some as large as 5 or 6 variables, to predict war onsets. Using these variable sets to predict boosts correct prediction rates on out of sample data from 78.98% to 98.05%. The application demonstrates gains in prediction rates for political phenomena like civil wars when including a research step for variable selection.

Chapter 1

Why significant variables aren't automatically good predictors

1.1 Abstract

Thus far, genome wide association studies (GWAS) have been disappointing in the inability of investigators to use the results of identified, statistically significant variants in complex diseases to make predictions useful for personalized medicine. Why aren't significant variables leading to good prediction of outcomes? We point out that this problem is prevalent in simple as well as complex data, in the sciences as well as the social sciences. We offer a brief explanation and some statistical insights on why higher significance cannot automatically imply stronger predictivity and illustrate through simulations and a real breast cancer example. We also demonstrate that highly predictive variables do not necessarily appear as highly significant, thus evading the researcher using significance-based methods. We point out that what makes variables good for prediction versus significance depends on different properties of the underlying distributions. If prediction is the goal, we must lay aside significance as the only selection standard. We suggest that progress in prediction requires efforts towards a new research agenda of searching for a novel criterion to retrieve highly predictive variables rather than highly significant variables. We offer an alternative approach that was not designed for significance, the Partition Retention method, that was very effective predicting on a

long studied breast dancer data set, by reducing the classification error rate from 30% to 8%.

1.2 Introduction

An early 2013 Nature Genetics article "Predicting the influence of common variants." 2013, "Predicting the influence of common variants", identified prediction as an important goal for current genome wide association studies (GWAS). However, a puzzle that has recently arisen in the GWAS related literature is that an increase in newly identified variants (variables) does not necessarily seem to lead to improvements in current predictive models. While intuitively it would seem that the addition of information (more statistically significant variants) should increase predictive powers, in recent models of prediction the power is not increased when adding more significant variants to classical significance test-based approaches (Clayton 2009; Campos, Gianola, and Allison 2010; Jakobsdottir et al. 2009; Janssens and Duijn 2008).[1]

A typical GWAS study collects data on a sample of subjects, cases, who have a disease, and controls, who are disease free. A very large list of single-nucleotide polymorphisms (SNPs) is evaluated for each individual where each SNP corresponds to a given locus on the genome, and can take on the value 0, 1, or 2 depending on how many copies of the "minor" allele show up. The SNPs are distributed over the whole genome. Typically the researcher wants to select a subgroup of the SNPs that is associated with the disease, so that she can study how the disease works. She may also be interested in predicting whether a new individual has the disease by analyzing the individual's selected SNPs.

Whether or not an individual has the disease is regarded as the dependent variable.[2] The SNP values are the explanatory variables. In a typical study there may be several thousand subjects and hundreds of thousands of SNPs. From the scientist's point of view there are two basic problems, complicated by the large size of the data set. These are variable selection and prediction. For variable selection, we wish to find a relatively small set of SNPs associated with the disease. For prediction we wish to find how a

[1] We refer to "statistically significant" variables throughout this paper as simply "significant".

[2] Here we focus on discrete outcomes, as is common in GWAS studies that are case-control.

small set of such variables can be used to predict whether the subject has the disease. The size of the data set is such that the typical approach to variable selection has been to see how well correlated each SNP value is with the disease, and to keep only those for which the statistical significance was very high. Only recently has there been serious consideration of the possible interactions among two or more SNPs by some investigators. The prediction problem has typically been approached by using some variation of linear regression based on the limited number of SNPs from the variable selection stage.

If predictivity is measured by how well the method works on the (training) data used to derive the predictions, we are almost bound to get overoptimistic results. Methods of cross validation will result in more accurate estimates. Alternatively one may use a separate test sample, independent of the data used to produce the prediction model. Much of our discussion is also relevant to large data sets in other fields of study. Indeed, this problem is not unique to genetic data; we find cases of similar problems in the social sciences. For instance, significant explanatory variables for civil wars serve nearly negligible input for predicting civil wars (Ward, Greenhill, and Bakke 2010). Likewise, variables found to be significant for fluctuations in the stock market index carry no predictive power (Welch and Goyal 2008). This phenomenon is pervasive across different types of data as well as different sample sizes. Thus the goal of this paper is to offer theoretical insight and illuminating examples to demonstrate precisely how finding highly significant variables is different from finding highly predictive ones — regardless of data type. For illustrative purposes however, we use the lens of prediction for genetic data throughout.

One might ask why one method of variable selection that works perfectly well for a significance-based research question might not work so well for a classification-based research question. Fundamentally, the main difference is that what constitutes a good variable for classification and what constitutes a good variable for significance depend on different properties of the underlying distributions. To test for significance is a test of the null hypothesis that the distributions of X under the two states are the same, whereas the classification error is a test of whether X belongs to one state or the other. Different properties of the distributions are involved. The tests used also may or may not be efficient. In fact significance was not originally designed for the purposes of

prediction.

Some might also comment that perhaps it is clear and intuitive why it is that some significant variables do not appear as highly predictive. After all, variables may be significantly associated with the outcome simply for a small group of individuals in the population, thereby leading to poor prediction on the population. This is true to an extent. However, there is still a fair amount of research using significant variables to predict, perhaps because of a lack of obvious alternative options for variable selection. For instance, currently, prediction oriented GWAS research uses genetic variants for constructing additive prediction models for estimating disease risk. A recent New England Journal of Medicine article illustrates one example of such an approach, whereby researchers constructed a model based on five genetic variants from GWAS results on prostate cancer; they report that the variants do not increase predictive power (Zheng et al. 2008). Likewise, Gransbo et al. show that chromosome 9p21, while significantly associated with cardiovascular disease, does not improve risk prediction (Gransbo et al. 2013).

In addition, while the intuition behind significant variables not appearing predictive might be reasonably obvious, the fact that highly predictive variables do not appear necessarily as highly significant is perhaps less so. We discuss and then demonstrate this phenomenon with both a theoretical explanation and a series of examples. Finally, while superficially we might reason that indeed, significance cannot be the same as predictivity, why this is precisely so and what makes for their differences is also not quite so obvious.

With this in mind, we provide a short theoretical explanation for the differences between highly significant and highly predictive variables. We then demonstrate, with a series of artificial examples of increasing relevance, how and why seeking significance and prediction can lead to very different decisions in variable selection. These examples are artificial, partly because they assume that the underlying probabilities are known, whereas the scientist can only infer these from the data. In these examples we compare significance and prediction, and show how the relatively simple I-score, defined in the Methods section, which we have used in our Partition Retention (PR) approach to variable selection (Chernoff, Lo, and Zheng 2009; Lo and Zheng 2004; Zheng et al.

2010; Zheng, Wang, and Lo 2006), seems to correlate well with predictivity. We offer the I-score as one possible useful tool in the study of increasing predictivity. We show a highly successful real application of the PR approach for increasing predictivity in the analysis of a longstanding data set on breast cancer, for which we show some results. Finally some conclusions are offered to aid in the study of improving predictivity in GWAS research.

There is a long established literature in Statistics on Classification with major applications to Biology. In recent years the fields of pattern recognition, machine learning and computer science became heavily involved, often with different terminology and new ideas adapted to the increasing size of the relevant data sets. In the last section (Methods), we present a very brief description of some of the techniques, approaches and terminology.

1.3 Highly significant vs highly predictive variables

Data has substantially grown in recent years with both exponential increases in the number of variables and, in many cases, increases in sample sizes as well. This has served as stimulation for a large number of applications via the novel retooling of well-known concepts. Two popular concepts, statistical significance and prediction (including classification), serve as the focus of this article. Historically, significance has played a larger role in statistical inference while prediction has served more in identifying future data behavior. The retooling of significance has found a role in data dimension reduction for prediction, that of guiding the feature selection/variable selection step (Guyon et al. 2003). We evaluate this retooling and consider how significance and predictivity are related in the goal of good prediction.

We have mentioned that a key difference between what makes a variable highly significant versus highly predictive lies in different properties of their underlying distributions. We elaborate on this point a bit more here.

Suppose a statistician is given a variable set denoted by X. It is assumed that among control observations X follows a distribution f_H and among cases X follows a distribution f_D. The statistician wishes to test the null hypothesis H_0 that $f_D = f_H$

against the alternative hypothesis H_a that $f_D \neq f_H$, where f_D is not specified, using observed data and assess the statistical significance of the observed data with respect to the null hypothesis. He also wishes to evaluate how strong a predictor based on this variable set could be in predicting the case/control label of future data. Particularly, in a case-control study, he is interested in whether case samples (from f_D) are significantly differently from control samples (from f_H).

To carry out a test between H_0 and H_a based on variable set x, the statistician chooses a test statistic T_n and, based on the observed values x of X for the n cases and n controls, calculates $t_n = T_n(x)$. Then one can claim that f_H and f_D are significantly different if the probability $P(T_n \geq t_n)|H_0)$, which we call the p-value, is sufficiently small.

To decide whether x, the observed value of X for a single individual, comes from the distribution f_D or from f_H, when the costs of false positives and false negatives are equal and both possibilities are equally likely, the appropriate Bayes decision rule is to decide in favor of the larger of $f_D(x)$ and $f_H(x)$. Then the corresponding error rates are $\sum_{x:f_D(x)<f_H(x)} f_D(x)$ and $\sum_{x:f_D(x)\geq f_H(x)} f_H(x)$. The average of these two is $0.5\sum_x \min(f_D(x), f_H(x))$ which, together with $0.5\sum_x \max(f_D(x), f_H(x))$ add to 1.[3] Thus we may write:

$$\text{prediction rate} = 0.5\sum_x \max(f_D(x), f_H(x)) \tag{1.1}$$

Here x represents the possibly multivariate observation that can assume a finite number of values; f_D and f_H are its probability distributions, under case and control respectively. Equation (1) defined above requires the knowledge of the true probability distributions, while, in practice, the statistician can only infer such knowledge from the data.

The key difference between finding a subset of variables to be highly significant versus finding it to be highly predictive is that the former uses assumptions on, but no knowledge of, the exact distributions of the variables, while the latter, as shown in Equation (1), requires knowledge of both f_D and f_H.

Should the statistician still wish to pursue the significance route to identify vari-

[3] We note that the prediction rate can be seen as equal to 1 minus the average error rate. For continuous distributions, Equation 1 would be written with integrals rather than summations.

ables that are highly predictive, he might wish to compare two subsets of explanatory variables, x and x', for their usefulness in the prediction problem. Here x' has distributions f'_D and f'_H. It is a current practice to carry out the comparison by testing the null hypotheses $f_H = f_D$ and $f'_H = f'_D$ and seeing which has a smaller p-value. Because of his limited knowledge on the underlying distributions he is restricted to use tests that are not necessarily powerful enough. Often he is reduced to using a chi-square technique, recommended, for example, in the studies of complex diseases, which is not very powerful for the multiple variable cases. The suboptimality of the test procedure makes the significance level an unreliable basis for comparing subsets of variables and for the usefulness in prediction. It is no surprise that searching for variables based on significance level and based on correct prediction rate can lead us in conflicting directions.

The statistician's p-value for the test is a random variable and here we have assigned the *significance* value to be the median of the p-values, which we may calculate, knowing the probability distributions. The statistician sees only the p-value. To make his prediction using x, in the case of equal sample sizes and equal costs of error, he can select for each observed value x, either D or H depending on whether there are more cases or controls in his samples corresponding to x. A naive estimate of the correct prediction rate, the training prediction rate, is obtained by simply using this method on the observed samples. It tends to be overoptimistic. Many sampling properties, such as the significance, the expected training prediction rate, and the median of the I-score, can often be calculated conveniently by simulation.

Our next section uses artificial examples to illustrate how highly significant variables and highly predictive variables might differ.

1.4 Three examples

Although we are concerned with large data, our first few examples use only a few observations to cleanly illustrate the issues. The three examples are followed by comparisons, based on a set of 546 more relevant and related examples, each involving 6 SNP's and many observations as summarized as Example 4. These examples will show how and why significance and predictivity can differ and that the I-score can serve as a

useful sign of predictivity. They also show that the problems we run into in prioritizing significance instead of predictivity in our variable selection stage can grow with the complexity of the data. The comparisons in the last example require many simulations and are meant to demonstrate a complicated data scenario, more akin to a GWAS.

1.4.1 Example 1

For Example 1, there is a single observation X, the distribution of which is normal with mean 0 and standard deviation 1 under a hypothesis H, which can be thought of as health. But there is an alternative hypothesis K, under which X has a normal distribution with mean 3 and standard deviation 3. We wish to use X to determine whether H or K is the correct hypothesis. Our problem can be thought of as predicting or classifying the state of an individual yielding the observation X. It is a standard problem of testing the hypothesis H and we may regard large values of X as favoring K and suggesting rejection of H.

Statistical theory tells us that the optimal test of H consists of rejecting H when the likelihood-ratio is large. For any choice c of what constitutes large enough, we have two error probabilities, $e(c,H)$ and $e(c,K)$ which are the probabilities of making the wrong decision under H and K respectively. Notice that if c increases it becomes harder to reject H and $e(c,H)$ decreases while $e(c,K)$ increases. It is possible to calculate the value of c which minimizes the average of $e(c,H)$ and $e(c,K)$ and to call this minimal value e_X, the minimal average error probability associated with X.

For this problem a plausible, if slightly suboptimal, test is to reject H when X is sufficiently large. For each possible value x of X, there is a probability $a(x)$, under H, that X will be as large as x or larger. Then $a(X)$ is called the p-value when X is observed. Before observing X, we know that X and the p-value are random variables. Under H, $a(X)$ is uniformly distributed between 0 and 1, but under K, $a(X)$ will have a different distribution. If X is very good at discriminating between H and K, $a(X)$ should be very small with large probability under K. We label the median value of $a(X)$ under K as the significance s_X associated with X. In this case $e_X = 0.174$ and $s_X = 0.0014$. Note that e_X is an optimal error rate, but we calculated s_X based on a suboptimal test, that a researcher, not knowing the underlying probability distributions, could reasonably have

decided to use. In that sense the significance was treated unfairly (see Figure 1.1). Note also that predictivity, measured by $1 - e_X$, is associated with a test of the hypothesis H against the alternative K, and is related to the classification problem of deciding which of several (in this case 2) situations applies. Thus prediction, classification, and hypothesis testing are different names for the same problem.

Now suppose that there is another variable Y which is also normally distributed with mean 0 and standard deviation 1 under H, but normally distributed with mean 0 and standard deviation 0.05 under K. Here we calculate $e_Y = 0.06$ and if we insist on using the silly test of rejecting H when Y is large, we obtain $s_Y = 0.5$. (Surprisingly, in this strange case a much better test would consist of rejecting H when the absolute value of Y is too small). Forgetting for the moment how silly the test is, let us consider the dilemma of the scientist who must decide, based on these numbers, whether to observe X or Y. He prefers Y if he decides on the basis of error rate or predictivity and X if the decision is based on significance. We refer to this situation where the preferred choice between X and Y depends on the use of significance or predictivity as a *reversal*.

There are several explanations for the reversal. One is that there was some arbitrariness in our choices of measures of predictability and significance (measures e_X and s_X). Another is that even though the two choices are aimed at measuring the force of inference, they depend on different properties of the probability distributions involved. Another important point is that because we know the probability distributions in this admittedly artificial example, we used that knowledge to calculate the ideal average error probabilities. On the other hand we did not use the optimal test procedure based on the likelihood-ratio for calculating the significance. This may be important because for real data sets we have to use the data to calculate significance levels and predictability. Our estimates may depend as much on the limited capability of our methods of analysis as on the unknown probabilities.

The following two examples, illustrated in Figure 1.2, are more relevant and show the same sort of reversal under considerably more reasonable circumstances. They are also more conventional examples of obtaining significance for the test of a null hypothesis.

1.4.2 Example 2

In Example 2 the outcome variable is case or control status. The explanatory variable X is the reading on one SNP for each of 500 cases and 500 controls, for which the probabilities under cases and controls are listed in the blue table. In this case the minor allele frequency (MAF) is 0.5 and the odds-ratio is close to 1 for each of the three possible observations 0, 1, and 2. For Y, based on the other SNP described in the red table, the MAF is between 0.1 and 0.2 depending on what proportion of the population is healthy. For Y, the odds ratio varies from 4 to 1. In this example we have $e_X = 0.476$ (prediction rate $= 0.524$) and $e_Y = 0.485$ (prediction rate $= 0.515$). We calculate the significance level using the standard chi-square test for the null hypothesis that the two distributions for case and control are the same. This yields $s_X = 0.06$ and $s_Y = 0.0035$. Once more we have a reversal since the smaller average error rate is not accompanied by the smaller median p-value. The Figure also lists the median I-score for both X and Y, which favors X as does the prediction rate.

1.4.3 Example 3

Example 3 is also presented in Figure 1.2. Here the variable X in the blue table consists of the outcome of two SNPs (two-way interaction effect). This outcome can fall in one of the $9 = 3^2$ cells $(0,0),(0,1),...(2,2)$. Again there is a reversal and the median I-score favors X as opposed to Y (in the red table) as does the prediction rate. While the prediction rates are comparable, the median p-values are wildly different. Note in both plots of distributions of the predictive variable sets (predictive VS) and significant variable sets (significant VS) in Examples 1 and 2, there is overlapping between variable sets but large portions of predictive variable sets are not significant and vice versa. In addition, in both examples the I-score follows the preferred prediction rate and not the significance (median p-values).

1.5 Comparing significance tests with the I-score

Before drawing conclusions from the 3 examples, we present a more complex data simulation for Example 4, which consists of a comparison of 546 related more relevant cases with large numbers of subjects.

In these cases we deal with 6 independent but similar SNPs (encapsulating six-way interaction effects), and the observation for a given subject falls into one of $3^6 = 729 =$ cells. The 546 levels of disease are controlled by 26 minor allele frequencies (MAF) and 21 odds ratios (OR). The results in Figure 3 present Truth, Training Prediction Rate and Significance. Truth is the ideal prediction rate given the MAF and OR. The training prediction rate is the overoptimistic rate based on deciding according to the observed number of cases and controls in each of the 729 cells. The significance level depends on the use of the chi-square test. The latter two are medians of measures based on observed data and their calculation requires extensive simulations. The graphs show how poorly these correlate with Truth until the number of subjects becomes very large. While the I-score and its median are also based on the data, Figure 4 shows that it is very well correlated with the Truth for modest sample sizes; at large sample sizes I is still better correlated with Truth than are the Training Prediction Rate and Chi-square test.

1.6 Applying the I-score to real breast cancer data

To reinforce the previous section we turn to a brief examination of real disease data. As noted before, our research team has made heavy use of the I measure in a variable selection method called Partition Retention. This method, applied to real disease data, has not only been quite successful in finding possibly interacting influential variable sets but has also resulted in variable sets that are very predictive and do not necessarily show up as significant through traditional significance testing (Chernoff, Lo, and Zheng 2009; Wang et al. 2012; Lo et al. 2008). Here "predictive" refers to both high in I-score as well as having high correct prediction rates as determined by k-fold cross-validation. We present examples of some discovered variable sets found to be highly predictive for a real data set on breast cancer (Veer et al. 2002) that are not highly

significant. When utilizing these newly found variable sets, the team was able to reduce the error rate on prediction from the literature standard of 30% to 8%. These results are found from the analysis and data used in (Wang et al. 2012).

In Table 1.9 we investigate the top 5-variable module (subset of interacting variables) in the breast cancer data found to be predictive through both top I-score and performance in prediction in cross-validation and an independent testing set in (Wang et al. 2012). To find how significant these variables are, we calculate the individual, marginal association of each variable in the marginal p-value. When testing 1,000 variables having no effect, it is likely that some will have p-values of around 0.001. Here, we have 4,918 variables and therefore desire a p-value of $7 * 10^{-5}$, the family-wise threshold, to announce significance. None of these variables show up as statistically significant. Measuring the joint influence of all 5 variables does not have a p-value that is significant either.

1.7 Comments and Conclusion

In our exposition of the differences between highly predictive versus highly significant variable sets, we use artificial examples. We need to know the true relevant underlying probability distributions in order to treat the problem as one of testing a simple hypothesis against a known alternative for which statistical theory can calculate optimal tests and predictive rates. Our four simulated examples can demonstrate with clarity the reversals we see in choosing significant versus predictive variable sets. Real examples are more difficult because the researcher must rely on a limited number of individuals to infer the relevant distributions and the number of possible variables is huge. However, in order to demonstrate the potential usefulness of our proposed measure, we additionally provided the highly promising results of applying the I-score to the real and well-known van't Veer breast cancer data set.

One may wonder whether the shortcoming of using significance is due to the custom of using marginal significance and not taking into account the possible interaction effects of groups of variables. In our examples the problem of reversals seems to increase when using significance-based measures on routine tests when dealing with

groups of interacting variables. In Example 4, six-way interactions are considered and traditional significance approaches do not capture predictive variable sets. However, using the PR approach based on the measure I for the variable selection stage does well for prediction. Finally, even when we can capture joint effects that are highly predictive, as in the case of the captured variable sets in the van't Veer example, *these groups of variables were not significant*. Seeking highly predictive groups of variables through significance alone would not have retrieved these variable sets.

If that is the case how did we manage to get good results in the breast cancer problem? We used the PR approach, relying heavily on the I-score for the variable selection aspect. For reasons we only partly understand the I-score seems to correlate well with predictivity. Having selected the relatively small number of candidate "influential" variables, an intensive use of a variety of known techniques in classification were applied. These were more sophisticated than simple linear regressions.

The issue of obtaining high predictivity from large data demands study. We encourage exploration away from significance-based methodologies and towards prediction oriented ones. We propose the I-score and the PR method of variable selection as candidate tools for the latter.

1.8 Acknowledgements

Chapter 1, in full, is a reprint of the material as it appears in "Why significant variables aren't automatically good predictors" 2015, Lo, Adeline; Chernoff, Herman; Lo, Shaw-hwa; Zheng, Tian., Proceedings of the National Academy of Sciences. This paper was coauthored amongst Adeline Lo, Herman Chernoff, Shaw-Hwa Lo, and Tian Zheng.

1.9 Tables and Figures

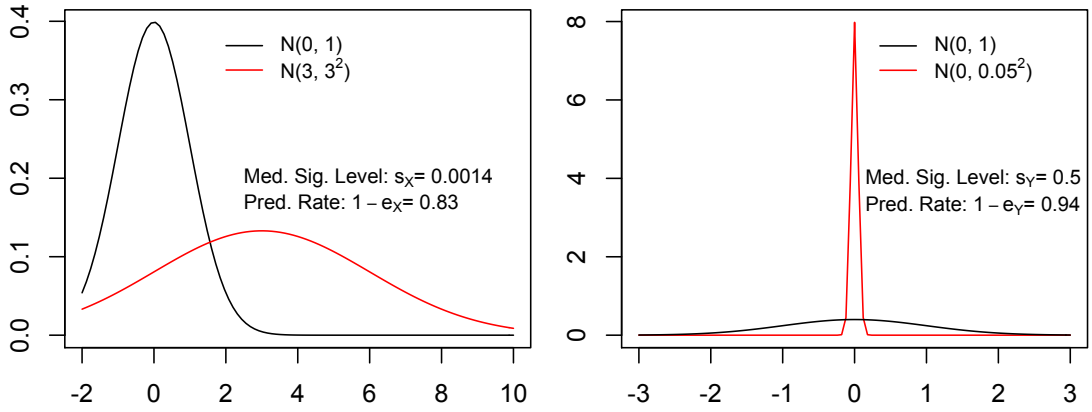
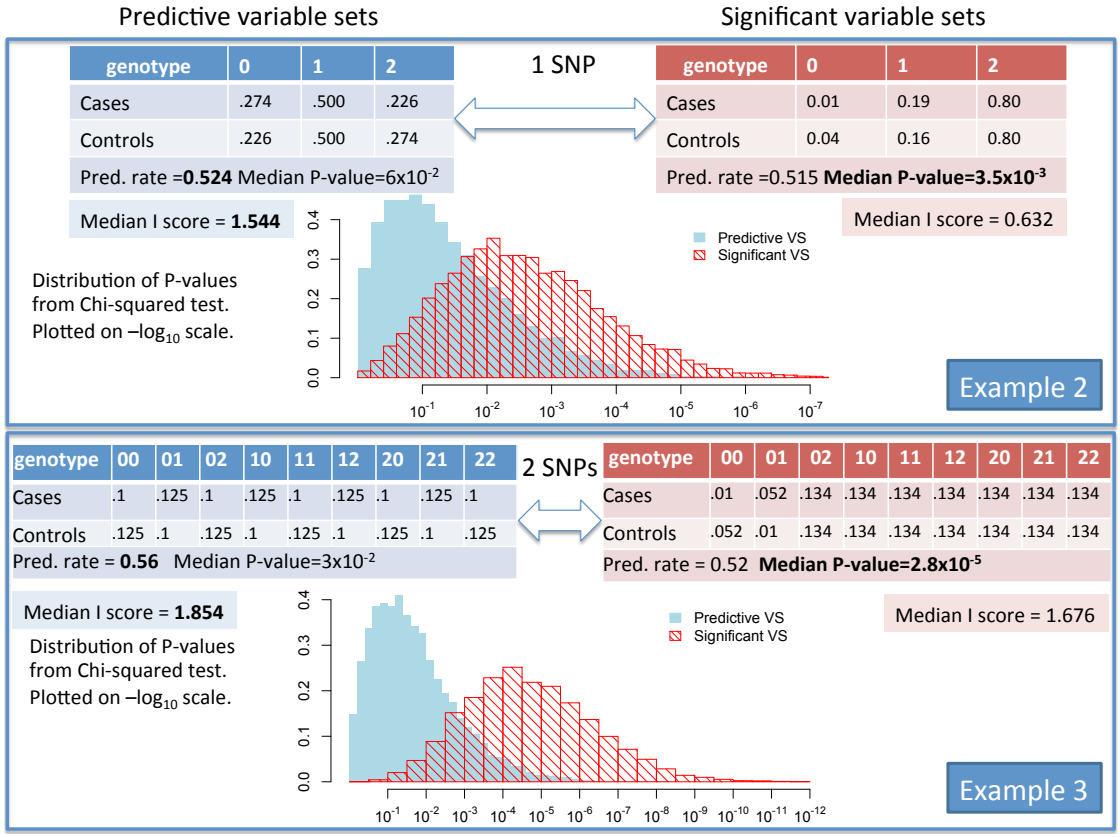


Figure 1.1: **Simple Example of Reversals.** The variable on the left (X) would be favored under a significance criterion but the variable on the right (Y) is favored under a predictivity criterion.



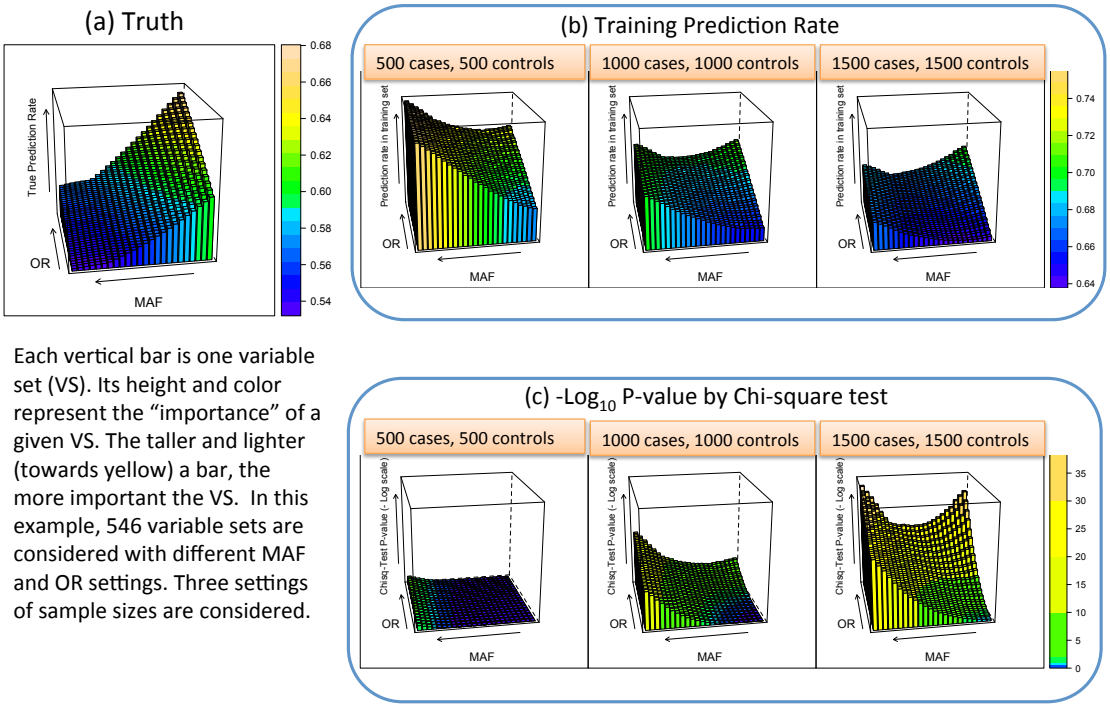
genotype	0	1	2
Cases	.274	.500	.226
Controls	.226	.500	.274
Pred. rate =**0.524** Median P-value=6x10^{-2}			

genotype	0	1	2
Cases	0.01	0.19	0.80
Controls	0.04	0.16	0.80
Pred. rate =0.515 **Median P-value=3.5x10^{-3}**			

genotype	00	01	02	10	11	12	20	21	22
Cases	.1	.125	.1	.125	.1	.125	.1	.125	.1
Controls	.125	.1	.125	.1	.125	.1	.125	.1	.125
Pred. rate = **0.56** Median P-value=3x10^{-2}									

genotype	00	01	02	10	11	12	20	21	22
Cases	.01	.052	.134	.134	.134	.134	.134	.134	.134
Controls	.052	.01	.134	.134	.134	.134	.134	.134	.134
Pred. rate = 0.52 **Median P-value=2.8x10^{-5}**									

Figure 1.2: **Reversals of Predictive and Significant variable sets in SNP examples**. Example 2 has one explanatory variable (1 SNP) for which the probabilities under cases and controls are listed in the tables. Example 3 has two explanatory variables (2 SNPs) for which the probabilities under cases and controls are listed in the tables. Left hand side tables (in blue) are for more predictive variable sets, while right hand side tables (in red) are for more significant variable sets. The prediction rate (proportion of correct predictions) of each variable set (of size 1 or 2) can be directly computed using the genotype frequencies specified. Using sample sizes of 500 cases and 500 controls, we simulate $B = 1000$ random case-control data sets by simulating genotype counts among cases and controls using the genotype frequencies specified. I score and the Chi-square test statistic were computed for each simulated data set. Simulation details can be found in the Supplementary materials.



Each vertical bar is one variable set (VS). Its height and color represent the "importance" of a given VS. The taller and lighter (towards yellow) a bar, the more important the VS. In this example, 546 variable sets are considered with different MAF and OR settings. Three settings of sample sizes are considered.

Figure 1.3: **Disconnect between true prediction power of a variable set and its empirical training set prediction rate and test-based significance**. 546 variable sets of 6 SNPs with varying levels of disease information are used (both minor allele frequencies (MAF) and odds ratios (OR)). This results in a partition of 729 cells, each corresponding to a genotype combination on the six SNPs represented by this variable set. Three levels of sample size are considered, 500 cases and 500 controls, 1000 cases and 1000 controls, and 1500 cases and 1500 controls. For each variable set, the theoretical Bayes rate is computed based on the population frequencies and odds ratios. 2000 independent simulations under each variable sets — given a sample size specification — were used to evaluate the average training prediction error, p-value from the chi-square test, and the I score prediction rate. Simulation details can be found in the Supplement.

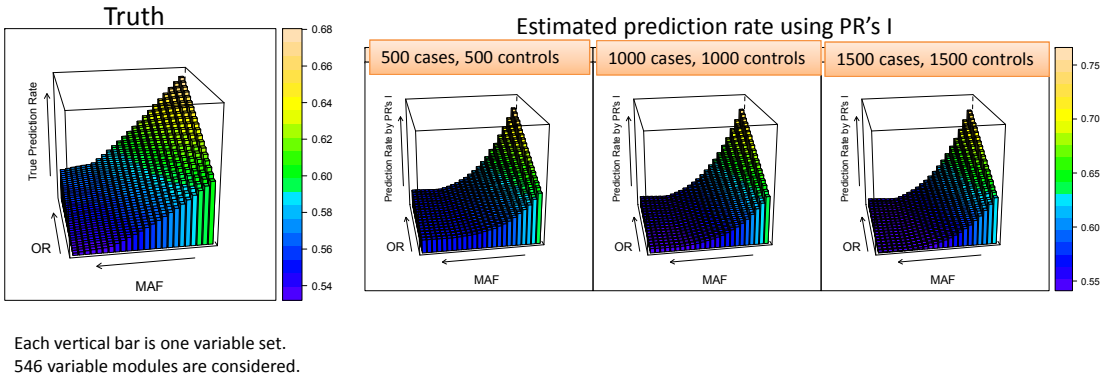


Figure 1.4: **The proposed estimated prediction rate based on I-scores correlates well with the Truth.**

Table 1.1: **Real breast cancer example: top returned predictive variable set from van't Veer data.**

	Systematic name	Gene name	Marginal p-value
1	Contig45347_RC	KIAA1683	0.008
2	NM_005145	GNG7	0.54
3	Z34893	ICAP-1A	0.15
4	NM_006121	KRT1	0.9
5	NM_004701	CCNB2	0.003

Joint *I*-score: 2.89;
Joint p-value: 0.005;
Family-wise threshold: $6.98\mathbf{x}10^{-5}$

Chapter 2

Making good prediction: a theoretical framework

2.1 Abstract

We propose approaching prediction from a framework grounded in the theoretical correct prediction rate of a variable set and define a novel measure of predictivity. It enables us to find variable sets that are highly predictive. While intuitively obvious, not nearly enough attention has been paid to the creation of such a framework. Motivated by the needs of current genome-wide association studies (GWAS), we provide such a discussion. We first describe the correct prediction rate for a variable set. We then consider and ultimately reject an estimator that approximates the correct prediction rate of a variable set using sample data due to the estimator's inability to distinguish between noisy and predictive variables which directly leads to an inability to estimate without inflated bias. In response, we offer an alternative parameter that describes the predictivity of a variable set — a lower bound to the correct prediction rate. We demonstrate that the Partition Retention method's I-score can be used to compute a measure that asymptotically approaches this lower bound. The I-score can effectively differentiate between noisy and predictive variables as well, making it helpful in variable selection. We offer simulations and an application of the I-score on real data to demonstrate the statistic's predictive performance on sample data. These show that the I-score can capture highly

predictive variable sets, estimates a lower bound for the theoretical correct prediction rate and correlates well with the *out of sample correct rate*. We conjecture that using the Partition Retention and I-score can aid in finding variable sets with promising prediction rates, however, further research in the avenue of sample-based measures of predictivity is much desired.

2.2 Introduction

Prediction is a highly important goal for many scientists and has become increasingly difficult as the quantity and complexity of available data has grown. Complex and high dimensional data particularly demand attention. However, prediction does not have a clear theoretical framework that allows for characterizing a variable's predictivity directly. Rather, variable selection (or VS as it is referred to throughout this paper) for variable sets cthat have high predictivity is currently conducted in two common ways. The first is VS through identification of variables correlated with the outcome, measured through tests of statistical significance — such as the chi-square test. The second is through VS of variables that appear to do well in an independent set of test data, as measured through testing sample error rates. The first approach is still very much in use for predicting health outcomes (see Gransbo et al. 2013 among others) but its prediction performance has been disappointing. A recent New England Journal of Medicine article illustrates one example of such an approach, whereby researchers constructed a model based on five statistically significant genetic variants from genome-wide association studies (GWAS) results on prostate cancer; they report that the variants do not increase predictive power Zheng et al. (2008). Likewise, Gränsbo et al. show that chromosome 9p21, while significantly associated with cardiovascular disease, does not improve risk prediction Gransbo et al. (2013). Using variants discovered to have statistically significant associations with the disease outcome to build predictive models does not seem to imply an automatic increase in the ability to predict.

We point out in Lo et al. (2015) that this first approach suffers from the problem that significant variables are not necessarily predictive and vice versa, so targeting significant variables might miss the goal of VS for higher predictivity. Indeed, this prob-

lem is prevalent in simple as well as complex data. The second way for VS sets aside testing (or validation) data to determine how well selected predictors might do on "new data". However, as is in the case of GWAS data, researchers often lack large enough sample sizes for this approach to be efficient. Reuse of training data in the form of cross-validation is often adopted in practice.

An alternative, and perhaps logical, approach to VS should start with understanding the theory behind prediction, such as defining theoretical prediction rates of variables as a parameter of interest. It would be productive then to create measures designed to directly measure such a parameter, rather than relying on the estimated prediction rate by cross-validation. We hope that such a logically driven approach — designing measures that directly estimate a variable set's true ability to predict — may prove to be both fruitful and efficient in the use of sample data for guiding VS.

In this paper, we propose such a prediction-based framework. Grounded in statistical theory, we highlight a new avenue of research towards creating sensible measures that target highly predictive variable sets.[1] We emphasize genetic data, though we will show that the methods proposed can be easily tailored to other high-dimensional data in the natural and social sciences. Our paper proceeds as follows. The first section considers the related literature of variable (or feature) selection in machine learning. While the literature is rich in techniques designed to evaluate prediction, most techniques are through independent test data or cross-validation. These require setting aside some part of valuable sample data (either from the training data or testing data) for the purpose of validation. We emphasize the need for a theoretically motivated measure of predictivity. The second section introduces the set-up for considering maximally predictive variable sets in a non-sample constrained, theoretical world. We first introduce variable set's correct prediction rate, θ_c, and consider a measure that would attempt to directly estimate this parameter using sample data. This measure is rejected for its lack of practical use. We propose an alternative measure, the I-score, and demonstrate that the I of a given variable set asymptotically approaches a parameter, θ_I, which is closely related to the correct prediction rate of the same variable set. The third section demonstrates the effectiveness of the I-score as a measure of predictivity in simulations and real data

[1] We refer mostly to variable sets here, which include, of course, individual variables as variable sets of size one.

applications. Finally, we offer some concluding remarks.

