Contents

Foreword

The work for this thesis consisted for the most part of coding, executing, interpreting and writing down the results of a simulation study. Conceptual foundation of this study, as well as guidance in planning and executing, was given by professor Thas, for which I am thankful. Unless explicitly specified, all code and text is written by myself. The data for the simulation study was generated by myself on my laptop with its own limitations. The data used in the example analysis comes from a publicly available data set that can be downloaded from http://bioinf.wehi.edu.au/limma/data/ecoli-lrp.zip.

The most important people to thank are my wife and daughter, for their patience and understanding. They have believed in me and have supported me from day one. Tine and Oona, completing the mastat while making the combination with work and family life has been an utter challenge and I could not have done it without you.

Summary

As we currently reside in the era of big data, artificial intelligence and machine learning methods are on a rising tide. Interest in these methods are ever increasing since constraints in terms of computing power, storage capacity and connectivity are less of a concern. However, in certain settings, data gathering is still limited in sample size due to high costs and limited resources. More specifically, in early drug development research high throughput techniques are often used in small-sized experiments. The high dimensional data sets that are produced by these experiments often include tens of thousands of features and only 10 to 50 observations. Researchers in this field hope they can exploit the power of machine learning and AI to make more and better discoveries that would eventually lead to new and better drugs or diagnostic tests. Yet, since the main application of machine learning is in settings where data sets consist of thousands or even millions of observations and only a fraction of features, little is known about how these techniques empirically perform in such extreme situations. As such, it is unclear how these data sets in early drug discovery are best approached.

To gain insight in the limitations of these methods we performed a simulation study wherein 5 methods (t-tests with FDR control (FDR), t-tests with an empirical Bayes based correction based on Tweedie's formula (Tweedie), LASSO logistic regression (LASSO), Boruta, and SVM-RFE (SVM)) were tested in various circumstances.

Overall, the results indicate that there certainly is 'no free lunch.' That is to say in many situations methods fail or perform poorly. However, some guidelines are presented. The approach of the analyst will depend on their intentions. FDR is a very conservative method that often fails to discover true features. SVM on the other hand often discovers many features but has the drawback of finding many false features. Tweedie often performs somewhere in between, which can be a good thing. If the aim is to build a good prediction model LASSO is often a better choice.

1 Introduction

Artificial intelligence (AI) and machine learning (ML) are gaining popularity because of the successes booked by the application of these techniques in the last decades. Scientists from various fields believe in the power of these methods and thus want to apply those methods to their own problems in their respective fields. In doing so, they carry high hopes this will eventually result in more discoveries in less time.

The left-hand panel of figure 1 shows the number of publications that are selected from the Web of Science Core Collection via a search with the search term 'machine learning,' per year from 1997 to 2021 (the current year, 2021, being incomplete). The right panel of figure 1 shows the same statistics for the more general search term 'statistics.' From these graphs we can see that although the absolute numbers are less for machine learning than for statistics, the number of publications that mention machine learning are growing more rapidly. The right-hand panel demonstrates that rise in number of research articles that mention machine learning cannot solely be explained by the growth of the number of published research articles in general.

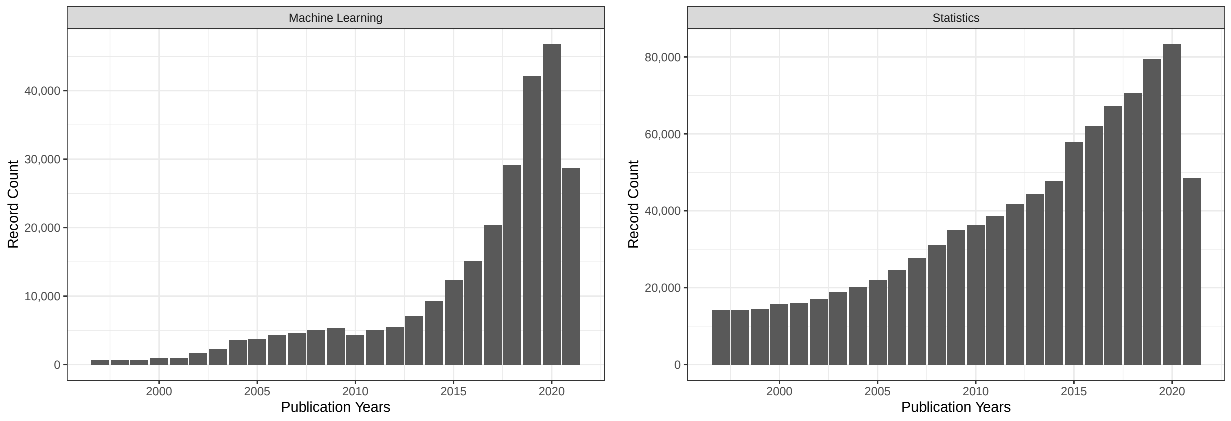


Figure 1: *Left panel: Number of publications selected from the Web of Science Core Collection with the search term 'machine learning', per year. Right panel: Number of publications selected from the Web of Science Core Collection with the search term 'statistics', per year. Searches were carried out on 08/14/2021. Note that the year 2021 is incomplete.*

We have showed that AI and ML are increasingly popular within research. AI and ML are also increasingly effective. This is not only because of the continuing development of new and better algorithms, but also because of advances in computing power and the ever increasing amounts of data that can be fed into these machine learning models. Indeed, a recent study showed empirically that model selection gets less important as training sets take on extreme proportions, suggesting that *"[...] the scientific community should perhaps focus more on data curation than on algorithm development"* (Halevy, Norvig, and Pereira 2009), [very_large_corpora], (Géron 2020).

We are now in the era of 'big data,' where the amount of data that are created is unimaginably high. However, while it is true that for example high-throughput technologies such as **single cell sequencing** produce huge amounts of data, knowledge extraction from these data sources still remains a difficult task. Why that is has to do with what is called the curse of dimensionality which is discussed in the following subsection.

1.1 The curse of dimensionality

In this thesis we want to focus on the analysis of data that are used for biomarker discovery within pre-clinical pharmaceutical research. In this field, the researcher is typically interested in finding results that ultimately will allow for quick and cost-effective identification or prediction of clinically relevant outcomes. Examples of such outcomes are the presence of a disease, the acceptance or rejection of a transplanted organ, the presence of severe side effects, etcetera. In this thesis we will focus on binary classification problems.

Biomarker candidates are typically found in 'omics' data. This could be genomic, transcriptomic, proteomic etc. data or even a combination. The datasets in these settings are typically (ultra) high dimensional, i.e. data where the number of features far exceeds the number of instances. This is often denoted as p » n where p is the number of features and n is the number of samples. In the case of biomarker discovery research the dimensionality of datasets are typically in the range of thousands to ten thousands.

This high dimensionality introduces serious challenges for the analyst who wants to extract knowledge from these datasets. These challenges are often referred to as the *curse of dimensionality*. The curse of dimensionality is called this way because there are a number problems that arise when the dimensionality of a dataset increases. More so, these problems get worse very quickly with further increasing dimensionality.

For many people, this could be counterintuitive as it apparently goes against what people remember from their statistics classes, that is: the adage that more data is better. The confusion could come from the faulty belief that 'more dimensions' means the same as 'more information.' While it is true that there are more entries in a 100-dimensional dataset than in a 10-dimensional dataset, for a constant number of instances there are, however, an equal amount of data points in both datasets. The reason this causes problems has to do with the geometry of high dimensional spaces. Two important reasons are briefly discussed below. This discussion is not meant to be exhaustive.

Firstly, as the dimensionality increases, the volume of the data-space increases exponentially. This means, with the same number of samples, an increase in dimensions leads to a sparser populated data-space compared to lower dimensions. The sampling density is proportional to $n^{1/p}$, so one needs n^p samples in p-dimensional space in order to have the same sampling density as in one-dimensional space (Hastie, Tibshirani, and Friedman 2009). This is a problem because in realistic settings where the cost of biological samples is very high, one cannot gather much more data than 10 to 50, maybe at most 100 samples. Contrasting this number with the number of features (often thousands or more) shows that the data-space is immensely sparse populated. Lower sampling density will then result in higher variance of estimates and thus in decreased performance.

A second property of high dimensional space that is related to decreased performance of statistical methods is the fact that in high dimensional space most of the data-points are rather 'extreme' in one way or another. Hastie, Tibshirani, and Friedman (2009) showed that the median distance from the origin to each of 500 uniformly distributed data-points in 10-dimensional unit space is more than halfway. They conclude that more than half of the points are closer to the boundary than to any other data point. This is a problem because prediction is much more difficult near the edges of the training space because that relies more on extrapolation rather than interpolation.

Now it is clear there are challenges with high dimensional data, we now shift our attention to different methods to analyze these kinds of data.

1.2 Methods for biomarker discovery with high dimensional data

In this thesis we want to compare 5 different statistical methods for the analysis of omics data in biomarker discovery. The selection of these methods was partially based on what is actually used in research departments of pharmaceutical companies (FDR, LASSO), partially on theoretical properties (Tweedie), and partially on a brief scan of the literature (Boruta, SVMRFE).

1.2.1 T-tests with false discovery rate control

The Welch t-test is a relatively simple method that is used to find out if 2 groups have different means at the population level. In the case of biomarker discovery one t-test for every feature is performed. Because of the vast amount of features in these kinds of datasets, this strategy without any adjustment is problematic. Typically a researcher wants to control the type-I error rate at a certain level (say 0.05). While Welch's t-test controls the type-I error well at the level of the individual test, the type-I error rate inflates when multiple tests are performed within one study at the level of the study which is more of interest. This well known phenomenon is known as the multiple testing problem. Apart from the inflation at the 'family' level, there is also the practical problem of having on average 500 false positive results out of 10,000 comparisons (when the null hypothesis holds for every comparison).

A lot of strategies to cope with this problem have been proposed, of which Bonferroni correction is a well known example. Due to its very conservativeness that method is less suited for large scale hypothesis testing. Most often one choses not to control the family wise error rate but the false discovery rate (FDR) instead, and use the method of Benjamini and Hochberg (1995). There are other FDR methods but we will focus on Benjamini and Hochberg (1995).

This method is based on the knowledge that p-values that come from null-comparisons are uniformly distributed and that the distribution of p-values that come from true differences are skewed with a mode close to 0. This knowledge makes it possible to adjust the p-values so that only 5% of the discoveries are expected to be false.

An example of a study that use this method is Marcišauskas et al. (2019).

1.2.2 T-tests with emperical Bayes based correction using Tweedie's formula

Selection bias is the phenomenon of overestimating population parameters based on extreme observations. This phenomenon is also known as regression to the mean. If one would throw many dice, we would expect the mean number of pips to be around 3.5. If one would then rethrow those dice that resulted a 6, the mean would again expected to be 3.5. This is an example of regression to the mean. The mean of the selected dice (6) regresses back to the overall mean (3.5). Expecting a higher mean would be an example of selection bias.

Efron (2011) describes Tweedie's formula as a means to deal with this selection bias. Tweedie's formula was introduced in Robbins (1956). It assumes that observations z have a true mean μ, come from a distribution $N(\mu, \sigma^2)$ and that every μ has an unknown prior distribution $g(\mu)$. The formula is $E\{\mu|z\} = z + \sigma^2 l'(z)$ and an estimate $\hat{\mu}$ can be obtained by applying following estimator: $z + \sigma^2 \hat{l}'(z)$

where $l'(z)$ is the first derivative of the marginal log-density of z, evaluated at z. Efron (2011) points out that the formula is practical because of the fact that one only needs an estimate of the marginal (log-) density of z, and no estimate of the prior. Efron (2011) suggests estimating the log-density by applying Lindsey's method. Lindsey's method is described in Efron (2008) and comes down to fitting a Jth degree polynomial poisson regression to the counts of the observations z in a number of bins. In other words, Lindsey's method tries to fit a Jth degree polynomial to a histogram. Because the Jth degree polynomial is of form $f(z) = exp\{\sum_{j=0}^{J} \beta_j z^j\}$, it is convenient to take the logarithm and calculate the derivative from there (Efron 2011).

Efron (2011) shows there is a connection between Tweedie's formula and the FDR method of Benjamini and Hochberg (1995) and the local false discovery rate.

Because of the selection-bias eliminating characteristics of Tweedie's formula it is interesting to see how it performs next to more popular methods such as FDR control and LASSO.

To our knowledge no implementation of the above is available in the form of R-code so one aim of this thesis is to implement this ourselves.

1.2.3 L1-penalized logistic regression

Logistic regression is a commonly used method to model the conditional probability of an observation belonging to one group (vs. an other group), given the values of the independent variables for that observation. That model and its probability as output is then often used to make predictions on the group status of new observations.

The benefits of regularization are well known and include among others protection against overfitting which is especially of importance in high dimensional settings. Different kinds of regularization are known among which L1-penalization or LASSO is a popular choice. LASSO, introduced in Tibshirani (1996), adds an L1-norm penalty term to the loss function (which is log loss for logistic regression) which penalizes large values for the coefficients and effectively shrinks them towards 0. Note that the intercept is kept out of this term and thus does not contribute to the penalization. A penalization parameter λ is introduced to allow control of the amount of penalization. This parameter can vary from 0 (which results in no regularization) to infinity (which results in a model with only an intercept) and can be chosen through cross validation.

The extra benefit of the LASSO compared to for example L2-norm penalization is that it not only shrinks coefficients towards 0, it also sets some coefficients at exactly 0. As a result, LASSO usually results in a more parsimonious model as it 'selects' only those features that are deemed necessary. For this reason, this method is attractive for use in biomarker discovery.

1.2.4 Boruta

Boruta (Kursa and Rudnicki 2010) is a random forest based feature selection method. Random forest (L 2001) is a method in which an ensemble of decision trees, in which each tree is built on a bootstrap sample of the original dataset and in which at each branch only a random subset of the variables are considered. In each ensemble, a fixed number of random trees are built, hence 'random forest.' A property of the random forest algorithm is that it introduces repeated randomness to mitigate the fact that a single decision tree is a high variance method.