2.3 Variable selection literature

Variable Selection (VS) or Feature Selection refers to the practice of selecting a subset of an original group of variables that is later used to construct a model. Often VS is used on data of large dimensionality with modest sample sizes Saeys et al. (2007). In the context of high dimensional data, such as GWAS, with potentially a large number of redundant or irrelevant variables, this dimensionality reduction can be a crucial step. Unlike projection or compression based approaches (such as principal component analysis or usage of information theory), VS methods do not change the variables themselves.

The types of approaches and tools developed for feature selection are both diverse and varying in degrees of complexity. However, there is general agreement that three broad categories of feature selection methods exist: *filter*, *wrapper* and *embedded* methods. *Filter* approaches tend to select variables through ranking them by various measures (correlation coefficients, entropy, information gains, chi-square, etc.). *Wrapper* methods use "black box" learning machines to ascertain the predictivity of groups of variables; since wrapper methods often involve retraining prediction models for different variable sets considered, they can be computationally intensive. *Embedded* techniques search for optimal sets of variables via a built-in classifier construction. A popular example of an embedded approach is the least absolute shrinkage and selection operator (LASSO) method for constructing a linear model, which penalizes the regression coefficients, shrinking many to zero. Often cross-validation is used to evaluate the prediction rates.

We are unaware of a measure that *directly* attempts to evaluate a variable set's theoretical level of predictivity, however. For a more comprehensive survey of the feature selection literature see, among others Guyon et al. (2003), Saeys et al. (2007), Hua et al. (2009), and Bolon-Canedo et al. (2013).

Although a spectrum of variable selection approaches exists, many scientists have taken the approach of tackling prediction through the usage of important and hard-

to-discover influential variables found to be statistically significant in previous studies. When these efforts are in the context of high dimensional data and alongside work investigating variables known to be influential, it might seem reasonable to hope that variables found to be significant can prove useful for predictive purposes as well. This approach is in some ways most similar to a univariate filter method, as it is independent of the classifier and has no cross-validation or prediction step for VS. We show in our related work Lo et al. (2015) how and why the popular filter approach of variable selection through statistical significance does not serve the purpose of prediction well. For an intuitive illustration of the relationship between predictive and significant sets of variables, see Figure 2.1. Under the context of a significance-test based search for variable sets, the set of variables found to be significant expands as the sample-size grows (see widening orange dotted ovals). However, the set of predictive variables (blue circle) are not susceptible to sample-size changes in the same way — as predictivity is a *population parameter* — and overlaps with, but not is not perfectly aligned with, significant sets. It is easy to see that in this scenario, targeting significant sets may miss the goal of prediction entirely. Instead, we suggest that emphasis must be placed on designing measures that directly evaluate variable sets' predictivity.

Many methods also use out of sample testing error rates or cross-validation (CV) to ascertain whether prediction is done well. This approach was not designed to specifically find a theoretic correct prediction rate for a given variable set; rather, this is simply a performance evaluation of future predictions from a pattern recognition technique on selected variable sets (trained on training data).Sometimes the variable sets in the training data are selected through statistics such as the adjusted R squared, Akaike information criterion (AIC) or Bayesian information criterion (BIC). When $p \gg n$ (or even in instances where $p > n$), a standard in big data, however, these statistics can fail to be useful.[2] Using out of sampling testing and/or cross-validation techniques additionally requires setting aside valuable sample data to make sure the variable sets selected under the training set are indeed highly predictive and are not just overfitting the data. It be-

[2]"Unfortunately, the C_p, AIC, and BIC approaches are not appropriate in the high-dimensional setting, because estimating $\hat{\sigma}^2$ [variance] is problematic. (...) Similarly, problems arise in the application of the adjusted R^2 in the high-dimensional setting, since one can easily obtain a model with an adjusted R^2 value of 1." James et al. 2013.

comes important then that we have a good screening mechanism when conducting VS for removing noisy variables (and thus finding influential ones), even with constrained amounts of sample data. We show in our simulations how poorly we can do in VS for prediction through training set compared to out of sample testing prediction rates (with "infinite" future testing data – a mostly unattainable, but ideal, scenario). An ideal measure for predictivity would guide our VS stage through screening out noisy variables and should correlate well with the out of sample correct prediction rate. We will demonstrate a potential candidate measure, the I-score, for evaluating the predictivity of a given variable set.

2.4 Toy example

To highlight some of our key issues, we consider a small artificial example. Suppose that an observed variable Y is defined as:

$$Y = \begin{cases} X_1 + X_2 \ \text{(modulo 2)} & \text{with probability } 1/2, \\ X_2 + X_3 + X_4 \ \text{(modulo 2)} & \text{with probability } 1/2, \end{cases} \tag{2.1}$$

where X_1, X_2, X_3 and X_4 are four of 50 observed and potentially influential variables $\{X_i; 1 \leq i \leq 50\}$. Each X_i can take on the values 0 and 1. A collection of several discrete variables S may be regarded as a discrete variable that takes on a finite number of values. Each value defined by S constitutes a *cell*. The collection of all cells forms a *partition*, Π_S, based on the discrete variables in S. We also assume that the X_i were selected independently to be 1 with probability 0.5, again the simplest case without affecting the general results. Clearly, none of the individual X_i have a marginal effect on Y.

Scenario I: A statistician knows the model and wishes to compute which variables or variable sets are predictive of Y, and how predictive, when $\mathbf{X} = (X_1, X_2, ..., X_{50})$ is a given. Because Y depends only on the first four X variables, it is obvious there are two clusters of variable sets $S_1 = \{X_1, X_2\}$ and $S_2 = \{X_2, X_3, X_4\}$ that are potentially useful in his prediction. In this paper, we treat the highest correct prediction rate possible for a given variable set as an important parameter and call this *predictivity* (θ_c). Using the knowledge of the model, we can compute the predictivity for S_1 as $\theta_c(S_1) = 0.75$.

The predictivity for S_2 is also $\theta_c(S_1) = 0.75$. Incidentally, the predictivity of the union of S_1 and S_2, $\theta_c(S_1 \cup S_2)$, is also 0.75.

The statistician realizes that using variable sets S_1 and S_2, he can predict Y correctly 75% of the time. This is indeed the case because, for instance, upon observing $\mathbf{X} = (X_1, ..., X_{50})$ the statistician predicts:

$$\hat{Y} = X_1 + X_2 \text{ (modulo 2)}.$$

It is easy to verify that the strategy of predicting with S_1 returns a 75% prediction accuracy in expectation. This is also the highest percent accuracy S_1 can theoretically achieve. We discuss this in depth shortly. This result extends to S_2 as well.

Scenario II: In practice, the statistician rarely has knowledge of the model and instead observes only the data. In this paper, we suggest that the statistician use the Partition Retention (PR) approach and its corresponding I-score to identify the influential variable sets. Suppose he observes 400 observations and wishes to identify variable sets with high predictivity and to infer their abilities to predict after being identified. Using the PR approach he can use the I-score to screen for variable sets with high potential predictivity. In this example, S_1 and S_2 are consistently returned with the highest I-scores of 23.71 and 12.79 in simulations. Using the inequality in Equation (2.10), which we derive in the following section, the lower bounds for the predictivity of $\theta_c(S_1)$ and $\theta_c(S_2)$ are calculated to be 67% and 62% respectively. Equation (2.10) does not require knowledge of the true model as defined in Equation (2.1).

2.5 Theoretical prediction rates

If the goal is to find highly predictive variable sets for a given outcome, a natural approach is to find methods that can evaluate the predictivity of different variable sets and compare them against one another in order to find variable sets with higher predictivity. We show in this paper that the Partition Retention (PR) method's I-score searches for variable sets that are highly predictive in high dimensional and sample-size constrained data like GWAS. It is thus a prime candidate for a method that tackles the big data prediction goal. This paper contributes to the prediction literature by introduc-

ing the *theoretical prediction rate* as a parameter to be directly estimated. We show that the I-score is a sample-based statistic that can be used to construct an asymptotically consistent lower bound for the theoretical prediction rate.

The set-up

To target variable sets with high predictivity, we must design measures that accurately reflect prediction rates for different variable sets. Consider GWAS data of the usual type, with cases and controls.[3] Assume that there are n_d cases and n_u controls.[4]

Using the traditional Bayesian binary classification setting, we ideally have a prior probability, $\pi(w = d)$, that the state of the next individual, w, is a disease case, d, and $\pi(w = u) = 1 - \pi(w = d)$ that the next individual is a control, u. In the following we shall assume that both d and u are equally likely and that the cost of an incorrect classification is the same for both possibilities.[5]

If we are lucky, when assessing the predictivity of a variable set, we know the joint distribution of the disease status and the feature value $\mathbf{X} = \mathbf{x}$, denoted as $P(\mathbf{x}, w)$, either through knowledge or through estimation via a reliable source. The joint distribution can be expressed as: $P(w, \mathbf{x}) = \pi(w|\mathbf{x}) \cdot P(\mathbf{x}) = P(\mathbf{x}|w) \cdot P(w)$, where $\pi(w|\mathbf{x})$ is the posterior distribution and $\pi(w)$ is the prior. It is easy to see that the best classification rule can be derived by Bayes' decision rule for minimizing the posterior probability of error: d if $\pi(d|\mathbf{x}) > \pi(u|\mathbf{x})$, otherwise u. Here the variable set $\mathbf{X} = (X_1, X_2, ..., X_m)$, with each X_i taking one of the values in $\{0, 1, 2\}$, corresponding to the 3 possible genotypes for each single-nucleotide polymorphism (SNP). In this way, $\mathbf{X}$ forms a partition, denoted by Π_X, with $3^m = m_1$ elements: $\Pi_{\mathbf{X}} = \{\mathbf{X} = \mathbf{x}_j, j = 1, \ldots, m_1 : \mathbf{x}_j = (x_{j1}, x_{j2}, \ldots, x_{jm}), x_{jk} \in \{0, 1, 2\}, 1 \leq k \leq m\}$.

A problem that emerges when dealing with case-control data like GWAS is that prior information on observing the next person as a disease case is unavailable or unknown and cannot be easily estimated from empirical data. Priors are defined by cir-

[3] The pooled sample of cases and controls serves as the training set.

[4] We consider prediction of a binary variable (or classification) here. Our framework can be extended for prediction of a continuous variable. In this case, the I-score still plays a role in measuring predictivity though we leave this for beyond the scope of this paper.

[5] We generalize to different cost functions and priors for d and u in the SI.

cumstances and contexts within which the case-control data are sampled — each dataset requires its own unique and unknown prior at that point in time.

To surmount this, we focus first on the classification problem based on the class-conditional probability: $P(\mathbf{x}|w=d)$ versus $P(\mathbf{x}|w=u)$. This is equivalent, implicitly, to the full Bayes' decision rule when an equal prior is adopted, $\pi(d)=\pi(u)=1/2$. General cases for arbitrary priors and unbalanced loss functions are discussed later.

Let $\mathbf{x}$ be a discrete random vector defined on the space $\Pi_{\mathbf{X}}$, with density denoted by $p_{\mathbf{X}}(\mathbf{x})$. Suppose n_d cases and n_u controls are independently selected from two discrete probabilities: $p_{\mathbf{X}_d}(\mathbf{x})$ and $p_{\mathbf{X}_u}(\mathbf{x})$, equivalent to $P(\mathbf{x}|w=d)$ and $P(\mathbf{x}|w=u)$ respectively. Note that $\{\mathbf{X}_d,\mathbf{X}_u\}$ always arrive as a pair when the m SNPs are fixed (fixing $\Pi_{\mathbf{x}}$); that is, $\mathbf{X}_d$ and $\mathbf{X}_u$ are defined on the common partition space, $\Pi_{\mathbf{X}}$. If the newly arrived subject has a 50% chance to be either case or control, the expected error of adopting the above Bayes' decision rule (under a 0/1 loss) is:

$$\theta_e[p_{\mathbf{X}_d},\, p_{\mathbf{X}_u}] = \frac{1}{2}\sum_{\mathbf{x}\in\Pi_{\mathbf{x}}} \min\{p_{\mathbf{X}_d}(\mathbf{x}),\, p_{\mathbf{X}_u}(\mathbf{x})\}.$$

The correct prediction rate θ_c on $\mathbf{X}$ becomes:

$$\theta_c(\mathbf{X}) = \theta_c[p_{\mathbf{X}_d},\, p_{\mathbf{X}_u}] = 1-\theta_e[p_{\mathbf{X}_d},\, p_{\mathbf{X}_u}] = \frac{1}{2}\sum_{\mathbf{x}\in\Pi_{\mathbf{x}}} \max\{p_{\mathbf{X}_d}(\mathbf{x}),\, p_{\mathbf{X}_u}(\mathbf{x})\}$$

where θ_e is the error rate. For simplicity of presentation, we can represent the above as:

$$\theta_c = \frac{1}{2}\sum_{j\in\Pi_{\mathbf{x}}} \max\{P(j|d), P(j|u)\} \tag{2.2}$$

where j is short for $\mathbf{x}_j$, a cell in the partition $\Pi_{\mathbf{X}}$ formed by the variables $\mathbf{X}$.

Finding variable sets with better predictivity

It is clear that one can achieve a better prediction rate by seeking the probability pair $\{p_{\mathbf{X}_d},\, p_{\mathbf{X}_u}\}$ that minimizes the expected predictive error $\theta_e[p_{\mathbf{X}_d},\, p_{\mathbf{X}_u}]$, or equivalently, maximizes the predictive rate $\theta_c[p_{\mathbf{X}_d},\, p_{\mathbf{X}_u}]$. Equivalently:

$$\frac{1}{2}\{\theta_c[p_{\mathbf{X}_d}, p_{\mathbf{X}_u}] - \theta_e[p_{\mathbf{X}_d}, p_{\mathbf{X}_u}]\} = \theta_c[p_{\mathbf{X}_d}, p_{\mathbf{X}_u}] - \frac{1}{2} = \frac{1}{4}\sum_{j\in\Pi_{\mathbf{X}}} |P(j|d) - P(j|u)|. \quad (2.3)$$

Therefore,

$$\theta_c[p_{\mathbf{X}_d}, p_{\mathbf{X}_u}] = \frac{1}{2} + \frac{1}{4}\sum_{j\in\Pi_{\mathbf{X}}} |P(j|d) - P(j|u)|. \quad (2.4)$$

This suggests that we can achieve better prediction rates by choosing variable sets corresponding to the probability pairs that lead to very large values of $\sum_{j\in\Pi_{\mathbf{X}}}|P(j|d) - P(j|u)|$. We can consider the goal of prediction as simply trying to find variable sets $\mathbf{X}$ that make θ_c as large as possible. In this theoretical setting, it is easy to show that θ_c increases or stays the same when another variable is added to the current variable set. This means adding many noisy variables leads to maintaining the same θ_c. Therefore, when sample size is no constraint, we are never hurt in our search for highly predictive variables by simply adding explanatory variables to our current set. However, in the realistic world of sample size constraints, a direct search for a variable set with a larger *sample estimate* of θ_c will fail; we give a heuristic explanation as to why in the following section. We refer to this direct search of θ_c with sample data as the *sample analog* throughout.

Problems with the sample analog

First, as noted earlier, θ_c is always nondecreasing when more variables are added to the current variable set. Therefore, in principle, a group of predictive variables is difficult to identify as we would not be able to differentiate between adjoining a noisy or a predictive variable to a given variable set. Second, in reality, the actual θ_c are unknown and must be estimated. We may naturally turn to the sample estimate of its true theoretical values. Again, due to the similar problem with θ_c, the estimated values of θ_c (where the cell probabilities are replaced by the observed proportions) is nondecreasing with the addition of more variables to a given variable set under evaluation. To make this point clearly, we assume all variables X_j can take only two values, 0 or 1.

Suppose $\mathbf{X}_m = \{X_1, ..., X_m\}$ and $\mathbf{X}_{m+1} = \{X_1, ..., X_m, X_{m+1}\}$. The partition from $\mathbf{X}_m$ is $\Pi_{\mathbf{X}_m} = \{A_1, ..., A_{m_1}\}$, while the partition formed by $\mathbf{X}_{m+1}$ is $\Pi_{\mathbf{X}_{m+1}} = \{A_1 \cap B, ..., A_{m_1} \cap B, A_1 \cap B^c, ..., A_{m_1} \cap B^c\} = \{\Pi_{\mathbf{X}_m} \cap B, \Pi_{\mathbf{X}_m} \cap B^c\}$ where $B = \{\mathbf{X}_{m+1} = 1\}$. Let $\Pi^1_{\mathbf{X}_m} = \Pi_{\mathbf{X}_m} \cap \{X_{m+1} = 1\}$ and $\Pi^0_{\mathbf{X}_m} = \Pi_{\mathbf{X}_m} \cap \{\mathbf{X}_{m+1} = 0\}$, where $\Pi^1_{\mathbf{X}_m}$ and $\Pi^0_{\mathbf{X}_m}$ form two subpartitions of $\Pi_{\mathbf{X}_{m+1}}$, i.e. , $\Pi_{\mathbf{X}_{m+1}} = \Pi^0_{\mathbf{X}_m} \cup \Pi^1_{\mathbf{X}_m}$. Then

$$|\hat{p}_{\Pi_{\mathbf{X}_m}}(d) - \hat{p}_{\Pi_{\mathbf{X}_m}}(u)| \leq |\hat{p}_{\Pi^0_{\mathbf{X}_m}}(d) - \hat{p}_{\Pi^0_{\mathbf{X}_m}}(u)| + |\hat{p}_{\Pi^1_{\mathbf{X}_m}}(d) - \hat{p}_{\Pi^1_{\mathbf{X}_m}}(u)|,$$

where $\hat{p}(\cdot)$ is the sample estimator. It is thus easy to see that the sample analog inherently favors an increase in number of partition cells (or in other words, favoring more variables).

As the partition becomes increasingly finer, there reaches a point where there is at maximum a single observation within each partition cell and 100% correct sample prediction rate is attained. This is true regardless of the true prediction rate. As a result, the final estimated prediction rate is equivalent to 100%, rendering it useless as a method for searching for highly predictive variable sets and screening out noisy variable sets. This is a direct result of a sparsity problem that does not occur in our theoretical world but certainly plagues the sample-size constrained real world. In the latter setting, the sample analog of θ_c favors ever-increasing the variable set with both truly influential as well as noisy and un-influential variables. We continue accepting both types of variables until our partition experiences complete sparsity. What is needed is a sample-based measure that can discern adding noisy versus influential variables and identify the variable set(s) with large prediction rates under a given sample size.

Alternative measure: *I*-score

We consider this obstacle and suggest the *I*-score of the PR method (first presented in Chernoff, Lo, and Zheng 2009) as a possible statistic in lieu of the sample analog of the correct prediction rate. We suggest an alternative measure, a lower bound to θ_c, for which we can use the I statistic to estimate in sample data. We can show that

the I-score converges asymptotically to a constant multiple of:

$$\theta_I(\Pi_{\mathbf{X}}) = \sum_{j \in \Pi_{\mathbf{X}}} (P(j|d) - P(j|u))^2.$$

To understand how the above constant relates the I-score to θ_c defined in Equation (2.4), we first examine the following Lemma 1, which is derived in the SI with the other proofs.

Lemma 1. *For K real values $\{z_j; 1 \leq i \leq K\}$, $\sum_{j=1}^{K} z_j = a$ and $\sum_{j=1}^{K} |z_j| = b$, we have*

$$\sum_{j=1}^{K} z_j^2 \leq \frac{a^2 + b^2}{2}. \tag{2.5}$$

In the case of $z_j = (P(j|d) - P(j|u)$ for $j \in \Pi_{\mathbf{X}}$, we have $a = 0$. It then follows that

$$\sqrt{2 \sum_{i=1}^{k} (P(j|d) - P(j|u))^2} \leq \sum_{i=1}^{k} |P(j|d) - P(j|u)|.$$

This suggests a strategy that seeks variable sets with a larger value of θ_I can have the parallel effect of encouraging selection of variable sets with larger values of θ_c, yielding better predictors. In addition, the I-score, the measure for which we suggest for this alternative setup, can also discern noisy variables from influential ones (for theoretical work supporting this important characteristic of the I-score, see Chernoff, Lo, and Zheng 2009).

2.6 The I-score

We can show that, as sample sizes increase, identifying a cluster of variables with a larger influential score, or I-score, can simultaneously yield a cluster with high predictivity.

The Influential score (I-score) is a statistic derived from the Partition Retention (PR) method. Several forms and variations were associated with the PR method before it was finally coined with this name in 2009 by Chernoff et al. We introduce the PR method and the I-score briefly here.[6]

[6]We use GWAS data to motivate our presentation of the I-score and PR method, but the approach

Consider a set of n observations of a disease phenotype Y (dichotomous or continuous) and a large number S of SNPs, $X_1, X_2, ..., X_S$. Randomly select a small group, m, of the SNPs. Following the same notation as in previous sections, we call this small group $\mathbf{X} = \{X_k, k = 1, ..., m\}$. Recall that X_k takes values 0, 1, and 2 (corresponding to 3 genotypes for a SNP locus: AA, A/B and B/B). There are then $m_1 = 3^m$ possible values for $\mathbf{X}$'s. The n observations are partitioned into m_1 cells according to the values of the m SNPs (X_k's in $\mathbf{X}$), with n_j observations in the jth cell. We refer to this partition as $\Pi_{\mathbf{X}}$. The proposed I-score (denoted by $I_{\Pi_{\mathbf{X}}}$) is designed to place greater weight on cells that hold more observations:

$$I_{\Pi_{\mathbf{X}}} = \sum_{j=1}^{m_1} \frac{n_j}{n} \cdot \frac{(\bar{Y}_j - \bar{Y})^2}{s_n^2/n_j} = \frac{\sum_{j=1}^{m_1} n_j^2 (\bar{Y}_j - \bar{Y})^2}{\sum_{i=1}^{n} (Y_i - \bar{Y})^2} \tag{2.6}$$

where $s_n^2 = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \bar{Y})^2$.

We present Theorem 1 and Corollary 1:

Theorem 1. *Under the assumptions that $\frac{n_d}{n} \to \lambda$, a value strictly between 0 and 1, and an equal prior $\pi(d) = \pi(u) = 1/2$, then:*

$$\lim_{n\to\infty} \frac{s_n^2 I_{\Pi_{\mathbf{X}}}}{n} \stackrel{\mathcal{P}}{=} \lambda^2(1-\lambda)^2 \sum_{j\in\Pi_{\mathbf{X}}} [P(j|d) - P(j|u)]^2 \tag{2.7}$$

where $\stackrel{\mathcal{P}}{=}$ indicates that the left hand side converges in probability to the right hand side.

We show in the following $\theta_I = \theta_I(\Pi_{\mathbf{X}}) = \sum_{j\in\Pi_{\mathbf{X}}}(P(j|d) - P(j|u))^2$ is a parameter relevant to $\theta_c(\mathbf{X})$. Together with Lemma 1, we can use the I-score to derive a useful asymptotic lower bound to the correct prediction rate of a variable set $\mathbf{X}$, $\theta_c(\mathbf{X})$, as presented in the following Corollary 1.

Corollary 1. *Under the assumptions in Theorem 1, the following is an asymptotic lower bound for the correct predictive rate:*

$$\theta_c(\mathbf{X}) \stackrel{\mathcal{P}}{\geq} \frac{1}{2} + \frac{1}{4}\sqrt{2 \lim_{n\to\infty} \frac{I_{\Pi_{\mathbf{X}}}}{n\lambda(1-\lambda)}}. \tag{2.8}$$

applies to any discrete data.

Proof. From Equation (2.4),

$$\begin{aligned}
\theta_c(\mathbf{X}) &= \frac{1}{2} + \frac{1}{4} \sum_{j \in \Pi_{\mathbf{X}}} |P(j|d) - P(j|u)| \\
\text{(Lemma 1)} \ &\geq \frac{1}{2} + \frac{1}{4} \sqrt{2 \sum_{j \in \Pi_{\mathbf{X}}} (P(j|d) - P(j|u))^2} \\
&= \frac{1}{2} + \frac{1}{4} \sqrt{2\theta_I(\Pi_{\mathbf{X}})} \\
\text{(Theorem 1)} &\overset{\mathcal{P}}{=} \frac{1}{2} + \frac{1}{4} \sqrt{2 \lim_{n \to \infty} \frac{s_n^2 I_{\Pi_{\mathbf{X}}}}{n\lambda^2(1-\lambda)^2}} \\
&\overset{\mathcal{P}}{=} \frac{1}{2} + \frac{1}{4} \sqrt{2 \lim_{n \to \infty} \frac{I_{\Pi_{\mathbf{X}}}}{n\lambda(1-\lambda)}}. \qquad (2.9)
\end{aligned}$$

□

Using sample data, the estimated lower bound for θ_c is then:

$$\frac{1}{2} + \frac{1}{4} \sqrt{\frac{2I_{\Pi_{\mathbf{X}}}}{n\lambda(1-\lambda)}}. \qquad (2.10)$$

The lower bounds presented in the toy example were obtained using the above Equation (2.10). We also extend to an arbitrary prior in Corollary 2.

Corollary 2. *Under the assumptions of an arbitrary prior $\pi(d)$ and $\frac{n_d}{n} \to \lambda$ as $n \to \infty$, the correct prediction rate is:*

$$\theta_c^*[p_{\mathbf{X}_d}, p_{\mathbf{X}_u}] = \frac{1}{2} + \frac{1}{2} \sum_{j \in \Pi_{\mathbf{X}}} \left| P(j|d)\pi(d) - P(j|u)\pi(u) \right| \qquad (2.11)$$

The last generalization of the proposed framework involves the possibility of incurring different costs (or losses) when making incorrect predictions. We leave this to the SI for the interested reader.

Note that searching for $\mathbf{X}$ with larger I-scores is asymptotically equivalent to searching for larger values of the lower bound in Equation (2.8) which is closely related to the correct predictivity of a given variable set $\mathbf{X}$, $\theta_c(\mathbf{X})$. For example, if a variable set $\mathbf{X}$ has a large value of I-score (substantially larger then 1, see Chernoff, Lo, and Zheng

2009), it is a strong indication that **X** itself could be a variable set with high predictivity. This stands in contrast to many current approaches to prediction (e.g. Random Forest, LASSO) that are evaluated for predictivity via cross-validation.

Desirable properties of the *I*-score

We note that the *I*-score is one possible approach to approximating the prediction rate in the sample analog form, and that the search for other potential scores is desirable and needed. Nevertheless, several properties of I are particularly appealing.

First, I requires no specification of a model for the joint effect of $\{X_1, X_2, ..., X_m\}$ on Y. It is designed to capture the discrepancy between the conditional means of Y on $\{X_1, X_2, ..., X_m\}$ and the mean of Y. Second, when a variable independent of the dependent variable is added to a group of variables which may have an influence (adding a noisy variable to an influential variable set), the *I*-score tends to *decrease*. In other words, I can screen out noisy variables. The Pearson chi-square statistic does not have this property; rather, it tends to increase, leading to a tendency to adjoin useless variables to those considered. As demonstrated earlier, the sample analog of θ_c also suffers from this problem.

Third, as mentioned earlier, the *I*-score does not monotonically increase with the addition of any and all variables as would the sample analog form of θ_c. Rather, given a variable set of size m with $m-1$ truly influential variables, the *I*-score is typically higher under the influential $m-1$ variables than under all m variables. If $m-1$ variables are influential in the sense that any smaller subset of variables is less influential, then removal of a variable to size $m-2$ will decrease the *I*-score in expectation. This natural tendency of the *I*-score to "peak" at variable set(s) that lead to high predictivity in the face of noisy variables under the current sample size is crucial.

Most importantly, we showed that the *I*-score can help find variables with high θ_c by identifying variables that have high values of θ_I (recall $\theta_I = \sum_{j \in \Pi_{\mathbf{X}}} (P(j|d) - P(j|u))^2$), which is related to the lower bound of θ_c. An important step to finding these highly predictive variable sets and discarding noisy ones through finding high *I*-scores is using the *Backwards Dropping Algorithm* (BDA) developed in Chernoff et al. (2009). The algorithm requires drawing many starting sets of variables and recursively dropping

random variables and calculating I-scores. For more information, see Chernoff et al. (2009) or the SI.

2.7 Using the I-score in sample-constrained settings

We have shown that the I-score asymptotically approaches a constant multiple of θ_I (which is related to a lower bound of θ_c) and has several desirable properties. We take this opportunity to explore and illustrate to the reader an application of the I-score measuring predictivity with sample data. To provide additional evidence of the I-score's ability to measure true predictivity, we consider a set of simulations for which we know the "true" levels of predictivity for all variable sets. We also provide a real data application on breast cancer for which the I-score approach has done very well in predicting.

We take a moment to comment that evaluating a variable (or variable set) for predictivity, which is the variable selection (VS) stage, is different from evaluating a given classifier, which is the prediction stage. The latter considers evaluating the predictivity of $f(\mathbf{x})$, a function $f(\cdot)$ applied to a particular set of explanatory variables $\mathbf{x}$, for a given outcome variable y, while the former considers the potential predictivity of the set of explanatory variables $\mathbf{x}$ for that outcome y. Often we see the two stages conducted simultaneously, as is common in approaches such as LASSO or other regression-based approaches. Our work here focuses simply on the VS stage. Variables selected as highly predictive in our framework can then be flexibly used in various models for prediction purposes as pleases the researcher.

We are now in an odd situation where we have identified variable sets that could not have been found using conventional approaches and yet we would like to evaluate the predictivity of our identified variable sets against these conventional approaches. Nevertheless, we endeavor to do so. A couple options arise for approaches to compare against: training prediction rate and out of sample testing prediction rate.We compare our approach against these two rates. We will show that the I-score based measure provides a useful estimated lower bound to the correct prediction rate and correlates well with the out of sample test rate, while the training rate statistic, the sample analog

of θ_c, does not. As such, our approach has an important benefit to prediction research: compared with methods such as cross-validation of error rates, the I-score is efficient in the usage of sample data, in the sense that it uses all observations instead of separating data into testing and training.

Simulations

We offer simulations to illustrate how (1) the I-score can serve as a lower bound to the true predictivity of a given variable set even as noisy variables are adjoined (2) thereby serving as a screening mechanism, and (3) finding the maximum I-score when conducting BDA leads to finding the variable set with the highest corresponding level of predictivity.

The simulation is based on a 3-SNP disease model, depicted in Figure 2.3. We also present a 6 SNP model in the SI. In this model, a disease SNP group composed of three SNPs X_1, X_2, and X_3 can jointly affect whether an individual has a disease. The remaining variables, (X_4, ..., X_{12}), are all noisy and unrelated to the disease. The frequency for the minor allele (coded as 0) of each SNP are all 0.5. The population baseline odds is set at $1/20$ (the ratio of the probability of an event occurring over the probability of that event not occurring). The high and low risk genotype combinations can be found in Figure 2.2.

In this set up, if an individual carries no risk genotype, his/her odds of having the disease are reduced by three fold; with a risk genotype, his/her odds are twice that of the baseline. It is clear of course that we still do have random chance; we cannot predict perfectly as the population baseline is 10% for having the disease.

We simulate data at three sample size levels: 250 cases and 250 controls, 500 cases and 500 controls, and 1000 cases and 1000 controls. For each possible variable set, we create a partition Π and calculate the $\hat{p}_{id}$ and $\hat{p}_{iu}$ (the estimated probability that an individual in cell j is a case or a control), respectively: $\frac{n_{id}}{n_d}$ and $\frac{n_{iu}}{n_u}$ where $i = 1...m$ and $m = |\Pi|$ where $|\Pi|$ is the size of the partition Π. We conducted 300 simulations and evaluated a set of statistics on each of the variable sets for each simulation. These were: training prediction rate, Bayes' prediction rate, out of sample prediction rate, and the I-score derived lower bound estimate of the predictivity rate; see Figure 2.3. Throughout,

we assume an equal prior. The statistics are detailed below:

1. Training prediction rate is defined as the following:

$$\frac{1}{2}\sum_{j=1}^{m_1}\max(\hat{p}_{jd},\hat{p}_{ju})$$

2. Bayes' rate: recall this rate is constant across all variable sets that are inclusive of the truly influential variables, regardless of how many noisy variables are also included. This is the best predictivity one can achieve if knowledge of the influential variables is available. It is defined as:

$$\frac{1}{2}\sum_{j=1}^{m_1}\max(p_{jd},p_{ju})$$

3. Out of sample prediction rate: this is conducted on the "infinite" future data to find p_{jd} and p_{ju} for the rate. The "infinite" future data is often unrealistic with real data but we present it for the purposes of this simulation and to clearly provide a golden standard against which to compare. It is defined as:

$$\sum_{j=1}^{m} p_{jd}\cdot\hat{Y}_j + p_{ju}\cdot(1-\hat{Y}_j)$$

4. I-score lower bound predictivity rate as defined from Equation (2.10).

We have demonstrated that the $\frac{I}{n\lambda(1-\lambda)}$ estimates[7] θ_I, which is related to an asymptotic lower bound (Equation (2.8)) for θ_c, as $n\to\infty$. It would be helpful to see how I performs at fixed, reasonable sample sizes. We compare the I-score derived prediction rate and compare it against the Bayes' theoretical prediction rate in our simulations to illustrate this. The out of sample correct prediction rate is presented in the simulations here as a further benchmark against which the I-score can be compared when data is limited, as is the case in real world applications. The out of sample correct prediction rate is derived from the most optimistic context we can achieve in the real world, whereby future testing data is infinite. In all the simulations, the I-score of a set of influential variables drops

[7]This assumes that $s_n^2\to\lambda(1-\lambda)$ as $n\to\infty$.

when a noisy variable is added. This drop is subsequently seen in the I-score derived bound for the correct prediction rate. The I-score has the benefit of screening out noisy variables, which makes it useful in practical data applications.

To illustrate how these statistics fare in accurately capturing the level of predictivity of each variable set under consideration, we considered their performance given already having found X_2 and X_3 as important. We then add X_1 (also an important variable), which should ideally correspond with an increase in the statistic. We continue to add the remaining noisy variables one at a time to this "good" set of variables and observe how the statistics evaluate the new, larger set of variables for predictivity.[8]

Next, we consider the output of the simulations. The first row of plots in Figure 2.3 correspond to comparing the (red) I-score lower bound correct prediction rate against the (light blue) out of sample test (correct) prediction rate. Recall the out of sample test prediction rate is not attainable in the real world as it requires an infinite supply of future samples of data to calculate. We use it here as a standard with which to compare against. The lower row of plots correspond to comparing each of the (dark blue) training prediction rate against the (light blue) out of sample test prediction rate. The graphs show an increase in sample size from left to right. The thick black line is the true Bayes' rate. The Bayes' rate rises when adding X_1 to the variable set X_2 and X_3 and then continues to remain flat with the addition of noisy variables so long as the informative variables (X_1, X_2, and X_3) are included. This is because the Bayes' rate is defined purely by the partition formed from the informative variables, and does not change when adjoining noisy variables (X_7,..., X_{12}) and creating finer partitions.

The x-axes depict the variable sets under consideration. Variables in red are influential for the disease (X_1, X_2, X_3). Variables in black are noisy ($X_7, \ldots, X_{12}$).The y-axes depict the correct prediction rate, θ_c. The graphs use violin plots to show the distribution of the statistic under each setting across the simulations. Violin plots are similar to box-plots but display more distributional information (such as standard errors), as the plots are created using a density curve of the correct prediction rate bound from random samples.

[8]We simulated 9 variables — 3 influential and 6 noisy — making a total of 512 subsets in total. For ease of presentation we do not consider all subsets here, only what occurs when adding noisy variables to see how the statistics differ in their screening capabilities.

Several patterns emerge in these simulations. First, and most importantly, the I-score-derived prediction rate appears to be a reasonable lower bound to the Bayes' rate. This holds even in moderate sample sizes. The peak value of the I-score lower bound is associated with the variable set that is inclusive of all influential variables (X_1, X_2, and X_3) and *no* additional noisy variables.

The second pattern observed is that the estimated I score lower bound peaks at the variable set that includes only X_1, X_2, X_3 (the influential variables). This is a characteristic of the out of sample correct prediction rate as well. For instance, if we consider the top row of Figure 2.3 and start from the right of the x-axes in each of the three plots with the largest set of variables inclusive of both influential and noisy variables ($X_1, X_2, X_3, X_7, ..., X_{12}$), continual removal of the noisy variables (sliding to the left of the x-axis) until we reach the variable set (X_1,X_2, X_3) results in higher predictivity as measured by the I-score lower bound. We can note that the I-score lower bounds drop upon further removing the influential X_1 variable from the set (X_1, X_2, X_3). Thus the variable set that appears with the maximum I-score derived lower bound here both identifies the largest possible variable set of influential variables with no noisy variables and is also reflective of a conservative lower bound of the correct prediction rate for that variable set. We note here that once we have found the variable sets with the highest I-scores and calculated the corresponding lower bound of the correct prediction rate, we can adjust this lower bound rate for its bias to derive an improved estimate of the correct prediction rate.

A third pattern that emerges is that the training rate will suffer from overfitting with the adjoining of noisy variables even when the variable set includes a true influential subset of variables. If the variable set is irreducible however, the training rate estimator reflects the Bayes' correct prediction rate well; thus the training rate estimator can perform reasonably well conditional on already identifying (X_1, X_2, X_3). The training rate estimator cannot be used to screen to that variable set first, however.

Finally, and as we might expect, the training set rate explodes due to overfitting in high dimensions as more noisy variables are adjoined to the partition formed by the informative variables (X_1, X_2, X_3). Thus while the training set prediction rate seems to improve as the sample size increases, it cannot be used to screen out noisy variables,

and is therefore difficult to use as a statistic to *select* highly predictive variable sets. The predictivity rates found through this statistic also dramatically departs from the out of sample testing rate. It tends to ever-optimistically evaluate the variable sets for their future predict even as only noisy variables are added. This stands in stark contrast to the out of sample prediction rate as it lowers in prediction rate with the addition of useless variables. We notice that there is a trend that the I-score prediction rate does not remain flat. The score increases when removing a noisy variable and reducing to a variable set of only influential variables, indicating the additional advantage of the I-score as a lower bound; the I-score prefers a simpler model even when the Bayes' rate remains the same, thus selecting for more parsimonious partitions that attain the Bayes' rate, which simultaneously is a closer reflection of the out of sample prediction rate.

Recall the correct prediction rate is based on an absolute difference of probabilities summed over all Xs, an L_1 norm. Suppose we start with influential variables only, with θ_c^* correct prediction rate, the highest we can attain out of all possible variable sets. Adding noisy variables to this set, variables that add no signal but simply create a finer partition, still returns θ_c^*. When estimating the correct prediction rate using sample data, though, the estimated θ_c value generally keeps increasing if noisy variables are added; the researcher does not know when to stop the search for influential variables, making VS impossible. Ideally, we would like to "punish" adding such noisy variables to our variable set, so having a measure that balances between favoring coarser partitions but still recognizing actual new variables with strong enough signals (non noisy variables) is important. The I-score seems to support such an effect — preferring coarser partitions unless an additional variable (and therefore finer partition) provides enough signal in the data to justify keeping it.

Noisy variables in sample data may be indicative of actually noisy variables or influential variables with weak signals due to the sample size. Thus, we note there are cases where the I-score might not recognize these variables when their signals would require unrealistic sample sizes to be found through the measure. An example of this would be if a good predictor is highly complex (perhaps a combination of very many variables) and the observations are sparse in the partition. Since the I-score places greater weight on where the data tends to appear (note the n_i^2 term in the score), when

most of the partition cells contain no observations or at most 1 observation, this can often look like noise.

The main draw, however, of the I-score is its ability to screen for influential variable sets. The variable sets inclusive of all three actually influential variables (X_1, X_2, and X_3) alone display the highest I-scores. Searching for variable sets with the highest I-scores thus tends to return highly influential variables only. Using the training prediction rate as a guiding measure for screening, however, would continually seek for ever larger variable sets, regardless of whether they include noisy variables or not.

Real data applications

Application to the vant Veer breast cancer data

To reinforce the previous sections, we turn to a brief analysis of real disease data. As noted before, part of this research team has discovered that applying the PR approach to real disease data has not only been quite successful in finding variable sets (thus encompassing higher order interactions, traditionally rather tricky in big data), but has also resulted in finding variable sets that are very predictive[9] that do not necessarily show up as significant through traditional significance testing. We present one discovered variable set (a total of 18 variable sets were found in Wang et al. (2012)) found to be highly predictive for a real data set on breast cancer that is not highly significant using a chi-square test.[10] In Table 2.1 we investigate the top, 5-variable set (or in this case, group of five genes) found to be predictive through both top I-score and performance in prediction in cross-validation and an independent testing set in Wang et al. (2012). To find how significant these variables are, we calculate the individual, marginal association of each variable in the marginal p-value. Given the family-wise p-value threshold of $6.98\text{x}10^{-5}$, none of these variables show up as statistically significant. Measuring the

[9]Here "predictive" refers to both high in I-score as well as having high correct prediction rates in k-fold cross-validation testing rates.

[10]We note an inherent difficulty the presentation of the reverse situation, that of finding the most significant variable sets in the breast cancer data and determining their predictivity rates. This is precisely because the PR approach allows for higher order interaction searches which is more difficult using current common approaches. While it is possible to use common approaches to discover marginally significant variables, or possibly two-way interactions, and then determine their predictivity rates, capturing up to 5-way (as shown in our presentation here using the PR approach) interactions is not yet feasible as of the date of this writing.

joint influence of all five variables does not have a p-value that is significant either. Using the variable sets (all 18 in Wang et al. 2012) that appeared to have the highest I-scores to predict on this dataset resulted in an out-of-sample testing error rate of 8%, in direct comparison with the literature's best error rates of 30%. Using only the variable set displayed in Table 2.1 and the lower bound in Equation (2.6) we can calculate the asymptotic lower bound of the correct prediction rate for this variable set as 59%. Thus, only using this variable set alone, we can achieve a 59% correct classification rate at minimum. For details on the final predictors, see Wang et al. (2012).

2.8 Concluding remarks

Prediction has become more important in recent decades and, with it, the need for tools appropriate for good prediction. A first step in prediction can be to find variable sets that are highly predictive, which we have called the VS stage in this paper. We show in other work that approaching VS through selection of variables from a statistical significance criterion (for instance, using the chi-square test statistic) is not ideal Lo et al. 2015. A currently popular alternative solution is to conduct VS via sample-based, out-of-sample testing error rates. This approach is ad hoc in nature, sample-based, and is not measuring some theoretical underlying level of predictivity for a given variable set. Often validation of selected candidate variable sets requires setting aside valuable sample data in out of sample testing or cross-validation. Sometimes the sample size may not suffice for validating variable set sizes larger than one or two variables, as is often the case in big data like GWAS. As such, prediction research would benefit from a theoretical framework towards finding highly predictive variables. We believe our work here is a preliminary and important effort in that direction, by considering what theoretically maximally predictive variable sets are, and how we might try to find them. In fact, using measures like the I-score could be an important new direction in the prediction literature as it neither uses the training sample prediction rate at all nor does it require an artificial or an ad hoc regularization choice.

We identify the equation for the theoretical correct predictivity of variable sets (θ_c) in Equation (2.4) and then demonstrate that unfortunately, the sample analog for

it is quite useless. As such, we offer an alternative measure. We show that the I-score asymptotically approaches a lower bound to Equation (2.4), θ_I, and is thus correlated with the correct predictivity rate of a given variable set. Importantly, we show that the I-score has a natural tendency to discard noisy variables, keep influential ones, and asymptotically approach this lower bound to θ_c. The I-score does well in identifying predictive variable sets in both our complex simulations as well as real data application.

We note that other measures with such desirable properties may also exist, and we encourage rigorous research in this direction. As a new field of inquiry, the search for measures that maximize predictivity may do much in the way of living up to the hopes of advancing predicting outcomes of interest, such as disease status. In some ways, this work is motivated by a practical consideration of finite samples. As noted in the set-up of our framework, in a theoretical world of limitless data, we can in fact find the variable sets with highest values of θ_c. However, our real world of finite sample sizes requires other sample-appropriate measures that may approximate but not achieve the θ_c. This occurs during the variable selection stage. In other words, based on available sample size, the I-score, and any other such measure, detects not necessarily the maximum θ_c but some $\theta_{c,n}^H$, the largest θ_c correct prediction rate for which the corresponding X variables can be selected given n. Consider a situation where the true set of variables $\mathbf{X}^*$ that provide the theoretical maximum θ_c^* is very large. Suppose we have a sample of data that is quite modest. Selecting all variables $\mathbf{X}^*$ is not possible given the n size (the partition of the data is too fine) and so a measure like the I-score retrieves a set $\mathbf{X}'$ that provides potentially the largest θ_c achievable given the sample constraint. This in some ways mirrors the common issue of not detecting true effects when the sample size is too small in statistical significance testing.

We leave the important discussion of how to combine identified predictive variable sets in different final prediction models outside the scope of this paper.

2.9 Acknowledgements

The material in Chapter 2 is currently being prepared for submission for publication of the material. This material was coauthored amongst Adeline Lo, Herman

Chernoff, Shaw-Hwa Lo, and Tian Zheng.

2.10 Tables and Figures

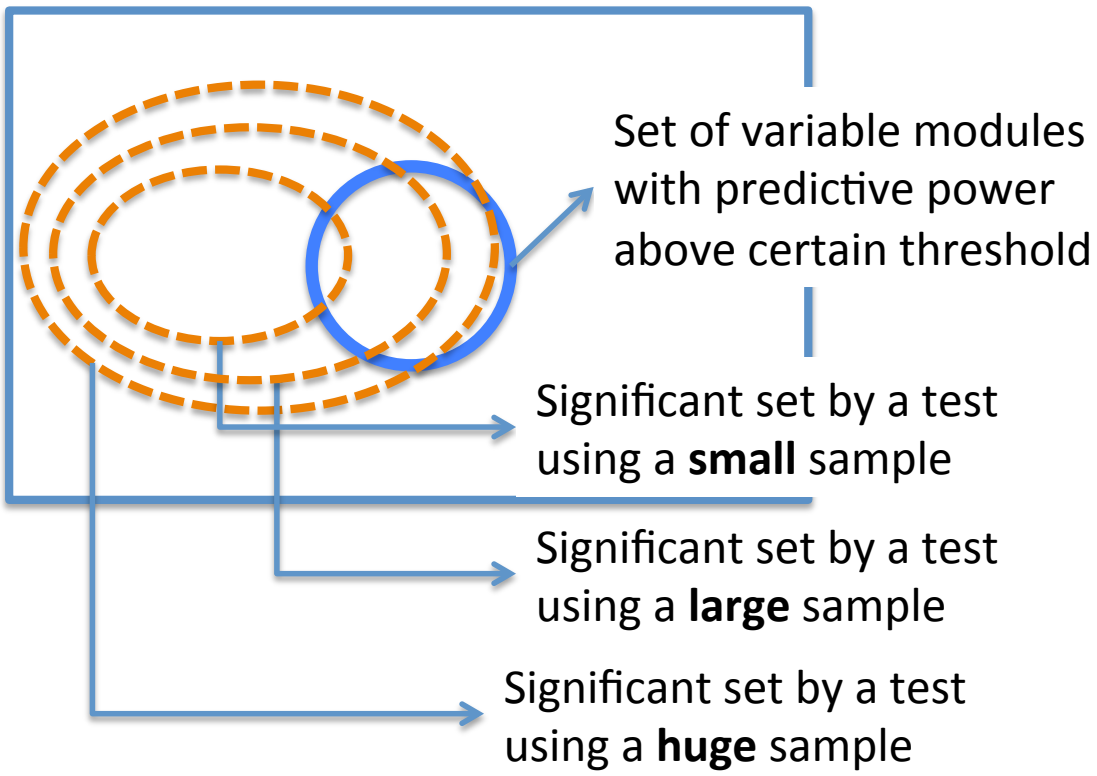


Figure 2.1: **Illustration of the relationship between predictive and significant sets of variable sets.** Rectangular space denotes all candidate variable sets. Significant sets are identified through traditional significance-tests.

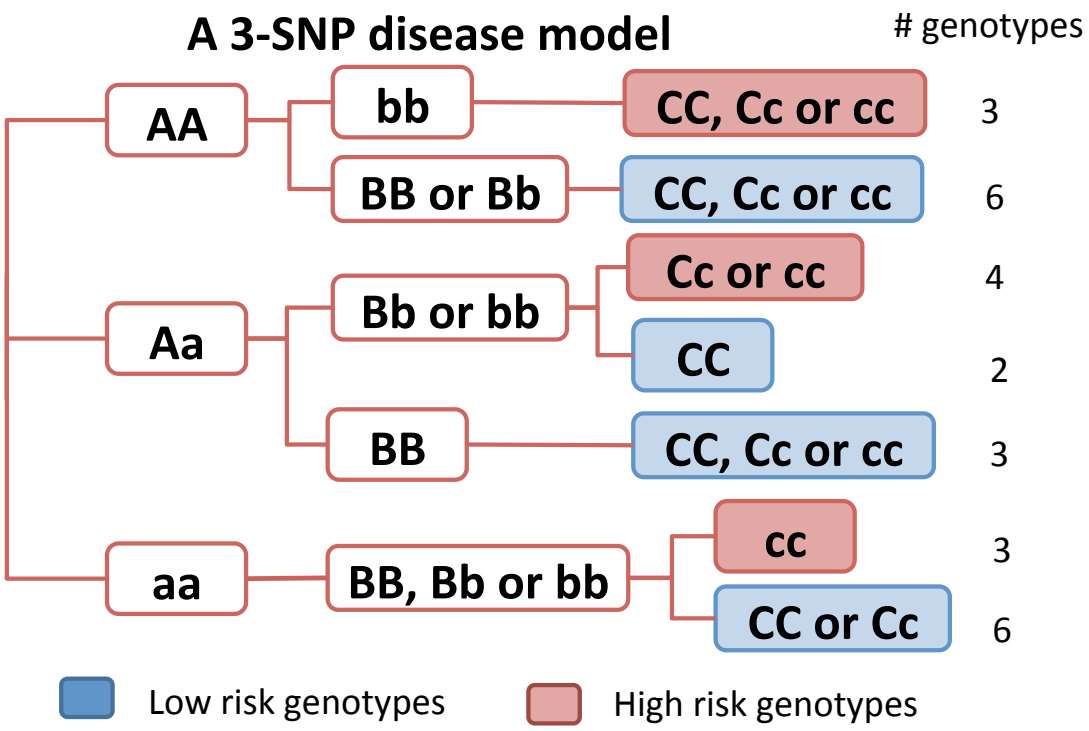


Figure 2.2: **3 SNP disease model**. High risk genotypes are shaded in red, while low risk genotypes are shaded in blue.

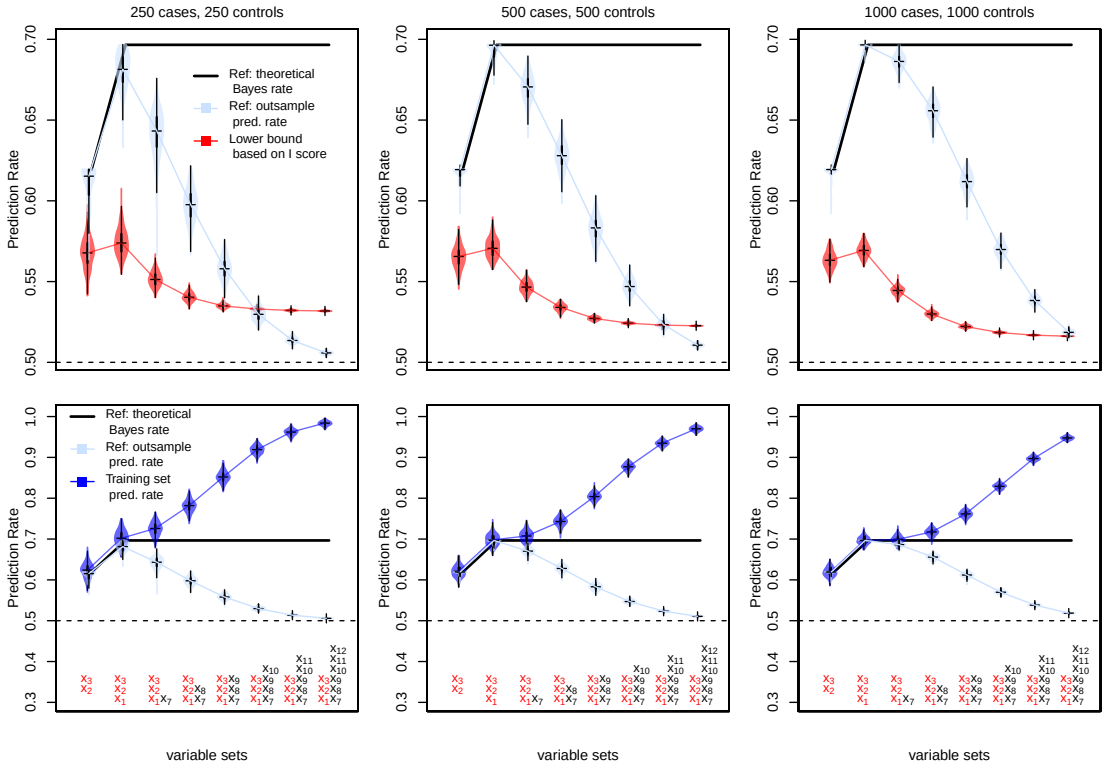


Figure 2.3: **Variable set size 3: Comparison of the training rate and I-score against the out of sample prediction rate** Here we compare two statistics, I score lower bound and the training set prediction rate against the out of sample prediction rate. Lower bound from the I score is provided in red, training set prediction rate in blue, and the out of sample prediction rate is in light blue. The thick black line in all six graphs is the true Bayes rate. All x-axes correspond to variable sets (described in red for important variables and black for noisy ones) while all y-axes correspond to (correct) prediction rate. There are three important variables in this example, x_1, x_2, and x_3. The top row of graphs compares the (red) I score statistics against the (light blue) out of sample prediction rate. The lower row of graphs compares the (dark blue) training set prediction rate against the (light blue) out of sample prediction rate. From left to right the graphs increase in sample size from 500 cases and 500 controls, to 1000 cases and 1000 controls in the middle, to 2000 cases and 200 controls on the right.

Table 2.1: **Real data example: van't Veer breast cancer data.** A variable set of five interacting genes is presented.

	Systematic name	Gene name	Marginal p-value
1	Contig45347_RC	KIAA1683	0.008
2	NM_005145	GNG7	0.54
3	Z34893	ICAP-1A	0.15
4	NM_006121	KRT1	0.9
5	NM_004701	CCNB2	0.003
	Joint *I*-score: 2.89	**Joint p-value:** 0.005	Family-wise threshold: 6.98×10^{-5}

Chapter 3

Variable selection with Partition Retention to predict civil war onset

3.1 Abstract

I propose a new approach to predicting civil war onset that emphasizes variable selection. I argue that variables and their interactions should be selected based on their predictive power. Selecting variables based on theoretical importance and statistical significance does not guarantee good prediction. Additionally, using only information from non-interacted variables may result in discarding important information embedded in high order interactions (interactions between several variables). I show that important higher order interactions can be selected by the Partition Retention (PR) method. I illustrate the approach with simulations and an application to civil wars data, comparing the results with alternative approaches such as random forests, neural networks and lasso. The PR approach performs favorably in simulations, and in the real data application, conducting variable selection with PR boosts correct prediction rates on out of sample testing sets from 78.98% to 98.05%. The application demonstrates possible gains in correct prediction rates for civil wars when including a research step for identifying predictive variable sets.

3.2 Introduction

Civil wars are highly consequential and complex processes. There are over two million scholarly articles on this topic as of the writing of this paper.[1] In particular, the forecasting of conflicts, such as civil wars, has grown in prominence in the conflict literature (see Beck, King, and Zeng 2000; Ward, Greenhill, and Bakke 2010; Hegre et al. 2013; Goldstone et al. 2010; Gleditsch and Ward 2013; Muchlinski, Siroky, and Kocher 2015, among many others). Predicting conflict is important due to the widespread humanitarian consequences (Burke et al. 2009) and because conflicts are becoming more prevalent and lethal (Ward, Greenhill, and Bakke 2010). Accurate prediction is integral in the preparation against and amelioration of conflict. A recent Center for Strategic and International Studies special project briefing noted that "early warning models" that can predict conflict are critical for policymakers to anticipate, analyze and plan for a conflict. When policymakers are unprepared, they respond hastily or even exacerbate the conflict by delaying responses (Barton et al. 2008).

Prediction is a useful exercise that should be part of the analyst's research process. Researchers can move beyond generating models that are reflective of their theories to testing whether these models are reflective of data that is not yet at hand. This is likely to be especially important given that many studies are based on observational data and are thus susceptible to bias based on data availability. We can also improve our substantive understanding of social science phenomena by understanding where prediction is especially poor when applying our models to new contexts.

The prediction of conflicts like civil war is characterized by two approaches. The first is primarily guided by a rich literature on the origins and causes of civil wars (see Sambanis 2002; Blattman and Miguel 2010 for reviews). Variables known to capture these origins and causes are considered important for the prediction of civil wars and are selected for use in statistical models. A more recent approach has pursued explorations of different models on conflict data; after selecting the input variables either through guidance from the theoretical work surrounding civil wars or through data mining, the researcher picks amongst models that might have different abilities to predict the data. This work contributes to the latter approach, emphasizing in particular the importance

[1] Date of search: July 30, 2015.

of variable selection.

This paper provides several contributions. The first is to demonstrate that however rich the theoretical literature on civil wars may be, if the goal is predicting wars then conducting variable selection based on statistical significance is not as effective as conducting variable selection based on a criterion of good prediction. I refer a variable set's ability to predict as predictivity. The second approach in the literature on conflict prediction, selecting models based on different abilities to predict the data, can be improved by including a variable selection stage that accounts for interacting variables and selects based on predictivity.

The paper's second contribution, is justification for variable selection as a premodeling stage, especially for large or big datasets, where the number of variables and variable interactions exceeds the sample size. The last few decades have seen an enormous growth political science dataset size, in both the "length" (number of observations) and "width" (number of variables) of the dataframes (Grimmer 2015; Nagler and Tucker 2015). Including interactions between variables will exponentially expand the width of the dataframe. Methods that are able to account for the larger dimensions of the available data are highly desired for prediction. More attention should be devoted towards identifying variable selection methods for prediction. These variable selection methods should be able to accurately screen amongst variables that identify the highly predictive single and grouped variables. It might be particularly difficult to identify variable sets with high predictivity when the variables in the sets interact to predict well but the individual components to the set do not appear predictive for the outcome of interest.

The third contribution is the proposal of a new candidate approach, the Partition Retention (PR), for predicting civil wars. The PR approach relies heavily on a measure called the I-score in variable selection. It can both handle interactions through the screening algorithm as well as assess the predictivity of variable sets. I argue for a fundamental shift in our approach towards prediction by rethinking how we identify the influential variables we use in our models for prediction. This includes consideration of higher order interactions (here defined as any two-way or higher interaction between variables) between explanatory variables when defining our influential variable sets. I propose the PR method as one such potential methodological candidate; upon recover-

ing influential variable sets, the researcher can incorporate the predictive variable sets in a model of her choice. The PR approach can thus be considered a pre-modeling analytical tool for excavating highly influential variables.

I demonstrate the usefulness of PR for variable selection using simulations and a real data application to a civil wars dataset. The method allows for identifying previously undertheorized influential variables, which may aid civil war onset theory-building when there is contention over the composition of the set of influential variables amongst a larger set of political variables. While the aim of this paper is not to build a theory behind civil war onset, one positive externality of excavating higher-order interactions may be in contributing new variable interaction possibilities in the pre-theory building process of the causes of civil war. I consider this in brief in the civil war data application of this paper.

The paper proceeds as follows. The first section provides a literature review on variable selection specific to the conflict literature. The second section reviews a few common variable selection methods from the machine learning literature that have been applied to conflict data. The third section presents the Partition Retention approach. The fourth section offers simulations to illustrate some of the variable selection attributes of PR when compared with methods such as the lasso and random forests. The fifth section is an application of the Partition Retention approach to a real civil wars dataset. The sixth section concludes.

3.3 Variable selection in the conflict literature

When predicting conflict, we need to know what variables to use in the prediction model. This process of identifying good variables for prediction is called variable selection (referred to throughout as VS).[2] A common VS approach to predict conflict is to select variables previously identified to be theoretically relevant for use in prediction models. These explanatory variables are often tested in conflict data for statistically significant associations with the conflict outcome, which is taken as supportive evidence of their theoretical value. The reasoning is simple: if certain variables have been identified

[2]Sometimes referred to as feature selection in the machine learning literature.

to be significantly associated with civil wars, then it seems reasonable to conclude that conducting VS by using statistically significant variables should help us predict civil wars. This approach is not to be confused with the practice of using statistical significance as a way of evaluating the statistical model for prediction. It is well accepted that the gold standard for evaluating the latter is out-of-sample testing (Goldsmith et al. 2013).

Theory-guided prediction, particularly through usage of variables that appear significantly related to conflict, can be misdirected, however. This point is made explicit in Ward, Greenhill & Bakke (2010), where the authors caution against the usage of variables that have low p-values for predicting civil conflicts, demonstrating the poor performance of such an approach with both Fearon and Laitin's (2003) data and Collier and Hoeffler's (2004) data.[3] That statistical significance is a poor criterion for good prediction is neither new nor specific to political data; Welch and Goyal (2008) note the robust inability for variables well-known in the literature to be statistically significant and related to the equity premium to predict well in a variety of models. In the sciences, genes that are statistically and biologically significant for certain diseases do not immediately transfer to good prediction of those diseases (Clayton and Gleditsch 2014; Gransbo et al. 2013). In related work, Lo et al. (2015) explores the disconnect between statistical significance and predictiveness and finds that this is both prevalent in many types of data and worsens as the data become more complex.

The conflict forecasting literature still relies on this approach, even in recent work, despite heavy caution against this method. The most common approach is to collect variables previously theorized to be relevant in conflict and use these in various prediction models (see inter alia, Hegre et al. 2013; O'Brien 2010; Goldsmith et al. 2013). A related approach is to suggest a new set of variables that could be good predictors for conflict, justified by their theorized linkages to conflict. For instance, Rustad et al. (2011) note that local variables such as natural disasters or local elections could trigger different baseline risks for civil conflict and proceed to use subnational variables to predict the level of civil conflict risk. This usage of micro-level variables is echoed in Balcells and Justino (2014) and their call for linking macro conflict studies with micro

[3]The authors do not explain precisely how and why significance and predictivity differ, however. For an explanation, see Lo et al. (2015).

level variables in understanding civil conflict. While the goal remains good prediction of conflict, the selection and addition of new important variables is guided through a theoretically motivated approach. Bell et al. (2013) join in this vein of work by conducting VS guided by concepts they believe influence the probability and degree of political violence in a state, such as coercion, coordination and capacity. Weidmann and Ward (2010) emphasize the importance of incorporating geographic variables to the prediction of conflict as neighbors might affect each others' probability of conflict.[4] Importantly, these works also primarily devote their analyses to the marginal information (the effects of single variables) and do not appear to consider higher order interactions. In contrast, a contribution of this paper is to highlight the importance of VS, including higher order interactions, and using this to predict better.

The efforts of Goldstone et al. (2010) and Muchlinski et al. (2015) are closest to the contributions of this paper. Goldstone et al. (2010) avoid using statistical significance in choosing their explanatory variables for their models predicting political instability and instead rely on VS that balances between least prediction error and stability across different samples of data. Their models do not appear to conduct VS across higher order interactions, however, only amongst the marginals. Muchlinski et al. (2015) propose the usage of random forests in the prediction of civil war onset. While random forests have had a long history in the machine learning literature (see Ho 1995; Breiman 2001 for the introduction to random forests, and Hastie, Tibshirani, and Friedman 2008 for a review of its usage), the authors argue that it has been underused in the highly appropriate setting of predicting civil wars. This paper contributes to the literature in a similar fashion by highlighting the importance of VS, particularly VS for groups of variables, and presenting the Partition Retention as a candidate technique to conduct VS in conflict forecasting. Furthermore, I demonstrate in simulations and the civil war application that the PR approach is weakly better than random forests in predicting.

While variable selection with the intent of good prediction is not new, the emphasis on doing so in the application of conflict has not been given nearly enough at-

[4]Though we note that the authors also include the twin goal of exploring whether information on geography can help understand where and when conflict might arise and are not only interested in making better predictions of conflict.

tention, as evidenced by the sizable amount of research on conflict prediction that uses significance-based VS. A good VS approach for prediction should be able to not only choose a variable based on some measure of its ability to predict but should ideally allow for choosing sets of variables that interact in ways that are good for usage in prediction. This paper actively embraces this approach to VS for good prediction by proposing the PR approach for VS. PR is shown elsewhere to predict very well (Fan and Lo 2013; Lo et al. 2008; Wang et al. 2012) in big data applications. The PR's *I*-score is also related to the theoretical correct prediction rate, making it a good candidate measure of predictivity (Lo et al. 2016), and can be used to retrieve higher order interactions (Chernoff, Lo, and Zheng 2009).

A related literature in political methodology places emphasis on limiting the number of explanatory variables in regressions, suggesting that the researcher take caution in using only variables for which there are reasons to believe are associated with the outcome of interest and refraining from "kitchen sink" approaches of statistical equations (Achen 2002; Achen 2005; Ray 2003; Ray 2005). In fact, Achen (2005) discusses the implications and pitfalls of erroneously and zealously including as many "control" variables as possible in common regressions. This would seem to have the opposite advice espoused here — of exploring the predictivity of all variables and variable sets the researcher has at hand in order to conduct good prediction. The key difference, however, is again the goal of the researcher. In the case of testing hypotheses and finding unbiased and consistent effects of certain explanatory variables on dependent variables of interest, the researcher desires to identify the causal influence of these explanatory variables. Adding as many control variables as possible could bias these estimates.[5]

Here, we are primarily concerned with how a researcher might be able to predict well. As such, our variable sets and models are assessed based on their final abilities to lower out of sample error rates (or improve out of sample correct prediction rates). Since we are selecting variables based on their predictivity and not conducting VS from a theory-guided standpoint, we ought to include as many variables as possible when we begin our VS process and allow our prediction-based VS approach to do the selection for us. In this sense, the spirit of this work joins that of Ward and Bakke (2005) in

[5] See Kadera and Mitchell (2005) for an overview of how control variables can be erroneously used in international conflict research.

their call for greater incorporation of contextual factors for better prediction. Thus, the explanatory variables on which we might conduct VS should include all available variables— *and all of their possible interactions*. The VS stage will reduce the number of final predictors we use in our predictive model and using cross-validation and out-of-sample testing can alleviate the effects of overfitting.

This full set of possible variables and variable interactions (which I will refer to as variable sets in this paper) is not small, given the types of data we have available today. Higher order interactions become more difficult to identify as the number of explanatory variables grow due to the well-known curse of dimensionality. While the individual effects contributed by single variables are important, when answering political science questions we may require information contributed by interactions of variables. This is of particular importance when we care about predicting a complex social outcome, such as predicting whether a given country year will see civil war onset, outcomes of elections, or voter turnout, among others.

To see how the curse of dimensionality might quickly affect the size of variable sets amongst which we must select a smaller set of highly predictive variable sets, consider the following. We have a single binary outcome variable of interest, Y, and a set of m explanatory variables $\mathbf{X} = \{X_1, ..., X_m\}$. When m is small relative to the sample size, for example, five, with a reasonably sized sample we can utilize a logistic regression with all possible interactions between all variables. We would have a manageable model of 31 parameters (the sum of five choose all variable set sizes). Simply double the size of m, however, and the number of parameters the researcher must estimate becomes 1,023.[6] In the real world of sample constrained data, uncovering influential higher order interactions quickly becomes exponentially difficult.

This presents a dilemma — given the full set of possible explanatory variables and their interactions, and limited sample sizes, how might we identify the most predictive variable sets?

[6]Number of parameters given m variables is $2^m - 1$.

3.4 Variable selection in machine learning

Ideally we want VS methods for prediction to have several key capabilities. First, they should be able to conduct VS by screening for the predictivity of variables and variable sets. Second, the VS approach should be able to handle interactions, even when the data grows in variable size. Recent advances in big data prediction and machine learning have begun to focus on dealing with large numbers of explanatory variables for prediction, as well as uncovering higher order interactions. Some favored methods in the classification literature that have been applied to conflict prediction include lasso (least absolute shrinkage and selection operator), neural networks and random forests. I review these approaches in brief here.

3.4.1 Lasso (least absolute shrinkage and selection operator)

Lasso is a shrinkage model that penalizes certain explanatory variables to 0. A common representation of the model is:

$$\hat{\beta}^{lasso} = \arg\min_{\beta} \sum_{i=1}^{N} \left(\left(y - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 \right. \\ \text{subject to } \sum_{j=1}^{p} |\beta_j| \leq t. \tag{3.1}$$

where y is the outcome of interest and x are explanatory variables. The lasso has been favored for its simplicity of execution as efficient algorithms exist for applying the L_1 lasso penalty $\sum_1^p |\beta_j|$ to the minimization problem (Tibshirani 1996). The penalizing constraint allows for the lasso to "tune down" certain coefficients to zero; this is a favored ability when handling large numbers of parameters under a sparsity assumption. The lasso estimator can be used for variable selection by selecting variables with non-zero estimated coefficients (Belloni, Chernozhukov, and Hansen 2014).

3.4.2 Neural networks

Neural networks have been used in political science data for a couple decades (see Schrodt 1995; Zeng 2000; Beck, King, and Zeng 2000). Indeed, a major application of the approach has been in conflict prediction; Beck et al. (Beck, King, and Zeng

2000) note that a possible improvement to the classic logistic regressions used for predicting the binary outcome of conflict could be the inclusion of more interaction terms that would be able to overcome the drawback of the highly restrictive and nearly constant specifications of ex ante probabilities of conflict. However, they too are concerned with the inherent curse of dimensionality problem that arises with the inclusion of too many interactions in a finite sample. The authors turn to neural networks as a possible good alternative approach that can directly handle this problem and find that neural networks seem to predict marginally better than generalized linear regression models in forecasting conflict when using the same explanatory variables.[7]

Neural networks are two-stage regression or classification models. For regressions, the outcome Y is often a single binary output. Here the generalized neural network equations are presented for K-class classification (Y_k, $k = 1, ..., K$). The outcome Y_k is simply a function of the linear combinations of the features Z_m ($Z = (Z_1, ..., Z_m)$), which themselves are linear combinations of the explanatory variable inputs (X_p, $p = 1, ..., P$):

$$\begin{aligned} Z_m &= \sigma(\alpha_{0m} + \alpha_m^T X), m = 1, ..., M, \\ T_k &= \beta_{0k} + \beta_k^T Z, k = 1, ..., K, \\ f_k(X) &= g_k(T), k = 1, ..., K, \end{aligned} \tag{3.2}$$

where $T = (T_1, ..., T_k)$, and $\alpha(v)$ is an activation function often chosen to be either the sigmoid $\sigma(v) = \frac{1}{1+e^{-v}}$ or a Gaussian radial basis function (see Hastie, Tibshirani, and Friedman 2008 for a more in depth discussion of neural networks). The output function $g_k(T)$ is a final transformation of the inputs T; often the multilogit model transformation is favored:

$$g_k(T) = \frac{e^{T_k}}{\sum_{l=1}^{K} e^{T_l}} \tag{3.3}$$

Note that if the activation function σ is the identity function the model reduces to a linear model in the inputs.

[7] Though this is disputed by De Marchi et al. (2004).

3.4.3 Random forests

Random forests is a modification of the technique called bootstrap aggregation (or "bagging"), proposed by Breiman in (2001). Bagging is a technique that reduces the variance of an estimated prediction function by averaging amongst a class of noisy but approximately unbiased models and is particularly useful in high variance, low bias procedures such as trees. Random forests builds a large set of decorrelated trees and then averages amongst them, making the random forests approach a relative of boosting. Random forest can be particularly useful in capturing complex interaction information in the data.

The random forest procedure is as follows[8]:

1. For $b = 1$ to B:
   (a) Draw bootstrap sample $\mathbf{Z}^*$ of size N from the training data.
   (b) Grow a random-forest tree T_b to the bootstrapped data. This is done by recursively repeating the below steps for the tree until the minimum node size n_{min} is reached.
      i. Select m variables from p total variables at random.
      ii. Pick the best variable/split-point among the m.
      iii. Split the node into two nodes.
2. Output ensemble of trees $\{T_b\}_1^B$
3. To make a prediction with new x data point:
   (a) Regression approach:
   $$\hat{f}_{rf}^B(x) = \frac{1}{B}\sum_{b=1}^{B} T_b(x). \tag{3.4}$$
   (b) Classification approach: Let $\hat{C}_b(x)$ be the class prediction of the bth random forest tree. Then x is classified according to a majority vote:
   $$\hat{C}_{rf}^B(x) = \text{majority vote}\{\hat{C}_b(x)\}_1^B \tag{3.5}$$

[8]For a full treatment of the random forests approach see Hastie et al. (2008).

While random forests have proven to be computationally efficient and useful for high dimensional data with interactions, the approach has difficulties discovering interacting information when the data consists of high dimensional information that cannot be broken down to lower dimensional components. This is due to the forward seeking algorithm in the tree splitting process; a backwards approach would allow for searching of variable sets that interact at a higher dimension but whose component variables might only contribute noise on their own.

I propose adoption of another VS approach to the conflict prediction toolkit, the Partition Retention (PR) method. PR is designed to tackle big data with much larger numbers of variables. The approach is a backwards partitioning variable selection method that heavily relies on the usage of an influence measure, the I-score, that is calculated for each selected variable set, and which is used as a measure to gauge how predictive that particular variable set is. The backwards dropping feature additionally allows for discovery and retention of variable sets with noise at lower order interactions but higher order influence.

3.5 Partition retention

The Partition Retention method is desirable as a statistical tool for social scientists for several reasons. First, it flexibly captures higher order interactions for a very large number of explanatory variables. Second, the variable sets identified by PR predicts at worst similar to and at best better than common alternatives (Chernoff, Lo, and Zheng 2009). This is unsurprising; when the true relationship between a given outcome variable and a set of independent variables is based mostly on marginal effects, then methods that focus on marginal effects are likely to do just fine in prediction. However, should the relationship involve more complicated dependencies amongst the explanatory variables, then PR stands at an advantage (Chernoff, Lo, and Zheng 2009). The last appeal is important. For example, we may desire to know whether a complex network of variables or a simpler process influences a political phenomenon. Finally, the I-score has been shown to correlate very well with predictivity in highly complex and high dimensional data (Wang et al. 2012; Lo et al. 2015). More recently, we show in

Lo et al. (2016) that the reason the I-score appears to do quite well in prediction is because it shares a relationship with the theoretical correct prediction rate of a variable set. Indeed, the calculated I-score of a given variable set for a sample can be adjusted to directly reflect a conservative lower bound of the correct prediction rate for that variable set.