The Boruta method works as follows. For each feature in the dataset a 'shadow' feature is created. These shadow features are copies of the original features that are shuffled across samples. Once the shadow features are added to the dataset, a random forest is fitted. The variable importance of every feature is then evaluated against the mean shadow feature importance. This process is then repeated until all features are either categorized important or unimportant or until a certain number of runs is reached.

1.2.5 Support vector machine recursive feature elimination

Support vector machine recursive feature elimination (SVMRFE) is method that consists of iteratively fitting a support vector machine model to correctly classify the subjects. At the end of each iteration the variables with the lowest variable importance are discarded from the feature pool before a new iteration begins.

Support vector machines are a family of models that, for classification, try to find a maximally separating hyperplane between two classes in the data space. An other way to look at support vector machines is that they minimize hinge loss. To avoid severe overfitting, a form of regularization is used (L2 penalization). In using support vector machines, one needs to choose a kernel. In the simulation study we used a linear kernel.

2 Methods

In this section we follow the guidelines of Morris et al. in that we structure the methods section according to the ADEMP acronym, which stands for aims, data-generating mechanisms, estimands or targets, methods, and performance measures.

2.1 Aims

We want to examine, by means of a simulation study, several properties of the 5 aforementioned methods in the context of pre-clinical pharmacological research where high dimensional (omics) data on only very few subjects are to be analyzed. These properties are:

- the mean number of true / false detections
- the chance of doing at least one true / false detection
- the rate of null-results
- the duration of the analysis
- discriminative ability for the model based methods

In doing so we hope to identify:

- conditions under which the methods under investigations fail or break or, as we call it, boundary conditions
- relative strengths

2.2 Data-generating mechanisms

We will simulate data using a simple parametric model (normal distribution) that will allow to manipulate the data easily.

We will consider a number of different scenario's for this simulation study. To this end we will vary:

- the number of subjects, n
- the number of features, p
- the number of significant features, sig_p
- the difference in population mean between the 2 groups for the significant features, or effect size, d
- correlation structure

We have simulated a number of different scenarios using a low number of repetitions to explore which parameter settings yield the most interesting results. These can be found in appendix A. From this first round of simulations we could select scenarios in a more informed manner. These scenarios were then simulated one hundred times.

In each scenario there will be 2 groups (k, treatment vs. control) of equal size. In the scenarios with no correlation between features, the data will be simulated following $ X_{ij}^{k}$ distributed as $N(k\delta_j, 1)$, with i = 1 … n, j = 1 … p, k = 0, 1 and $\delta_j = d$ for j = 1…sig_p and $\delta_j = 0$ for j = sig_p+1…p.

In the scenarios with correlation we use a block correlation structure for the correlation matrix following the same architecture as in Acharjee et al. (2020), that is 6 groups of 10 features that are highly intra-group correlated and a seventh rest group with no correlation (see figure 2). To achieve the block correlation structure we use the simcor function from the MarkJBrewer/ICsims package (Brewer, n.d.). The intra-group correlations were set at 0.9, the other correlations were set at 0.

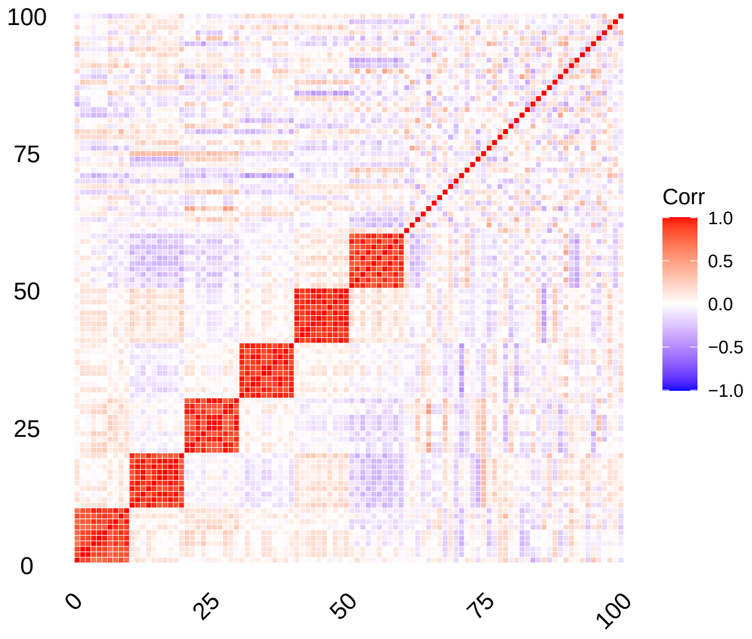


Figure 2: *First 100 features of an example of the block correlation structure used in the correlated scenarios.*

2.3 Targets and performance measures

The statistical task we are evaluating the methods on is model selection (for the models) and hypothesis testing (for the t-tests). We want to measure how good the methods can discriminate between significant and non-significant features. We want as many true positives as possible while keeping the false positives as low as possible.

The performance measures we are using are the mean number of true/false detections and the chance of true/false detection. Additionally we are also measuring the rate of null-results, and the duration of the analyses. We define null-results as results with no positives, true or false.

The discriminative ability of the model based methods will be measured with the AUC. After the feature selection, a model will be built with those features. That model will then be tested on a separate test set (see below). LASSO will just use its output model. Boruta results in a random forest model, and SVM-RFE results in a SVM.

2.4 Implementation of methods

This sections discusses the practicalities of the implementation of the different methods, allowing reproducibility. For a substantive discussion of the methods, please refer to sections 1.2.1 to 1.2.5.

We have 3 model based methods and 2 hypothesis test based methods. The hypothesis test based methods are performed using all available data. In the case of the model based methods, a separate train and test set is created using a 1:2 ratio (2/3 train set, 1/3 test set). This train-test split is done to reflect real life analyses and the 1/3 test set size is chosen because the more standard 20% test split size did not work in the scenarios with sample size of 10.

2.4.1 T-tests with false discovery rate control

In a first step the p t-tests are performed. Then, the corrections to the p-values are made, turning them into q-values, using the p.adjust function and using the BH (Benjamini and Hochberg 1995) method. After this correction, each feature for which holds that the q-value < 0.05 is stored as a detection.

2.4.2 T-tests with Tweedie correction

In a first step the p t-tests are performed. Each test statistic is stored and is binned into one of 30 bins. The number of test statistics per bin is counted. This comes essentially down to creating a histogram of the test statistics. Then a 5th degree polynomial poisson regression is fitted with the counts as dependent an the middle of each bin as independent variable. The result is a model that approximates the empirical probability density function of the test statistics. In a next step the variance of the test statistics is estimated and the derivative of the approximated density function is evaluated for each test statistic. At this point we have everything needed to apply Tweedie's formula $E\{\mu|z\} = z + \sigma^2 l'(z)$ and we calculate the corrected test statistics and according p-values. Each feature for which holds that the p-value < 0.05 is stored as a detection.

2.4.3 L1 penalized logistic regression

L1 penalized logistic regression has a hyperparameter (the penalization parameter λ) that needs to be tuned. The preffered way to do this is to do a grid search of possible values for λ in conjunction with cross validation. We did a 5-fold cross validation with the cv.glmnet function of the glmnet package (Tibshirani 1996), using the AUC as criterion. The cv.glmnet function provides parallel support which was used for $p > 1000$. This threshold was chosen after initial experimentation and profiling of the code. Once the optimal value for λ was chosen, i.e. the minimal value for lambda for which the AUC is maximized, a final model was fitted with this value for λ. The selected features were then stored as detections.

A construction to catch errors was necessary for the scenarios with extreme small sample sizes (10) because in those cases it happens that some folds have 0 observations in one of the 2 classes.

Table 1: Used packages related to methods

Method	Package	Citation
Boruta	Boruta	Kursa and Rudnicki (2010)
Logistic LASSO	glmnet	Friedman, Hastie, and Tibshirani (2010)
SVMRFE	SVM-RFE	Duan et al. (2005)

2.4.4 Boruta

Boruta is very easy to implement. The Boruta package (Kursa and Rudnicki 2010) provides a Boruta function to which you can pass the data and several other parameters. We set the critical p-value at 0.05 and the number of trees at 500. Only features that had a "confirmed" status after the algorithm was finished were treated as detected features.

2.4.5 SVMRFE

We used a variant of SVMRFE which uses cross validation to achieve more stability. As with the L1 penalized logistic regression we chose to do a 5-fold cross validation. To speed up the computation, it is possible to tell the algorithm to halve the number of features each iteration until a certain number of features is reached. We chose a number (250) that was not too high, so that we could effectively speed up the computation, nor too low, so that the risk of accidentally eliminating significant features early on was kept at a minimum. As the algorithm gives a ranking of the features it was necessary to employ a rule to select the features. Based on Heinemann et al. (2014) we chose to select the top 1% as detections.

2.5 Software, reproducibility, and code availability

The simulations and analyses were programmed in R v4.1.1 (R Core Team 2021). The packages that were used for the different methods are listed in table 1. To ensure reproducibility a different seed was used for the pseudo random number generator. These seeds are listed in in table 2. For reproducibility purposes, the session info can be found in appendix B.

T-tests and FDR adjustment (Benjamini and Hochberg 1995) are implementen in base R. The Tweedie-correction has, to our knowledge, no implementation and has been programmed by the author.

Table 2: Used seed per scenario

scenario	correlated	d	sig_p	n	seed
A1	no	1.1	10	50	2108191137
A2	no	0.7	10	50	2108191339
A3	no	0.3	10	50	2108191522
B1	no	0.9	1	10	2108191717
B2	no	1.81	1	10	2108191839
C1	no	0.95	3	30	2108191954
C2	no	1.3	3	30	2108200747
D	no	0	0	20	2108201715
block A1	block	1.1	10	50	2108201412
block C2	block	1.3	3	30	2108201023

3 Results

Ten scenarios are presented in this results section. An overview of the different scenarios can be seen in figure 3.

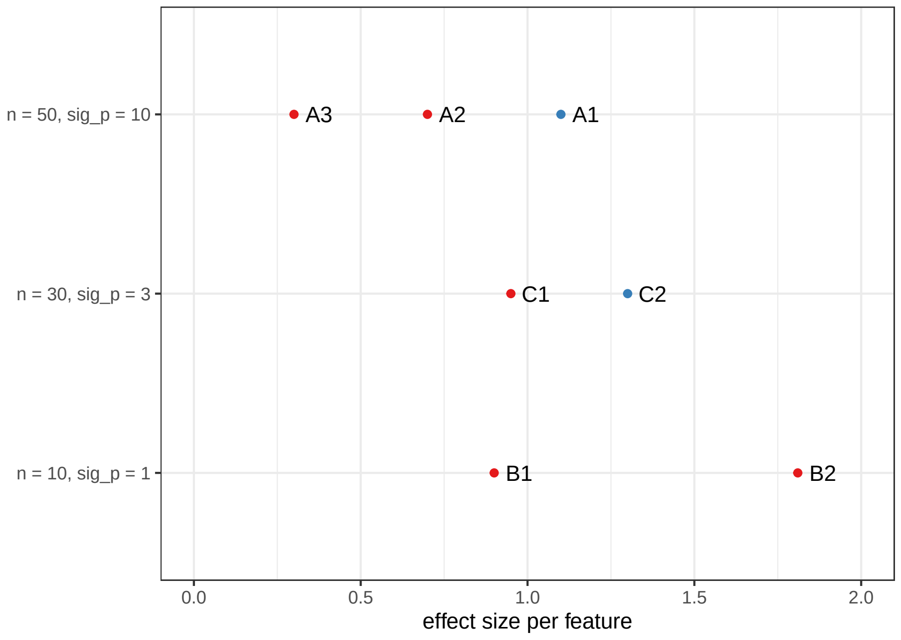


Figure 3: *Overview of the different scenarios. Blue dots: Scenarios was performed with and without correlation structure. Red dots: Scenario was only performed without correlation structure.*

3.1 Scenarios A - Optimistic

All scenarios 'A' are 'optimistic' scenarios in that they are scenarios with relatively many samples (50) and relatively many significant features (10) and no correlations between features. Within this optimistic setting, 3 different effect sizes are considered.

3.1.1 Scenario A1

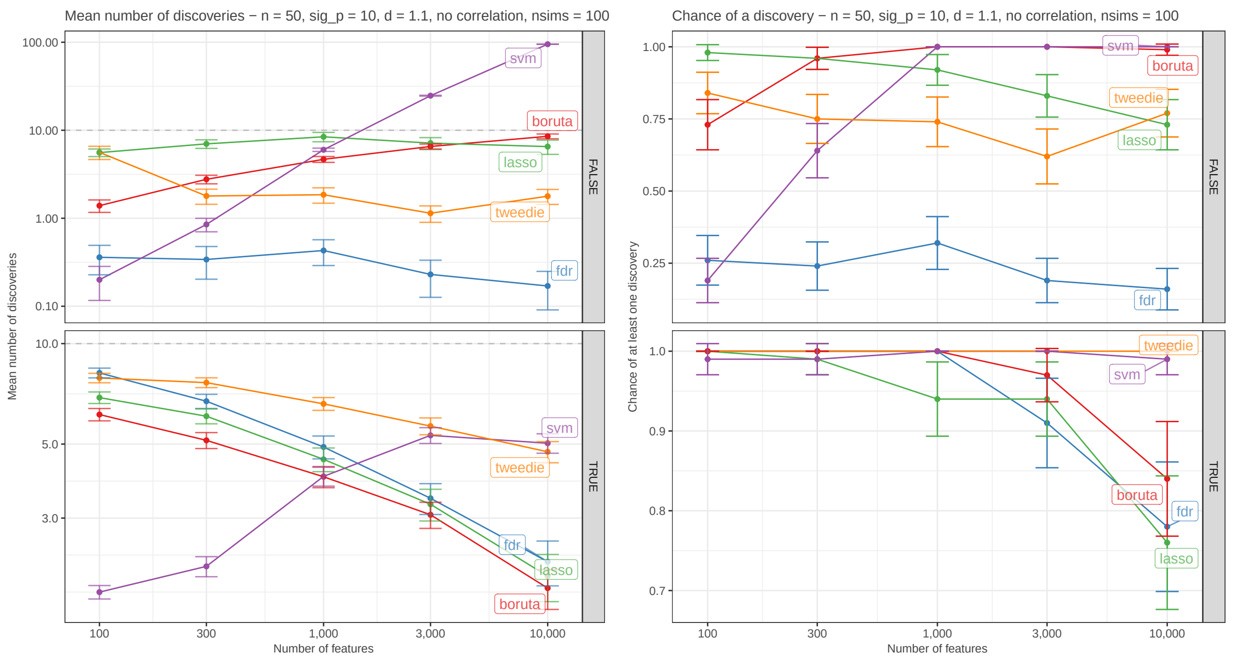
The first effect size considered is relatively large (d = 1.1, or AUC = 0.78), yet not overly large so that all methods would perform equally well. The results can be seen in figure 4.