Originally designed to tackle the overwhelming number of single nucleotide polymorphisms (SNPs) in the human genome that can jointly result in different disease phenotypes (and the resulting exponential boom in number of possible different interactions), the PR method has found success in selecting groups of genes highly predictive for diseases such as breast cancer, irritable bowel disease (IBD), and prostate cancer (Fan and Lo 2013; Lo et al. 2008; Wang et al. 2012). In brief, the PR approach involves randomly selecting a smaller subset of a full set of explanatory variables and analyzing if any of the variables in the subset are associated with the outcome variable. An influence measure, the I-score, attributes an amount of influence to the subset of variables and measures the association with the outcome variable. Next, a step-wise elimination decreases the subset of variables to a returned, smaller set of potentially influential explanatory variables. These steps are repeated many times and returned subsets are considered potentially highly influential towards the outcome variable (see Figure 3.1 for an illustration).

Two main innovations form the core of the approach. First, the PR method offers an alternative measure of influence, the I-score, to the classical significance-based measures (chi-square, F-statistics, etc.). The I-score is designed to retrieve variable sets that exhibit high predictivity and can be seen as relatable to a multiple correlation coefficient. The second innovation is in the approach of *backwards-dropping* variables from variable sets in order to identify maximally predictive variable sets. This is in contrast to the common forward searching (e.g. recursive partition methods), which are more likely to miss higher order interactions amongst variables if their lower order signals are weak or mild. I present the I-score first with an example.

Suppose we have a dataset of n observations with a binary observed variable Y of interest and a set S of explanatory variables measured as $X = \{X_1, ..., X_S\}$. If all X take at most 3 discrete values (e.g. 0, 1 or 2), then the idea is to partition the n observations

into 3^S partition elements identified with the values of X. There are n_i observations in each ith element. The I-score, or influence measure for how well the particular partition separates our observations into reasonably similar subsets (classifying between the two Y outcomes), is:

$$I = \frac{1}{n}\sum_{i=1}^{3^S} n_i^2(\bar{Y}_i - \bar{Y})^2 \tag{3.6}$$

where $\bar{Y} = \sum_i \frac{n_i \bar{Y}_i}{n}$ is the average of Y overall and $\bar{Y}_i$ is the average in the ith element. Suppose we wish to measure the influence of a single variable X_1 on I. We would simply consider the coarser partition formed by the $X = \{X_2, X_3, ..., X_S\}$ not including X_1. The I-score of this new, coarse partition, differenced from the partition including the X_1 variable is a measure of the influence of X_1 on Y when appearing with the other X variables in X. Should the new coarse partition produce a smaller I-score, we regard the X_1 variable as influential for Y. We repeat this process for all other X variables and consider the difference in I-scores, ultimately discarding the X variable that produces the lowest I-score. This procedure is repeated until discarding another X variable from the remaining set of variables produces only increases in the I-score; at this point the set of variables remaining are all kept. This is the *backwards-dropping* element to the PR approach.

When S becomes too large to estimate the finest partition, we can select a subset or group of m variables from $X = (X_1, X_2, ..., X_S)$ which defines a partition Π^* of the sample of n observations into $m_1 = 3^m$ partition elements. We denote these partition elements $\{A_1, A_2, ..., A_{m_1}\}$. These are all possible values taken by our subset of m variables. For ease of presentation, let $\{X_1, X_2, ..., X_m\}$ denote the group of m variables selected from the original X. Each A_j is a subset of $n_j Y$ values and $\sum n_j = n$. Every non-empty A_j has a mean value $\bar{Y}_j$. The overall mean is $\bar{Y} = \sum \frac{n_j \bar{Y}_j}{n}$. The I-score is then:

$$I_{\Pi^*} = \frac{1}{n}\sum_{j=1}^{3^m} n_j^2(\bar{Y}_j - \bar{Y})^2 \tag{3.7}$$

Again, we can compare a coarser partition of $\{X_2, X_3, ..., X_m\}$ against the original partition $\{X_1, X_2, ..., X_m\}$ by comparing the difference in I-scores under the full set variables, including X_1, that form the original partition against the I-score retained under

the coarser partition leaving out X_1. This difference is a measure of how much X_1 contributes in influence on Y in the presence of $X = \{X_1, X_2, ..., X_m\}$. The equation for the difference between a coarse and finer partition when X_1 can take a finite set of values is:

$$D_I = -\frac{1}{n}\sum_i \sum_{j<k} n_{ij} n_{ik} (\bar{Y}_{ij} - \bar{Y})(\bar{Y}_{ik} - \bar{Y}) \tag{3.8}$$

where A_{ij} is the subset in A_i where $X_1 = j$ and has n_{ij} elements averaging to $\bar{Y}_{ij}$. Asymptotic properties of the I-score and D_I can be found in Chernoff et al. 2009.

With this new influence score in hand, we turn to the second arm of the PR approach, the Backwards Dropping Algorithm (BDA). This step essentially involves repeated random sampling of subsets of variables from the total number of independent variables, finding joint I-scores, finding differences in I-scores and dropping variables from the subsets and rescoring until a maximum I-score is retrieved. The resulting subsets of variables are then ranked by I-score to find the most influential variable subsets. Below are the steps to BDA (also illustrated in Figure 3.1).

3.5.1 Steps in the PR approach

1. Randomly draw a subgroup m of variables from the total number S variables. Calculate the I-score of this group of m variables. Call this $I(m)$.

2. From m, randomly draw a single variable from the group and calculate the I-score for the remaining $m-1$ variables. Do this for each variable in m. Compare I-scores in (2) with the I-score in (1), $I(m)$. If $I(m)$ is larger than all the I-scores in (2), keep all m variables as influential, and the process ends here. If not, find the variable that, when removed, results in the largest positive I-score difference. This variable is regarded as not influential towards the outcome and is discarded; there remain $m-1$ variables.

3. Repeat step 2 until removal of any remaining variables from a set of size m^* results in a lower I-score than the I-score of set m^*. m^* is the set of interacting and influential variables for the outcome of interest.

4. Repeat steps 1-3 many times for coverage of the full set of combinations of set m variables.

5. Repeat steps 1-4 for other random draws from S of size m.

A key characteristic of this process is its recursive search for influential variable sets. Forward seeking algorithms (inter alia, random forests) rely on starting with single variables and deciphering further partitions of the data according to measures of marginal effects; the resulting variable sets are thus partitioned along specific types of dependency amongst the variables chosen. The boon of backwards dropping is that removing variables (not adding them) leave the joint dependencies amongst the remaining variables in the variable set intact (Chernoff, Lo, and Zheng 2009).

Although a returned influential variable set may be of higher order, lower-order influences in the same variable set may also exist, albeit masked by the higher I-score of the entire variable set. That is, suppose a variable set composed of 3 variables, $\{X_1, X_2, X_3\}$ is returned by the PR backwards dropping algorithm as highly predictive. While the three-way interaction is clearly important, the variable set may be masking (slightly less) influential interactions at the lower order; any two-way combination of the three variables or their marginal effects may also be important. Consequently, when incorporating the returned variables and variable sets into our model of choice for prediction, we should include all combinations of lower-order interactions as well. While this will result in more parameters to estimate, a) we have already heavily reduced the number of parameters, compared to the parameters for the original universe of possible variables and their interactions, and b) we can easily harness common models such as lasso or other shrinkage and selection methods for our prediction modeling stage.[9]

3.6 Simulations

Simulations are presented in this section to illustrate the PR's VS ability compared against other favored approaches such as lasso or random forests. Five models of

[9]This paper focuses on the VS aspect of prediction and leaves the equally important aspect of model selection outside the scope of the paper.

data are introduced with differences in both the existence of information amongst interactions and whether or not the data is linear. For the first four models, the outcome variable is y and there are 12 possible x explanatory variables, $\mathbf{x} = \{x_1, x_2, ..., x_{12}\}$. Model 1 is linear and additive, where the influential variables are $\{x_1\}$, $\{x_2\}$, and $\{x_3\}$. Model 2 is linear and interactive, where the influential variable set is $\{x_1, x_2, x_3, x_4\}$. Model 3 is a nonlinear model without interactions where three variables are influential, $\{x_1\}$, $\{x_2\}$ and $\{x_3\}$. Finally, Model 4 is a nonlinear and interactive model where $\{x_1, x_2, x_3\}$ are influential together. The remaining variables are always noisy and unrelated to the outcome y. For the fifth model, the outcome variable is y, and there are 20 possible x explanatory variables. The variable set $\{x_1, x_2, x_3\}$ is important, while the remaining 17 variables are noisy. This last model is designed to illustrate how the different VS approaches might fare as the dimensionality grows while the sample size (n) remains constant. The n size for each simulation is 100 observations.

Below are the five model specifications:

1. Linear additive:
$$y = 0.2x_1 + 0.5x_2 - 0.1x_3$$

2. Linear interactive:
$$y = x_1 * x_2 * x_3 * x_4$$

3. Nonlinear:
$$y = (x_1 + x_2 + x_3) \mod 2$$

4. Nonlinear interactive:
$$y = (x_1 * x_2 * x_3) \mod 2$$

5. Nonlinear interactive (with 20 possible variables):
$$y = (x_1 * x_2 * x_3) \mod 2$$

I simulate 100 datasets for each model and conduct VS on each dataset for every model using lasso, random forests, and evaluating variable set I-scores. The VS task for each

approach is to retrieve the true influential variable sets amongst the remaining noisy ones. To analyze the simulated data with lasso and random forests, I use the open-source R packages "glmnet" and "randomForests". As VS is conducted within the models for these two approaches, some work must be done to retrieve the variables found to be important by lasso and random forest.

For lasso, this is more straightforward. All variable sets are introduced into the lasso model. In the case of Models 1 through 4, this is the set of all possible variable interactions amongst the 12 variables — a total of 4,095 unique variable sets. For the last model, this is a total of 263,949 unique variable sets if we restrict our variable sets to up to eight-way interactions. Lasso carries out VS by simply penalizing the parameters of variable sets that are discarded to zero (after 10-fold cross validation within the dataset to find the appropriate tuning parameter value). Variable sets for which parameters are not penalized to zero are considered important and kept as predictors.

For random forests, if the focus is prediction, variable importance can be measured through the "mean decrease in gini" results from each random forest model run. This measure is used for prediction using out-of-bag (OOB) samples, where the jth variable's importance is measured by random permutation of the variables in the bth tree. For details, see Hastie et al. (2008, p.503). Unfortunately, as of the writing of this paper, there are no obvious approaches to conducting the same type of variable set importance ranking for neural networks; as such, the VS comparisons in the simulations do not present neural networks.

The PR approach is also presented. As discussed, the I-score can be calculated for each variable set under consideration. The I-scores of different variable sets, given a common dataset, can be directly compared against one another; searching for the largest I-scores should return the most influential variable sets. For a full replication of the simulation, see simulation details in the Appendix.

3.6.1 Lasso VS

We consider how well using the lasso for VS in the first four models fares. Here VS is amongst the 12 candidate variables and all their interactions. I conduct lasso VS among all possible variable sets in Models 1-4. For Models 1 and 4, the lasso

retrieves the correct variables and variable sets, $\{x_1\}$, $\{x_2\}$, and $\{x_3\}$ for Model 1 and $\{x_1, x_2, x_3\}$ for Model 4, for all 100 simulations. Models 2 and 3 show some variation in VS, however. These are presented in Figures 3.2 and 3.3, respectively. The x-axes span the 100 simulated datasets. The y-axes are a list of any variable sets retained in any of the 100 simulations. White cells indicate that the variable sets were penalized to zero in the corresponding simulations. For instance, in Model 2, the variable set $(x_1 * x_2 * x_3 * x_4 * x_8 * x_{12})$ was penalized to zero and not selected in simulation 100. Red shaded cells indicate that the variable sets were kept in the corresponding simulations and had positive coefficients (with darker shading for larger coefficients). Blue shaded cells indicate that the variable sets were kept in corresponding simulations, but had negative coefficients (again with darker shading for larger, more negative coefficients). Correctly selected variable sets are highlighted in yellow throughout.

In Model 2, the lasso manages to select the correct variable set, $\{x_1, x_2, x_3, x_4\}$ (or (x_1, x_2, x_3, x_4) in Figure 3.2), through a good number of the simulations but also selects six other variable sets with similar frequency throughout the simulations. In Model 3, the nonlinear model, many variable sets are retained (see Figure 3.3). The top 50 variable sets are shown (only a fraction of the lasso-retained variable sets); the approach seems to have difficulties in finding the truly influential variables ($\{x_1\}$,$\{x_2\}$, and $\{x_3\}$). The 50 top variable sets span many noisy variable sets, are not consistent across the 100 simulations, and do not include x_1, x_2, and x_3 as individually selected variables.

For Model 5, the lasso is unable to run on all possible variable sets. Reducing to eight-way interactions (263,949 variable sets) is still problematic due to (1) memory insufficiencies in an average home PC desktop and (2) an inability to estimate the model with an overly small number of observations n compared to the number of variable sets (here we can consider these as parameters p, and in this case, $\frac{n}{p} = \frac{100}{263{,}949} = 0.00038$). The lasso thus faces some difficulties in conducting VS with interactions and nonlinear models, though is promising with linear models. It can also can face some computational ceilings when $\frac{n}{p}$ becomes too small.

3.6.2 Random forests VS

We consider VS with random forests. Figures 3.4,3.5,3.6, and 3.7 presents heat maps of the random forests variable set importance measure "mean decrease in gini" across the first four models. Note, these maps have slightly different interpretations with regard to their heat colorings when compared to the lasso VS. Here variable sets are ranked based on their variable set importance measures directly. Thus, the more red a cell appears, the *more important* random forest has determined the variable set to be for that simulated dataset, while the more blue the cell appears the *less important* the variable set. White cells are of medium importance. Note here only the top 50 variable sets (out of a total of 4095) are shown for presentation clarity. Again, the x-axes across the four Figures depict the 100 simulated datasets. The y-axes are the variable sets under consideration. Random forests appears to be able to select the important variables in Models 1 and 3, $\{x_1\}, \{x_2\}, \{x_3\}$ and $\{x_1\}, \{x_2\}, \{x_3\}$ respectively, and to discard the remaining noisy variables (see Figures 3.4 and 3.6, respectively). In Figure 3.5, which depicts Model 2, the linear interactive model, random forests retrieves the individual variables $\{x_1\}$, $\{x_2\}$, $\{x_3\}$ and $\{x_4\}$ as the most important. These are indeed the variables that are influential for the outcome, but the random forests approach does not indicate that these variables should be interacted as a four-way interaction in order to predict the outcome well. For Model 4 (see Figure 3.7), random forests selects $\{x_1\}$, $\{x_2\}$, and $\{x_3\}$ as individually important but does not indicate that these variables should be interacted to be influential for the outcome.

In Model 5, which is a scenario where the ratio of $\frac{n}{p}$ is quite small (0.00038), random forests appears to be able to select the individual important variables x_1,x_2, x_3 though does not return these variables as a set for the top three returned variable sets (see Figure 3.8).

3.6.3 Partition retention VS

We turn finally to the PR VS. The first four models are shown in Figures 3.9, 3.10, 3.11 and 3.12. All variable set I-scores were calculated for each of the 100 simulations and across all four models. The average top ranked 50 I-scores for each model are

presented in the heat maps. For instance, in Model 4 (Figure 3.12), the variable set with the highest I-score on average across all 100 simulated datasets was $(x_1 * x_2 * x_3)$ and is thus ranked as the top row. Incidentally, this is also the correct variable set to select for Model 4. The more red the cell appears, the higher the I-score for the corresponding variable set and simulation. The more blue the cell appears, the lower the I-score associated with that cell.

In Figure 3.9, which depicts Model 1, the top two variable sets include the variables (x_1), (x_2), and (x_3). While these appear in a two-way and three-way interaction variable set, as noted earlier, in the PR approach the backwards partitioning requires that lower orders of returned variable sets are kept as well, as these might be influential, and merely appear less influential than the higher order in the sample. Thus the PR approach would retrieve the individual variables, (x_1), (x_2), and (x_3), which are the truly influential ones for Model 1. The same holds for Model 3 (Figure 3.11) — the top retrieved variable set is $(x_1 * x_2 * x_3)$, which when allowing for retention of lower order variable sets, also returns the truly influential variables (x_1), (x_2), and (x_3).

In Model 2 (Figure 3.10), the top returned variable set across simulations, based on I-score ranking is $(x_1 * x_2 * x_3 * x_4)$, which is the truly influential variable set. Figure 3.12, which shows Model 4, likewise depicts the PR returning the truly influential variable set $(x_1 * x_2 * x_3)$ as the top variable set across the simulations.

In Model 5, the I-scores of variable sets up to eight-way interactions (for a total of 263,949 variable sets) were calculated. The average top ranked 50 I-scores are presented in Figure 3.13. The first ranked variable set is $(x_1 * x_2 * x_3)$, which is the true set of important variables. The PR VS approach thus reliably returns the important variables and variable sets throughout linear, nonlinear and interactive scenarios.

Finally, I provide a depiction of the I-score dropping process across the five models in Figures 3.14, 3.15, 3.16, 3.17, and 3.18. In each of the models, the I-score is plotted along the y-axes, while the x-axes are the variable sets under consideration. The boxplots depict the distribution of I-score values of the given variable set across the 100 simulated datasets. For instance, in the first model depicted in Figure 3.14, the variable set "$x1 - x3$" (or $\{x_1, x_2, x_3\}$) has an average I-score across the simulated datasets of about 14. This is the highest I-score in the graph and also corresponds to the group of

truly influential variables for the model. For each model, regardless of functional form or the existence of interactions, the highest I-score as variables are removed from the starting set of all 12 variables identifies the group of influential variables for each model (the fifth model has a starting set of 20 variables but is also able to identify the group of influential variables for the model). Finally the I-score dropping process is presented for Model 5 in Figure 3.18, where the highest average I-score is also at the correct variable set "$x1 - x3$".

It seems the I-score can comfortably select correct influential variables and is able to distinguish these variables apart from noisy ones. When the model does not include interactions and nonlinearities, lasso and random forests are also reasonable VS approaches. Random forests and PR can conduct VS when $\frac{n}{p}$ decreases to 0.00038, but lasso faces some difficulties at that stage. To further illustrate the PR approach, we turn to a real data application on civil war onsets in the next section.

3.7 Application: civil war onset

In this section, I analyze the Fearon and Laitin dataset from their 2003 paper on civil wars using the PR approach as well as some comparison models, including Fearon and Laitin's main specified model. Ward et al. (2010) used the same data gathered by Fearon and Laitin (2003) as well as Collier and Hoeffler (2004) to demonstrate the poor predictive abilities of the models built from statistically significant variables. I build on Ward et al.'s work to show: first, high statistical significance does not necessarily lead to high predictivity. I discuss this phenomenon in further detail in Lo et al. (2015). In the results section to follow, I show that a model including a PR VS stage to predict compares favorably with Fearon and Laitin's theoretically motivated main specified model, increasing out of sample correct prediction rates from 78.98% to 98.05%. Second, important information helpful towards predicting may be lost when only using marginal effects of independent variables. I present top returned variable sets in the PR VS stage to illustrate the highly interactive and nonlinear effects of the explanatory variables on civil war onset.

The original Fearon and Laitin dataset included a total of 82 variables. After

removing variables that contained more than two thirds missing values, 39 variables were included in the variables list including the dependent variable for civil war onset, "onset". I used median imputation on the remaining variables with missing values.[10] I create lagged versions of up to five periods of variables that change over time (for instance, country-years coded as 1 for "Asia" do not change over time). This creates a total of 96 variables, inclusive of the onset dependent variable (see Tables 3.1, 3.2 and 3.3). If we wish to search amongst all variables as well as interactions up to say, 8-way interactions between variables, then this amounts to evaluating 121,550,931,645 variable sets. Given the size of the sample, $n = 6610$ observations, and the need to create folds of data with testing data set aside to evaluate our predictions without overfitting, we can already see that the variable space to explore is quite overwhelming and certainly vastly larger than the n size.

The number of values each variable can take directly factors into how much the data can be partitioned across a variable space. I have recoded continuous variables so they take a maximum of 3 possible discrete values (here, 0, 1 and 2) in order to keep the partition space manageable for the VS process. While this surely removes some information from each continuous variable, the retained information boosts the number of variables we can analyze and the number of interactions we can consider. The recoded variables are only used in the VS process. After identifying groups of variables, the researcher can turn to the original variable forms when constructing final predictors for the prediction process.

In order to discretize continuous variables, I used k-means clustering, where $k = 3$. For instance, the variable "popl", which is lagged population, is a variable that can take any positive integer value.

Many papers have approached predicting civil wars using a training set for in-sample variable selection and prediction and leaving aside an independent testing set to find an out of sample prediction rate. Cross-validation is a common approach for tackling the problem of model overfitting (Geisser 1993, Kohavi 1995) and is a desirable way to identify out-of-sample prediction rates. Currently the literature is short on a

[10]Imputation is a highly important and well-studied field in and of itself but I do not venture into this here and simply use median imputation for ease of presentation. Converting to mean imputation produces similar outcomes.

widely agreed upon cross-validation techniques for time-series or panel data, however.[11] For the purposes of this analysis, training data always occurs prior to testing data in each of the three folds so that no future information is used in the training portion of the analysis. This exercise attempts to mimic the "real world" where the researcher might have data on the past years at hand and wishes to predict the future. I divide the data into three sequential, but overlapping folds, so that there are three training sets and three corresponding testing sets (see Table 3.10)[12]. Fold 1 is composed of a training set spanning the years 1945-1984 inclusive, with a testing set of 1985-1986. Fold 2's training set spans 1952-1991 while the testing set is from 1992-1993. Fold 3's training set covers 1958-1997 while 1998-1999 serve as the testing data. This is to allow for a little more confidence in the reported out of sample error rates as three separate independent testing sets are laid aside to conduct the testing errors. The average out of sample error rates are taken across the three testing set folds.

For each of the training sets, I run the backwards dropping algorithm (PR steps 1 through 4) 100,000 times with a maximum starting variable set size of 8. This is equivalent to allowing for capture of up to 8-way interactions, or 8-variable sets. The starting partition size is thus $3^8 = 6,561$ partition cells. Sample size and computational power constraints determine the maximum starting variable set size in the backwards dropping process. Given the training set sizes in each fold, choosing from 8 to 10 as maximum variable set sizes are feasible. I choose 8 ultimately as this made the processing time faster (dropping from nearly 3 hours per training set to 30 minutes on a home PC desktop) and also because the top returned influential variable sets were very similar when using 8, 9 or 10 starting variable set sizes.

I retain variable sets with the top normalized I-scores to create predictors. These predictors are then used on the testing data in a logistic lasso regression with onset as the outcome variable.[13]

[11] The question of whether to validate the model or the variables in time series dependent data comes to play in a way that does usually plague the researcher when conducting cross validation on otherwise iid sample data.

[12] Since the onset variable appears as 1 in only roughly 10% of the sample, we require a large enough n size in both training and testing sizes for each fold in order to have a reasonable number of onsets equal to 1. Three folds were chosen for this reason as increasing folds after this quickly decreases the number of onsets across folds to minimal or no onsets.

[13] I use logistic lasso as my model. Logistic regression is a common binary choice model for the civil

3.7.1 PR Variable Selection results

Table 3.5 displays the top returned variable sets from each of the three training sets, ranked from row 1 to row 5 in the Table where row 1 has the highest I-score. Recall that this is the result of the novel variable selection stage I propose as helpful in prediction efforts. Using the PR's I-score and backwards dropping from randomly drawn 8-variable groups, I identify variable sets with the highest I-scores and keep these as variable sets of interest for prediction. Most if not all of the variables are unsurprising; after all, the population of variables collected by Fearon and Laitin were specifically meant to either control for or be significantly associated with civil wars. However, what is perhaps most interesting is that the top returned variable sets are all composed of groups of variables which suggests that there is information from higher order interactions and nonlinearities in how our explanatory variables are related with civil war onset. This is due to the highly complex nature of social phenomena like conflict, of which previous methodologies have not fully accounted.

While the civil war literature stresses the importance of how new a state might be or how fragile (as measured in the Fearon and Laitin data through variables like "nwstate" and "instab") on the probability of seeing a civil war onset, what Table 3.5 tells us is there are important nonlinearities in how these variables affect onset. Indeed, Hegre et al. (2001) note the importance of the "newness" of a state in affecting the potential for civil war outbreak. That some version of the "nwstate" variable ("nwstate", followed by "l2" to denote lagged 2 years, for example) is present in 11 out of the 15 variable sets in Table 3.5 provides some evidence into the nonlinear and time-dependent ways in which the proximity to independence of a country can affect the onset of civil war. Likewise, variations of the nwstate variable seem to appear frequently in these variable sets with

war literature. Because of the inclusion of all lower orders of each higher-order influential interaction from the backwards dropping process however, we are left with a slightly large number of parameters to estimate (though this is still orders of magnitude smaller than the number of parameters needed to estimate all variable interactions). The lasso is a common shrinkage and selection approach (see Hastie, Tibshirani, and Friedman 2008 and Tibshirani 1996). I also try to estimate the lasso on a comparison model to demonstrate that it is not the lasso itself that brings higher prediction rates, but lasso on all higher order interactions up to 8-way interactions is unable to run due to memory insufficiencies and overly small $\frac{n}{p}$. Reducing to 4-way interactions still faces the same problems. I discuss this further in the "Feasibility and Costs" section. Correct prediction (and error rates) rates are found for each fold's testing sets, using 0-1 loss.

variations of the instability ("instab") variable. This further suggests that not only are the two variables nonlinearly affecting the onset of civil wars differently over time, they ought to be considered together. The interested researcher could also certainly take this suggestive evidence as preliminary material for developing a theory that considers how a country's proximity to independence and its recent history of instability affect the onset of civil war.

Similarly, Fearon and Laitin (2003) argue that an insurgent group's ability to mount an insurgency is key in whether a civil war occurs. One of their hypotheses postulates that increases in proxies for strength of the insurgent band should increase the probability of civil war. These proxies include: newly independent states ("nwstate"), political instability at the center ("instab"), a regime that mixes democratic and autocratic features ("anoc"), and a large country population ("pop"). The authors test these variables individually and find that these variables are all significantly associated with civil war. What I find is that these variables (short of the population variable) all appear in interacting ways to predict civil war onset, as evidenced by their appearances across many of the variable sets in Table 3.5. Lagged variations of these variables are picked up by the top variables but that these variables return in groups suggests that they likely work together to predict civil war onset, and not merely on their own. Interestingly, Fearon and Laitin find that the percent of Muslims in each country ("muslim") is not significantly associated with civil war onset and thus leave out this variable in their main regression specification. I find that the "muslim" variable appears in various lagged forms in nine of the fifteen top variable sets across the three training sets, but only in conjunction with other variables. As these top variable sets demonstrate, it is highly possible that the percent of Muslims in each country nonlinearly and interactively predict civil war with other variables. It is possible that the variable carries little to no marginal information but requires interaction with other variables such as "new state" and "instability", which always appear alongside "muslim" in the table.

In this way, we might be able to take the exercise of variable selection as both an important step to prediction with high dimensional data, but also as a preliminary theory-building step. After conducting variable selection via PR, we turn to modeling.

3.7.2 Prediction results

For each of the three fold training sets, I incorporate returned influential variable sets into model form and test out-of-sample prediction error rates. Here each training set and its associated set of influential variables (found through PR) are subjected to the logistic lasso with a 10-fold cross validation within the logistic lasso model in order to determine the tuning parameter value that minimizes training error. No testing sets have been touched at this stage. I refer to this main model as "PR Lasso" henceforth. This is because a model is required after VS by PR, and the PR approach itself does not dictate a model. As previously noted, since the PR-returned variable sets allow for the possibility of lower order interactions (that have possibly slightly less signal than the higher-order variable set returned), it is prudent to account for all subsets of variables in a returned variable set. Again, the reduction in variable sets, and thus dimensionality, should be significant enough to more than make up for the extra estimation efforts required for the lower-order variable sets. It is thus reasonable to apply a model that has a penalizing element such as lasso.

I also run the following comparison models: random forests ("RF"), neural networks ("NN"), lasso on the variables selected in Fearon and Laitin's main model (see Fearon and Laitin 2003; "FL lasso"), and a logistic regression on the variables selected in the Fearon and Laitin model ("FL logistic"), which is a replica of Fearon and Laitin's main model applied to each of the three training data.

Feasibility and costs

I briefly discuss the feasibility and costs of running each of the prediction models. All computing times are considered for a standard Dell PC home desktop. The PR lasso requires first conducting PR VS (here on up to 8-way interactions), which for each fold requires 15 minutes of computing time, followed by the prediction model itself which requires a couple minutes per model. Neural networks ("NN") and random forests ("RF") conduct VS and model fitting at the same time and require roughly 10-15 minutes for each model. The "FL logistic" model similarly requires very little computation time (only a few minutes), in part because the model only estimates eleven coefficient parameters.

The reader might be interested in the performance of lasso on all variable sets as a relevant comparison to the PR lasso model. The lasso model on all variable sets up to 8-way interactions (to mimic the order of interactions PR can handle) is unable to run due to the overly small proportion of sample size to parameters estimated (p is too much larger than n; $\frac{n}{p} = 3.295\text{x}10^{-8}$). Reducing the order to 6-way interactions limits the parameters somewhat ($\frac{n}{p} = 4.729\text{x}^{-6}$) but lasso is still unable to run.[14] Thus, the lasso model is excluded in the application due to an inability to estimate the model. This should not be overly surprising, as Model 5 in the Simulations section tells us that when $\frac{n}{p} = 0.00038$ the lasso already starts running into a computational ceiling. Thus, a shortcoming to the lasso approach is that it seems to buckle under computation power constraints earlier than random forests, neural networks and PR. When $\frac{n}{p}$ is of reasonable size and the true data generating process is not extremely nonlinear, lasso can be very good for prediction.

Results

Table 3.6 presents the error rates across the different models and three testing data sets. Recall that VS and model training are all conducted on the training data; testing data are set aside to determine out of sample prediction error rates. For each of the three folds, the lowest error rates are boldfaced; what should be noted is that the PR Lasso always does quite well in that it has the lowest prediction error rates across the three folds.

The PR approach has the lowest average error rate of 1.95% (or 98.05% correct prediction rate) for predicting the onset of a civil war. Random forests and neural networks also perform reasonably well, with slightly higher average error rates of 2.72% and 2.70% respectively. On the other hand, the average error rate across the three folds for the "FL Logistic" model, which recall is the model that mirrors Fearon and Laitin's main specified model, applied to each of the training data) is 26.02% (average correct prediction rate=73.98%). The "FL Lasso" model the error rate is 30.13% and the average correct prediction rate is 69.87%. What is clear is that using significance as a

[14]The models were run in R on a standard 64-bit PC (Dell) home desktop. The memory required for the matrix sizes also greatly exceeded the capacity available.

criterion to variable select (both FL models) leads to poorer prediction performance.

In Table 3.7, I break down the errors into false positives and negatives. False positives occur when the model predicts civil war onset when no onset occurs, while false negatives are when the model predicts no onset of civil war when in reality an onset did occur. The PR model produce the least false positives and false negatives across the three folds, while the FL lasso and FL logistic models produce the most. While the FL lasso and FL logistic also produced fewer false negatives in Fold 2 than the remaining four models, the total number of mistakes (false positives and false negatives together) the two models make are nearly tenfold that of the PR, RF, and NN models.

3.8 Conclusion

Predicting civil war onsets requires both identifying highly influential variables and variable sets and selecting good models that minimize prediction errors in out-of-sample testing. Since the main goal is minimizing error rates, and not establishing significance of key independent variables as in the case of theory-building and testing, we should approach predicting civil wars differently. Current efforts towards prediction often turn to theoretical work on the causes of civil war as a method of identifying variables to use in models for prediction. Selecting significant variables for prediction does not seem to automatically lead to better predictions, however. I argue in this paper that it is in fact the universe of collected independent variables — *and all of their possible interactions* — that compose the full set of possible variables to consider for prediction. This approach to finding variables and variable sets is more appropriate for a prediction-oriented goal.

However, given the large number of variables and their interactions, selecting for influential variables without using significance as a criterion and taking into account higher order interactions is a difficult feat. This paper proposes applying the PR method as a possible candidate for these two goals and illustrates the method using simulations and an application to Fearon and Laitin's 2003 civil wars data. Results from the simulations and data application are promising. In the simulations, the PR I-score is able to locate influential variable sets even when the underlying data generating process is

nonlinear and interactive. Other favored approaches such as random forests and lasso appear to perform well, but are less reliable when the data is both nonlinear and interactive. In addition, the lasso begins to break down as the number of parameters p increases too much compared to the number of observations n ($\frac{n}{p}$ decreases substantially).

In the data application, compared to similar models that use significance-based selection of variables, the model that uses variable sets returned through PR searching ("PR lasso") improves prediction rates from 69.87% and 73.98% to 98.05%. The PR approach performs weakly better than well-known machine learning approaches such as random forests and neural networks. As such, it can be regarded as a candidate technique in predicting conflict and a helpful addition to the prediction toolkit.

The real data application conducted in this paper demonstrates the applicability of the PR method. It is highly likely that important existing variables not collected in the Fearon and Laitin dataset may also capture important information for the prediction of civil war. As our datasets grow in political science, researchers can and should consider the big data problems of dimensionality reduction as not simply baggage but also opportunity; it is very possible that large amounts of information can be found in these higher dimensions and that these interactions can be found using appropriate VS strategies.

3.9 Acknowledgements

This work benefited from the feedback and suggestions of Rachel Fan, Shaw-Hwa Lo, Margaret Roberts, Tian Zheng. Chien Hsun Huang suggested efficient programming. I am grateful to Héctor Pifarré i Arolas, Roy Allen, Maya Duru, Erin Giffin, Veena Jeevanandam, Michael Levy, Shanthi Manian, as well as the participants of the Human Nature Group workshop, UCSD Methodology workshop, and the Penn State New Faces of Methodology conference for commentary and discussion.

3.10 Tables and Figures

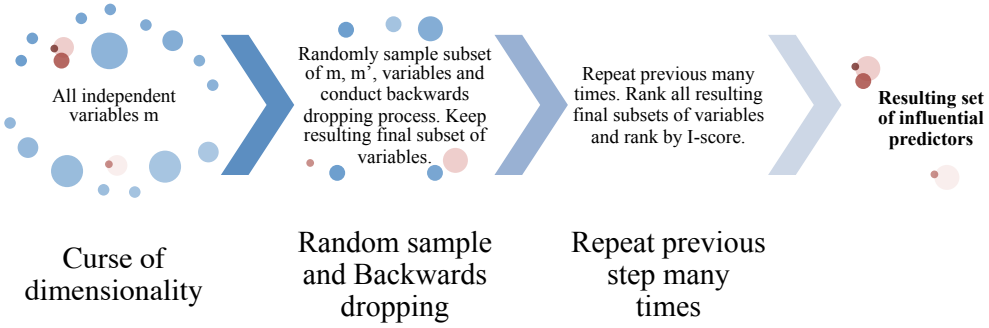


Figure 3.1: **Backwards Dropping Process**. The BDA process involves randomly drawing a subset of starting variables and calculating *I*-scores as random variables are dropped from the subset.

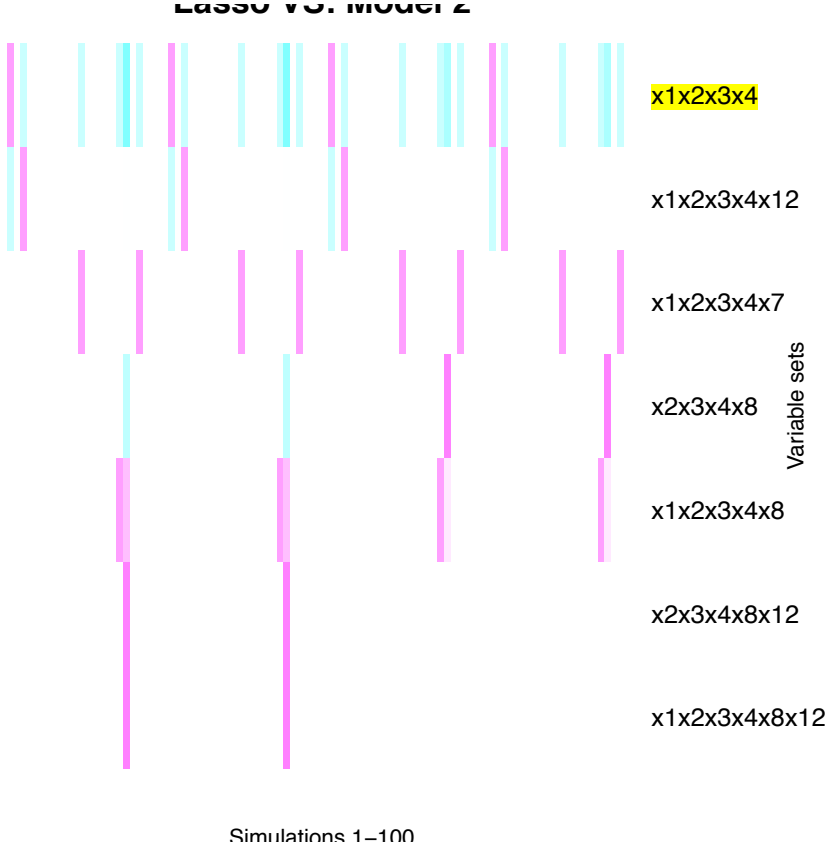


Figure 3.2: **Lasso VS for Model 2**. The x-axes represent the 100 simulations, and the y-axes depict the variable sets returned that were not penalized to 0 by the lasso. White cells indicate that some variable sets were penalized 0 in the corresponding simulations. Red cells indicate variable sets retained by lasso with positive coefficients; blue cells indicate variable sets retained by lasso with negative coefficients. The correctly selected variable set is highlighted in yellow.

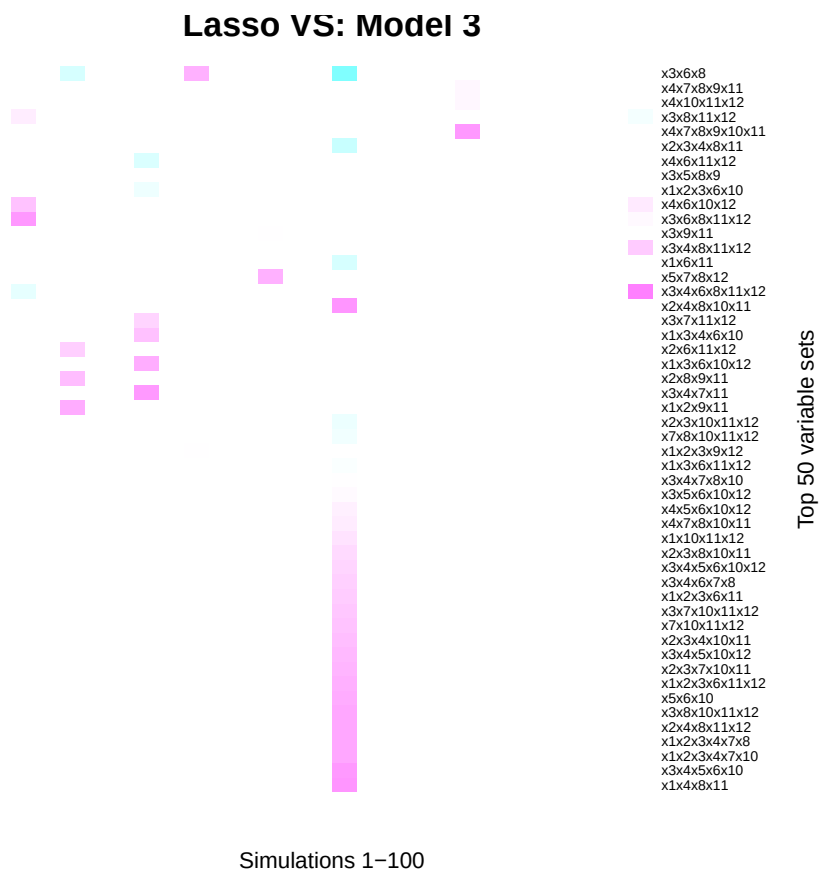


Figure 3.3: **Lasso VS for Model 3**. The x-axes represent the 100 simulations, and the y-axes depict the variable sets returned that were not penalized to 0 by the lasso. White cells indicate that some variable sets were penalized 0 in the corresponding simulations. Red cells indicate variable sets retained by lasso with positive coefficients; blue cells indicate variable sets retained by lasso with negative coefficients. No variable sets are highlighted as the correct variable set was not selected in the top 50.

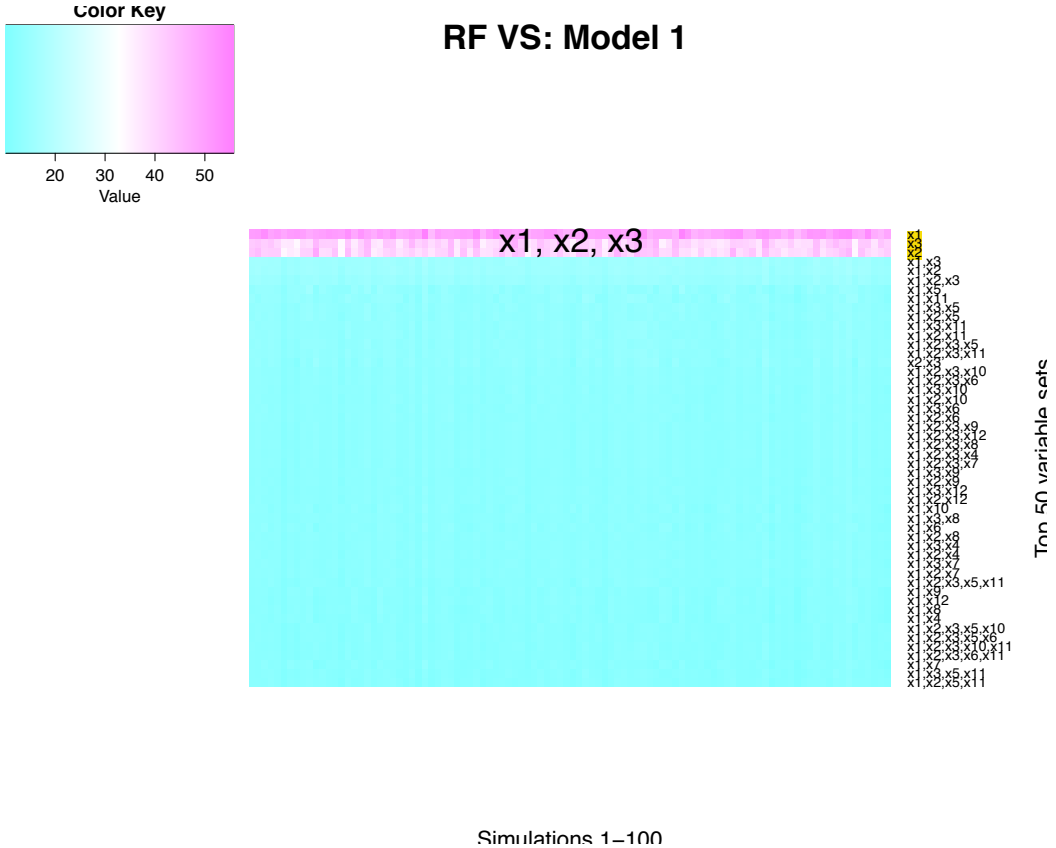


Figure 3.4: **Random forests VS for Model 1**. The x-axis depicts the 100 simulations, and the y-axis depicts the top 50 variable sets in descending order of variable set importance from the top to bottom. The correctly selected variable sets are highlighted in yellow.

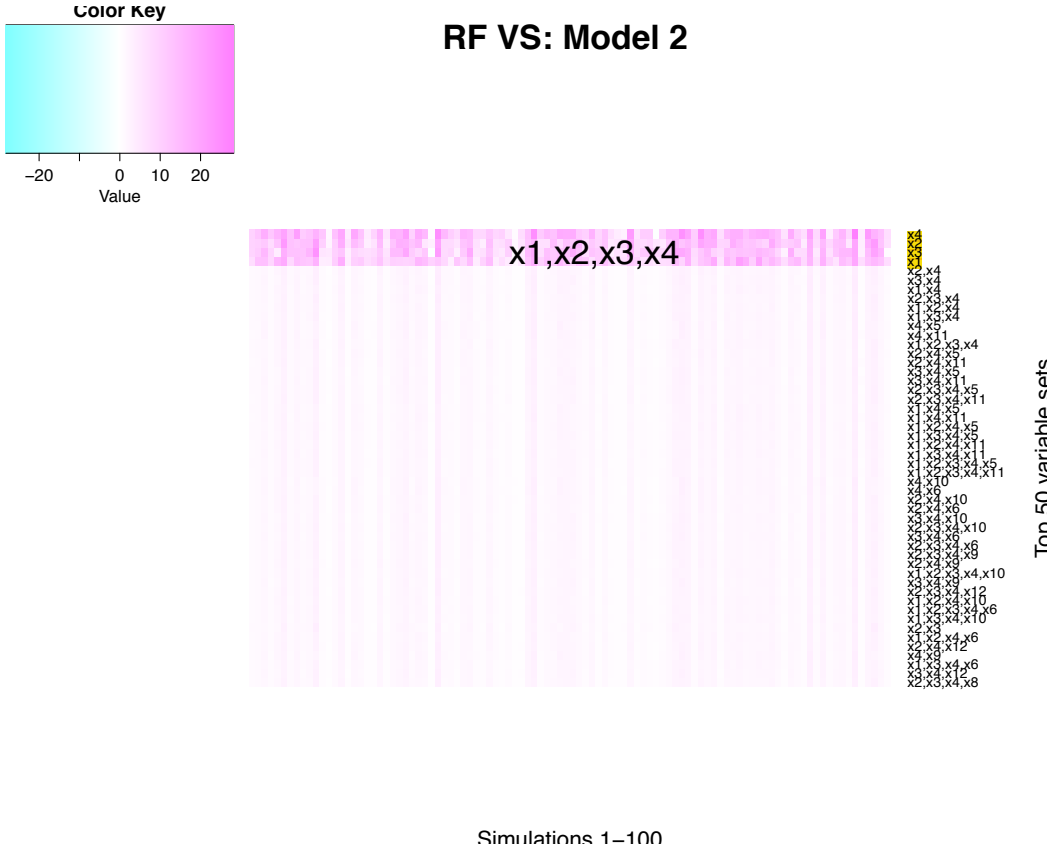


Figure 3.5: **Random forests VS for Model 2**. The x-axis depicts the 100 simulations, and the y-axis depicts the top 50 variable sets in descending order of variable set importance from the top to bottom. The correctly selected variable sets are highlighted in yellow.

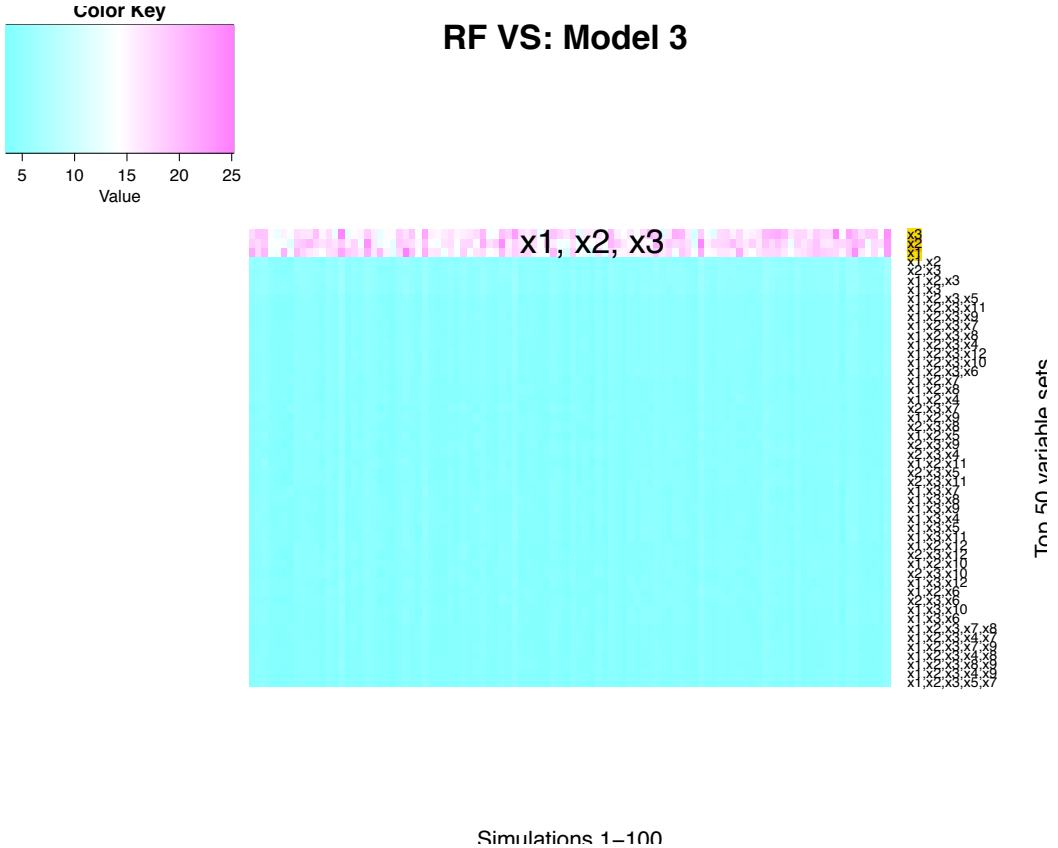


Figure 3.6: **Random forests VS for Model 3**. The x-axis depicts the 100 simulations, and the y-axis depicts the top 50 variable sets in descending order of variable set importance from the top to bottom. The correctly selected variable sets are highlighted in yellow.

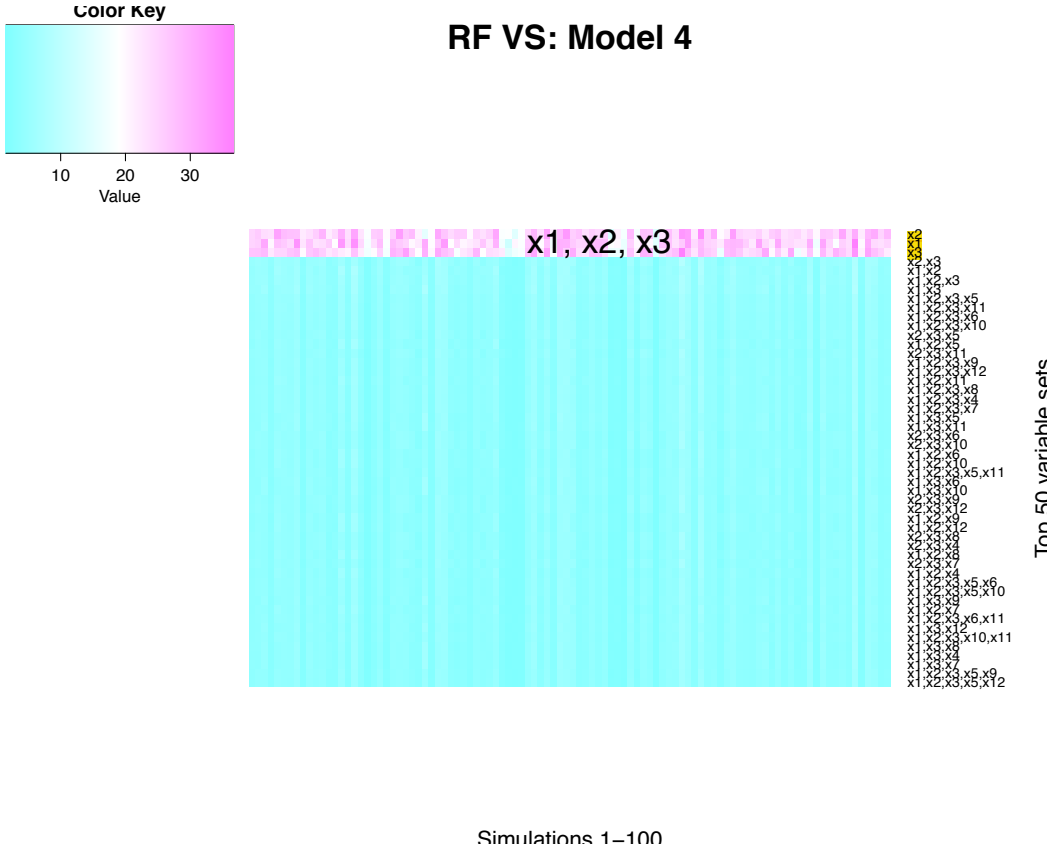


Figure 3.7: **Random forests VS for Model 4**. The x-axis depicts the 100 simulations, and the y-axis depicts the top 50 variable sets in descending order of variable set importance from the top to bottom. The correctly selected variable sets are highlighted in yellow.

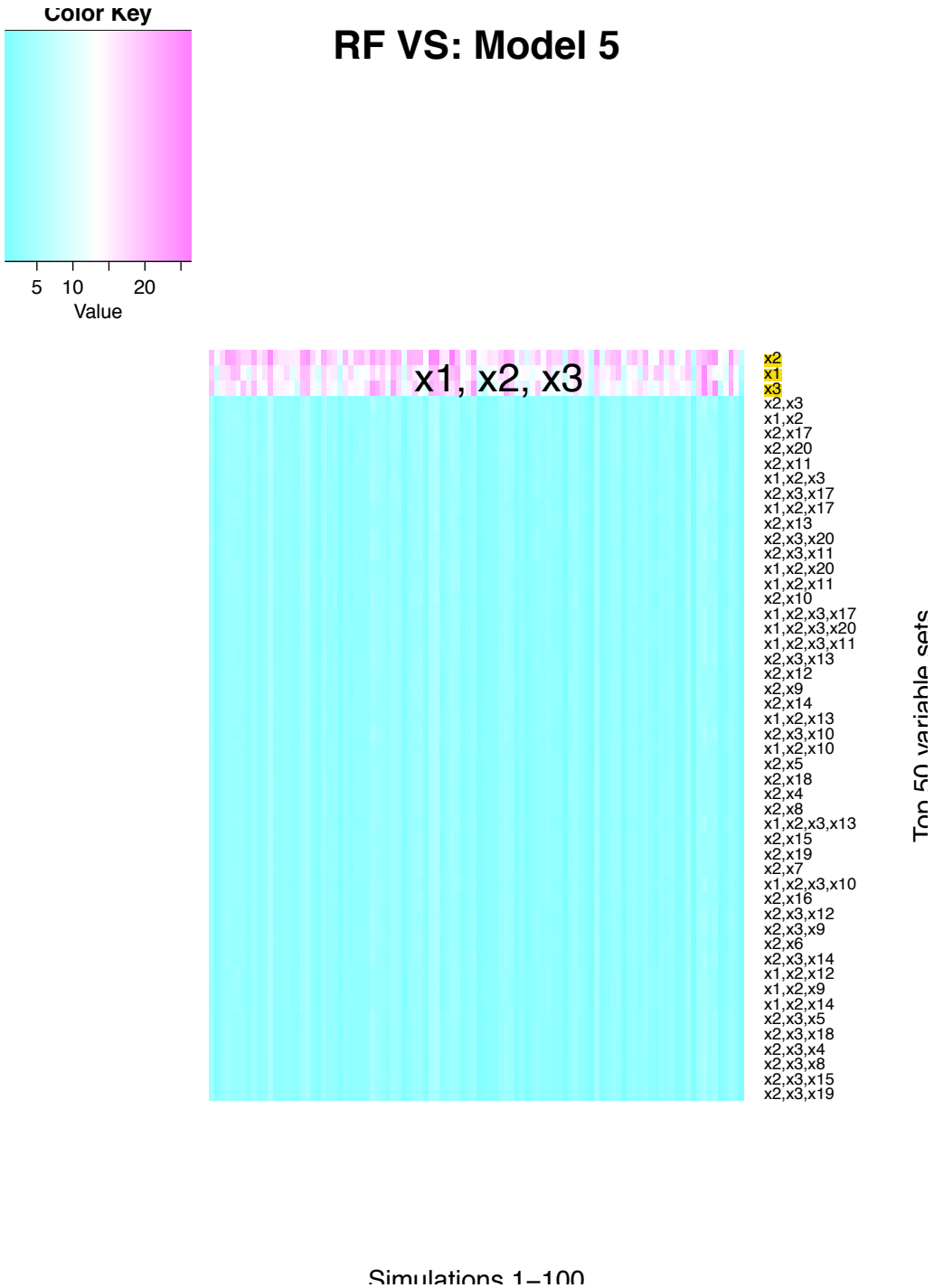


Figure 3.8: **Random forests VS for Model 5**. The x-axis depicts the 100 simulations, and the y-axis depicts the top 50 variable sets in descending order of variable set importance from the top to bottom. The correctly selected variable sets are highlighted in yellow.

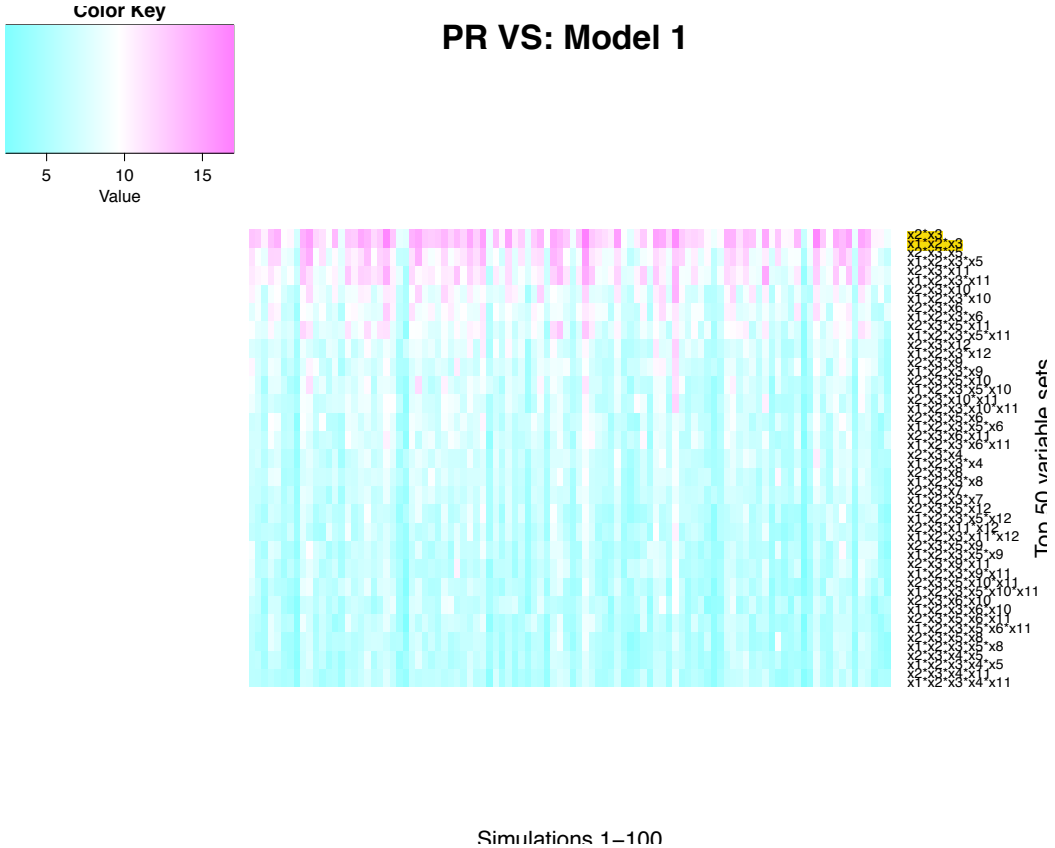


Figure 3.9: **I-score VS for Model 1**. Variables (x_1, x_2, x_3) are influential in Model 1. The x-axis depicts the top 50 variable sets ordered by I-score. The y-axis depicts the corresponding I-scores of each variable set. The correctly selected variable sets are highlighted in yellow.

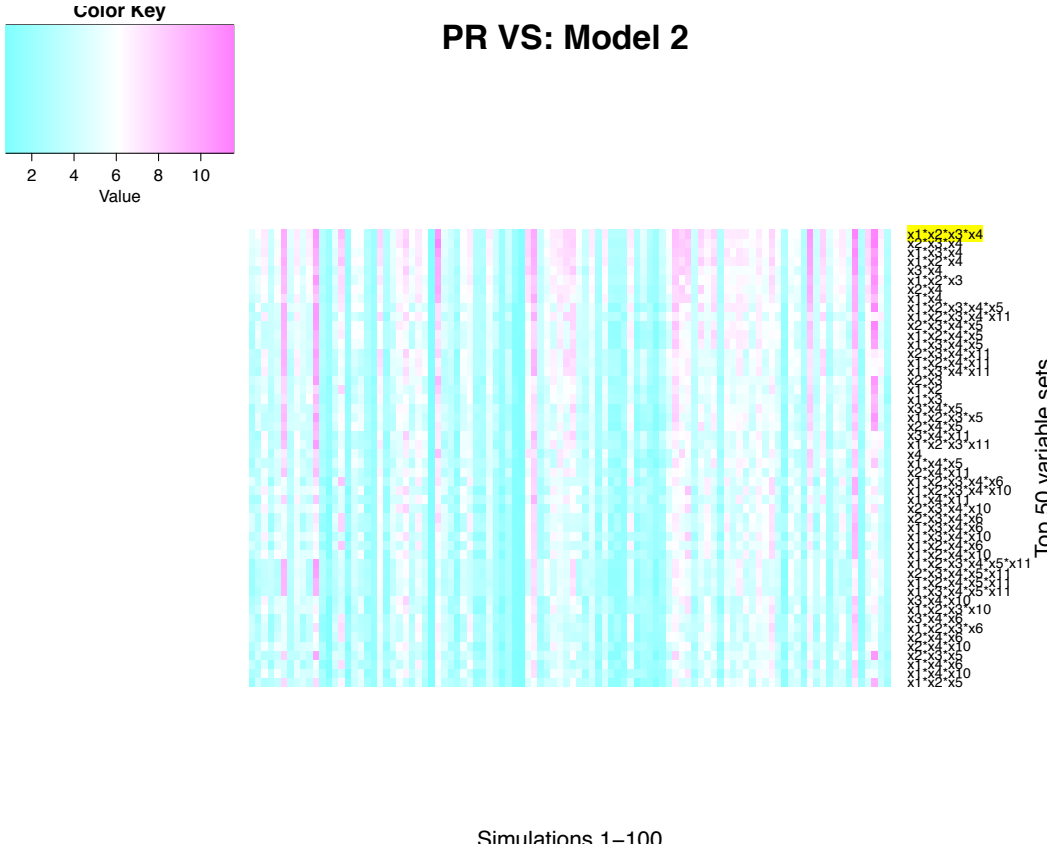


Figure 3.10: **I-score VS for Model 2**. Variable set (x_1, x_2, x_3, x_4) is influential in Model 2. The x-axis depicts the top 50 variable sets ordered by I-score. The y-axis depicts the corresponding I-scores of each variable set. The correctly selected variable set is highlighted in yellow.

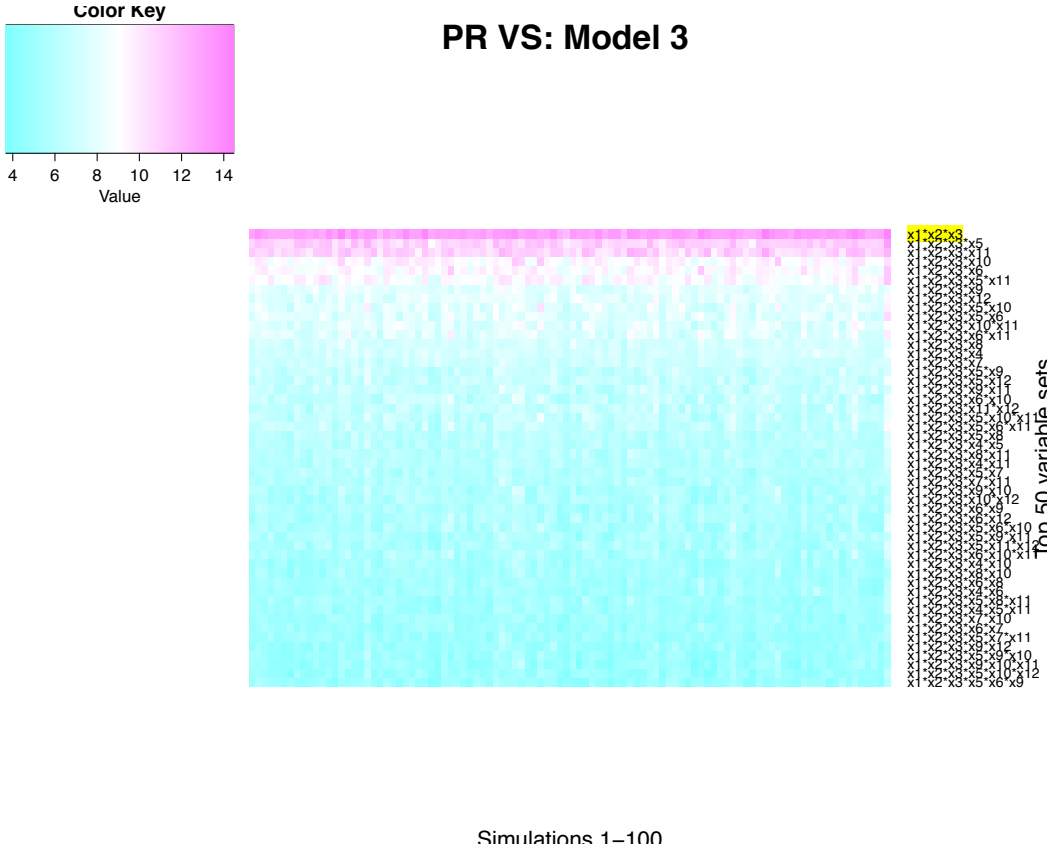


Figure 3.11: **I-score VS for Model 3**. Variables (x_1, x_2, x_3) are influential in Model 3. The x-axis depicts the top 50 variable sets ordered by I-score. The y-axis depicts the corresponding I-scores of each variable set. The correctly selected variable set is highlighted in yellow.

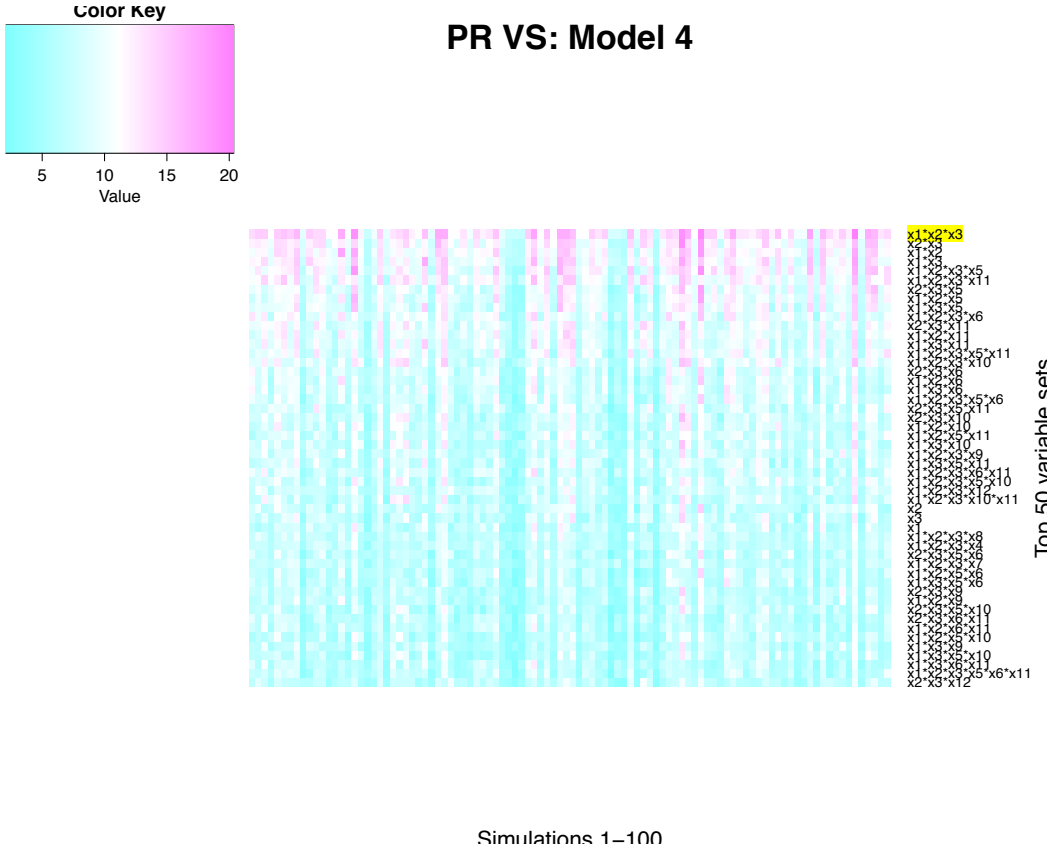


Figure 3.12: **I-score VS for Model 4**. Variable set (x_1, x_2, x_3, x_4) is influential in Model 4. The x-axis depicts the top 50 variable sets ordered by I-score. The y-axis depicts the corresponding I-scores of each variable set. The correctly selected variable set is highlighted in yellow.

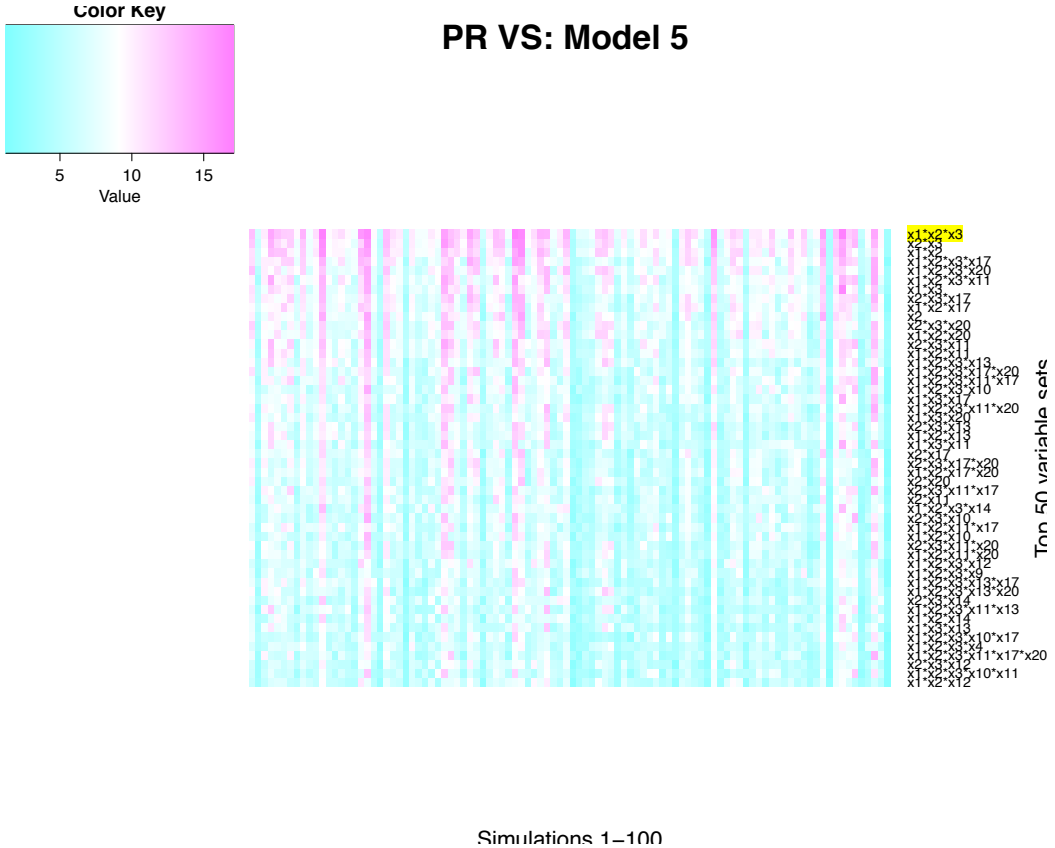


Figure 3.13: **I-score VS for Model 5**. The variable set (x_1, x_2, x_3) is influential in Model 5. The x-axes depict the top 50 variable sets ordered by I-score. The y-axes depict the corresponding I-scores of each variable set. The correctly selected variable set is highlighted in yellow.

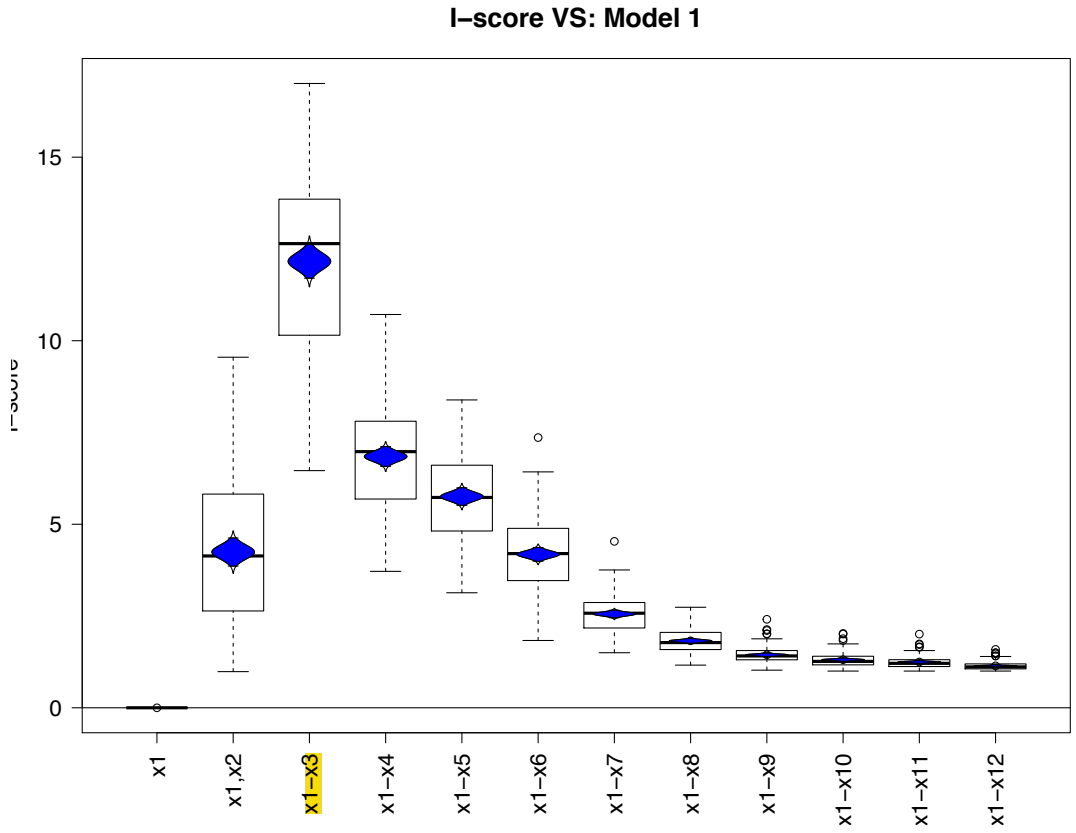


Figure 3.14: **I-score drop in Model 1**. Variables (x_1, x_2, x_3) are influential in Model 1. The x-axis depicts variable sets considered. The y-axis depicts the corresponding I-scores of each variable set. Boxplots indicate the distribution of I-scores for each variable set across simulations. The correctly selected variable set is highlighted in yellow.