In terms of mean number of false discoveries SVM performs the worst and shows a different behavior than most other methods with respect to growing number of features p. SVM shows a linear increase of mean number of false discoveries with growing number of features, on a logarithmic scale, whereas LASSO, Tweedie, and FDR stay relatively stable across number of features, or even show a decrease (FDR). Boruta, like SVM, shows a linear increase in mean number of false discoveries with growing p, though less pronounced. The best performing method is FDR with mean number of false discoveries less than 1 throughout the entire range of p.

In terms of chance of discoveries we can more or less draw the same conclusions. That is, with regard to false discoveries FDR is the best performing method. SVM and Boruta are the worst performing methods.

With respect to null results we see that FDR and LASSO are more prone to delivering null results than the other methods.

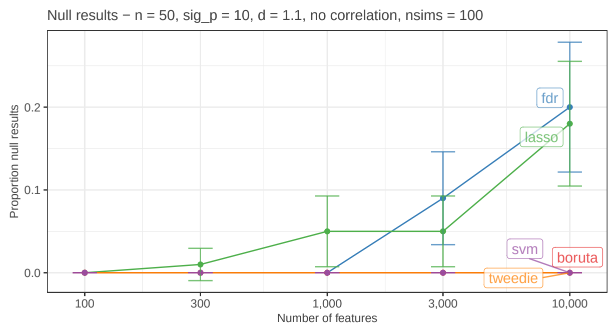
For the model based method we calculated the AUC of the models trained on the training set and tested on the test set. We can see that LASSO is consistently better than the other 2 methods.



3.1.2 Scenario A2

In this scenario the effect size is 0.7 (or AUC = 0.69) per feature. This scenario reflects slightly worse than optimistic conditions. The results can be seen in figure 5.

In terms of mean number of false discoveries SVM again performs the worst, while FDR is again the best method. In between lies (in order from better to worse) Tweedie, LASSO and Boruta. In terms of mean



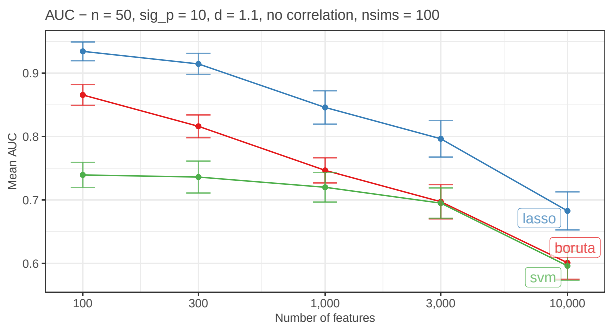


Figure 4: *Upper left: Mean number of false and true discoveries for various numbers of features for every method. Upper right: Chance of at least one false and true discovery for various numbers of features for every method. Bottom left: Proportion of null results for various numbers of features for every method. Bottom right: Mean AUC of models built on selections by the model based methods for various numbers of features. 95% confidence intervals are shown. In the case an interval is lacking, this has to do with the log scale of some of the plots.*

number of true discoveries we see again SVM as the best method and FDR as the worst. It is worth noting that SVM is the only method with more than 1 mean true discovery in the p = 10,000 condition. This translates to the same pattern of chances of at least one discovery, which is close to 1 for SVM but under 0.5 for all other methods (for p = 10,000). Leaving SVM out of the discussion, Tweedie is consistently among the best methods throughout the entire range of p, with Boruta as a close contender.

In terms of chance of false discoveries we see that SVM and Boruta are the worst methods and FDR is the best method.

In terms of null results we can see that FDR is the worst method across all values for p. LASSO does a better job but still has a considerable amount of null results. Tweedie has yet a smaller amount of null results, except at p = 10,000 where the amount of null results is the same as for LASSO. Boruta and SVM delivered the least number of null results.

In terms of discriminative ability the LASSO performs better than the other methods for low numbers of p (100, 300). For larger numbers of features the performances are very close with a slight advantage for SVM.

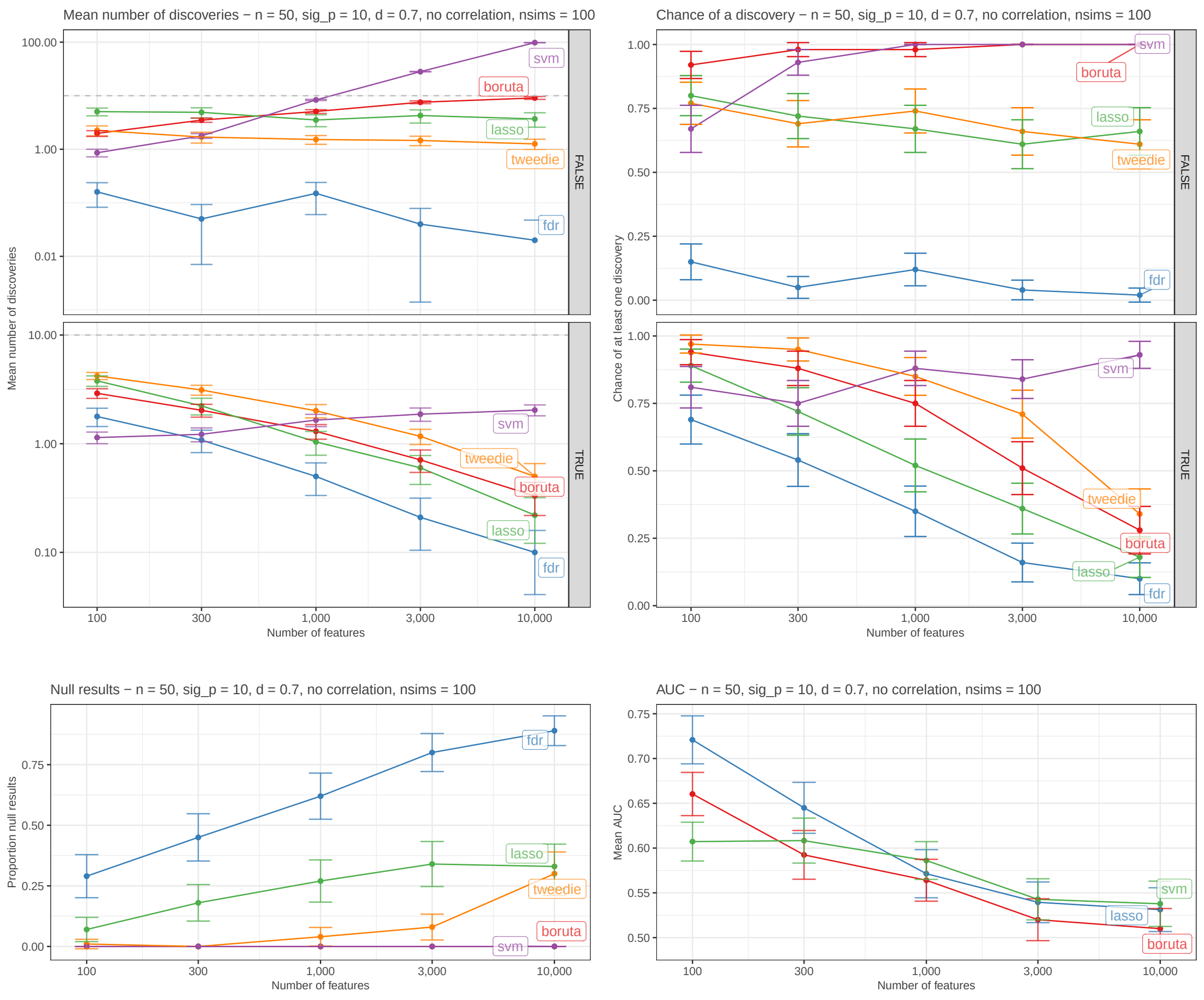


Figure 5: *Upper left: Mean number of false and true discoveries for various numbers of features for every method. Upper right: Chance of at least one false and true discovery for various numbers of features for every method. Bottom left: Proportion of null results for various numbers of features for every method. Bottom right: Mean AUC of models built on selections by the model based methods for various numbers of features. 95% confidence intervals are shown. In the case an interval is lacking, this has to do with the log scale of some of the plots.*

3.1.3 Scenario A3

In this scenario the effect size is brought down to only 0.3 (or AUC = 0.58) per feature. This scenario reflects a boundary condition in which most methods fail to deliver any useful results. The results can be seen in figure 6.

In terms of mean number of false discoveries we again see that SVM performs worst and FDR performs best. SVM is again the best performing method and FDR the worst in terms of mean number of true discoveries. It is interesting to see that under these poor conditions (small effect size) SVM shows a stable performance across all values for p. However, this performance is relatively poor (0.3, SE 0.05 at p = 10,000), indicating that this order of magnitude of effect size may represent a boundary.

More or less the same pattern of results can be seen in the results regarding chances of at least one detection. That is SVM and Boruta are the least performing methods with regard to false discoveries, and FDR is the best performing method. With regard to chance of at least one discovery SVM is the best performing method while all the other methods perform equally bad. Again the caveat about the relativity of what 'best performing' means should be made. With a chance of at least one true discovery of 0.26 (SE 0.04) one could argue this is not a good performance at all.

With respect to null results FDR is again the worst method with consistently values larger than 0.75 across the values for p. SVM and Boruta have the least null-results.

For the model based method we calculated the AUC of the models trained on the training set and tested on the test set. We can see that for almost all values for p, no model has an AUC higher than 0.5. Although only marginally relevant, the only exception is Boruta at p = 300 with.

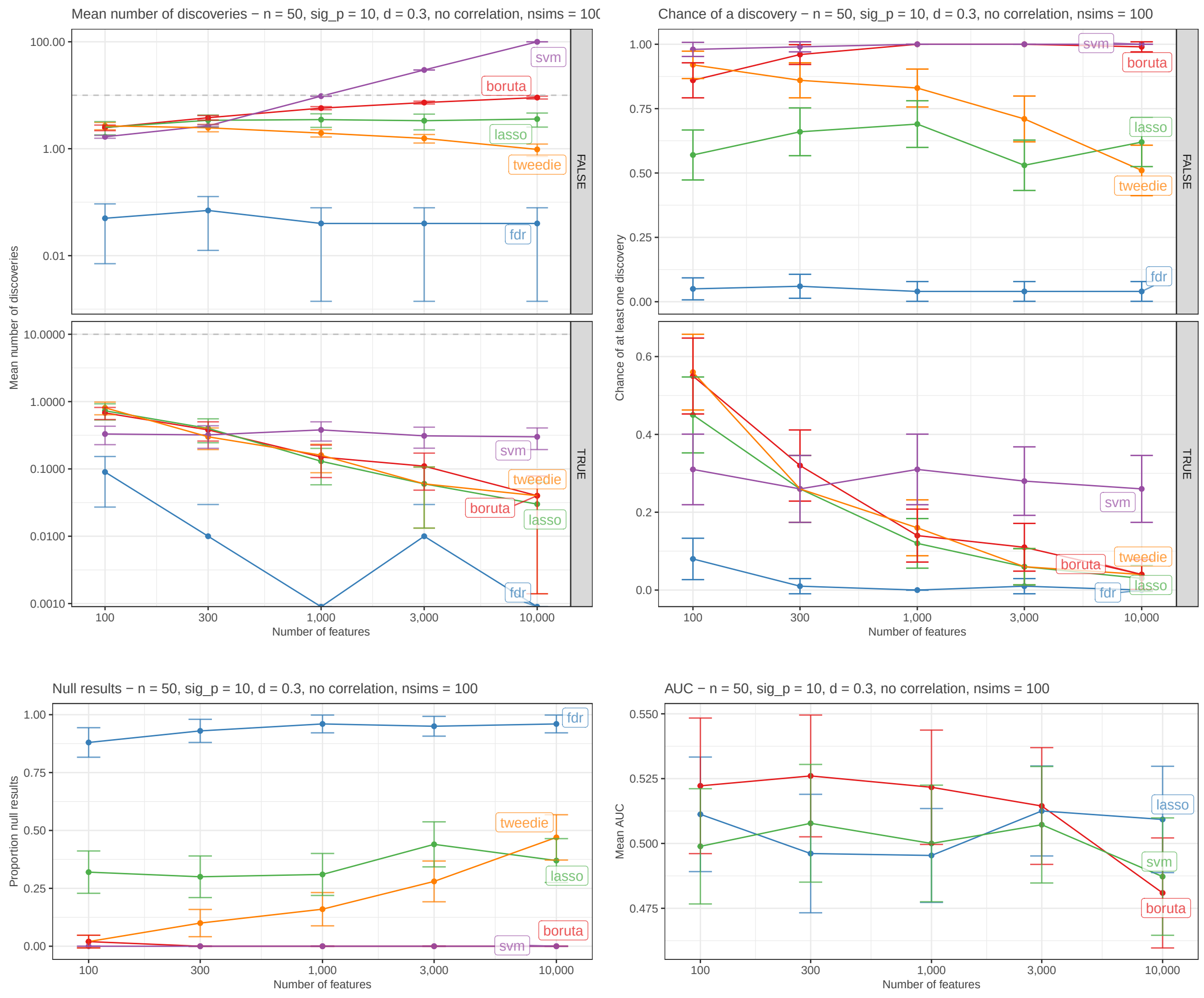


Figure 6: *Upper left: Mean number of false and true discoveries for various numbers of features for every method. Upper right: Chance of at least one false and true discovery for various numbers of features for every method. Bottom left: Proportion of null results for various numbers of features for every method. Bottom right: Mean AUC of models built on selections by the model based methods for various numbers of features. 95% confidence intervals are shown. In the case an interval is lacking, this has to do with the log scale of some of the plots.*

3.2 Scenarios B - Pessimistic

All scenarios 'B' are 'pessimistic' scenarios in that they are scenarios with relatively few samples (10) and relatively few significant features (1) and no correlations between features. Within this pessimistic setting, 2 different effect sizes are considered.