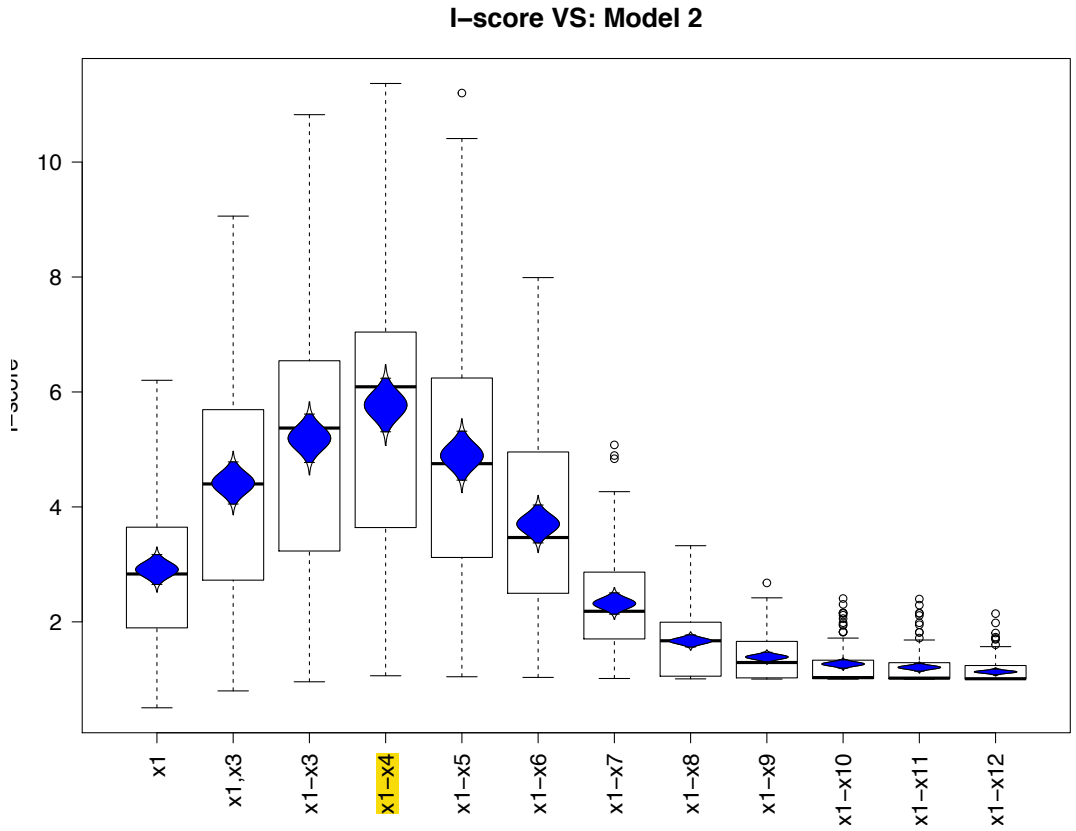


Figure 3.15: **I-score drop in Model 2**. Variable set (x_1, x_2, x_3, x_4) is influential in Model 2. The x-axis depicts variable sets considered. The y-axis depicts the corresponding I-scores of each variable set. Boxplots indicate the distribution of I-scores for each variable set across simulations. The correctly selected variable set is highlighted in yellow.

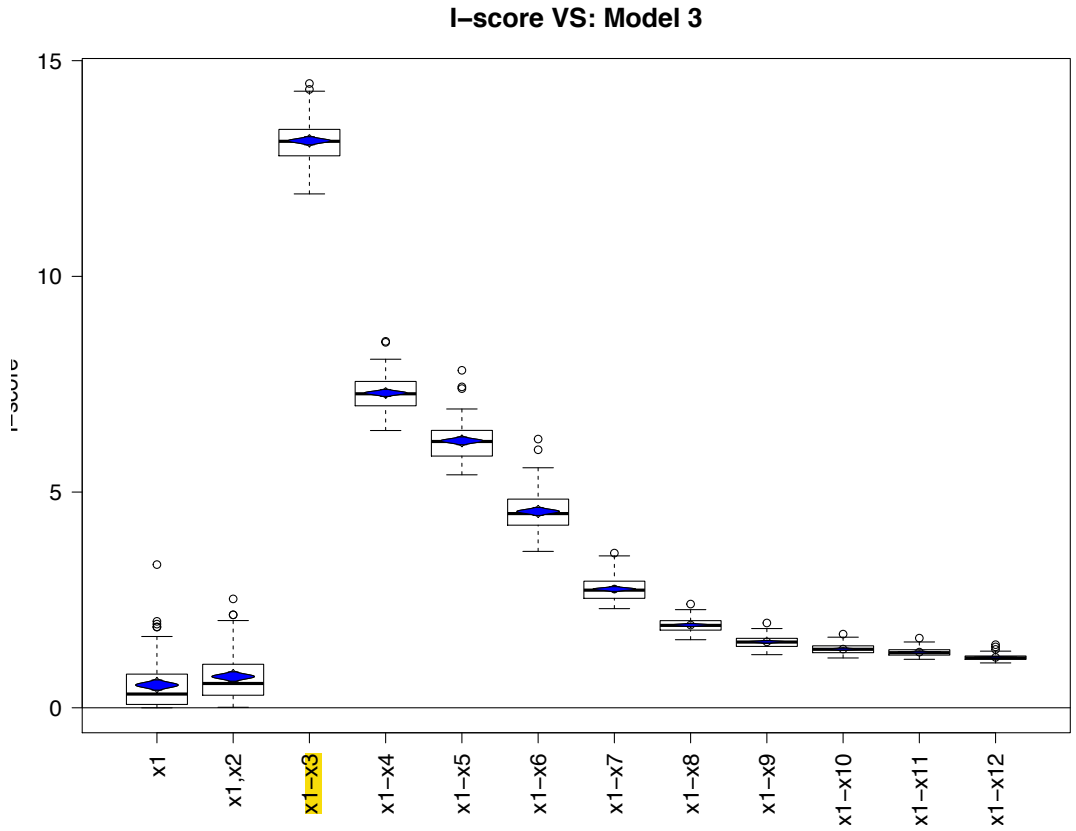


Figure 3.16: **I-score drop in Model 3**. Variables (x_1, x_2, x_3) are influential in Model 3. The x-axis depicts variable sets considered. The y-axis depicts the corresponding I-scores of each variable set. Boxplots indicate the distribution of I-scores for each variable set across simulations. The correctly selected variable set is highlighted in yellow.

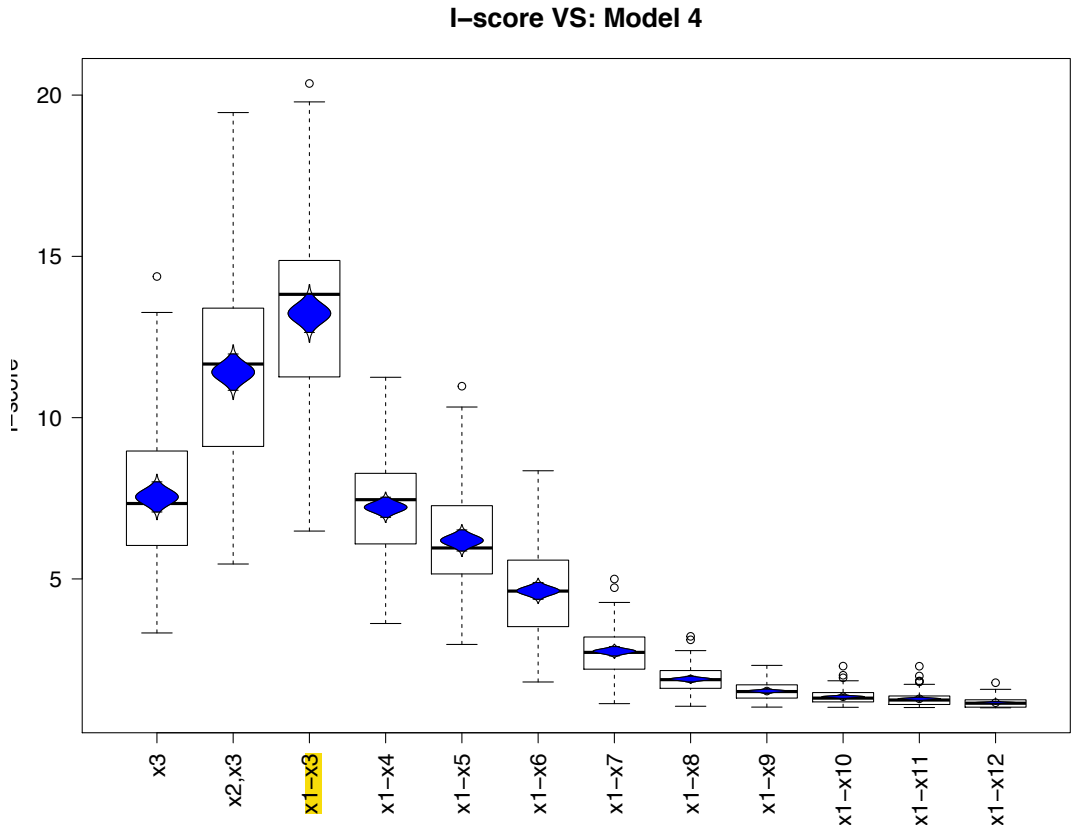


Figure 3.17: **I-score drop in Model 4**. The variable set (x_1, x_2, x_3) is influential in Model 4.The x-axis depicts variable sets considered. The y-axis depicts the corresponding I-scores of each variable set. Boxplots indicate the distribution of I-scores for each variable set across simulations. The correctly selected variable set is highlighted in yellow.

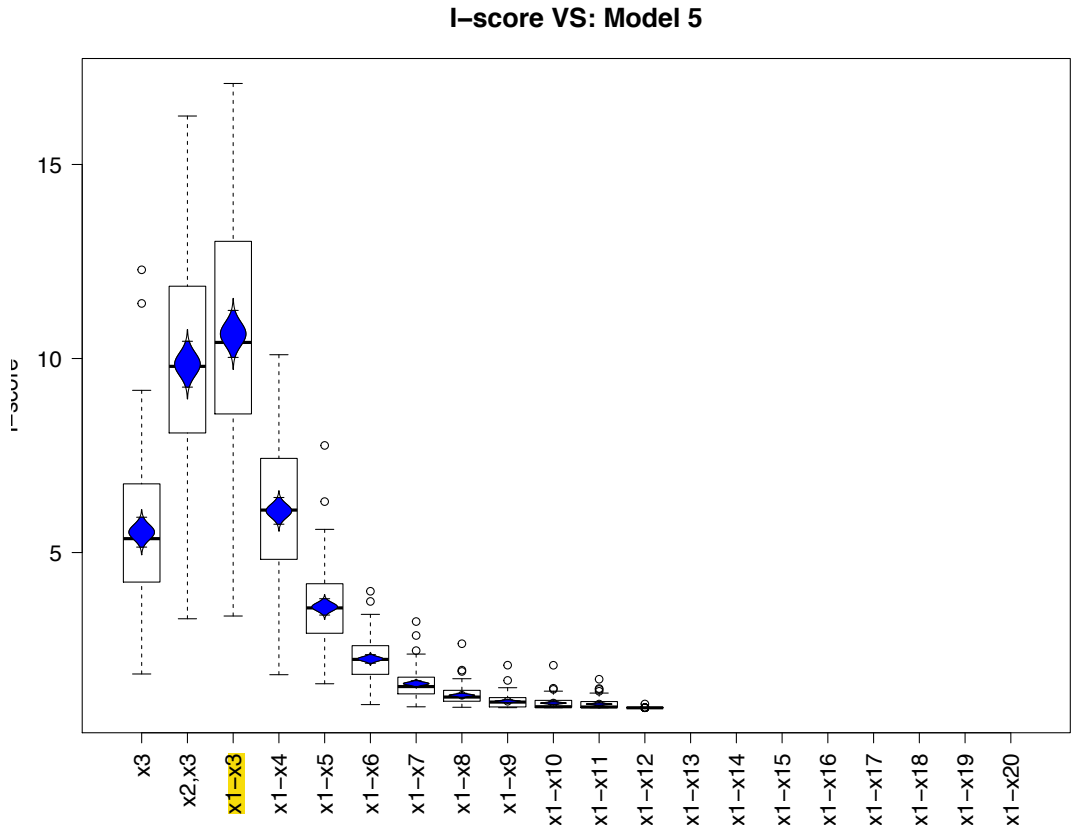


Figure 3.18: **I-score drop in Model 5**. Variable set (x_1, x_2, x_3) is influential in Model 5. The x-axes depict variable sets considered. The y-axes depict the corresponding I-scores of each variable set. Boxplots indicate the distribution of I-scores for each variable set across simulations. The correctly selected variable set is highlighted in yellow.

Table 3.1: **Fearon & Laitin variables "ccode" through "plurall5"**

Statistic	Mean	St. Dev.	Min	Max
ccode	450.622	248.143	2	950
year	1,975.548	15.075	1,945	1,999
popl	31,149.740	101,250.800	222.000	1,238,599.000
popl2	30,448.010	99,351.290	222.000	1,227,177.000
popl3	29,756.370	97,460.450	222.000	1,215,414.000
popl4	29,073.640	95,578.420	222.000	1,203,324.000
popl5	28,403.760	93,704.140	222.000	1,190,918.000
lpopl1	9.049	1.455	5.403	14.029
polity2l	−0.506	7.480	−10	10
gdpenl	3.592	4.465	0.048	66.735
gdptypel	0.842	1.357	0	4
gdptypel2	0.798	1.349	0	4
gdptypel3	0.756	1.341	0	4
gdptypel4	0.713	1.331	0	4
gdptypel5	0.670	1.320	0	4
lgdpenl1	7.649	1.038	3.871	11.108
mtnest	18.088	20.966	0.000	94.300
lmtnest	2.177	1.404	0.000	4.557
elevdiff	3,180.187	2,008.973	53	9,002
ethfracl	0.383	0.282	0.001	0.925
ethfracl2	0.381	0.279	0.001	0.925
ethfracl3	0.380	0.276	0.001	0.925
ethfracl4	0.378	0.273	0.001	0.925
ethfracl5	0.374	0.269	0.001	0.925
efl	0.460	0.264	0.002	1.000
efl2	0.461	0.261	0.002	1.000
efl3	0.461	0.258	0.002	1.000
efl4	0.462	0.255	0.002	1.000
efl5	0.462	0.252	0.002	1.000
plurall	0.661	0.243	0.004	0.999
plurall2	0.661	0.240	0.004	0.999
plurall3	0.662	0.237	0.004	0.999
plurall4	0.663	0.234	0.004	0.999
plurall5	0.664	0.231	0.004	0.999

Table 3.2: **Fearon & Laitin variables "secondl" through "warl"**

Statistic	Mean	St. Dev.	Min	Max
secondl	0.153	0.110	0.000	0.440
secondl2	0.153	0.108	0.000	0.440
secondl3	0.153	0.107	0.000	0.440
secondl4	0.152	0.106	0.000	0.440
secondl5	0.152	0.104	0.000	0.440
numlangl	6.782	7.144	1	46
numlangl2	6.715	7.072	1	46
numlangl3	6.648	6.999	1	46
numlangl4	6.581	6.925	1	46
numlangl5	6.514	6.849	1	46
relfracl	0.367	0.216	0.000	0.783
relfracl2	0.366	0.213	0.000	0.783
relfracl3	0.365	0.211	0.000	0.783
relfracl4	0.364	0.208	0.000	0.783
relfracl5	0.363	0.205	0.000	0.783
plurrell	72.969	20.163	25	100
plurrell2	73.156	19.934	25	100
plurrell3	73.344	19.700	25	100
plurrell4	73.532	19.461	25	100
plurrell5	73.720	19.218	25	100
minrelpcl	18.190	12.786	0	50
minrelpcl2	18.110	12.626	0	50
minrelpcl3	18.030	12.463	0	50
minrelpcl4	17.950	12.298	0	50
minrelpcl5	17.748	12.153	0	50
musliml	24.964	37.146	0.000	100.000
musliml2	24.335	36.834	0.000	100.000
musliml3	23.707	36.509	0.000	100.000
musliml4	23.078	36.170	0.000	100.000
musliml5	22.450	35.816	0.000	100.000
warsl	0.150	0.407	0	4
warsl2	0.145	0.400	0	4
warsl3	0.140	0.393	0	4
warsl4	0.135	0.386	0	4
warsl5	0.129	0.378	0	4
warl	0.135	0.341	0	1

Table 3.3: **Fearon & Laitin variables "western" through "onset"**

Statistic	Mean	St. Dev.	Min	Max
western	0.175	0.380	0	1
eeurop	0.098	0.297	0	1
lamerica	0.183	0.387	0	1
ssafrica	0.241	0.428	0	1
asia	0.166	0.372	0	1
nafrme	0.138	0.345	0	1
colbrit	0.287	0.452	0	1
colfra	0.171	0.377	0	1
oill	0.126	0.332	0	1
oill2	0.123	0.328	0	1
oill3	0.119	0.324	0	1
oill4	0.115	0.319	0	1
oill5	0.111	0.314	0	1
ncontig	0.173	0.379	0	1
nwstate	0.030	0.170	0	1
instabl	0.142	0.349	0	1
instabl2	0.138	0.345	0	1
instabl3	0.134	0.341	0	1
instabl4	0.131	0.337	0	1
instabl5	0.125	0.331	0	1
anocl	0.223	0.416	0	1
deml	0.329	0.470	0	1
cowwarl	0.067	0.251	0	1
sdwarl	0.124	0.330	0	1
colwarl	0.081	0.273	0	1
onset	0.017	0.137	0	4

Table 3.4: **3-fold training and testing sets**

	Training set years	**Testing set years**	**# of cases**	**# of controls**
Fold 1	1945-1984	1985-1986	71	4319
Fold 2	1952-1991	1992-1993	75	4786
Fold 3	1958-1997	1998-1999	90	5210

Table 3.5: **Example of top returned variable sets**. Returned variable sets with the highest I-scores each of the three training sets are illustrated here. Variables with numbers at the end of their names are lagged the corresponding number of years.

	Fold 1	**Fold 2**	**Fold 3**
1	musliml3, warsl2, asia, nwstate, instabl5, sdwarl (6)	musliml5, asia, nwstate, instabl, instabl5, sdwarl (6 var)	asia, nwstate, instabl4, anocl, sdwarl, colwarl (6 var)
2	asia, nafrme, nwstate, sdwarl (4 var)	musliml2, asia, nwstate, instabl3, instabl4, sdwarl (6 var)	musliml5, asia, nwstate, instabl, sdwarl (5 var)
3	musliml4, asia, nwstate, instabl,sdwarl (5 var)	musliml5, asia, nwstate, instabl4, anocl (5 var)	musliml5, warsl3, nwstate, instabl3, instabl5, sdwarl (6 var)
4	popl, musliml5, nwstate, instabl, sdwarl (5 variables)	popl2, musliml4, nwstate, instabl4, anocl (5 var)	warsl, instabl4, anocl, sdwarl (4 var)
5	asia, nafrme, sdwarl, colwarl (4 variables)	numlangl2, ssafrica, colfra, instabl, cowwarl, colwarl (6 var)	warl, instabl4, anocl, sdwarl (4 var)

Table 3.6: **Model error rates**: The "PR Lasso" model includes searching for interaction terms of up to 8-way interactions. The "FL Lasso" and "FL Logistic" models are run on explanatory models from Fearon and Laitin's (2003) main model specification: war lagged, gdp, population, mountainous region, non-contiguous state, oil, new state, instability, democracy, ethnic fractionalization, and religious fractionalization.

Testing error rate	**PR Lasso**	**RF**	**NN**	**FL Lasso**	**FL Logistic**
Fold 1	**0.36%**	0.73%	**0.36%**	20.81%	19.71%
Fold 2	**4.21%**	5.83%	6.47%	29.45%	33.98%
Fold 3	**1.28%**	1.6%	**1.28%**	40.13%	24.36%
Average error rate	**1.95%**	2.72%	2.70%	30.13%	26.02%
Average correct prediction rate	**98.05%**	97.28%	97.3%	69.87%	73.98%

Table 3.7: **False positives (+) and False negatives (-).**

	Fold 1		**Fold 2**		**Fold 3**	
	False +	False -	False +	False -	False +	False -
PR Lasso	0	1	0	13	0	4
RF	1	1	7	13	1	4
NN	0	1	5	13	0	4
Lasso	0	1	0	13	0	4
FL Lasso	56	1	102	3	87	1
FL Logistic	53	1	88	3	75	1

References

Achen, Christopher H. (2002). “Toward a new political methodology: microfoundations and ART”. In: *Annual Review of Political Science* 5.1, pp. 423–450.

Achen, Christopher (2005). “Let’s Put Garbage-Can Regressions and Garbage-Can Probits Where They Belong”. In: *Conflict Management and Peace Science* 22.4, pp. 327–339.

Balcells, Laia and Patricia Justino (2014). “Bridging Micro and Macro Approaches on Civil Wars and Political Violence: Issues, Challenges, and the Way Forward”. In: *Journal of Conflict Resolution* 58.8, pp. 1343–1359.

Barton, Frederick et al. (2008). *Early Warning? A Review of Conflict Prediction Models and Systems*. Tech. rep., pp. 0–19.

Beck, Nathaniel, Gary King, and Langche Zeng (2000). “Improving Quantitative Studies of International Conflict: A Conjecture”. In: *The American Political Science Review* 94.1, pp. 21–35.

Bell, Sam R. et al. (2013). “Coercion, capacity, and coordination: Predictors of political violence”. In: *Conflict Management and Peace Science* 30.3, pp. 240–262.

Belloni, Alexandre, Victor Chernozhukov, and Christian Hansen (2014). “on Structural and Treatment Effects âĂă”. In: *Journal of Economic Perspectives* 28.2, pp. 29–50.

Blattman, Christopher and Edward Miguel (2010). “Civil war”. In: *Journal of Economic Literature* 48.1, pp. 3–57.

Bolón-Canedo, Verónica, Noelia Sánchez-Maroño, and Amparo Alonso-Betanz (2013). “A review of feature selection methods on synthetic data”. In: *Knowledge and Information Systems* 34.3, pp. 483–519.

Breiman, Leo (2001). “Random forests”. In: *Machine learning* 45.1, pp. 5–32.

Burke, Marshall B et al. (2009). “Warming increases the risk of civil war in Africa.” In: *Proceedings of the National Academy of Sciences of the United States of America* 106, pp. 20670–20674.

Campos, Gustavo de los, Daniel Gianola, and David B Allison (2010). “Predicting genetic predisposition in humans: the promise of whole-genome markers.” In: *Nature reviews. Genetics* 11.12, pp. 880–6.

Chernoff, Herman, Shaw-Hwa Lo, and Tian Zheng (2009). “Discovering influential variables: A method of partitions”. In: *The Annals of Applied Statistics* 3.4, pp. 1335–1369.

Clayton, David G (2009). “Prediction and interaction in complex disease genetics: experience in type 1 diabetes.” In: *PLoS genetics* 5.7, e1000540.

Clayton, Govinda and Kristian Skrede Gleditsch (2014). “Will we see helping hands? Predicting civil war mediation and likely success”. In: *Conflict Management and Peace Science* 31.3, pp. 265–284.

Collier, Paul and Anke Hoeffler (2004). “Greed and grievance in civil war”. In: *Oxford Economic Papers* 56.4, pp. 563–595.

De Marchi, Scott, Christopher Gelpi, and Jeffrey D Grynaviski (2004). “Untangling Neural Nets”. In: *American Political Science Review* 98.02, pp. 371–378.

Fan, Ruixue and Shaw-Hwa Lo (2013). “A robust model-free approach for rare variants association studies incorporating gene-gene and gene-environmental interactions.” In: *PloS one* 8.12, e83057.

Fearon, James D. and David D. Laitin (2003). “Ethnicity Insurgency and Civil war”. In: *American political science review* 97.1, pp. 75–90.

Geisser, Seymour (1993). *Predictive Inference*. New York, NY: Chapman and Hall.

Gleditsch, Kristian S. and Michael D. Ward (2013). “Forecasting is difficult, especially about the future: Using contentious issues to forecast interstate disputes”. In: *Journal of Peace Research* 50.1, pp. 17–31.

Goldsmith, Benjamin E. et al. (2013). “Forecasting the onset of genocide and politicide: Annual out-of-sample forecasts on a global dataset, 1988-2003”. In: *Journal of Peace Research* 50.4, pp. 437–452.

Goldstone, Jack A et al. (2010). “A global model for forecasting political instability”. In: *American Journal of Political Science*. Vol. 54. 1, pp. 190–208.

Gransbo, K et al. (2013). “Chromosome 9p21 genetic variation explains 13% of cardiovascular disease incidence but does not improve risk prediction.” In: *Journal of internal medicine* 274.3, pp. 233–40.

Grimmer, Justin (2015). “We Are All Social Scientists Now: How Big Data, Machine Learning, and Causal Inference Work Together”. In: *PS: Political Science & Politics* 48.01, pp. 80–83.

Guyon, I et al. (2003). “An introduction to variable and feature selection”. In: *Journal of Machine Learning Research* 3, pp. 1157–1182.

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman (2008). *The Elements of Statistical Learning*. 2nd. Springer.

Hegre, Havard et al. (2001). “Toward a democratic civil peace? Democracy, political change, and civil war, 1816-1992”. In: *American Political Science Association* 95.01, pp. 33–48.

Hegre, Havard et al. (2013). “Predicting Armed Conflict, 2010-2050”. In: *International Studies Quarterly* 57.2, pp. 250–270.

Ho, Tin Kam (1995). “Random Decision Forests Tin Kam Ho Perceptron training”. In: *Proceedings of the 3rd International Conference on Document Analysis and Recognition*, pp. 278–282.

Hua, Jianping, Waibhav D. Tembe, and Edward R. Dougherty (2009). “Performance of feature-selection methods in the classification of high-dimension data”. In: *Pattern Recognition* 42.3, pp. 409–424.

Jakobsdottir, Johanna et al. (2009). “Interpretation of genetic association studies: markers with replicated highly significant odds ratios may be poor classifiers.” In: *PLoS genetics* 5.2, e1000337.

James, Gareth et al. (2013). *An introduction to statistical learning*. Springer. New York, p. 241.

Janssens, a Cecile J W and Cornelia M van Duijn (2008). "Genome-based prediction of common diseases: advances and prospects." In: *Human molecular genetics* 17.R2, R166–73.

Kadera, Kelly and Sara Mitchell (2005). "Manna from Heaven or Forbidden Fruit? The (Ab) Use of Control Variables in Research on International Conflict". In: *Conflict Management and Peace Science* 22.4, pp. 273–275.

Kohavi, Ron (1995). "A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection". In: *International Joint Conference on Artificial Intelligence* 14.12, pp. 1137–1143.

Lo, Adeline et al. (2015). "Why significant variables aren't automatically good predictors". In: *Proceedings of the National Academy of Sciences* 112.45, pp. 13892–13897.

Lo, Adeline et al. (2016). "Making good prediction: a theoretical framework".

Lo, Shaw-Hwa and Tian Zheng (2004). "A demonstration and findings of a statistical approach through reanalysis of inflammatory bowel disease data." In: *Proceedings of the National Academy of Sciences of the United States of America* 101.28, pp. 10386–91.

Lo, Shaw-Hwa et al. (2008). "Discovering interactions among BRCA1 and other candidate genes associated with sporadic breast cancer." In: *Proceedings of the National Academy of Sciences of the United States of America* 105, pp. 12387–12392.

Muchlinski, David, David Siroky, and Matthew Kocher (2015). "Comparing Random Forest with Logistic Regression for Predicting Class-imbalanced Civil War Onset Data". In: *Political Analysis* 24, pp. 87–103.

Nagler, Jonathan and Joshua a. Tucker (2015). "Drawing Inferences and Testing Theories with Big Data". In: *PS: Political Science & Politics* 00.00, pp. 84–88.

O'Brien, Sean P. (2010). "Crisis Early Warning and Decision Support: Contemporary Approaches and Thoughts on Future Research". In: *International Studies Review* 12.1, pp. 87–104.

"Predicting the influence of common variants." (2013). In: *Nature genetics* 45.4, p. 339.

Ray, James Lee (2003). "Explaining interstate conflict and war: what should be controlled for?*". In: *Conflict Management and Peace Science* 20.1, pp. 1–31.

— (2005). "Constructing Multivariate Analyses (of Dangerous Dyads)". In: *Conflict Management and Peace Science* 22.4, pp. 277–292.

Rustad, Siri C. A. et al. (2011). "All Conflict is Local: Modeling Sub-National Variation in Civil Conflict Risk". In: *Conflict Management and Peace Science* 28.1, pp. 15–40.

Saeys, Yvan, Iñaki Inza, and Pedro Larrañaga (2007). "A review of feature selection techniques in bioinformatics". In: *Bioinformatics* 23.19, pp. 2507–2517.

Sambanis, Nicholas (2002). "A Review of Recent Advances and Future Directions in the Quantitative Literature on Civil War". In: *Defence and Peace Economics* 13.3, pp. 215–243.

Schrodt, Philip A. (1995). "Patterns, Rules and Learning: Computational Models of International Behavior". In: *Unpublished manuscript available at http://polmeth. calpoly. edu*.

Tibshirani, Robert (1996). "Regression Shrinkage and Selection via the Lasso". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 58.1, pp. 267–288.

Veer, Laura J van't et al. (2002). "Gene expression profiling predicts clinical outcome of breast cancer." In: *Nature* 415.6871, pp. 530–6.

Wang, Haitian et al. (2012). "Interaction-based feature selection and classification for high-dimensional biological data." In: *Bioinformatics* 28.21, pp. 2834–42.

Ward, Michael D. and Kristin Bakke (2005). "Predicting civil conflicts: on the utility of empirical research".

Ward, Michael D., Brian D. Greenhill, and Kristin M. Bakke (2010). "The perils of policy by p-value: Predicting civil conflicts". In: *Journal of Peace Research* 47.4, pp. 363–375.

Weidmann, Nils B. and Michael D. Ward (2010). “Predicting Conflict in Space and Time”. In: *Journal of Conflict Resolution* 54.6, pp. 883–901.

Welch, Ivo and Amit Goyal (2008). “A comprehensive look at the empirical performance of equity premium prediction”. In: *Review of Financial Studies* 21.4, pp. 1455–1508.

Zeng, Langche (2000). “Neural Network Models and Political Data Analysis”. In: *Political Complexity: Nonlinear Models of Politics*. Ed. by Diana Richards. Ann Arbor: University of Michigan Press, pp. 239–268.

Zheng, S. Lilly et al. (2008). “Cumulative association of five genetic variants with prostate cancer.” In: *The New England Journal of Medicine* 54.2, pp. 460–1.

Zheng, Tian, Hui Wang, and Shaw Hwa Lo (2006). “Backward genotype-trait association (BGTA)-based dissection of complex traits in case-control designs”. In: *Human Heredity* 62.4, pp. 196–212.

Zheng, Tian et al. (2010). *Handbook of Computational Statistics: Statistical Bioinformatics*. Ed. by H. H. S. Lu, B. Scholkopf, and H. Zhao. New York: Springer-Verlag.

Appendix A

Appendix for Chapter 1

A.1 Partition Retention and *I* score

The partition retention (PR) approach to variable selection depends heavily on the I-score applied to small groups of explanatory variables. Suppose we have n observations on a disease phenotype Y. When dealing with a small group of m SNP's, each individual is represented by a value Y of the dependent variable and one of $m_1 = 3^m$ possible cells into which the m variables fall. Then the value of I is given by

$$I = \sum_{j=1}^{m_1} \frac{n_j}{n} \frac{(\bar{Y}_j - \bar{Y})^2}{s^2/n_j} = \frac{\sum_{j=1}^{m_1} n_j^2 (\bar{Y}_j - \bar{Y})^2}{\sum_{i=1}^{n} (Y_i - \bar{Y})^2},$$

where Y_i corresponds to the i-th individual, $\bar{Y}$ is the mean of all n Y values, s is the standard deviation of all n Y values, $\bar{Y}_j$ is the mean of the Y values in cell j, n_j is the number of individuals in cell j, and n is the total number of individuals. The measure I is a statistic which may be calculated from the observed data, and does not involve knowing the underlying distributions, as did Truth in Example 4.

The I-score has several desirable properties. First it does not require specification of a model for the joint effect of the m SNPs on Y. It is designed to capture the discrepancy between the conditional means of Y given the values of the SNPS and the overall mean of Y. Unlike odds ratios as a measure of effect in assessing simple 2×2 tables, I captures and aggregates all discrepancy (signals) from all m_1 cells and forms

a flexible measure. It can be used as a measure to assess joint influence or effect sizes, and, importantly, is well-correlated with predictivity.

Second, under the null hypothesis that the subset has no effect on Y, the expected value of I remains non-increasing when dropping variables from the subset. In other words, the I score is robust to changes to the number of SNPs, m. And I has the property that adjoining to the group another variable which is independent of Y will tend to decrease I, the PR method is based on selecting a group at random and sequentially eliminating those variables which diminish I the most, and retaining those for which I can no longer be diminished. Those variables, that are retained most often from many randomly chosen groups are candidates for variable selection. The fact that I does not automatically increase as more variables are added to the group being measured is a good property of the I-score.

Finally, under the null hypothesis of no effect I acts like a weighted average of independent chi-squares with one degree of freedom. Therefore, I values substantially larger than 1 are worth noting.

Appendix B

Appendix for Chapter 2

Proof of Lemma 1:

It is obvious that $|a| \leq b$. Let S_1 be the sum of the positive values of z_j and S_2 the sum of the negative values. Let T_1 be the sum of the squares of the positive values and T_2 the sum of the squares of the negative values. It follows that $S_1 + S_2 = a$ and $S_1 - S_2 = b$ and thus $S_1 = {}^{(a+b)}/_2$ and $S_2 = {}^{(a-b)}/_2$. Then clearly $T_1 \leq {S_1}^2$ and $T_2 \leq {S_2}^2$. Consequently,

$$\sum_{j=1}^{K} {z_j}^2 = T_1 + T_2 \leq {S_1}^2 + {S_2}^2 = \frac{a^2 + b^2}{2} \tag{B.1}$$

which is equivalent to the inequality in Equation (2.5) and equality is attained when there are at most one positive and one negative component if $|a| < 1$.

B.1 Proof of Theorem 1:

We prove that the I-score approaches a constant multiple of θ_I asymptotically.

Under the null hypothesis of no association between $\mathbf{X} = \{X_k, k = 1, ..., m\}$ and Y, $I_{\Pi_\mathbf{X}}$ can be asymptotically expressed as $\sum_{j=1}^{m_1} \lambda_j \chi_j^2$ (a weighted average), where λ_j is between 0 and 1 and $\sum_{j=1}^{m_1} \lambda_j$ is equal to $1 - \sum_{j=1}^{m_1} {p_j}^2$, where p_j is the cell j's probability. $\{\chi_j^2\}$ are m_1 chi-squares, each with degree of freedom, df= 1 (see Chernoff, Lo, and Zheng 2009).

Furthermore, the above formulation and properties of $I_{\Pi_\mathbf{X}}$ apply to the specified

Y model with case-control study (where $Y = 1$ designates case and $Y = 0$ designates control) as demonstrated in Chernoff, Lo, and Zheng 2009. More specifically, in a case-control study with n_d cases and n_u controls (letting $n = n_d + n_u$), $ns_n^2 I_{\Pi_{\mathbf{X}}}$ can be expressed as the following:

$$\begin{aligned} ns_n^2 I_{\Pi_{\mathbf{X}}} &= \sum_{j \in \Pi_{\mathbf{X}}} n_j^2 (\bar{Y}_j - \bar{Y})^2 \\ &= \sum_{j \in \Pi_{\mathbf{X}}} (n_{d,j}^m + n_{u,j}^m)^2 \left(\frac{n_{d,j}^m}{n_{d,j}^m + n_{u,j}^m} - \frac{n_d}{n_d + n_u} \right)^2 \\ &= \left(\frac{n_d n_u}{n_d + n_u} \right)^2 \sum_{j \in \Pi_{\mathbf{X}}} \left(\frac{n_{d,j}^m}{n_d} - \frac{n_{u,j}^m}{n_u} \right)^2 \end{aligned}$$

where $n_{d,j}^m$ and $n_{u,j}^m$ denote the numbers of cases and controls falling in jth cell, and $\Pi_{\mathbf{X}}$ stands for the partition formed by m variables in $\mathbf{X}$. Since the PR method[1] seeks the partition that yields larger I-scores, one can decompose the following:

$$ns_n^2 I_{\Pi_{\mathbf{X}}} = \sum_{j \in \Pi_{\mathbf{X}}} n_j^2 (\bar{Y}_j - \bar{Y})^2 = A_n + B_n + C_n$$

where, $A_n = \sum_{j \in \Pi_{\mathbf{X}}} n_j^2 (\bar{Y}_j - \mu_j)^2$, $B_n = \sum_{j \in \Pi_{\mathbf{X}}} n_j^2 (\bar{Y} - \mu_j)^2$, and $C_n = \sum_{j \in \Pi_{\mathbf{X}}} -2n_j^2 (\bar{Y}_j - \mu_j)(\bar{Y} - \mu_j)$. Here, μ_j and μ are the local and grand means of Y, that is, $E(\bar{Y}_j) = \mu_j$; $\bar{Y} = \mu = \frac{n_d}{n_d + n_u}$ for fixed n. It is easy to see that both terms A_n and C_n, when divided by n^2 converge to 0 in probability as $n \longrightarrow \infty$. We turn to the final term, B_n. Note that:

$$\lim_n \frac{B_n}{n^2} \overset{\mathcal{P}}{=} \lim_n \sum_{j \in \Pi_{\mathbf{X}}} \left(\frac{n_j^2}{n^2} \right) (\mu_j - \mu)^2$$

In a case-control study, we have:

$$\mu_j = \frac{n_d P(j|d)}{n_d P(j|d) + n_u P(j|u)}$$

and

$$\mu = \frac{n_d}{n_d + n_u}$$

[1]The PR method encompasses a backwards dropping algorithm (BDA) that is introduced in Wang et al. 2012; we directly cite and present the BDA in the SI.