3.2.1 Scenario B1

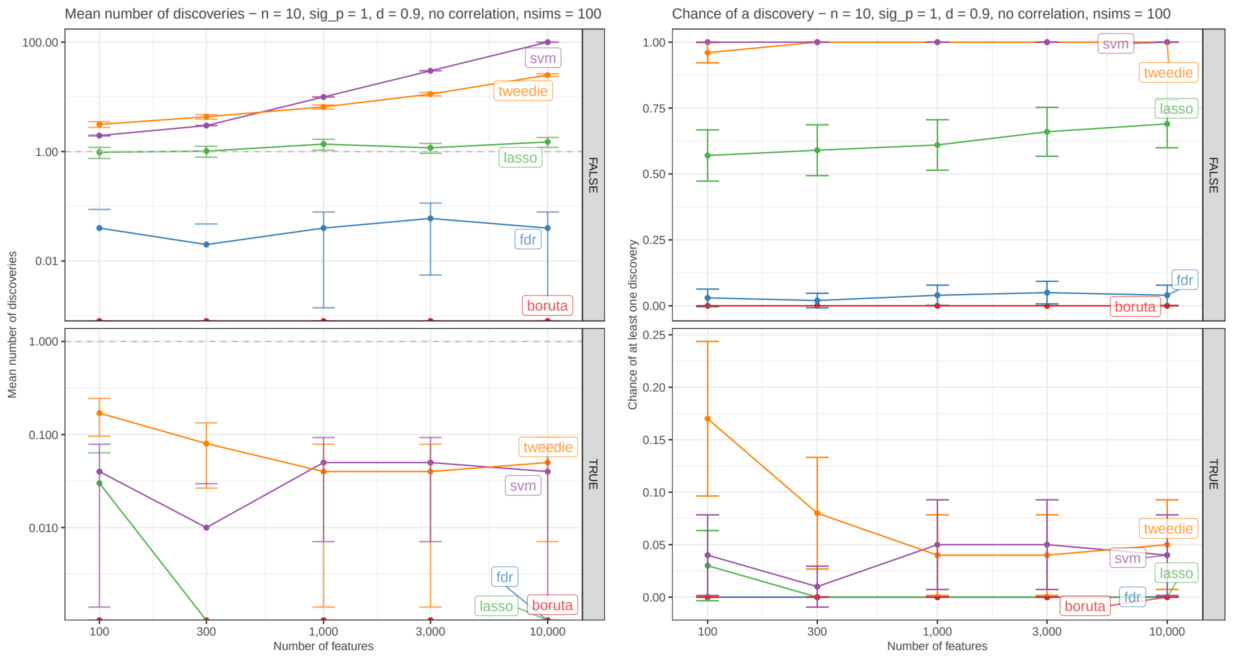
This scenario the effect size is relatively moderate (0.9, or AUC = 0.74) per feature. The results can be seen in figure 7.

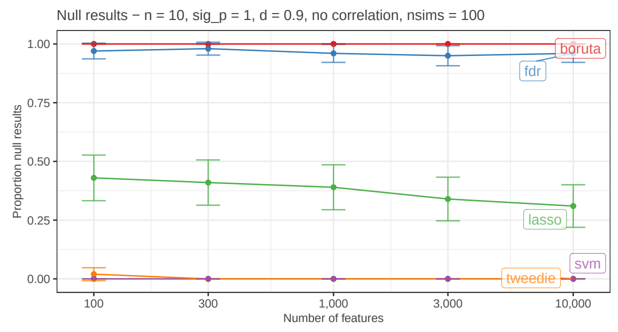
One major observation is that Boruta totally fails in this scenario, that is it never yields any discoveries regardless of their status (true or false). In a way it is the least performing method overall. However, in the current section we will make a further distinction and discuss the best and worst methods among the other 4 methods.

With respect to mean number of false discoveries FDR is the best and SVM the least performing method. In terms of mean number of true discoveries Tweedie and SVM are the best performing methods. FDR and LASSO are virtually at the same level as Boruta (that is no true discoveries) for the entire range of p. Although Tweedie and SVM are the best performing methods, they don't perform well with their mean below 0.1. This poor performance is reflected by the chance of true discovery that is below 0.1 for both methods, which means this scenario is indicative of a boundary condition.

In terms of null results FDR delivers poor results with a proportion of null results above of 0.9. Tweedie and SVM show the best results in this regard.

The model based methods perform poorly in terms of predictive ability with all AUC not different from 0.5.





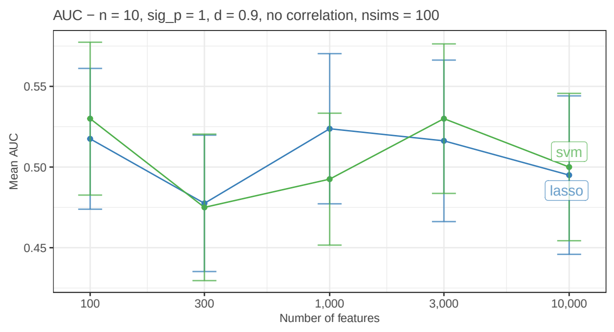


Figure 7: *Upper left: Mean number of false and true discoveries for various numbers of features for every method. Upper right: Chance of at least one false and true discovery for various numbers of features for every method. Bottom left: Proportion of null results for various numbers of features for every method. Bottom right: Mean AUC of models built on selections by the model based methods for various numbers of features. 95% confidence intervals are shown. In the case an interval is lacking, this has to do with the log scale of some of the plots.*

3.2.2 Scenario B2

This scenario the effect size is relatively large (1.81, or AUC = 0.9) per feature. This scenario tries to mimic a best case of worst case scenarios. The results can be seen in figure 8.

Again we see that Boruta totally fails in this scenario. The same remark as in the previous scenario is made, that is that although Boruta is evidently the least performing method overall we will make a further distinction and discuss the best and worst methods among the other 4 methods.

In terms of mean number of false discoveries FDR is again the best performing method and SVM the worst performing method. In terms of true discoveries Tweedie and SVM are the best performing methods while FDR and LASSO perform the least. The mean number of true discoveries for the best methods is small (less than 0.2).

In terms of number of null results we see that, apart from Boruta, FDR is the method with the most null results while Tweedie and SVM are the methods with the least number of null results.

The model based methods were evaluated on their predictive ability. However, all mean AUC values are not different from 0.5.

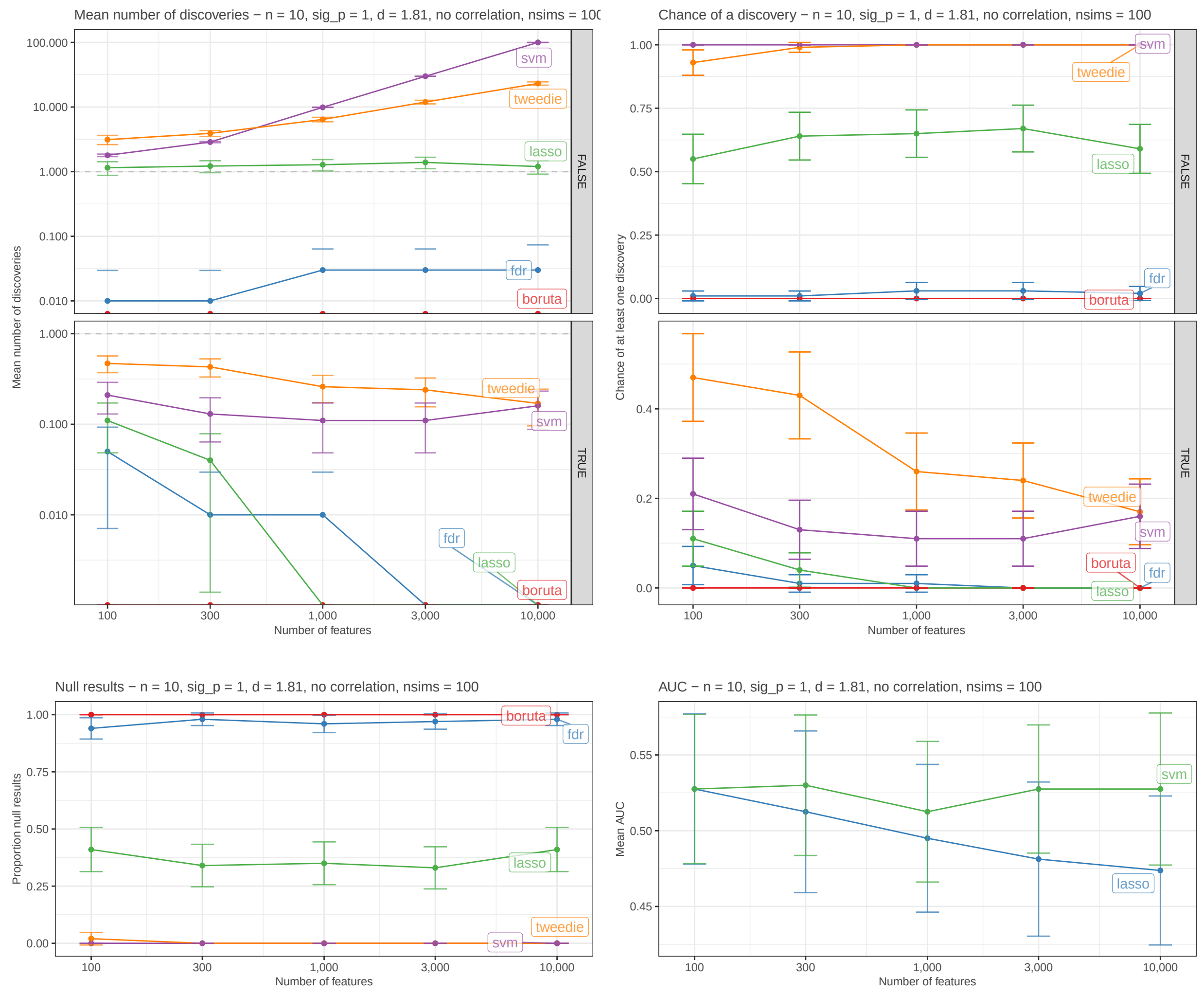


Figure 8: *Upper left: Mean number of false and true discoveries for various numbers of features for every method. Upper right: Chance of at least one false and true discovery for various numbers of features for every method. Bottom left: Proportion of null results for various numbers of features for every method. Bottom right: Mean AUC of models built on selections by the model based methods for various numbers of features. 95% confidence intervals are shown. In the case an interval is lacking, this has to do with the log scale of some of the plots.*

3.3 Scenarios C - In between

Whereas the scenarios 'A' and 'B' were both extremes in the 'optimistic-pessimistic-spectrum,' the scenarios 'C' are somewhat in between with 30 samples and 3 significant features. There are no correlations between features. Within this setting, 2 different effect sizes are considered.

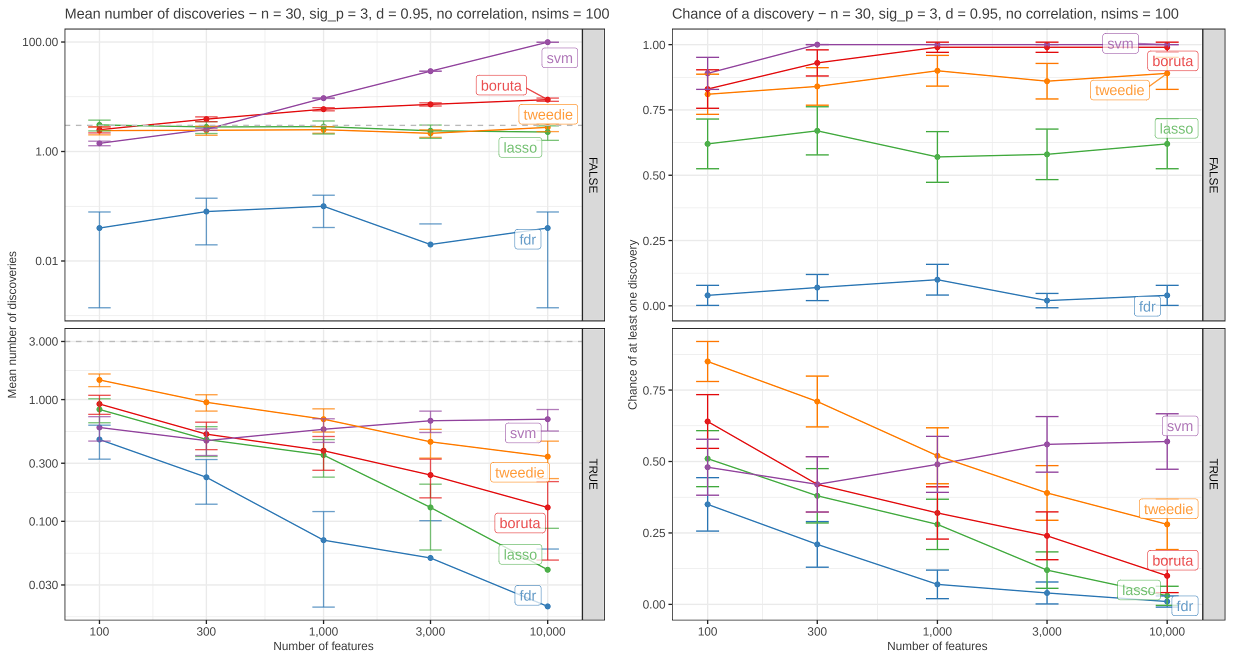
3.3.1 Scenario C1

This scenario has a relatively moderate effect size (0.95, or AUC = 0.75) per feature and reflects relatively poor conditions in this 'in between' setting. The results can be seen in figure 9.