Since for every j, $\frac{n_j}{n}$ converges (in probability) to $p_j = \lambda P(j|d) + (1-\lambda)P(j|u)$ as $n \to \infty$, if $\lim_n \frac{n_d}{n} = \lambda$, a fixed constant between 0 and 1, it follows that:

$$\begin{aligned}
\frac{B_n}{n^2} &= \sum_{j \in \Pi_{\mathbf{X}}} (\frac{n_j^2}{n^2})(\mu_j - \mu)^2 \xrightarrow{\mathcal{P}} \sum_{j \in \Pi_{\mathbf{X}}} p_j^2 \big(\frac{\lambda P(j|d)}{\lambda P(j|d) + (1-\lambda)P(j|u)} - \lambda \big)^2 \qquad \text{as } n \to \infty \\
&= \sum_{j \in \Pi_{\mathbf{X}}} \big\{ \lambda P(j|d) - \lambda [\lambda P(j|d) + (1-\lambda)P(j|u)] \big\}^2 \\
&= \sum_{j \in \Pi_{\mathbf{X}}} \big\{ \lambda (1-\lambda) P(j|d) - [\lambda (1-\lambda) P(j|u)] \big\}^2 \\
&= \lambda^2 (1-\lambda)^2 \sum_{j \in \Pi_{\mathbf{X}}} [P(j|d) - P(j|u)]^2 \\
&= \lambda^2 (1-\lambda)^2 \sum_{j \in \Pi_{\mathbf{X}}} [P(j|d) - P(j|u)]^2
\end{aligned}$$

Thus, ignoring the constant term in the above equation, the I-score can guide a search for X partitions which leads to finding larger values of the summation term $\sum_{j \in \Pi_{\mathbf{X}}} [P(j|d) - P(j|u)]^2$. We have proven Theorem 1.

B.2 Proof of Corollary 1:

The asymptotic lower bound of Equation (2.6) is a consequence of Lemma 1 and Theorem 1. In theory, the above corollary allows us to apply a useful lower bound for identifying good variable sets with large I-scores. In practice, however, once the variable sets are found (through their large I-scores), the true prediction rates can be greater than the identified lower bounds. Theorem 1 provides a simple asymptotic behaviors of I-score under some strict assumptions. We offer similar derivations below following two levels of relaxations of the constraints.

We remark that with additional work, one can show that the convergence given in Equation (2.6) can be extended to be uniformly over all partitions $\{\Pi\}$ with bounded number of cells and for all λ that stay away from 0 and 1.

B.3 Proof of Corollary 2:

COROLLARY 2 **Under the assumptions of an arbitrary prior $\pi(d)$ and $\frac{n_d}{n} \to \lambda$ as $n \to \infty$, the correct prediction rate can be easily seen as:**

$$\theta_c^*[p_{\mathbf{X}_d}, p_{\mathbf{X}_u}] = \frac{1}{2} + \frac{1}{2} \sum_{j \in \Pi_{\mathbf{X}}} \left| P(j|d)\pi(d) - P(j|u)\pi(u) \right| \tag{B.2}$$

Let the modified score $I^*_{\Pi_n}$ be defined as

$$n s_n^2 I^*_{\Pi_n} = \frac{1}{4} \sum_{j \in \Pi_{\mathbf{X}}} n_j^2 \left[\bar{y}_j (\frac{\pi(d)}{\lambda}) - (1 - \bar{y}_j)(\frac{\pi(u)}{1-\lambda}) \right]^2. \tag{B.3}$$

Then we have:

$$\lim_{n \to \infty} \frac{s_n^2 I^*_{\Pi_n}}{n} \overset{\mathcal{P}}{=} \frac{1}{4} \sum_{j \in \Pi_{\mathbf{X}}} \left[P(j|d)\pi(d) - P(j|u)\pi(u) \right]^2. \tag{B.4}$$

Similar lower bounds to Corollary 1 can then be derived as:

$$\theta_c^*[p_{\mathbf{X}_d}, p_{\mathbf{X}_u}] = \frac{1}{2} + \frac{1}{2} \sum_{j \in \Pi_{\mathbf{X}}} \left| P(j|d)\pi(d) - P(j|u)\pi(u) \right| \tag{B.5}$$

$$\geq \frac{1}{2} + \frac{1}{2} \sqrt{\lim_{n \to \infty} \frac{\lambda(1-\lambda) I^*_{\Pi_n}}{2n} - a^2} \tag{B.6}$$

where $a = \sum_{j \in \Pi_{\mathbf{X}}} (P(j|d)\pi(d) - P(j|u)\pi(u)) = \pi(d) - \pi(u)$.

Similar to Corollary 1, Equation (B.5) is a direct consequence of Equation (B.4) and Lemma 1 (but with z_j replaced by $|P(j|d)\pi(d) - P(j|u)\pi(u)|$).

B.4 Generalizing to different loss and cost functions:

Thus far, we have used a 0-1 loss on the binary classification problem. The 0-1 loss treats false negatives and false positives equally. In real applications, the scientist may wish to weigh the costs of different incorrect predictions differently. For instance, failing to detect a cancer patient may be deemed a more costly mistake to make than that of misclassifying a healthy patient because ameliorating the former mistake later on

can be more difficult. The different cost amounts in making a loan decision is another example. The cost of lending to a defaulter may be seen as greater than that of the loss-of-business cost of declining a loan to a non-defaulter due to some positive level of risk aversion. Let loss function L be defined as:

$$\mathrm{L}(d,u) = l_1, \mathrm{L}(u,d) = l_2 \tag{B.7}$$

and

$$\mathrm{L}(d,d) = L(u,u) = 0 \tag{B.8}$$

where l_1 and l_2 are the prices paid (or losses incurred) for misclassifying a diseased individual to the healthy class or a healthy person to a diseased class, respectively. We can derive the optimum Bayes' solution by minimizing the expected predicted loss, that is, to assign future observations to the class with less loss, given its j value. We simply assign a test sample with partition (predictor) j to d if:

$$P(j|d)\pi(d)\mathrm{L}(u,d) < P(j|u)\pi(u)\mathrm{L}(d,u)$$

otherwise, assign to u. Equivalently, choose d if

$$P(j|d)\pi(d)l_2 < P(j|u)\pi(u)l_1$$

otherwise u. In this way, the expected loss of adopting this rule is thus:

$$e^l = \frac{1}{2}\sum_{j\in\Pi_{\mathbf{X}}} \min\{a_j, b_j\},$$

where $a_j = P(j|d)\pi(d)l_2$ and $b_j = P(j|u)\pi(u)l_1$. The random rule of choosing d or u with equal probabilities has an expected loss of:

$$\gamma = \frac{1}{2}\sum(a_j + b_j) = \frac{1}{2}\left(\pi(d)l_2 + \pi(u)l_1\right),$$

a constant independent of partition Π_x. So, the "correct prediction gain", θ_c^l (interpreted as the improvement of correct prediction with respect to γ), can be defined as:

$$\theta_c^l = \frac{1}{2}\sum_{j\in\Pi_{\mathbf{X}}} \max\{a_j, b_j\} = \frac{1}{2}\sum_{j\in\Pi_{\mathbf{X}}} (a_j + b_j) - e^l = \gamma - e^l$$

Again we have

$$\begin{aligned}\theta_c^l &= \frac{\gamma}{2} + \frac{c^l - e^l}{2} \\ &= \frac{\gamma}{2} + \frac{1}{2}\sum_{j\in\Pi_{\mathbf{X}}} |a_j - b_j|\end{aligned}$$

After standardizing by γ, we obtain the correct prediction rate as:

$$\begin{aligned}\theta_c &= \frac{\theta_c^l}{\gamma} \\ &= \frac{1}{2} + \frac{1}{2\gamma}\sum_{j\in\Pi_{\mathbf{X}}} |a_j - b_j|\end{aligned}$$

Collecting the above discussion together, let the cost-based I-score $I^c_{\Pi_{\mathbf{X}}}$ be defined as:

$$\begin{aligned}n s_n^2 I^c_{\Pi_{\mathbf{X}}} &= \frac{1}{4\gamma^2}\sum_{j\in\Pi_{\mathbf{X}}} n_j^2 \left[\bar{y}_j \left(\frac{\pi(d)}{\lambda}\right) l_2 - (1-\bar{y}_j)\left(\frac{\pi(u)}{1-\lambda}\right) l_1\right]^2 \\ &\approx \frac{n^2}{4\gamma^2}\sum_{j\in\Pi_{\mathbf{X}}} \left[P(j|d)\pi(d)l_2 - P(j|u)\pi(u)l_1\right]^2. \qquad \text{(B.9)}\end{aligned}$$

B.5 Corollary 3:

We present the following lower bound in Corollary 3. Let

$$\sum_{j\in\Pi_{\mathbf{X}}} (P(j|d)\pi(d)l_2 - P(j|u)\pi(u)l_1) = \pi(d)l_2 - \pi(u)l_1 = a.$$

COROLLARY 3. Under the assumptions of Corollary 2 and using the loss function L **described in Equation (B.7), then**

$$\lim_{n\to\infty} \frac{s_n^2 I^c_{\Pi_{\mathbf{X}}}}{n} \overset{\mathcal{P}}{=} \frac{1}{4\gamma^2} \sum_{j\in\Pi_{\mathbf{X}}} [P(j|d)\pi(d)l_2 - P(j|u)\pi(u)l_1]^2 \tag{B.10}$$

Furthermore, one can derive a similar lower bound for the correct prediction rate θ_c **as:**

$$\begin{aligned}
\theta_c &= \frac{1}{2} + \frac{1}{2\gamma} \sum_{j\in\Pi_{\mathbf{X}}} \left|a_j - b_j\right| \\
&\overset{\mathcal{P}}{\geq} \lim_{n\to\infty} \left(\frac{1}{2} + \frac{1}{2\gamma} \sqrt{\frac{\lambda(1-\lambda) I^c_{\Pi_{\mathbf{X}}}}{n} - a^2} \right) \\
&= \frac{1}{2} + \frac{1}{2\gamma} \sqrt{\lim_{n\to\infty} \frac{\lambda(1-\lambda) I^c_{\Pi_{\mathbf{X}}}}{n} - a^2}
\end{aligned} \tag{B.11}$$

The proofs for Equations (B.10) and (B.11) are quite similar to that for Corollary 2 given above; we shall omit them.

B.6 Backwards Dropping Algorithm

The backward dropping algorithm (BDA) is a greedy algorithm to search for the variable subset that maximizes the I-score through stepwise elimination of variables from an initial subset sampled in some way from the variable space.[2] The details are as follows.

1. *Training set*: Consider a training set $\{(y_1, x_1), ..., (y_n, x_n)\}$ of n observations, where $x_i = (x_{1i}, ..., x_{pi})$ is a p-dimensional vector of explanatory variables. Typically p is very large. All explanatory variables are discrete.

2. *Sampling from variable space*: select an *initial* subset of k explanatory variables $S_b = \{X_{b_1}, ..., X_{b_k}\}$, $b = 1, ..., B$.

[2]The presentation of the backward dropping algorithm is taken directly from section 2.2 of Wang et al. (2012).

3. *Compute I-score*: $I(S_b) = \sum_{j \in P_k} n_j^2 (\bar{Y}_j - \bar{Y})^2$.

4. *Drop variables*: Tentatively drop each variable in S_b and recalculate the I-score with one variable less. Then drop the one that gives the highest I-score. Call this new subset S_b' which has one variable less than S_b.

5. *Return set*: Continue the next round of dropping on S_b' until only one variable is left. Keep the subset that yields the highest I-score in the whole dropping process. Refer to this subset as the *return set* R_b. Keep it for future use.

If no variable in the initial subset has influence on Y, then the values of I will not change much in the dropping process. On the other hand, when influential variables are included in the subset then the I-score will increase (decrease) rapidly before (after) reaching the maximum.

B.7 Simulation details for Variable set of size 6

In this 6-SNP model, we created two sets of 3 SNPs with the same disease model. Individuals with no risk genotypes on both sets have odds that is one third of the baseline odds. Individuals with risk genotypes on only one set have odds that is three times that of the baseline. Individuals with risk genotypes on both sets have odds that is 6 times that of the baseline. All other simulation parameters were set to be equivalent to the 3 SNP example.

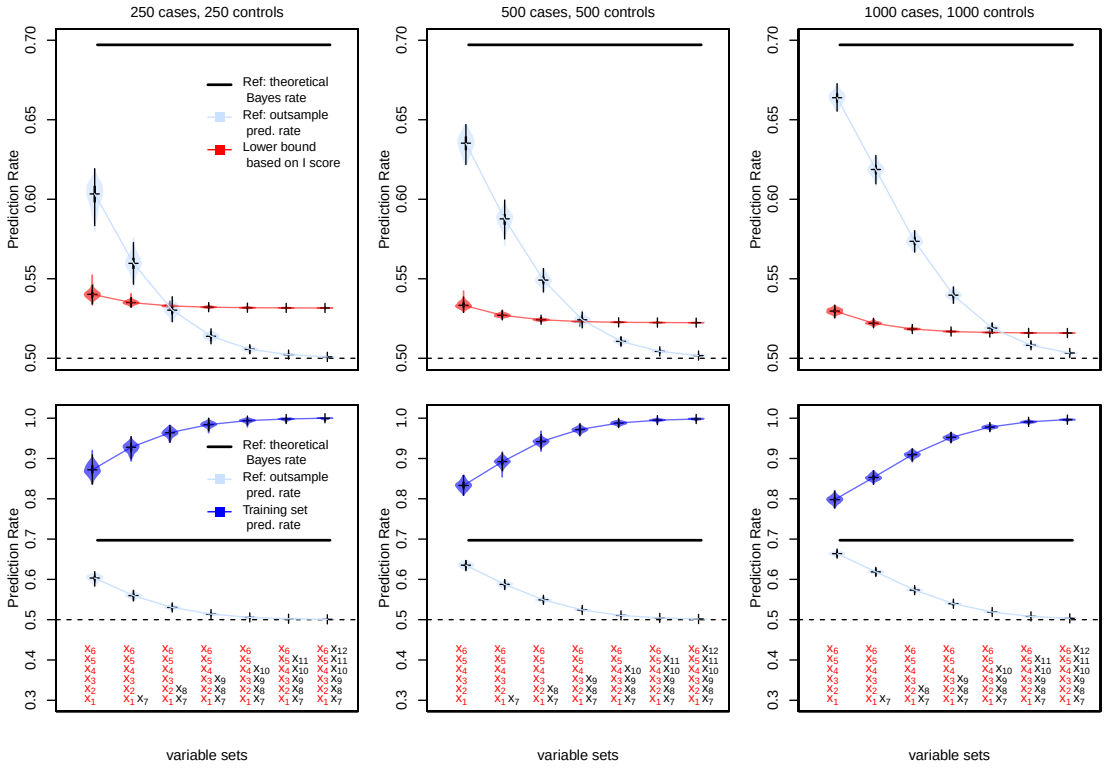


Figure B.1: **Variable set size 6: Comparison of the training rate and I-score against the out of sample prediction rate** Again we compare two statistics, I score lower bound and the training set prediction rate against the out of sample prediction rate. Lower bound from the I score is provided in red, training set prediction rate in blue, and the out of sample prediction rate is in light blue. The thick black line in all six graphs is the true Bayes rate. All x-axes correspond to variable sets (described in red for important variables and black for noisy ones) while all y-axes correspond to (correct) prediction rate. There are six important variables in this example, x_1, x_2, x_3, x_4, x_5, and x_6. The top row of graphs compares the (red) I score statistics against the (light blue) out of sample prediction rate. The lower row of graphs compares the (dark blue) training set prediction rate against the (light blue) out of sample prediction rate. From left to right the graphs increase in sample size from 500 cases and 500 controls, to 1000 cases and 1000 controls in the middle, to 2000 cases and 200 controls on the right.

Appendix C

Appendix for Chapter 3

C.1 Simulation details

Simulation details

Recall:

$$I = \sum_{i=1}^{n} (n_i^a - n_i^u)^2$$

Setting up the simulation. required libraries: glmnet, randomForest, psych, ggplot2, gplots, car, nnet, NeuralNetTools, Matrix

Functions

To compute I score

```
f.list.I=function(var.list, data.x, data.y){
    kk=length(var.list)
    if(kk>1){xx=data.x[,as.vector(var.list)]%*%as.vector((3^(0:(kk-1))))
        #values 1 3 9 for 3 variables
    }else{xx=as.matrix(data.x)[,var.list]}
    yy=unlist(data.y)
    dat.mat=table(xx,yy)
    n.d=dat.mat[,1]
    n.u=dat.mat[,2]
    nn.d=sum(n.d)
    nn.u=sum(n.u)
    i.score=nn.d*nn.u*sum((n.d/nn.d-n.u/nn.u)^2)/(nn.d+nn.u)
    return(c(var.list, i.score))
}
```

To compute group importance for RF

```
# uses library("randomForest")
var.share <- function(rf.obj, members) {
  count <- table(rf.obj$forest$bestvar)[-1]
  names(count) <- names(rf.obj$forest$ncat)
  share <- count[members] / sum(count[members])
  return(share)
}

group.importance <- function(rf.obj, groups) {
  var.imp <- as.matrix(sapply(groups, function(g) {
```

```
    sum(importance(rf.obj, 2)[g, ]*var.share(rf.obj, g))
  }))
  colnames(var.imp) <- "MeanDecreaseGini"
  return(var.imp)
}
```

Models 1-4 we assume we have 12 variables. ...

```
set.seed(323) #to reproduce
p.x=c(0.5, 0.5, 0.5, 0.4, 0.1,0.2,0.5,0.4,0.3,0.2,0.1,0.7)
xx=NULL
n=100
sim=100 #number of simulated datasets
x=vector("list", sim)
for(j in 1:sim){xx=NULL
  for(i in 1:length(p.x)){xx=cbind(xx, rbinom(n, 1, p.x[i]))}
  colnames(xx)=c("x1", "x2","x3","x4","x5","x6","x7","x8","x9",
                 "x10","x11","x12")
  xx=as.data.frame(xx)
  x[[j]]=xx
  rm(xx)
}
```

Generate Models 1-4

```
set.seed(323)
sim=100
y1=array(data=NA,c(sim,n,1))
#Model 1: linear additive x1+x2-x3
for(j in 1:sim){yy=(0.2*x[[j]][,1]+0.1*x[[j]][,2]-0.1*x[[j]][,3])
  y1[j,,]=yy
  rm(yy)}
```

Generate Y data for Models 1-4

```
set.seed(323)
sim=100
y1=y2=y3=y4=array(data=NA,c(sim,n,1))
#Model 1: linear additive x1+x2-x3
for(j in 1:sim){
  yy=(0.2*x[[j]][,1]+0.1*x[[j]][,2]-0.1*x[[j]][,3])
  y1[j,,]=yy
  rm(yy)
```

```
}
#Model 2: linear interactive x1*x2*x3*x4
for(j in 1:sim){
  yy=(x[[j]][,1]*x[[j]][,2]*x[[j]][,3]*x[[j]][,4])
  y2[j,,]=yy
  rm(yy)
}
#Model 3: nonlinear (x1+x2+x3)mod2
for(j in 1:sim){
  yy=(x[[j]][,1]+x[[j]][,2]+x[[j]][,3])%%2
  y3[j,,]=yy
  rm(yy)
}
#Model 4: nonlinear interactive (x1*x2*x3)mod2
for(j in 1:sim){
  yy=(5*x[[j]][,1]*x[[j]][,2]*x[[j]][,3])%%2
  y4[j,,]=yy
  rm(yy)
}
```

Generate Model 5 Now create 5th example where data is much noisier, and p>>n (not just p>n). We will have 100 datapoints (n=100), but there are 20 variables, with an interactive variable set predicting the outcome Y.

```
### 20 x variables
set.seed(323)
p.x2=c(0.5, 0.4, 0.5, 0.4, 0.6,0.5,0.5,0.4,0.3,0.2,0.1,0.7,0.2,
       0.3,0.4,0.5,0.1,0.4,0.5,0.1) #20variables
xx2=NULL
n=100
sim=100 #number of simulated datasets
x2=vector("list", sim)
for(j in 1:sim){
  xx2=NULL
for(i in 1:length(p.x2)){
  xx2=cbind(xx2, rbinom(n, 1, p.x2[i]))
}
  colnames(xx2)=c("x1", "x2","x3","x4","x5","x6","x7","x8","x9","x10","x11",
                  "x12","x13","x14","x15","x16","x17","x18","x19","x20")
  xx2=as.data.frame(xx2)
  x2[[j]]=xx2
  rm(xx2)
}
### Y outcome
```

```
set.seed(323)
sim=100
y5=array(data=NA,c(sim,n,1))
#Model 5:
for(j in 1:sim){
  yy2=(x2[[j]][,1]*x2[[j]][,2]*x2[[j]][,3])%%2
  y5[j,,]=yy2
  rm(yy2)
}
```

I-score VS for Models 1-4 #I-score VS Model 1

```
sim=100
list.c=NULL
  for(i in 1:12){
    list.c[[i]]=combn(12,i)}
# all 1way interactions
  iscore1=matrix(NA,sim,ncol(list.c[[1]]))
    for(i in 1:sim){
    for(j in 1:ncol(list.c[[1]])){
  iscore1[i,j]=f.list.I(c(list.c[[1]][1,j]), as.matrix(x[[i]]),
                        y1[i,,])[2]}}
  names=paste("x", format(1:length(p.x),trim=TRUE), sep="")
  colnames(iscore1)=names
#2way
  iscore2=matrix(NA,sim,ncol(list.c[[2]]))
  for(i in 1:sim){
    for(j in 1:ncol(list.c[[2]])){
  iscore2[i,j]=f.list.I(c(list.c[[2]][1,j],list.c[[2]][2,j]),
                        as.matrix(x[[i]]), y1[i,,])[3]}}
  #colnames2way
  n2=NULL
  for(i in 1:ncol(list.c[[2]])){
  n2[i]=paste(names[list.c[[2]][1,i]],names[list.c[[2]][2,i]],sep="*")}
  colnames(iscore2)=n2
#3way
    iscore3=matrix(NA,sim,ncol(list.c[[3]]))
  for(i in 1:sim){
    for(j in 1:ncol(list.c[[3]])){
  iscore3[i,j]=f.list.I(c(list.c[[3]][1,j],list.c[[3]][2,j],
            list.c[[3]][3,j]), as.matrix(x[[i]]), y1[i,,])[4]}}
      #colnames3way
  n3=NULL
  for(i in 1:ncol(list.c[[3]])){
```

```
  n3[i]=paste(names[list.c[[3]][1,i]],names[list.c[[3]][2,i]],
              names[list.c[[3]][3,i]],sep="*")}
  colnames(iscore3)=n3
### use above code to similarly create iscore4,...,iscore12 ###

#plot
dat=data.matrix(cbind(iscore1,iscore2,iscore3,iscore4,iscore5,iscore6,
                iscore7,iscore8,iscore9,iscore10,iscore11,iscore12))
mns <- colMeans(dat, na.rm=TRUE)
dat2 <- dat[,order(-mns)]
dat3=t(dat2)
dat4=dat3[1:50,]
#plot
heatmap.2(dat4,density.info="none",trace="none",labCol=NA,
          main="PR VS: Model 1 ",xlab="Simulations 1-100",
          ylab="Top 50 variable sets", margins=c(10,10),
          col=cm.colors(256),dendrogram='none',Rowv=FALSE,Colv=FALSE)
#### Example of i-score dropping #####
sim=100
iscore=matrix(NA,sim,12) #for groups var
for(i in 1:sim){
  iscore[i,1]=f.list.I(c(1), as.matrix(x[[i]]), y1[i,,])[2]
  iscore[i,2]=f.list.I(c(1,2), as.matrix(x[[i]]), y1[i,,])[3]
  iscore[i,3]=f.list.I(c(1,2,3), as.matrix(x[[i]]), y1[i,,])[4]
  iscore[i,4]=f.list.I(c(1,2,3,4),as.matrix(x[[i]]),y1[i,,])[5]
  iscore[i,5]=f.list.I(c(1,2,3,4,5),as.matrix(x[[i]]),y1[i,,])[6]
  iscore[i,6]=f.list.I(c(1,2,3,4,5,6),as.matrix(x[[i]]),y1[i,,])[7]
  iscore[i,7]=f.list.I(c(1,2,3,4,5,6,7),as.matrix(x[[i]]),y1[i,,])[8]
  iscore[i,8]=f.list.I(c(1,2,3,4,5,6,7,8),as.matrix(x[[i]]),y1[i,,])[9]
  iscore[i,9]=f.list.I(c(1,2,3,4,5,6,7,8,9),as.matrix(x[[i]]),y1[i,,])[10]
  iscore[i,10]=f.list.I(c(1,2,3,4,5,6,7,8,9,10),
                        as.matrix(x[[i]]),y1[i,,])[11]
  iscore[i,11]=f.list.I(c(1,2,3,4,5,6,7,8,9,10,11),
                        as.matrix(x[[i]]),y1[i,,])[12]
  iscore[i,12]=f.list.I(c(1,2,3,4,5,6,7,8,9,10,11,12),
                        as.matrix(x[[i]]),y1[i,,])[13]
}

var.names=c("x1","x1,x2","x1-x3","x1-x4","x1-x5","x1-x6","x1-x7",
            "x1-x8","x1-x9","x1-x10","x1-x11","x1-x12")
colnames(iscore)=var.names
#plot
boxplot(iscore,notch=FALSE,main="I-score VS: Model 1",
ylab="I-score",las=2)
```

```
error.bars(iscore,ylab="I-score",xlim=c(0,5),
           labels=var.names, add=TRUE)
abline(h=0)
dev.off()
```

I-score VS Model 2

```
sim=100
list.c=NULL
  for(i in 1:12){
    list.c[[i]]=combn(12,i)}
#all 1way interactions
  iscore1=matrix(NA,sim,ncol(list.c[[1]]))
    for(i in 1:sim){
    for(j in 1:ncol(list.c[[1]])){
  iscore1[i,j]=f.list.I(c(list.c[[1]][1,j]),
              as.matrix(x[[i]]), y2[i,,])[2]}}
  names=paste("x", format(1:length(p.x),trim=TRUE), sep="")
  colnames(iscore1)=names
#2way
  iscore2=matrix(NA,sim,ncol(list.c[[2]]))
  for(i in 1:sim){
    for(j in 1:ncol(list.c[[2]])){
  iscore2[i,j]=f.list.I(c(list.c[[2]][1,j],list.c[[2]][2,j]),
              as.matrix(x[[i]]), y2[i,,])[3]}}
  #colnames2way
  n2=NULL
  for(i in 1:ncol(list.c[[2]])){
  n2[i]=paste(names[list.c[[2]][1,i]],names[list.c[[2]][2,i]],sep="*")}
  colnames(iscore2)=n2
#3way
    iscore3=matrix(NA,sim,ncol(list.c[[3]]))
  for(i in 1:sim){
    for(j in 1:ncol(list.c[[3]])){
  iscore3[i,j]=f.list.I(c(list.c[[3]][1,j],list.c[[3]][2,j],
              list.c[[3]][3,j]),as.matrix(x[[i]]), y2[i,,])[4]}}
  #colnames3way
  n3=NULL
  for(i in 1:ncol(list.c[[3]])){
  n3[i]=paste(names[list.c[[3]][1,i]],names[list.c[[3]][2,i]],
              names[list.c[[3]][3,i]],sep="*")}
  colnames(iscore3)=n3
```

```
### use above code to similarly create iscore4,...,iscore12 ###

#plot
dat=data.matrix(cbind(iscore1,iscore2,iscore3,iscore4,iscore5,iscore6,
                iscore7,iscore8,iscore9,iscore10,iscore11,iscore12))
mns <- colMeans(dat, na.rm=TRUE)
dat2 <- dat[,order(-mns)]
dat3=t(dat2) #dat3 dim=4095x100
dat4=dat3[1:50,]
#plot
heatmap.2(dat4,density.info="none",trace="none",labCol=NA,
          main="PR VS: Model 2 ",xlab="Simulations 1-100",
          ylab="Top 50 variable sets", margins=c(10,10),
          col=cm.colors(256),dendrogram='none',Rowv=FALSE,Colv=FALSE)

###################
sim=100
iscore=matrix(NA,sim,12)
for(i in 1:sim){
  iscore[i,1]=f.list.I(c(1), as.matrix(x[[i]]), y2[i,,])[2]
  iscore[i,2]=f.list.I(c(1,3), as.matrix(x[[i]]), y2[i,,])[3]
  iscore[i,3]=f.list.I(c(1,2,3), as.matrix(x[[i]]), y2[i,,])[4]
  iscore[i,4]=f.list.I(c(1,2,3,4),as.matrix(x[[i]]),y2[i,,])[5]
  iscore[i,5]=f.list.I(c(1,2,3,4,5),as.matrix(x[[i]]),y2[i,,])[6]
  iscore[i,6]=f.list.I(c(1,2,3,4,5,6),as.matrix(x[[i]]),y2[i,,])[7]
  iscore[i,7]=f.list.I(c(1,2,3,4,5,6,7),as.matrix(x[[i]]),y2[i,,])[8]
  iscore[i,8]=f.list.I(c(1,2,3,4,5,6,7,8),as.matrix(x[[i]]),y2[i,,])[9]
  iscore[i,9]=f.list.I(c(1,2,3,4,5,6,7,8,9),as.matrix(x[[i]]),y2[i,,])[10]
  iscore[i,10]=f.list.I(c(1,2,3,4,5,6,7,8,9,10),
                        as.matrix(x[[i]]),y2[i,,])[11]
  iscore[i,11]=f.list.I(c(1,2,3,4,5,6,7,8,9,10,11),
                        as.matrix(x[[i]]),y2[i,,])[12]
  iscore[i,12]=f.list.I(c(1,2,3,4,5,6,7,8,9,10,11,12),
                        as.matrix(x[[i]]),y2[i,,])[13]
}
var.names=c("x1","x1,x3","x1-x3","x1-x4","x1-x5","x1-x6","x1-x7",
            "x1-x8","x1-x9","x1-x10","x1-x11","x1-x12")
colnames(iscore)=var.names
plot(c(0,3), c(0,14), xlab="variable sets",
                              ylab="I-score", type="n",
                              xaxt="n",
                              main="I-score VS: Model 2", cex=1.2)
boxplot(iscore,notch=FALSE,main="I-score VS: Model 2"
        ,ylab="I-score",las=2)
```

```
error.bars(iscore,ylab="I-score"#,xlab="Variable sets"
           ,xlim=c(0,5),labels=var.names, add=TRUE)
abline(h=0)
dev.off()
```

I-score VS Model 3

```
###################
sim=100
list.c=NULL
  for(i in 1:12){
    list.c[[i]]=combn(12,i)}
# all 1way interactions
  iscore1=matrix(NA,sim,ncol(list.c[[1]]))
    for(i in 1:sim){
    for(j in 1:ncol(list.c[[1]])){
  iscore1[i,j]=f.list.I(c(list.c[[1]][1,j]), as.matrix(x[[i]]),
                        y3[i,,])[2]}}
  names=paste("x", format(1:length(p.x),trim=TRUE), sep="")
  colnames(iscore1)=names
#2way
  iscore2=matrix(NA,sim,ncol(list.c[[2]]))
  for(i in 1:sim){
    for(j in 1:ncol(list.c[[2]])){
  iscore2[i,j]=f.list.I(c(list.c[[2]][1,j],list.c[[2]][2,j]),
              as.matrix(x[[i]]),y3[i,,])[3]}}
  #colnames2way
  n2=NULL
  for(i in 1:ncol(list.c[[2]])){
  n2[i]=paste(names[list.c[[2]][1,i]],names[list.c[[2]][2,i]],sep="*")}
  colnames(iscore2)=n2
#3way
    iscore3=matrix(NA,sim,ncol(list.c[[3]]))
  for(i in 1:sim){
    for(j in 1:ncol(list.c[[3]])){
  iscore3[i,j]=f.list.I(c(list.c[[3]][1,j],list.c[[3]][2,j],
               list.c[[3]][3,j]), as.matrix(x[[i]]), y3[i,,])[4]}}
  #colnames3way
  n3=NULL
  for(i in 1:ncol(list.c[[3]])){
  n3[i]=paste(names[list.c[[3]][1,i]],names[list.c[[3]][2,i]],
              names[list.c[[3]][3,i]],sep="*")}
```

```
  colnames(iscore3)=n3
### use above code to similarly create iscore4,...,iscore12 ###

#plot
dat=data.matrix(cbind(iscore1,iscore2,iscore3,iscore4,iscore5,iscore6,
                 iscore7,iscore8,iscore9,iscore10,iscore11,iscore12))
mns <- colMeans(dat, na.rm=TRUE)
dat2 <- dat[,order(-mns)]
dat3=t(dat2)
dat4=dat3[1:50,]
#plot
heatmap.2(dat4,density.info="none",trace="none",labCol=NA,
          main="PR VS: Model 3",xlab="Simulations 1-100",
          ylab="Top 50 variable sets",
margins=c(10,10),col=cm.colors(256),dendrogram='none',
    Rowv=FALSE,Colv=FALSE)
######################
sim=100
iscore=matrix(NA,sim,12)
for(i in 1:sim){
  iscore[i,1]=f.list.I(c(1), as.matrix(x[[i]]), y3[i,,])[2]
  iscore[i,2]=f.list.I(c(1,2), as.matrix(x[[i]]), y3[i,,])[3]
  iscore[i,3]=f.list.I(c(1,2,3), as.matrix(x[[i]]), y3[i,,])[4]
  iscore[i,4]=f.list.I(c(1,2,3,4),as.matrix(x[[i]]),y3[i,,])[5]
  iscore[i,5]=f.list.I(c(1,2,3,4,5),as.matrix(x[[i]]),y3[i,,])[6]
  iscore[i,6]=f.list.I(c(1,2,3,4,5,6),as.matrix(x[[i]]),y3[i,,])[7]
  iscore[i,7]=f.list.I(c(1,2,3,4,5,6,7),as.matrix(x[[i]]),y3[i,,])[8]
  iscore[i,8]=f.list.I(c(1,2,3,4,5,6,7,8),as.matrix(x[[i]]),y3[i,,])[9]
  iscore[i,9]=f.list.I(c(1,2,3,4,5,6,7,8,9),as.matrix(x[[i]]),y3[i,,])[10]
  iscore[i,10]=f.list.I(c(1,2,3,4,5,6,7,8,9,10),
                        as.matrix(x[[i]]),y3[i,,])[11]
  iscore[i,11]=f.list.I(c(1,2,3,4,5,6,7,8,9,10,11),
                        as.matrix(x[[i]]),y3[i,,])[12]
  iscore[i,12]=f.list.I(c(1,2,3,4,5,6,7,8,9,10,11,12),as.matrix(x[[i]]),
                        y3[i,,])[13]
}
var.names=c("x1","x1,x2","x1-x3","x1-x4","x1-x5","x1-x6","x1-x7",
            "x1-x8","x1-x9","x1-x10","x1-x11","x1-x12")
colnames(iscore)=var.names
plot(c(0,3), c(0,14), xlab="variable sets", ylab="I-score", type="n",
            xaxt="n",main="I-score VS: Model 3", cex=1.2)
boxplot(iscore,notch=FALSE,main="I-score VS: Model 3"
        ,ylab="I-score",las=2)
error.bars(iscore,ylab="I-score"#,xlab="Variable sets"
```

```
          ,xlim=c(0,5),labels=var.names, add=TRUE)
abline(h=0)
dev.off()
```