In terms of mean number of discoveries we see that SVM has the most false and true discoveries. FDR has the least both false and true discoveries. Tweedie and LASSO are able to keep the number of false discoveries relatively under control, that is more or less 3 throughout the range of p. Boruta has increasingly more false discoveries and ever less true discoveries with growing p. The same pattern is visible in terms of chances of at least one discovery. SVM is the only method that has a chance for at least one true discovery of more than 0.5 (0.57, SE 0.05) for p = 10,000. Tweedie comes in second at 0.28 (SE 0.04). These results indicate that this condition may be a boundary condition.

In terms of null results FDR is the worst method and SVM and Boruta the best methods. Tweedie has only a moderate amount of null results when p is larger than 300.

The AUC of the model based methods on the test set all behave in the same way. That is small values for AUC (around 0.6) for the smallest value for p and from there on decreasing with no distinction from 0.5 at p = 3,000 and more.



3.3.2 Scenario C2

This scenario has a relatively larger effect size (1.3, or AUC = 0.82 per feature) and reflects somewhat better conditions than in scenario C1. The results can be seen in figure 10.

In terms of mean number of discoveries wee see that SVM has the most false and true discoveries. FDR has the lowest mean number of false and true discoveries. As the second best method, Tweedie has a mean number of true discoveries of 0.94 (SE 0.09) for p = 10,000. In terms of chance of at least one false discovery, Boruta and SVM are the worst performing methods while FDR is the best method. In terms of chance of at least one true discovery FDR and LASSO are the worst performing methods and Tweedie and SVM the best performing methods.

In terms of null results FDR is again the worst performing method while SVM, Boruta and Tweedie are the

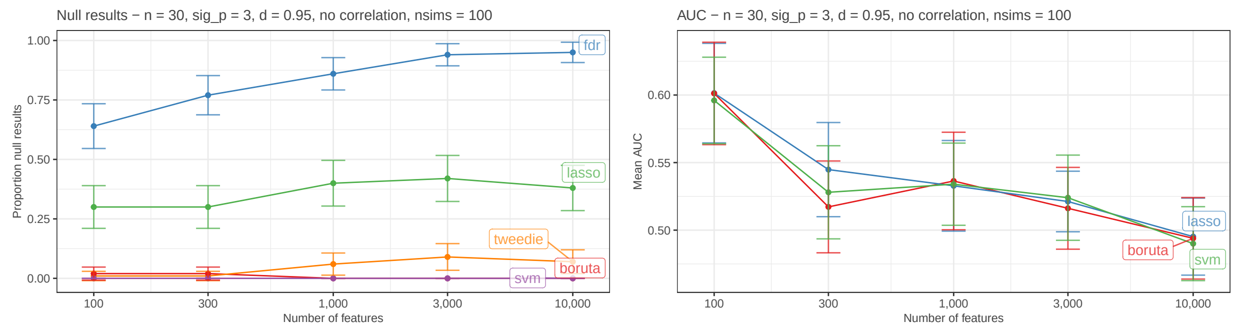
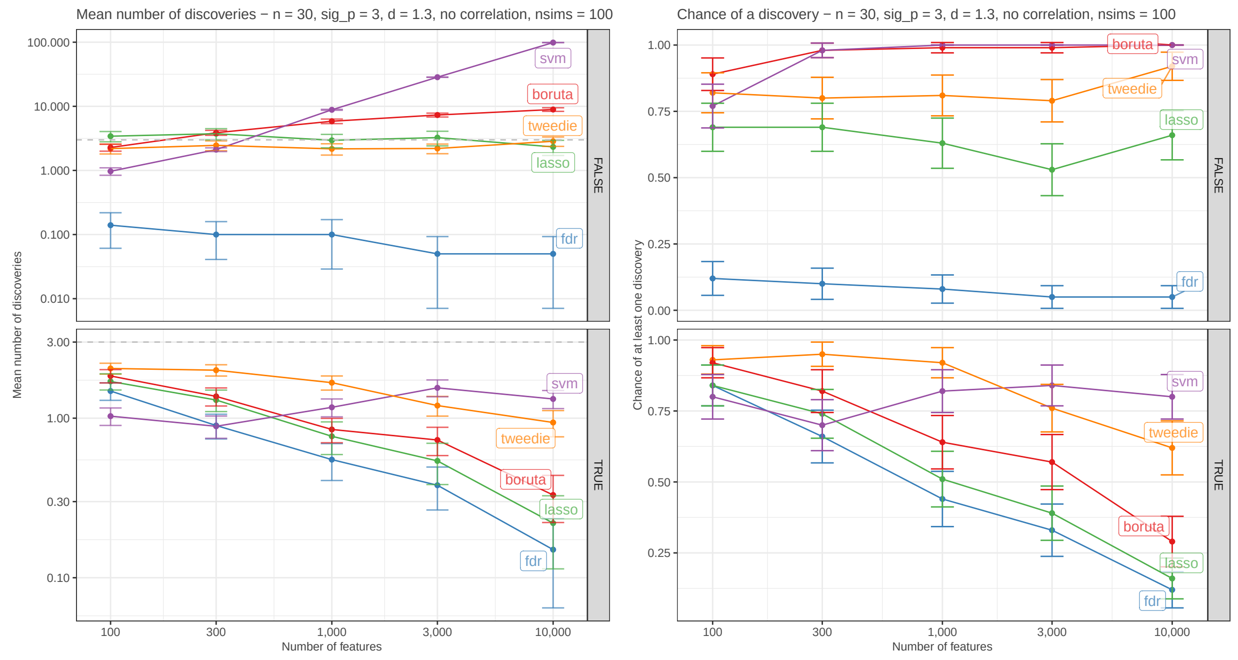


Figure 9: *Upper left: Mean number of false and true discoveries for various numbers of features for every method. Upper right: Chance of at least one false and true discovery for various numbers of features for every method. Bottom left: Proportion of null results for various numbers of features for every method. Bottom right: Mean AUC of models built on selections by the model based methods for various numbers of features. 95% confidence intervals are shown. In the case an interval is lacking, this has to do with the log scale of some of the plots.*

best performing methods.

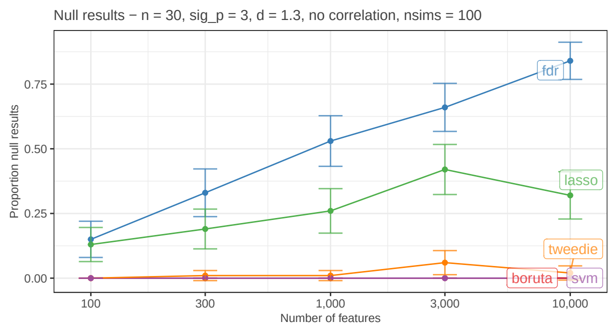
The AUC measured on the test set for the model based methods was highest for LASSO, except at p =10,000 where all methods perform equally poor.



3.3.3 Scenario D

This scenario is there to be able to evaluate the methods when there are no significant features. The number of samples is set at 20. The results can be seen in figure 11.

In terms of number of false discoveries FDR is the best method throughout the entire range of p. LASSO finds on average between 1 and 3 false discoveries and is stable across p. Tweedie and Boruta are close to one



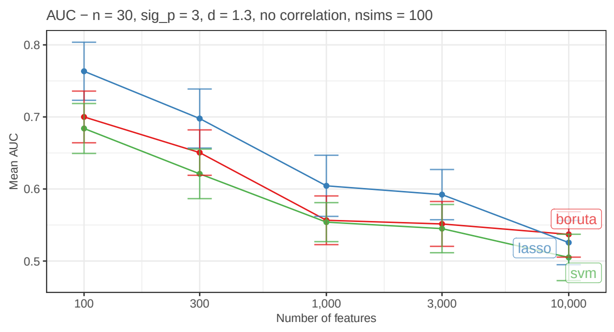


Figure 10: *Upper left: Mean number of false and true discoveries for various numbers of features for every method. Upper right: Chance of at least one false and true discovery for various numbers of features for every method. Bottom left: Proportion of null results for various numbers of features for every method. Bottom right: Mean AUC of models built on selections by the model based methods for various numbers of features. 95% confidence intervals are shown. In the case an interval is lacking, this has to do with the log scale of some of the plots.*

another with on average between 3 and 10 false discoveries. Both methods perform wors with increasing number of features. SVM always picks the top 1%, so there will be 1% of p false discoveries. More or less the same conclusions can be drawn from the chances of at least one false discovery plot.

3.4 Block correlation structure scenarios

The most 'forgiving' scenarios of scenarios A and C are selected and rerun with a block correlation structure in the data generating mechanism as described in the methods section. Only the most optimistic scenarios are selected because of the fact that we expect the performance to drop. In that way it is not very informative to try a condition with already very poor performances. This is also the reason why no scenario from the B scenarios is selected.

In these scenarios, data with at most 3,000 features were simulated because our system failed at p = 10,000.

3.4.1 Scenario A1 with block correlation structure

The results can be seen in figure 12.

In terms of mean number of false discoveries we see that Tweedie performs poorly at p = 100 and performs gradually better with growing number of features. At 3,000 features Tweedie is the second best performing method. LASSO performs more or less at the same level throughout the range of p. Boruta is among the best performing methods at p = 100 but gradually becomes worse and ends with the second to highest mean number of false discoveries. SVM also starts off among the best performing methods at p = 100 and gradually worsens and ends as poorest performing method. FDR is the best performing method throughout the range of p.

In terms of mean number of true discoveries we see that Tweedie is consistently the best or among the best performing methods. The decline in performance of Tweedie with growing p seems to be the smallest (apart from SVM, which has an increasing performance), on the log scale. SVM is among the least performing

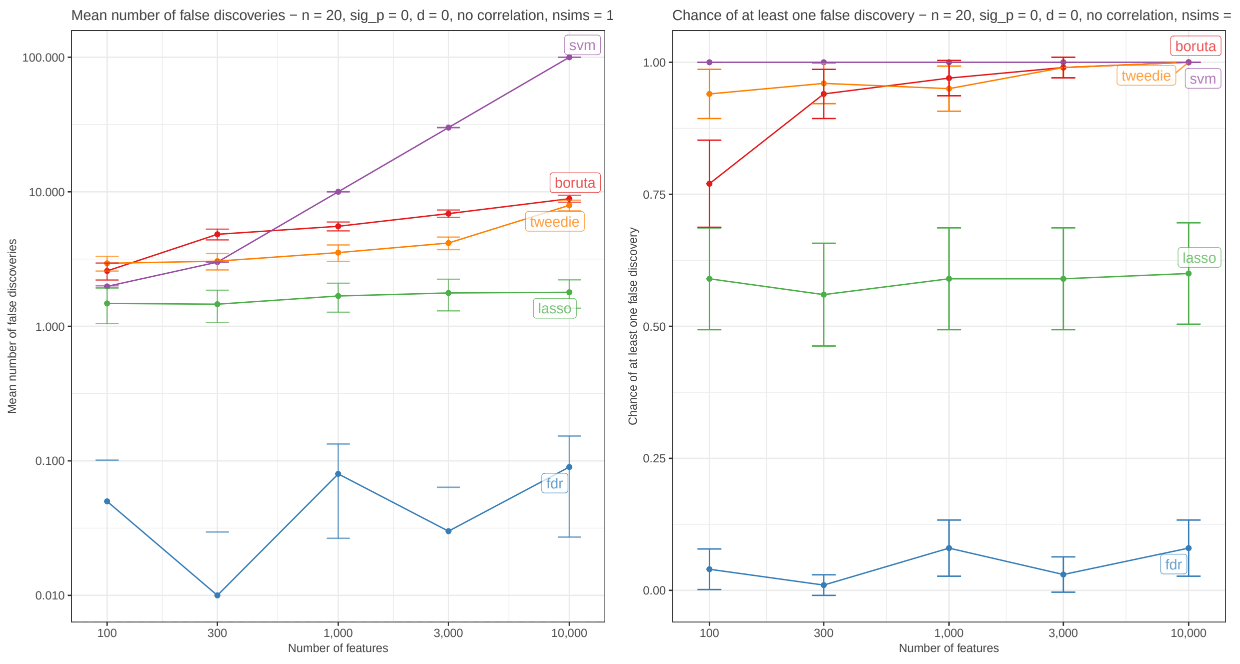


Figure 11: *Left: Mean number of false discoveries for various numbers of features for every method. Right: Chance of at least one false discovery for various numbers of features for every method.*

methods with smaller numbers of features and gets better at p = 1,000 and is at the level of Tweedie at p = 3,000. FDR and Boruta show a similar pattern with Boruta showing a poorer performance than FDR. LASSO doesn't do well in these conditions. It consistently drops in performance throughout the range of p while it started already at the second to last place at p = 100.

In terms of chances of at least one false discovery we see more or less the same pattern as discussed above, with the exception that LASSO seems a bit more conservative than Tweedie, however, the differences are not always meaningful (e.g. at p = 3,000 there is no difference). In terms of chances of at least one true discovery we see more or less the same pattern, except for FDR which performs worse than one would expect on the basis of the mean numbers of true discoveries and performs now worse than Boruta.

In terms of null results FDR is the worst performing method while Tweedie, SVM, and Boruta are the best performing methods. Of these, Tweedie has the most null results at higher numbers of features.

In terms of mean AUC we see that all models perform better than 0.5 for every value for p. All values are close to each other with the only exception being p = 100, where LASSO performs better than the 2 other methods.

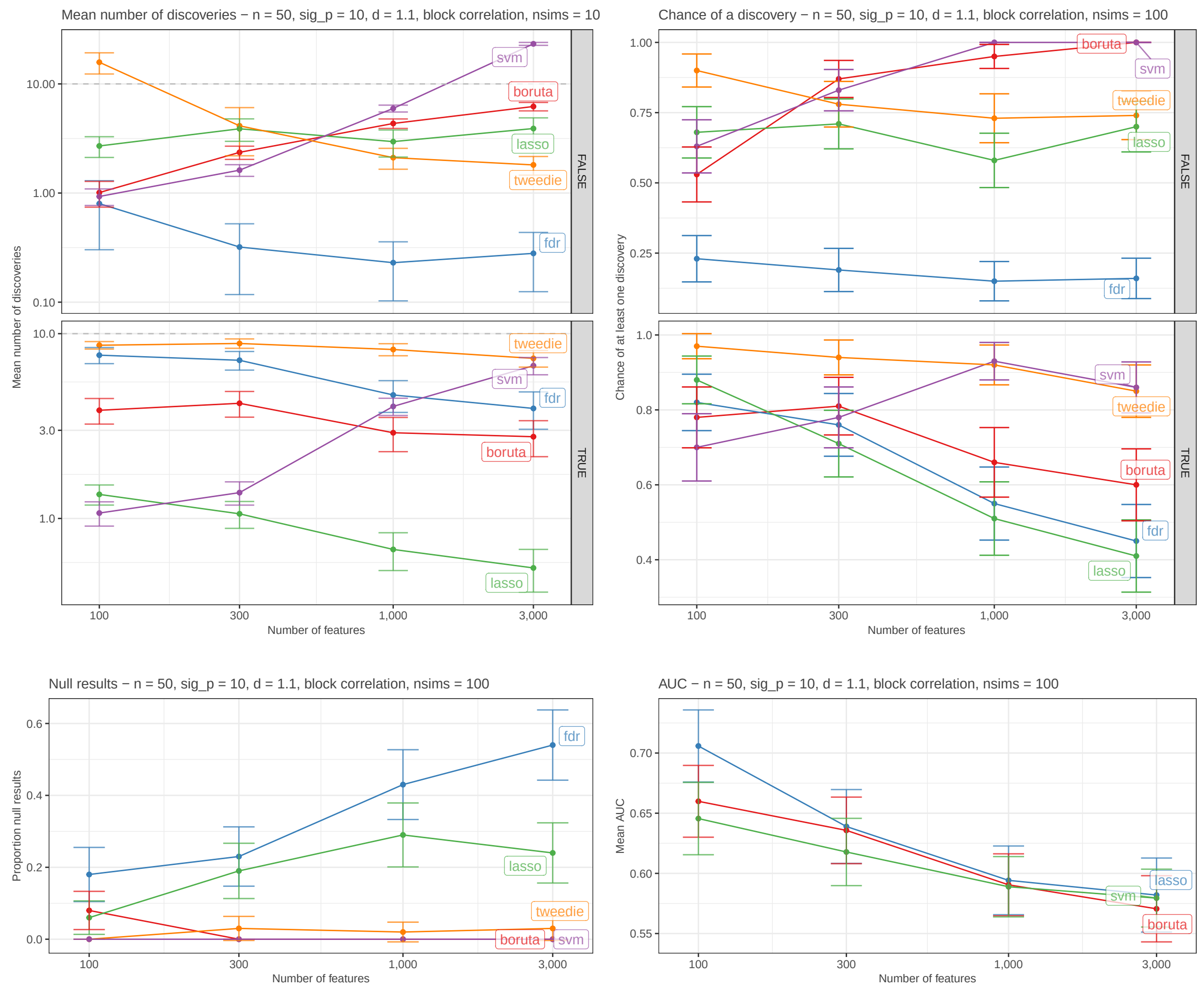


Figure 12: *Upper left: Mean number of false and true discoveries for various numbers of features for every method. Upper right: Chance of at least one false and true discovery for various numbers of features for every method. Bottom left: Proportion of null results for various numbers of features for every method. Bottom right: Mean AUC of models built on selections by the model based methods for various numbers of features. 95% confidence intervals are shown. In the case an interval is lacking, this has to do with the log scale of some of the plots.*

3.4.2 Scenario C2 with block correlation structure

The results can be seen in figure 13.

In terms of mean number of false discoveries we see that SVM performs better than most methods at p = 100 and 300 and performs the poorest at p = 1,000 and p = 3,000. Boruta shows the same pattern as SVM, although much less pronounced. LASSO and Tweedie perform consistently at around 3 mean false discoveries, while FDR performs the best with very few discoveries on average (aroudn 0.3 to 0.1).

In terms of mean number of true discoveries we see that Tweedie is the best or among the best performing methods, except at p = 3,000 where SVM is slightly better. SVM performs among the poorest performing methods for relatively small values for p (100 and 300) and is among the best methods at higher values for p

(1,000 and 3,000). After Tweedie, the best performing methods in order are Boruta, FDR, while LASSO comes in last.

In terms of chances of true/false discoveries we see more or less the same patterns. We see again that LASSO is a bit more conservative than Tweedie. In terms of true discveries we see that FDR and LASSO switched places compared to mean number of true disoveries.

In terms of null results FDR is the worst performing method while Tweedie, SVM, and Boruta are again the best performing methods. Of these, Tweedie has again the most null results at higher numbers of features.

In terms of mean AUC we see the same pattern of resutls as in A2 block correlated. All models perform better than 0.5 for every value for p. All values are close to each other with the only exception being p = 100, where LASSO performs better than the 2 other methods.

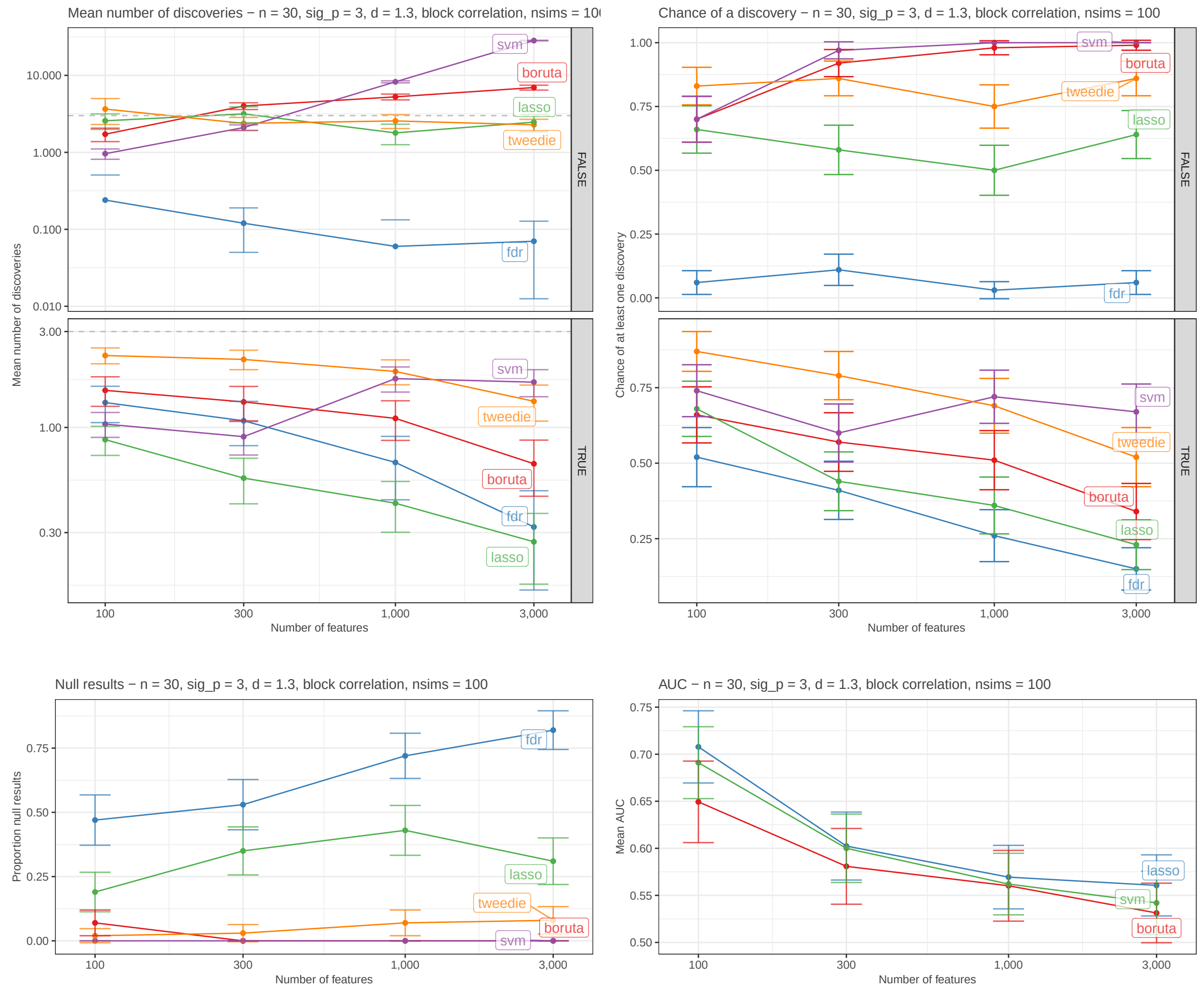


Figure 13: *Upper left: Mean number of false and true discoveries for various numbers of features for every method. Upper right: Chance of at least one false and true discovery for various numbers of features for every method. Bottom left: Proportion of null results for various numbers of features for every method. Bottom right: Mean AUC of models built on selections by the model based methods for various numbers of features. 95% confidence intervals are shown. In the case an interval is lacking, this has to do with the log scale of some of the plots.*

3.5 Computation time

Although this not of primary concern for method users, it can be interesting too see how well methods scale with growing number of p with regard to computation time. The results can be seen in figure 14.

All computation times are relatively short which adds to the lack of concern regarding computation time from the perspective of method users. Though, we can see that Boruta seems to have difficulties with scaling. One could expect that if the number of samples and number of features keep increasing, the computation time might become an issue. Tweedie, on the other hand, seems to scale very well with growing number of features.

There is one additional result that draws the attention and that is that of SVM at p = 1,000 which is slower than at p = 3,000. The reason of this has to do with the halving of the features up to the point at which there are 250 or less features in the feature pool. At a starting point of 3,000 one ends up at 188 before the procedure starts to eliminate one feature at a time. At a starting point of 1,000 this is 250, which is higher and that is why the analysis takes longer on average.

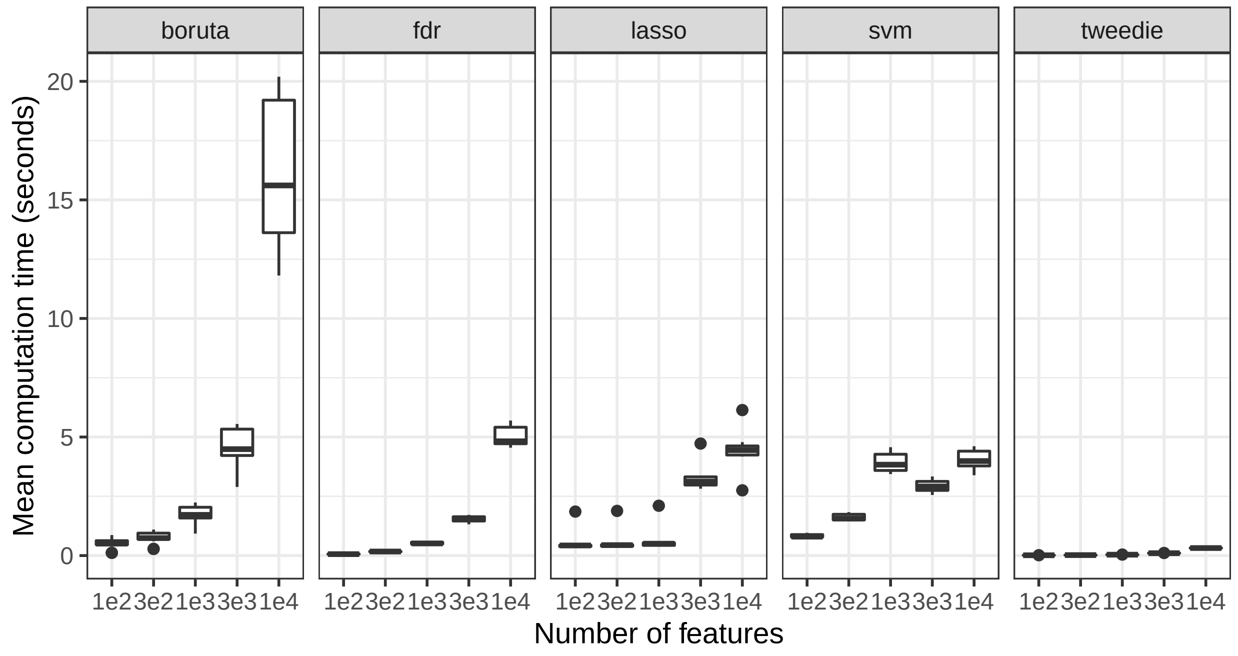


Figure 14: *Boxplots of computation time for every method and for every number of features, across all simulations*

4 Discussion

4.1 Boundary conditions

From the results of scenario A3 we learn that at n = 50, and sig_p = 10, d = 0.3 and at p larger than or equal to 1,000 virtually no method can deliver any satisfying results anymore. None of the methods have an expected number of true discoveries larger than 1 and this for the entire range of p. None of the methods have a chance of having a true discovery below 0.4 (except at p = 100), while for all methods except FDR the chance of at least one false discovery is above 0.5 for the entire range of p.

The same is true for the B scenarios, that is no satisfying performance, where n = 10 and sig_p = 1 for the entire range of p and both values for d. In these scenarios the Boruta method totally failed to work. The other methods delivered very poor performances. Except for FDR (and Boruta, see above), all other methods had a chance of having at least one false discovery of above 0.5 for the entire range of p while the chance of having at least one true discovery was below 0.25 for the entire range of p in when d = 0.9. Performances were slightly better at a high effect size per feature. However they were still far beneath satisfactory levels.

In scenario C1 with n = 30, sig_p = 3, d = 0.95 there were also very poor results when p was larger than or equal to 1,000. In terms of mean number of true discoveries only Tweedie was able to find on average more than one true discovery at p = 100, on average 1 true discovery at p = 300. From p = 1,000 on the mean number of true discoveries was less than 1 for every method while the mean number of false discoveries was larger than 1, except for FDR.