I-score VS Model 4

```
################
sim=100
list.c=NULL
  for(i in 1:12){
    list.c[[i]]=combn(12,i)}
#all 1way interactions
  iscore1=matrix(NA,sim,ncol(list.c[[1]]))
    for(i in 1:sim){
    for(j in 1:ncol(list.c[[1]])){
  iscore1[i,j]=f.list.I(c(list.c[[1]][1,j]), as.matrix(x[[i]]), y4[i,,])[2]
    }}
  names=paste("x", format(1:length(p.x),trim=TRUE), sep="")
  colnames(iscore1)=names
#2way
  iscore2=matrix(NA,sim,ncol(list.c[[2]]))
  for(i in 1:sim){
    for(j in 1:ncol(list.c[[2]])){
  iscore2[i,j]=f.list.I(c(list.c[[2]][1,j],list.c[[2]][2,j]),
                        as.matrix(x[[i]]), y4[i,,])[3]}}
  #colnames2way
  n2=NULL
  for(i in 1:ncol(list.c[[2]])){
  n2[i]=paste(names[list.c[[2]][1,i]],names[list.c[[2]][2,i]],sep="*")}
  colnames(iscore2)=n2
#3way
    iscore3=matrix(NA,sim,ncol(list.c[[3]]))
  for(i in 1:sim){
    for(j in 1:ncol(list.c[[3]])){
  iscore3[i,j]=f.list.I(c(list.c[[3]][1,j],list.c[[3]][2,j],
                list.c[[3]][3,j]),as.matrix(x[[i]]), y4[i,,])[4]}}
      #colnames3way
  n3=NULL
  for(i in 1:ncol(list.c[[3]])){
  n3[i]=paste(names[list.c[[3]][1,i]],names[list.c[[3]][2,i]],
              names[list.c[[3]][3,i]],sep="*")}
  colnames(iscore3)=n3
```

```
### use above code to similarly create iscore4,...,iscore12 ###

vars=data.matrix(cbind(iscore2[,1],iscore2[,22]))
colnames(vars)=c("x1*x2","x3*x4")
dat=data.matrix(cbind(vars,iscore1,iscore2[,2:21],
                      iscore2[,23:ncol(iscore2)],iscore3,
                      iscore4,iscore5,iscore6,iscore7,iscore8,
                      iscore9,iscore10,iscore11,iscore12))
test=dat[,3:ncol(dat)]
mns <- colMeans(test, na.rm=TRUE)
test2 <- test[,order(-mns)]
test3 =cbind(vars,test2)
test4=t(test3)
test5=test4[1:50,]
#plot
heatmap.2(test5,density.info="none",trace="none",labCol=NA,
          main="PR VS: Model 4",xlab="Simulations 1-100",
          ylab="Variable sets",
margins=c(10,10),col=cm.colors(256),dendrogram='none',
  Rowv=FALSE,Colv=FALSE)
## plot used for descending
dat=data.matrix(cbind(iscore1,iscore2,iscore3,iscore4,iscore5,iscore6,
                      iscore7,iscore8,iscore9,iscore10,iscore11,iscore12))
mns <- colMeans(dat, na.rm=TRUE)
dat2 <- dat[,order(-mns)]
dat3=t(dat2)
dat4=dat3[1:50,]
#plot
heatmap.2(dat4,density.info="none",trace="none",labCol=NA,
          main="PR VS: Model 4",xlab="Simulations 1-100",
          ylab="Top 50 variable sets",
margins=c(10,10),col=cm.colors(256),dendrogram='none',Rowv=FALSE,Colv=FALSE)
##################
sim=100
iscore=matrix(NA,sim,12)
for(i in 1:sim){
  iscore[i,1]=f.list.I(c(3), as.matrix(x[[i]]), y4[i,,])[2]
  iscore[i,2]=f.list.I(c(2,3), as.matrix(x[[i]]), y4[i,,])[3]
  iscore[i,3]=f.list.I(c(1,2,3), as.matrix(x[[i]]), y4[i,,])[4]
  iscore[i,4]=f.list.I(c(1,2,3,4),as.matrix(x[[i]]),y4[i,,])[5]
  iscore[i,5]=f.list.I(c(1,2,3,4,5),as.matrix(x[[i]]),y4[i,,])[6]
  iscore[i,6]=f.list.I(c(1,2,3,4,5,6),as.matrix(x[[i]]),y4[i,,])[7]
  iscore[i,7]=f.list.I(c(1,2,3,4,5,6,7),as.matrix(x[[i]]),y4[i,,])[8]
  iscore[i,8]=f.list.I(c(1,2,3,4,5,6,7,8),as.matrix(x[[i]]),y4[i,,])[9]
```

```
  iscore[i,9]=f.list.I(c(1,2,3,4,5,6,7,8,9),as.matrix(x[[i]]),y4[i,,])[10]
  iscore[i,10]=f.list.I(c(1,2,3,4,5,6,7,8,9,10),
                        as.matrix(x[[i]]),y4[i,,])[11]
  iscore[i,11]=f.list.I(c(1,2,3,4,5,6,7,8,9,10,11),
                        as.matrix(x[[i]]),y4[i,,])[12]
  iscore[i,12]=f.list.I(c(1,2,3,4,5,6,7,8,9,10,11,12),
                        as.matrix(x[[i]]),y4[i,,])[13]}
var.names=c("x3","x2,x3","x1-x3","x1-x4","x1-x5","x1-x6","x1-x7","x1-x8",
            "x1-x9","x1-x10","x1-x11","x1-x12")
colnames(iscore)=var.names
plot(c(0,3), c(0,14), xlab="variable sets",ylab="I-score", type="n",
            xaxt="n",main="I-score VS: Model 4", cex=1.2)
boxplot(iscore,notch=FALSE,main="I-score VS: Model 4"
        ,ylab="I-score",las=2)
error.bars(iscore,ylab="I-score"#,xlab="Variable sets"
           ,xlim=c(0,5),labels=var.names, add=TRUE)
abline(h=0)
dev.off()
```

I-score VS for Model 5 (noisier dataset with smaller n/p)

I-score VS for Model 5

```
################
sim=100
list.c=NULL
  for(i in 1:20){
    list.c[[i]]=combn(20,i)}
#all 1way interactions
  iscore1=matrix(NA,sim,ncol(list.c[[1]]))
    for(i in 1:sim){
    for(j in 1:ncol(list.c[[1]])){
  iscore1[i,j]=f.list.I(c(list.c[[1]][1,j]), as.matrix(x2[[i]]),
                        y5[i,,])[2]}}
  names=paste("x", format(1:length(p.x2),trim=TRUE), sep="")
  colnames(iscore1)=names
#2way
  iscore2=matrix(NA,sim,ncol(list.c[[2]]))
  for(i in 1:sim){
    for(j in 1:ncol(list.c[[2]])){
  iscore2[i,j]=f.list.I(c(list.c[[2]][1,j],list.c[[2]][2,j]),
                        as.matrix(x2[[i]]), y5[i,,])[3]}}
```

```
  #colnames2way
  n2=NULL
  for(i in 1:ncol(list.c[[2]])){
  n2[i]=paste(names[list.c[[2]][1,i]],
              names[list.c[[2]][2,i]],sep="*")}
  colnames(iscore2)=n2
#3way
    iscore3=matrix(NA,sim,ncol(list.c[[3]]))
  for(i in 1:sim){
    for(j in 1:ncol(list.c[[3]])){
  iscore3[i,j]=f.list.I(c(list.c[[3]][1,j],list.c[[3]][2,j],
              list.c[[3]][3,j]),as.matrix(x2[[i]]), y5[i,,])[4]}}
  #colnames3way
  n3=NULL
  for(i in 1:ncol(list.c[[3]])){
  n3[i]=paste(names[list.c[[3]][1,i]],names[list.c[[3]][2,i]],
              names[list.c[[3]][3,i]],sep="*")}
  colnames(iscore3)=n3
### use above code to similarly create iscore4,...,onwards ###

#########################################
dat=data.matrix(cbind(iscore1,iscore2,iscore3,iscore4,iscore5,iscore6,
    iscore7,iscore8))#,
    #iscore9,iscore10,iscore11,iscore12,iscore13,iscore14,iscore15,
    #iscore16,iscore17,iscore18,iscore19,iscore20))
test=cbind(dat[,211],dat[,15512],dat[,1:210],dat[,212:15511],
           dat[,15513:ncol(dat)])
colnames(test)[1]="x1*x2*x3"
colnames(test)[2]="x4*x5*x6*x7*x8"
mns <- colMeans(test, na.rm=TRUE)
test2 <- test[,order(-mns)]
test3=t(test2)
test4=test3[1:50,]
#plot
heatmap.2(test4,density.info="none",trace="none",labCol=NA,
          main="PR VS: Model 5 ",xlab="Simulations 1-100",
          ylab="Top 50 variable sets",margins=c(10,10),
col=cm.colors(256),dendrogram='none',Rowv=FALSE,Colv=FALSE)
##################
sim=100
iscore=matrix(NA,sim,20)
for(i in 1:sim){
  iscore[i,1]=f.list.I(c(3), as.matrix(x2[[i]]), y5[i,,])[2]
  iscore[i,2]=f.list.I(c(2,3), as.matrix(x2[[i]]), y5[i,,])[3]
```

```
  iscore[i,3]=f.list.I(c(1,2,3), as.matrix(x2[[i]]), y5[i,,])[4]
  iscore[i,4]=f.list.I(c(1,2,3,4),as.matrix(x2[[i]]),y5[i,,])[5]
  iscore[i,5]=f.list.I(c(1,2,3,4,5),as.matrix(x2[[i]]),y5[i,,])[6]
  iscore[i,6]=f.list.I(c(1,2,3,4,5,6),as.matrix(x2[[i]]),y5[i,,])[7]
  iscore[i,7]=f.list.I(c(1,2,3,4,5,6,7),as.matrix(x2[[i]]),y5[i,,])[8]
  iscore[i,8]=f.list.I(c(1,2,3,4,5,6,7,8),as.matrix(x2[[i]]),y5[i,,])[9]
  iscore[i,9]=f.list.I(c(1,2,3,4,5,6,7,8,9),as.matrix(x2[[i]]),y5[i,,])[10]
  iscore[i,10]=f.list.I(c(1,2,3,4,5,6,7,8,9,10),
                        as.matrix(x2[[i]]),y5[i,,])[11]
  iscore[i,11]=f.list.I(c(1,2,3,4,5,6,7,8,9,10,11),
                        as.matrix(x2[[i]]),y5[i,,])[12]
  iscore[i,12]=f.list.I(c(1,2,3,4,5,6,7,8,9,10,11,12),
                        as.matrix(x2[[i]]),y5[i,,])[13]
  iscore[i,13]=f.list.I(c(1,2,3,4,5,6,7,8,9,10,11,12,13),
                        as.matrix(x2[[i]]),y5[i,,])[14]
  iscore[i,14]=f.list.I(c(1,2,3,4,5,6,7,8,9,10,11,12,13,14),
                        as.matrix(x2[[i]]),y5[i,,])[15]
  iscore[i,15]=f.list.I(c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15),
                        as.matrix(x2[[i]]),y5[i,,])[16]
  iscore[i,16]=f.list.I(c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16),
                        as.matrix(x2[[i]]),y5[i,,])[17]
  iscore[i,17]=f.list.I(c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17),
                        as.matrix(x2[[i]]),y5[i,,])[18]
  iscore[i,18]=f.list.I(c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18),
                        as.matrix(x2[[i]]),y5[i,,])[19]
  iscore[i,19]=f.list.I(c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19),
                        as.matrix(x2[[i]]),y5[i,,])[20]
iscore[i,20]=f.list.I(c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20),
                        as.matrix(x2[[i]]),y5[i,,])[21]
}
var.names=c("x3","x2,x3","x1-x3","x1-x4","x1-x5","x1-x6","x1-x7","x1-x8",
            "x1-x9","x1-x10","x1-x11","x1-x12","x1-x13","x1-x14","x1-x15",
            "x1-x16","x1-x17","x1-x18","x1-x19","x1-x20")
colnames(iscore)=var.names
plot(c(0,3), c(0,14), xlab="variable sets",
                              ylab="I-score", type="n",
                              xaxt="n",
                              main="I-score VS: Model 5", cex=1.2)
boxplot(iscore,notch=FALSE,main="I-score VS: Model 5"#,xlab="variable sets"
        ,ylab="I-score",las=2)
error.bars(iscore,ylab="I-score"#,xlab="Variable sets"
           ,xlim=c(0,5),labels=var.names, add=TRUE)
abline(h=0)
dev.off()
```

Lasso VS for Models 1-4

Lasso VS Model 1

```
########model 1#######
mylasso1=vector("list",100)
las=vector("list",100)
coefs=NULL
for(i in 1:100){
  dat.int=model.matrix(~(x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11+x12)^12-1,
                       x[[i]])# with interactions up to 12way
  names=colnames(dat.int)
  names=gsub(":","",names)
  colnames(dat.int)=names
  dat.int=as.data.frame(dat.int)
  dat.int2=sparse.model.matrix(~.,dat.int)
  mylasso1[[i]] = cv.glmnet(as.matrix(dat.int2),y1[i,,],
                            nfolds=10,family="gaussian")
  las[[i]] = glmnet(as.matrix(dat.int2),y1[i,,],
                    lambda=mylasso[[i]]$lambda.min,family="gaussian")
  rm(dat.int,dat.int2)}
for(i in 1:100){
  sim.num[i]=paste("sim",i,sep="")}
sim.num=c("V1",sim.num)
#first merge
  a1=as.matrix(rownames(las1[[1]]$beta)[which(las1[[1]]$beta!=0)])
  b1=as.matrix((las1[[1]]$beta)[which(las1[[1]]$beta!=0)])
  c1=as.data.frame(cbind(a1,b1))
  a2=as.matrix(rownames(las1[[2]]$beta)[which(las1[[2]]$beta!=0)])
  b2=as.matrix((las1[[2]]$beta)[which(las1[[2]]$beta!=0)])
  c2=as.data.frame(cbind(a2,b2))
  coefs=merge(c1,c2,by="V1",all.x=TRUE,all.y=TRUE)
  colnames(coefs)=sim.num[1:3]
#remaining merges
for(i in 3:sim){
  a=as.matrix(rownames(las[[i]]$beta)[which(las[[i]]$beta!=0)])
  b=as.matrix((las[[i]]$beta)[which(las[[i]]$beta!=0)])
  c=as.data.frame(cbind(a,b))
  coefs=merge(coefs,c,by="V1",all.x=TRUE,all.y=TRUE)
  colnames(coefs)[ncol(coefs)]=sim.num[i+1]
  rm(a,b,c)}
coefs2=data.matrix(coefs[,2:ncol(coefs)])
```

```
cnames=coefs[,1]
row.names(coefs2)=cnames
mns <- rowMeans(coefs2, na.rm=TRUE)
coefs2 <- coefs2[order(-mns),]
# to plot:
heat_model=heatmap(coefs2,Rowv=NA,Colv=NA,col=cm.colors(256),
          scale="column",labCol=NA,main="Lasso VS: Model 1",
          xlab="Simulations 1-100",ylab="Variable sets",
          margins=c(5,10),keep.dendro=FALSE)
dev.off()
```

Lasso VS Model 2

```
mylasso1=vector("list", 100)
las=vector("list",100)
coefs=NULL
for(i in 1:100){#first 25 simulations
  dat.int=model.matrix(~(x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11+x12)^12-1,
                       x[[i]])# with interactions up to 12way
  names=colnames(dat.int)
  names=gsub(":","",names)
  colnames(dat.int)=names
  dat.int=as.data.frame(dat.int)
  dat.int2=sparse.model.matrix(~.,dat.int)
  mylasso1[[i]] = cv.glmnet(as.matrix(dat.int2),y2[i,,],
                  nfolds=10,family="gaussian") #4095 variable sets
  las[[i]] = glmnet(as.matrix(dat.int2),y2[i,,],
                    lambda=mylasso[[i]]$lambda.min,family="gaussian")
  rm(dat.int,dat.int2)}
sim.num=NULL #names holder for colnames 2:ncol
for(i in 1:100){ #names
  sim.num[i]=paste("sim",i,sep="")}
sim.num=c("V1",sim.num)
#first merge
a1=as.matrix(rownames(las1[[1]]$beta)[which(las1[[1]]$beta!=0)])
b1=as.matrix((las1[[1]]$beta)[which(las1[[1]]$beta!=0)])
c1=as.data.frame(cbind(a1,b1))
a2=as.matrix(rownames(las1[[2]]$beta)[which(las1[[2]]$beta!=0)])
b2=as.matrix((las1[[2]]$beta)[which(las1[[2]]$beta!=0)])
c2=as.data.frame(cbind(a2,b2))
coefs=merge(c1,c2,by="V1",all.x=TRUE,all.y=TRUE)
colnames(coefs)=sim.num[1:3]
```

```
#remaining merges
for(i in 3:sim){
  a=as.matrix(rownames(las[[i]]$beta)[which(las[[i]]$beta!=0)])
  b=as.matrix((las[[i]]$beta)[which(las[[i]]$beta!=0)])
  c=as.data.frame(cbind(a,b))
  coefs=merge(coefs,c,by="V1",all.x=TRUE,all.y=TRUE)
  colnames(coefs)[ncol(coefs)]=sim.num[i+1]
  rm(a,b,c)}
coefs2=data.matrix(coefs[,2:ncol(coefs)])
cnames=coefs[,1]
row.names(coefs2)=cnames
mns <- rowMeans(coefs2, na.rm=TRUE)
coefs2 <- coefs2[order(-mns),]
# to plot:
heat_model=heatmap(coefs2,Rowv=NA,Colv=NA,col=cm.colors(256),
            scale="column",labCol=NA,main="Lasso VS: Model 2",
            xlab="Simulations 1-100",ylab="Variable sets",
            margins=c(5,10),keep.dendro=FALSE)
dev.off()
```

Lasso VS Model 3

```
mylasso1=vector("list", 100)
las=vector("list",100)
coefs=NULL
for(i in 1:100){
  dat.int=model.matrix(~(x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11+x12)^12-1,
                       x[[i]])# with interactions up to 12way
  names=colnames(dat.int)
  names=gsub(":","",names)
  colnames(dat.int)=names
  dat.int=as.data.frame(dat.int)
  dat.int2=sparse.model.matrix(~.,dat.int)#save this as a sparse matrix
  mylasso1[[i]] = cv.glmnet(as.matrix(dat.int2),y3[i,,],
                  nfolds=10,family="gaussian") #4095 variable sets
  las[[i]] = glmnet(as.matrix(dat.int2),y3[i,,],
                    lambda=mylasso[[i]]$lambda.min,family="gaussian")
  rm(dat.int,dat.int2)}
sim.num=NULL
for(i in 1:100){ #names
  sim.num[i]=paste("sim",i,sep="")}
sim.num=c("V1",sim.num)
```

```
#first merge
a1=as.matrix(rownames(las[[1]]$beta)[which(las[[1]]$beta!=0)])
b1=as.matrix((las[[1]]$beta)[which(las[[1]]$beta!=0)])
c1=as.data.frame(cbind(a1,b1))
a2=as.matrix(rownames(las[[2]]$beta)[which(las[[2]]$beta!=0)])
b2=as.matrix((las[[2]]$beta)[which(las[[2]]$beta!=0)])
c2=as.data.frame(cbind(a2,b2))
coefs=merge(c1,c2,by="V1",all.x=TRUE,all.y=TRUE)
colnames(coefs)=sim.num[1:3]
#remaining merges
for(i in 3:sim){
  a=as.matrix(rownames(las[[i]]$beta)[which(las[[i]]$beta!=0)])
  b=as.matrix((las[[i]]$beta)[which(las[[i]]$beta!=0)])
  c=as.data.frame(cbind(a,b))
  coefs=merge(coefs,c,by="V1",all.x=TRUE,all.y=TRUE)
  colnames(coefs)[ncol(coefs)]=sim.num[i+1]
  rm(a,b,c)}
coefs2=data.matrix(coefs[,2:ncol(coefs)])
cnames=coefs[,1]
row.names(coefs2)=cnames
mns <- rowMeans(coefs2, na.rm=TRUE)
coefs2 <- coefs2[order(-mns),]
coefs3=coefs2[1:50,]
# to plot:
 heat_model=heatmap(coefs3,Rowv=NA,Colv=NA,col=cm.colors(256),
            scale="column",labCol=NA,main="Lasso VS: Model 3",
            xlab="Simulations 1-100",ylab="Top 50 variable sets",
            margins=c(5,10),keep.dendro=FALSE)
dev.off()
```

Lasso VS Model 4

```
mylasso1=vector("list", 100)
las=vector("list",100)
coefs=NULL
for(i in 1:100){
  dat.int=model.matrix(~(x1+x2+x3+x4+x5+x6+x7+x8+x9+x10+x11+x12)^12-1,
                       x[[i]])# with interactions up to 12way
  names=colnames(dat.int)
  names=gsub(":","",names)
  colnames(dat.int)=names
  dat.int=as.data.frame(dat.int)
```

```
  dat.int2=sparse.model.matrix(~.,dat.int)#save this as a sparse matrix
  mylasso1[[i]] = cv.glmnet(as.matrix(dat.int2),y4[i,,],nfolds=10,
                            family="gaussian") #4095 variable sets
  las[[i]] = glmnet(as.matrix(dat.int2),y4[i,,],
                    lambda=mylasso[[i]]$lambda.min,family="gaussian")
  rm(dat.int,dat.int2)}

#create dataframe rows are variables x1-x12,
#cols simulation numbers with first col holding var names.
#data are lasso coefficients not penalized to 0.
sim.num=NULL
for(i in 1:100){ #names
  sim.num[i]=paste("sim",i,sep="")}
sim.num=c("V1",sim.num)
#first merge
  a1=as.matrix(rownames(las[[1]]$beta)[which(las[[1]]$beta!=0)])
  b1=as.matrix((las[[1]]$beta)[which(las[[1]]$beta!=0)])
  c1=as.data.frame(cbind(a1,b1))
    a2=as.matrix(rownames(las[[2]]$beta)[which(las[[2]]$beta!=0)])
    b2=as.matrix((las[[2]]$beta)[which(las[[2]]$beta!=0)])
    c2=as.data.frame(cbind(a2,b2))
      coefs=merge(c1,c2,by="V1",all.x=TRUE,all.y=TRUE)
      colnames(coefs)=sim.num[1:3]
#remaining merges
for(i in 3:sim){
  a=as.matrix(rownames(las[[i]]$beta)[which(las[[i]]$beta!=0)])
  b=as.matrix((las[[i]]$beta)[which(las[[i]]$beta!=0)])
  c=as.data.frame(cbind(a,b))
  coefs=merge(coefs,c,by="V1",all.x=TRUE,all.y=TRUE)
  colnames(coefs)[ncol(coefs)]=sim.num[i+1]
  rm(a,b,c)}
coefs2=data.matrix(coefs[,2:ncol(coefs)])
cnames=coefs[,1]
row.names(coefs2)=cnames
mns <- rowMeans(coefs2, na.rm=TRUE)
coefs2 <- coefs2[order(-mns),]
coefs3=coefs2[1:50,]
#plot
 heat_model=heatmap(coefs3,Rowv=NA,Colv=NA,col=cm.colors(256),
            scale="column",labCol=NA,main="Lasso VS: Model 3",
            xlab="Simulations 1-100",ylab="Top 50 variable sets",
            margins=c(5,10),keep.dendro=FALSE)
dev.off()
```

Simulations for Model 5 do not run (too much computational burden)

RF VS for Models 1-4

RF VS Model 1

```
rf=vector("list", sim)
imp=vector("list",sim)
######## for marginals (1way)#####
for(i in 1:sim){
rf[[i]]=randomForest(x[[i]],as.factor(y1[i,,]),importance=TRUE)}
dat=NULL
for(i in 1:length(imp)){
  dat=cbind(dat, data.matrix(importance(rf[[i]],type=1)))}
  c=NULL
  for(i in 1:12){
    c[[i]]=combn(12,i)}
  names=paste("x", format(1:length(p.x),trim=TRUE), sep="")
########2way#####
ind=matrix(NA,nrow=2,ncol=ncol(c[[2]]))#all 2way
for(j in 1:ncol(c[[2]])){
a=c[[2]][,j][1]
b=c[[2]][,j][2]
ind[1,j]=names(x[[i]])[a]
ind[2,j]=names(x[[i]])[b]}
groups2=NULL
for(j in 1:ncol(ind)){
    groups2[j]=list(c(ind[1,j],ind[2,j]))}
a=as.data.frame(NA,nrow=length(groups2),ncol=1)
colnames(a)="var2"
for(i in 1:length(groups2)){
  aa=strsplit(groups2[[i]],'"')
  bb=paste(aa[[1]],aa[[2]],sep=",")
  a[i,1]=bb
  rm(aa,bb)}
g2.imp=matrix(NA,nrow=length(groups2),ncol=sim)
for(i in 1:sim){
  g2.imp[,i]=group.importance(rf[[i]], groups2)}
rownames(g2.imp)=a[,1]
dat=rbind(dat,g2.imp)
  rm(a,dat2,dat3,dat4,ind,b)
########3way########
ind=matrix(NA,nrow=3,ncol=ncol(c[[3]]))
```

```
for(j in 1:ncol(c[[3]])){
        a1=c[[3]][,j][1]
        a2=c[[3]][,j][2]
        a3=c[[3]][,j][3]
        ind[1,j]=names[a1]
        ind[2,j]=names[a2]
        ind[3,j]=names[a3]}
groups=NULL
for(j in 1:ncol(ind)){
    groups[j]=list(c(ind[1,j],ind[2,j],ind[3,j]))}
a=as.data.frame(NA,nrow=length(groups),ncol=1)
colnames(a)="var3"
for(i in 1:length(groups)){
  aa=strsplit(groups[[i]],'"')
  bb=paste(aa[[1]],aa[[2]],aa[[3]],sep=",")
  a[i,1]=bb
  rm(aa,bb)}
g3.imp=matrix(NA,nrow=length(groups),ncol=sim)
for(i in 1:sim){
  g3.imp[,i]=group.importance(rf[[i]], groups)}
rownames(g3.imp)=a[,1]
dat=rbind(dat,g3.imp)
### use above code to similarly create g4.imp,...,g12.imp
###plot
mns <- rowMeans(dat, na.rm=TRUE)
dat2 <- dat[order(-mns),]
dat3=dat2[1:50,]
#plot
heatmap.2(dat3,density.info="none",trace="none",labCol=NA,
          main="RF VS: Model 1 ",xlab="Simulations 1-100",
          ylab="Top 50 variable sets",margins=c(10,10),
          col=cm.colors(256),dendrogram='none',Rowv=FALSE,Colv=FALSE)
```

RF VS Model 2

```
rf=vector("list", sim)
imp=vector("list",sim)
######## for marginals (1way)#####
for(i in 1:sim){
rf[[i]]=randomForest(x[[i]],as.factor(y2[i,,]),importance=TRUE)}
dat=NULL
```

```
for(i in 1:length(imp)){
  dat=cbind(dat, data.matrix(importance(rf[[i]],type=1)))}
  c=NULL
  for(i in 1:12){
    c[[i]]=combn(12,i)}
  names=paste("x", format(1:length(p.x),trim=TRUE), sep="")
########2way#####
      ind=matrix(NA,nrow=2,ncol=ncol(c[[2]]))#all 2way
      for(j in 1:ncol(c[[2]])){
        a=c[[2]][,j][1]
        b=c[[2]][,j][2]
        ind[1,j]=names(x[[i]])[a]
        ind[2,j]=names(x[[i]])[b]}
groups2=NULL
for(j in 1:ncol(ind)){
    groups2[j]=list(c(ind[1,j],ind[2,j]))}
a=as.data.frame(NA,nrow=length(groups2),ncol=1)
colnames(a)="var2"
for(i in 1:length(groups2)){
  aa=strsplit(groups2[[i]],'"')
  bb=paste(aa[[1]],aa[[2]],sep=",")
  a[i,1]=bb
  rm(aa,bb)}
g2.imp=matrix(NA,nrow=length(groups2),ncol=sim)
for(i in 1:sim){
  g2.imp[,i]=group.importance(rf[[i]], groups2)}
rownames(g2.imp)=a[,1]
dat=rbind(dat,g2.imp)
  rm(a,dat2,dat3,dat4,ind,b)
########3way########
ind=matrix(NA,nrow=3,ncol=ncol(c[[3]]))#all 3way
      for(j in 1:ncol(c[[3]])){
        a1=c[[3]][,j][1]
        a2=c[[3]][,j][2]
        a3=c[[3]][,j][3]
        ind[1,j]=names[a1]
        ind[2,j]=names[a2]
        ind[3,j]=names[a3]}
groups=NULL
for(j in 1:ncol(ind)){
    groups[j]=list(c(ind[1,j],ind[2,j],ind[3,j]))}
a=as.data.frame(NA,nrow=length(groups),ncol=1)
colnames(a)="var3"
for(i in 1:length(groups)){
```

```
  aa=strsplit(groups[[i]],'"')
  bb=paste(aa[[1]],aa[[2]],aa[[3]],sep=",")
  a[i,1]=bb
  rm(aa,bb)}
g3.imp=matrix(NA,nrow=length(groups),ncol=sim)
for(i in 1:sim){
  g3.imp[,i]=group.importance(rf[[i]], groups)}
rownames(g3.imp)=a[,1]
dat=rbind(dat,g3.imp)
### use above code to similarly create g4.imp,...,g12.imp
###plot
mns <- rowMeans(dat, na.rm=TRUE)
dat2 <- dat[order(-mns),]
dat3=dat2[1:50,]
#plot
heatmap.2(dat3,density.info="none",trace="none",
          labCol=NA,main="RF VS: Model 2 ",xlab="Simulations 1-100",
          ylab="Top 50 variable sets", margins=c(10,10),
          col=cm.colors(256),dendrogram='none',Rowv=FALSE,Colv=FALSE)
```

RF VS Model 3

```
rf=vector("list", sim)
imp=vector("list",sim)
######## for marginals (1way)#####
for(i in 1:sim){
rf[[i]]=randomForest(x[[i]],as.factor(y3[i,,]),importance=TRUE)}
dat=NULL
for(i in 1:length(imp)){
  dat=cbind(dat, data.matrix(importance(rf[[i]],type=1)))}
  c=NULL
  for(i in 1:12){
    c[[i]]=combn(12,i)}
  names=paste("x", format(1:length(p.x),trim=TRUE), sep="")
########2way#####
      ind=matrix(NA,nrow=2,ncol=ncol(c[[2]]))#all 2way
      for(j in 1:ncol(c[[2]])){
        a=c[[2]][,j][1]
        b=c[[2]][,j][2]
        ind[1,j]=names(x[[i]])[a]
        ind[2,j]=names(x[[i]])[b]}
```

```
groups2=NULL
for(j in 1:ncol(ind)){
    groups2[j]=list(c(ind[1,j],ind[2,j]))}
a=as.data.frame(NA,nrow=length(groups2),ncol=1)
colnames(a)="var2"
for(i in 1:length(groups2)){
  aa=strsplit(groups2[[i]],'"')
  bb=paste(aa[[1]],aa[[2]],sep=",")
  a[i,1]=bb
  rm(aa,bb)}
g2.imp=matrix(NA,nrow=length(groups2),ncol=sim)
for(i in 1:sim){
  g2.imp[,i]=group.importance(rf[[i]], groups2)}
rownames(g2.imp)=a[,1]
dat=rbind(dat,g2.imp)
  rm(a,dat2,dat3,ind,b)
########3way########
ind=matrix(NA,nrow=3,ncol=ncol(c[[3]]))#all 3way
      ind=matrix(NA,nrow=3,ncol=ncol(c[[3]]))#all 2way
      for(j in 1:ncol(c[[3]])){
        a1=c[[3]][,j][1]
        a2=c[[3]][,j][2]
        a3=c[[3]][,j][3]
        ind[1,j]=names[a1]
        ind[2,j]=names[a2]
        ind[3,j]=names[a3]}
groups=NULL
for(j in 1:ncol(ind)){
    groups[j]=list(c(ind[1,j],ind[2,j],ind[3,j]))}
a=as.data.frame(NA,nrow=length(groups),ncol=1)
colnames(a)="var3"
for(i in 1:length(groups)){
  aa=strsplit(groups[[i]],'"')
  bb=paste(aa[[1]],aa[[2]],aa[[3]],sep=",")
  a[i,1]=bb
  rm(aa,bb)}
g3.imp=matrix(NA,nrow=length(groups),ncol=sim)
for(i in 1:sim){
  g3.imp[,i]=group.importance(rf[[i]], groups)}
rownames(g3.imp)=a[,1]
dat=rbind(dat,g3.imp)
### use above code to similarly create g4.imp,...,g12.imp
###plot
mns <- rowMeans(dat, na.rm=TRUE)
```

```
dat2 <- dat[order(-mns),]
dat3=dat2[1:50,]
#plot
heatmap.2(dat3,density.info="none",trace="none",
          labCol=NA,main="RF VS: Model 3 ",xlab="Simulations 1-100",
          ylab="Top 50 variable sets", margins=c(10,10),
          col=cm.colors(256),dendrogram='none',Rowv=FALSE,Colv=FALSE)
```

RF VS Model 4

```
rf=vector("list", sim)
imp=vector("list",sim)
######## for marginals (1way)#####
for(i in 1:sim){
rf[[i]]=randomForest(x[[i]],as.factor(y4[i,,]),importance=TRUE)}
dat=NULL
for(i in 1:length(imp)){
  dat=cbind(dat, data.matrix(importance(rf[[i]],type=1)))}
  c=NULL
  for(i in 1:12){
    c[[i]]=combn(12,i)}
  names=paste("x", format(1:length(p.x),trim=TRUE), sep="")
########2way#####
      ind=matrix(NA,nrow=2,ncol=ncol(c[[2]]))#all 2way
      for(j in 1:ncol(c[[2]])){
        a=c[[2]][,j][1]
        b=c[[2]][,j][2]
        ind[1,j]=names(x[[i]])[a]
        ind[2,j]=names(x[[i]])[b]}
groups2=NULL
for(j in 1:ncol(ind)){
    groups2[j]=list(c(ind[1,j],ind[2,j]))}
a=as.data.frame(NA,nrow=length(groups2),ncol=1)
colnames(a)="var2"
for(i in 1:length(groups2)){
  aa=strsplit(groups2[[i]],'"')
  bb=paste(aa[[1]],aa[[2]],sep=",")
  a[i,1]=bb
  rm(aa,bb)}
g2.imp=matrix(NA,nrow=length(groups2),ncol=sim)
for(i in 1:sim){
```

```
  g2.imp[,i]=group.importance(rf[[i]], groups2)}
rownames(g2.imp)=a[,1]
dat=rbind(dat,g2.imp)
  rm(a,dat2,dat3,ind,b)
########3way########
ind=matrix(NA,nrow=3,ncol=ncol(c[[3]]))#all 3way
      ind=matrix(NA,nrow=3,ncol=ncol(c[[3]]))#all 2way
      for(j in 1:ncol(c[[3]])){
        a1=c[[3]][,j][1]
        a2=c[[3]][,j][2]
        a3=c[[3]][,j][3]
        ind[1,j]=names[a1]
        ind[2,j]=names[a2]
        ind[3,j]=names[a3]}
groups=NULL
for(j in 1:ncol(ind)){
    groups[j]=list(c(ind[1,j],ind[2,j],ind[3,j]))}
a=as.data.frame(NA,nrow=length(groups),ncol=1)#each row is name of three vars
colnames(a)="var3"
for(i in 1:length(groups)){
  aa=strsplit(groups[[i]],'"')
  bb=paste(aa[[1]],aa[[2]],aa[[3]],sep=",")
  a[i,1]=bb
  rm(aa,bb)}
g3.imp=matrix(NA,nrow=length(groups),ncol=sim)
for(i in 1:sim){
  g3.imp[,i]=group.importance(rf[[i]], groups)}
rownames(g3.imp)=a[,1]
dat=rbind(dat,g3.imp)
### use above code to similarly create g4.imp,...,g12.imp
###plot
mns <- rowMeans(dat, na.rm=TRUE)
dat2 <- dat[order(-mns),]
dat3=dat2[1:50,]
#plot
heatmap.2(dat3,density.info="none",trace="none",labCol=NA,
          main="RF VS: Model 4",xlab="Simulations 1-100",
          ylab="Top 50 variable sets", margins=c(10,10),
          col=cm.colors(256),dendrogram='none',Rowv=FALSE,Colv=FALSE)
```

RF VS for Model 5 (noisier dataset with smaller n/p)

RF VS for Model 5

```
rf=vector("list", sim)
imp=vector("list",sim)
######## for marginals (1way)#####
for(i in 1:sim){
rf[[i]]=randomForest(x2[[i]],as.factor(y5[i,,]),importance=TRUE)}
dat=NULL
for(i in 1:length(imp)){
  dat=cbind(dat, data.matrix(importance(rf[[i]],type=1)))}
  c=NULL
  for(i in 1:20){
    c[[i]]=combn(20,i)}
  names=paste("x", format(1:length(p.x2),trim=TRUE), sep="")
########2way#####
      ind=matrix(NA,nrow=2,ncol=ncol(c[[2]]))#all 2way
      for(j in 1:ncol(c[[2]])){
        a=c[[2]][,j][1]
        b=c[[2]][,j][2]
        ind[1,j]=names(x2[[i]])[a]
        ind[2,j]=names(x2[[i]])[b]}
groups2=NULL
for(j in 1:ncol(ind)){
    groups2[j]=list(c(ind[1,j],ind[2,j]))}
a=as.data.frame(NA,nrow=length(groups2),ncol=1)
colnames(a)="var2"
for(i in 1:length(groups2)){
  aa=strsplit(groups2[[i]],'"')
  bb=paste(aa[[1]],aa[[2]],sep=",")
  a[i,1]=bb
  rm(aa,bb)}
g2.imp=matrix(NA,nrow=length(groups2),ncol=sim)
for(i in 1:sim){
  g2.imp[,i]=group.importance(rf[[i]], groups2)}
rownames(g2.imp)=a[,1]
dat=rbind(dat,g2.imp)
  rm(a,dat2,dat3,ind,b)
########3way########
ind=matrix(NA,nrow=3,ncol=ncol(c[[3]]))#all 3way
      ind=matrix(NA,nrow=3,ncol=ncol(c[[3]]))#all 2way
      for(j in 1:ncol(c[[3]])){
        a1=c[[3]][,j][1]
        a2=c[[3]][,j][2]
```

```
        a3=c[[3]][,j][3]
        ind[1,j]=names[a1]
        ind[2,j]=names[a2]
        ind[3,j]=names[a3]}
groups=NULL
for(j in 1:ncol(ind)){
    groups[j]=list(c(ind[1,j],ind[2,j],ind[3,j]))}
a=as.data.frame(NA,nrow=length(groups),ncol=1)
colnames(a)="var3"
for(i in 1:length(groups)){
  aa=strsplit(groups[[i]],'"')
  bb=paste(aa[[1]],aa[[2]],aa[[3]],sep=",")
  a[i,1]=bb
  rm(aa,bb)}
g3.imp=matrix(NA,nrow=length(groups),ncol=sim)
for(i in 1:sim){
  g3.imp[,i]=group.importance(rf[[i]], groups)}
rownames(g3.imp)=a[,1]
dat=rbind(dat,g3.imp)
### use above code to similarly create g4.imp,...,g12.imp
###plot
mns <- rowMeans(dat, na.rm=TRUE)
dat2 <- dat[order(-mns),]
dat3=dat2[1:50,]
#plot
heatmap.2(dat3,density.info="none",trace="none",labCol=NA,
          main="RF VS: Model 5",xlab="Simulations 1-100",
          ylab="Top 50 variable sets", margins=c(10,10),
          col=cm.colors(256),dendrogram='none',Rowv=FALSE,Colv=FALSE)
```