The conditions discussed above are what we call boundary conditions in which it is to be expected to have no or very few useful results in general. With in general, we mean that it is the case for all methods.

4.2 Formulation of guidelines

There is no one method that is best at everything, so what method to choose will depend on what is important to the problem at hand. If one values conservativeness more, then it is clear that FDR is the best choice. In all scenarios FDR was the best performing method with respect to keeping the number of false discoveries within reasonable limits. The method to avoid is the SVM. Because of the top 1% rule, it always picks 100 features at p = 10,000. In this way there will be a lot of false discoveries when sig_p is rather small. The same conclusion can be made on the basis of the results regarding the chance of at least one false discovery. This is not that surprising. Another method to avoid when striving for conservativeness is Boruta as it performed poorly with regard to the chance of at least one false discovery in scenarios A (optimistic), C (in between), D (no sig_p) and both block correlation scenarios. In the B scenarios (pessimistic) Tweedie was also among the least conservative methods.

If one, on the other hand, values many true discoveries at the cost of more false discoveries, then other methods are in place. SVM is almost consistently among the best performing methods in this regard. Only in A1block (optimistic, large effect size), did SVM fail to be among the best methods. The downside of SVM is of course a high number of false discoveries. Other methods that deliver good results in terms of number of true discoveries is Tweedie (A1, B, C2, A1block, C2block). Most often, Tweedie is less 'aggressive' than SVM in that it doesn't result in that much false discoveries as SVM does. It is interesting that Tweedie is present in the list of well performing methods in the block correlation scenarios while SVM is not in both. This could perhaps indicate that Tweedie is better suited when the data gets more realistic properties.

Methods to avoid in this regard are FDR and LASSO. FDR is consistently among the worst methods. This is not surprising since it is the most conservative method (see above). LASSO didn't do well in this regard in scenarios A1, B1, C2, A1block, and C2block. It is interesting to see that LASSO was among the worst performing methods in the block correlation scenarios. The fact that it has difficulties in dealing with highly correlated features is a well known weakness of the LASSO. Highly correlated features are a problem for coefficient estimation. In such cases multicollinearity, the presence of highly intercorrelated features, results in highly variable estimates. The LASSO can deal with these kinds of features by removing all but one. However, the method is agnostic of which feature is a true or false feature. Thus, in highly correlated situations the lasso will discard many correlated features. And it does so unknowingly of feature status (true or false).

This downside of the LASSO has also a benefit. Getting rid of multicollinearity by setting some coefficients to zero will benefit prediction. When prediction is of main interest, the LASSO is often the better choice compared to the other model based methods. It was the best method in scenarios A1 (optimistic, relatively large effect size), A3 (optimistic, yet small effect size), C2 (in between, larger effect size), A1block, and C2block. SVM performed best in scenarios A2 and C2block. As such Boruta is often to be avoided.

If one really wants to avoid null results, then one thing to consider is using SVM. SVM will always produce results because of the top 1% rule. The added benefit is that we have seen that SVM is often good at capturing the significant features. Yet, it comes with the cost of a lot of false discoveries as well. Other methods to consider are Tweedie (A1, B1, B2, C2) and Boruta (A1, A2, A3, C1, C2, A1block, and C2block). However, Boruta did not perform very well with regard to other measures.

All in all, the results of Tweedie look promising because the method often performs quite well with regard to true discoveries and 'not so bad' with regard to false discoveries. It is not possible to have a method that is optimal in both. Yet, Tweedie seems to find some sort of middle ground in many scenarios.

One other thing to consider is the ease of use. While the application of Tweedie's formula is in essence not that complicated, it is a fact that no off-the-shelf implementation is available. This can and will be an obstacle for some, if not many, people. Maybe such a solution could become available in the future, for example in the form of an R-package.

4.3 Limitations of the current study

In the current study a lot of simulation parameters were manipulated: number of features, number of significant features, sample size, effect size per feature, correlation structure. Because there were so many parameters it was impossible to implement a full factorial design in which all combinations of values are explored. In this study, only a tiny fraction of the entire simulation parameter space was explored. We chose not to use a full factorial design because it would sooner rather than later become practically unfeasible to do the entire simulation study.

Yet, it might be that a full factorial design with a tighter focus would be feasible and interesting. Now we chose to entangle number of samples with number of significant features and compose them together as optimistic, pessimistic and in between. While this was a means to keep things workable, it leaves the question what the effect is of the one or the other. Which of the 2 is more important in terms of boundary conditions? Maybe the disentanglement of sample size and number of significant features could lead to other guidelines.

Even with the current entanglement, it is still the case that we only explored a tiny fraction of that subspace of the simulation parameter space. And it may be that we did not choose the most interesting values. The exploratory simulations were intended to mitigate this somewhat, but the problem isn't solved by that. The problem is only shifted.

Another possible remark are the included methods. In ideal circumstances one would include all the most relevant methods. However, this is practically not possible. In this study there were 2 classes of methods: methods that involve hypothesis tests and feature selection methods that are based on prediction models. As we have seen, not all performance measures are applicable to every method, which is not ideal.

An other weakness is the simplicity of the data generating mechanism. Simple data generating mechanisms allow for easy interpretation. On the other hand, it could be that because of that simplicity the results aren't applicable to real analyses. Methods like SPsimSeq (Assefa, Vandesompele, and Thas 2020) could be used to enhance the realism of the simulated data and as a consequence also enhance the applicability of the results.

A last thing to consider is the difference in parameter tuning of the LASSO in comparison with SVM and Boruta. Although both SVM and Boruta have hyperparameters that can be tuned, we only used the default values.

5 Example analysis

In this last section we will perform an example analysis. The data that we will use come from Hung, Baldi, and Hatfield (2002) and found it via the userguide of the limma bioconductor package. In this user guide there is example code given that allows one to set up the data. This example code was used only to import and set up the data so that it was in a format that was familiar.

Hung, Baldi, and Hatfield (2002) explain that

> the purpose of the work presented here is to identify the network of genes that are differentially regulated by the global E. coli regulatory protein, leucine-responsive regulatory protein (Lrp), during steady state growth in a glucose supplemented minimal salts medium. Lrp is a DNA-binding protein that has been reported to affect the expression of approximately 55 genes.

We will approach the data only from a statistical point of view.

5.1 Data set description and exploratory analysis

This dataset consists of 7312 features measured on 8 samples. There are 2 classes, each with 4 samples. Mean, standard deviation, median, inter quartile range (iqr), minimum (min), and maximum (max) was calculated for every gene and strain combination. This allows to visualize the distributions of these statistics across genes for every strain. This visualization can be seen in figure 15. The distributions of the statistics of the 2 strains look very alike. There is a small difference noticable in the distribution of standard deviation and inter quartile range, meaning that expression levels of lrp- are slightly more variable than expression levels of lrp+.

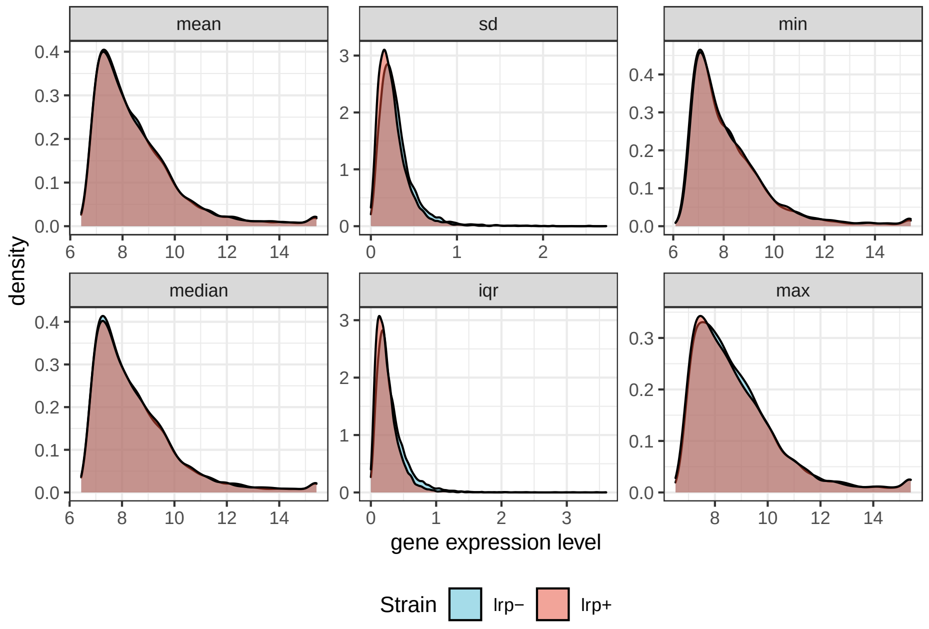


Figure 15: *Distributions of descriptive statistics of gene expression levels of both strains.*

5.2 Application of Tweedie's formula

For this analysis, we will use Tweedie as method. This is because Tweedie performed best in the more realistic scenarios (block correlation). Also, there is little data at hand, only 8 samples. In order to make the most use out of the data it makes more sense to use all data for t-tests than to split the data in a training and test set like we would when using model based methods.

Because FDR is far too concervative and we don't want to risk missing potential biomarkers we opt for Tweedie.

In order to find out which genes are differentially expressed between strains the following null hypotheses were tested against the following alternative hypotheses:

$$\left.\begin{array}{l} H_{0i} : \mu_{lrp-i} = \mu_{lrp+i} \\ H_{ai} : \mu_{lrp-i} \neq \mu_{lrp+i} \end{array}\right\} i = \{1, \dots, 7312\}$$

In these hypotheses, μ_{lrp-i} and μ_{lrp+i} are the population means of the gene expression level of the ith gene in the lrp- and lrp+ strain respectively.

After the t-tests are performed, a histogram of the test statistics is constructed. The middle of each bin and the counts are used to perform a 4th degree polynomial poisson regression (log link). The coefficients can be found in table 3

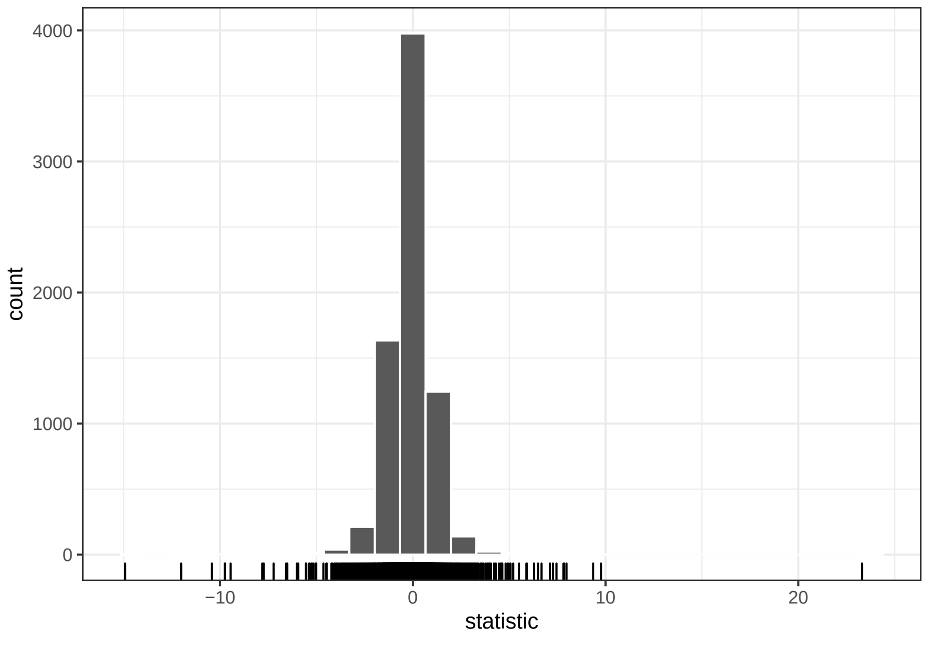


Figure 16: *Histogram of the test statistics. The vertical dashes show the individual data points.*

After the log derivative and the variance of the test statistics is estimated we perform the correction. Figure 17 shows the curve of corrected test statistic vs. uncorrected test statistic.

Now we can use the corrected statistics to calculate corrected p-values. We evaluate which p-values are less

Table 3: Coefficients of 4th degree polynomial poisson regression on bin counts.

Coefficient	value
Intercept	8.0740841
Poly 1	-0.0440204
Poly 2	-0.3814665
Poly 3	-0.0083491
Poly 4	0.0010862

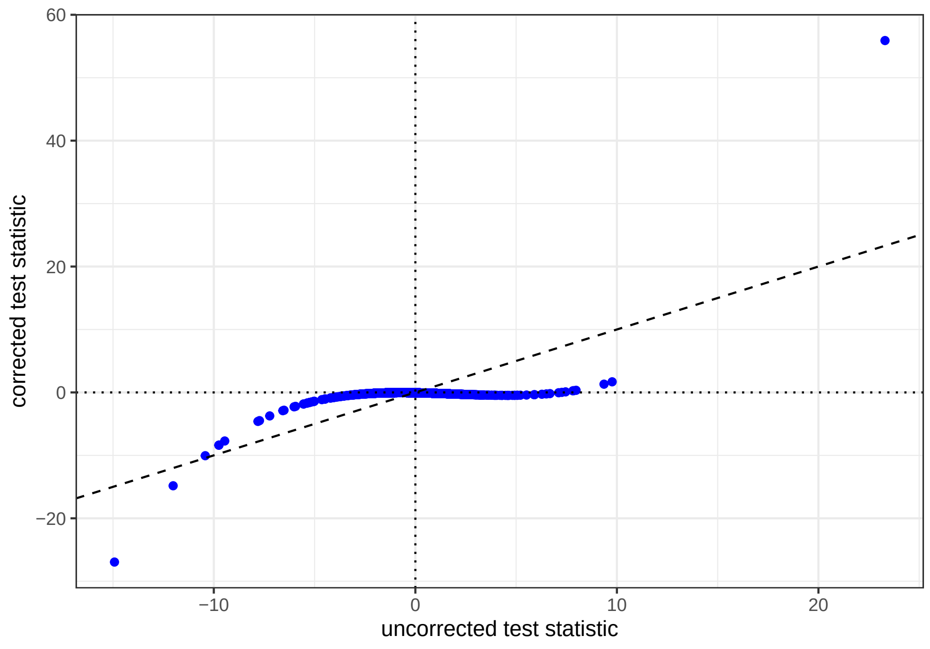


Figure 17: *Corrected vs. uncorrected test statistics plot. The dashed line shows y=x.*

Table 4: Results of example analysis

statistic	df	corrected_t	corrected_p	name
-9.743924	5.672275	-8.370091	0.0002120	gltB_b3212_st
-12.017442	5.808288	-14.831549	0.0000077	gltD_b3213_st
-7.225528	5.753797	-3.734709	0.0104526	IG_1643_2642304_2642452_rev_st
23.302899	5.944197	55.904190	0.0000000	IG_821_1300838_1300922_fwd_st
-7.810282	3.777262	-4.603318	0.0114482	ilvI_b0077_st
-9.751614	4.387994	-8.388041	0.0007306	livK_b3458_st
-10.423430	5.909518	-10.054092	0.0000617	lrp_b0889_st
-6.515208	5.028667	-2.830694	0.0364093	ndk_b2518_st
-7.734228	5.258478	-4.483707	0.0057558	pntA_b1603_st
-14.927422	4.162590	-26.952822	0.0000079	serA_b2913_st
-9.457497	5.741620	-7.719130	0.0003066	sodA_b3908_st
-6.574712	5.937519	-2.900350	0.0276515	stpA_b2669_st

than 0.05 and we retrieve the according gene names. The results can be consulted in table 4.

6 References

Acharjee, Animesh, Joseph Larkman, Yuanwei Xu, Victor Roth Cardoso, and Georgios V. Gkoutos. 2020. "A Random Forest Based Biomarker Discovery and Power Analysis Framework for Diagnostics Research." *BMC Medical Genomics* 13 (1). https://doi.org/10.1186/s12920-020-00826-6.

Assefa, Alemu Takele, Jo Vandesompele, and Olivier Thas. 2020. "Spsimseq: Semi-Parametric Simulation of Bulk and Single-Cell Rna-Sequencing Data." *Bioinformatics* 36 (10): 3276–78. https://doi.org/10.1093/bioinformatics/btaa105.

Benjamini, Yoav, and Yosef Hochberg. 1995. "Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing." *Journal of the Royal Statistical Society. Series B (Methodological)* 57 (1): 289–300. http://www.jstor.org/stable/2346101.

Brewer, Mark J. n.d. "MarkJBrewer/ICsims: Simulations for Studying Prediction Performance of Different INFORMATION CRITERIA." *GitHub*. https://github.com/MarkJBrewer/ICsims.

Duan, Kai-Bo, Jagath C Rajapakse, Haiying Wang, and Francisco Azuaje. 2005. "Multiple SVM-RFE for Gene Selection in Cancer Classification with Expression Data." *IEEE Transactions on Nanobioscience* 4 (3): 228–34.

Efron, Bradley. 2008. "Microarrays, Empirical Bayes and the Two-Groups Model." *Statistical Science* 23 (1): 1–22. https://doi.org/10.1214/07-sts236.

———. 2011. "Tweedie's Formula and Selection Bias." *Journal of the American Statistical Association* 106 (496): 1602–14. https://doi.org/10.1198/jasa.2011.tm11181.

Friedman, Jerome, Trevor Hastie, and Robert Tibshirani. 2010. "Regularization Paths for Generalized Linear Models via Coordinate Descent." *Journal of Statistical Software* 33 (1): 1–22. https://www.jstatsoft.org/v33/i01/.

Géron, Aurélien. 2020. *Hands-on Machine Learning with SCIKIT-LEARN, Keras, and TENSORFLOW: Concepts, Tools, and Techniques to Build Intelligent Systems.* O'Reilly.

Halevy, Alon, Peter Norvig, and Fernando Pereira. 2009. "The Unreasonable Effectiveness of Data." *IEEE Intelligent Systems* 24 (2): 8–12. https://doi.org/10.1109/mis.2009.36.

Hastie, Trevor, Robert Tibshirani, and J. H. Friedman. 2009. *The Elements of Statistical Learning Data Mining, Inference, and Prediction.* Springer.

Heinemann, Joshua, Aurélien Mazurie, Monika Tokmina-Lukaszewska, Greg J. Beilman, and Brian Bothner. 2014. "Application of Support Vector Machines to Metabolomics Experiments with Limited Replicates." *Metabolomics* 10 (6): 1121–28. https://doi.org/10.1007/s11306-014-0651-0.

Hung, She-pin, Pierre Baldi, and Wesley Hatfield. 2002. "Global Gene Expression Profiling in Escherichia Coli K12: The Effects of Leucine-Responsive Regulatory Protein." *J Biol Chem.*, 43th series, 25 (277): 40309–23. https://doi.org/doi:%2010.1074/jbc.M204044200.

Kursa, Miron B., and Witold R. Rudnicki. 2010. "Feature Selection with the Boruta Package." *Journal of Statistical Software* 36 (11): 1–13. http://www.jstatsoft.org/v36/i11/.

L, Breiman. 2001. "Random Forests." *Machine Learning* 45: 5–32.

Marcišauskas, Simonas, Benjamin Ulfenborg, Björg Kristjansdottir, Sofia Waldemarson, and Karin Sundfeldt. 2019. "Univariate and Classification Analysis Reveals Potential Diagnostic Biomarkers for Early Stage Ovarian Cancer Type 1 and Type 2." *Journal of Proteomics* 196: 57–68. https://doi.org/https://doi.org/10.1016/j.jprot.2019.01.017.

R Core Team. 2021. *R: A Language and Environment for Statistical Computing.* Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.

Robbins, Herbert. 1956. "An Empirical Bayes Approach to Statistics." *Proceedings of the Third Berkeley Symposium on Mathematical Statistics and Probability*, 1954-1955, I: 157–63. https://doi.org/10.1525/9780520313880-015.

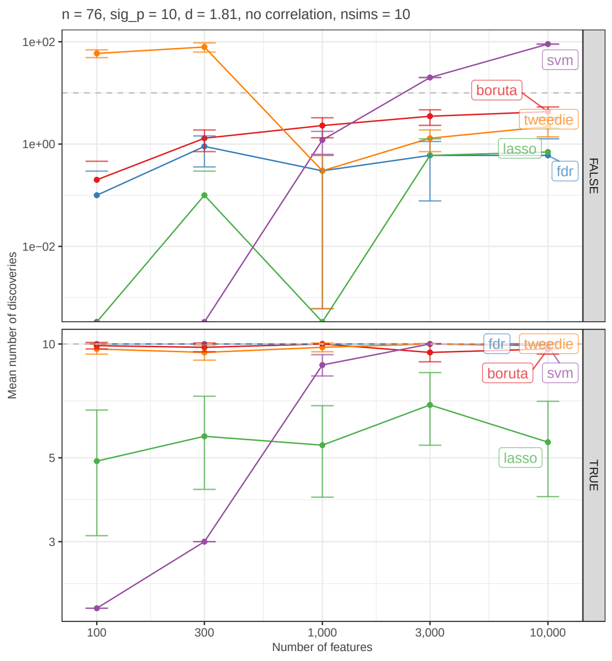
Tibshirani, Robert. 1996. "Regression Shrinkage and Selection via the Lasso." *Journal of the Royal Statistical Society. Series B (Methodological)* 58 (1): 267–88. http://www.jstor.org/stable/2346178.

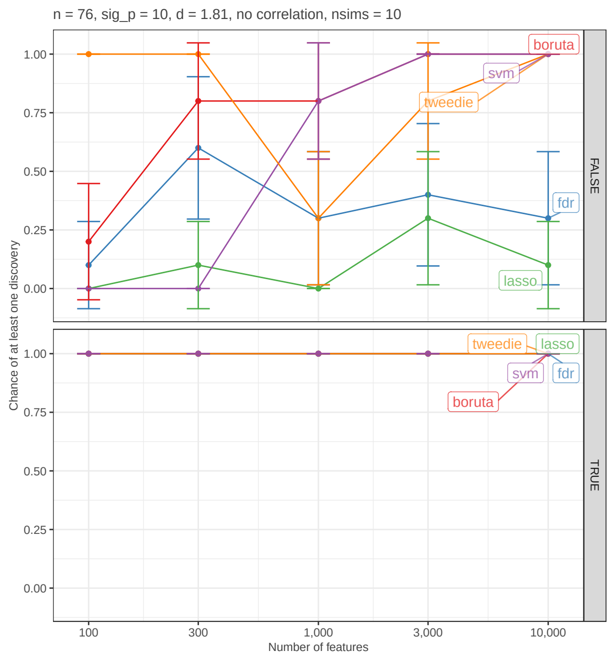
7 Appendices

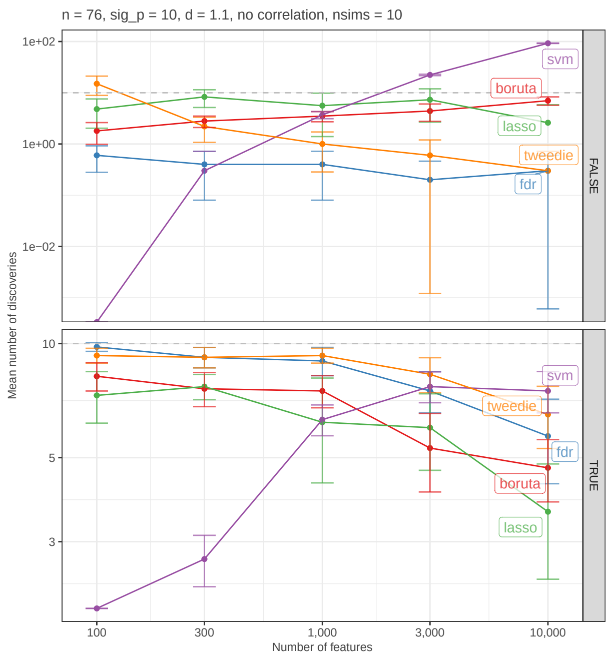
A Exploratory results

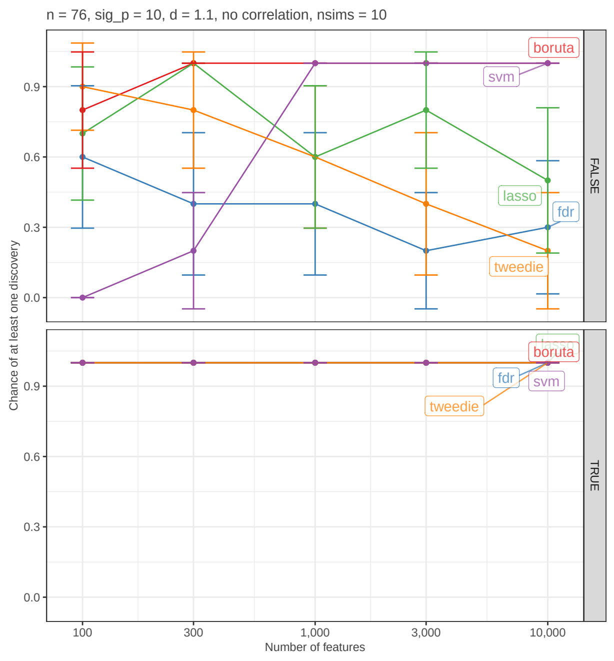
All plots in this section show 95% confidence intervals.

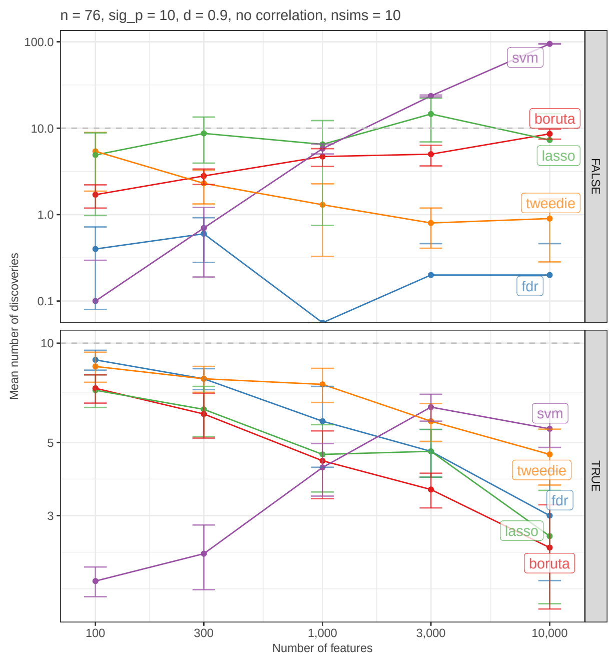
A.1 n = 76, sig_p = 10, uncorrelated, d = varying

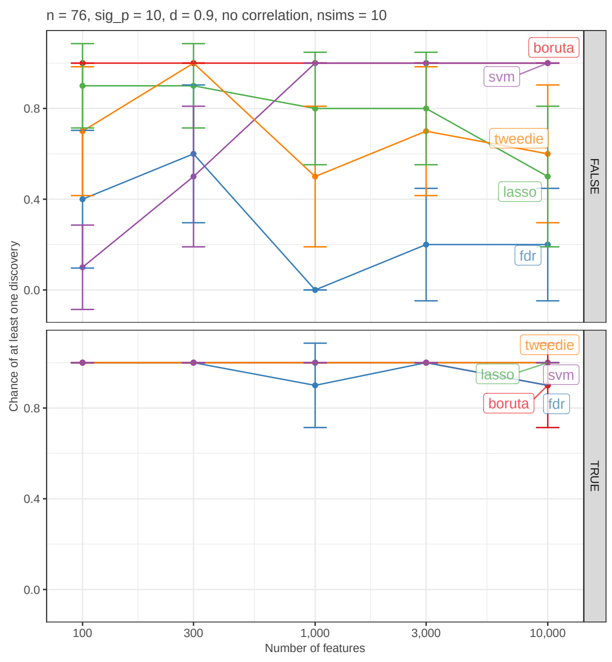


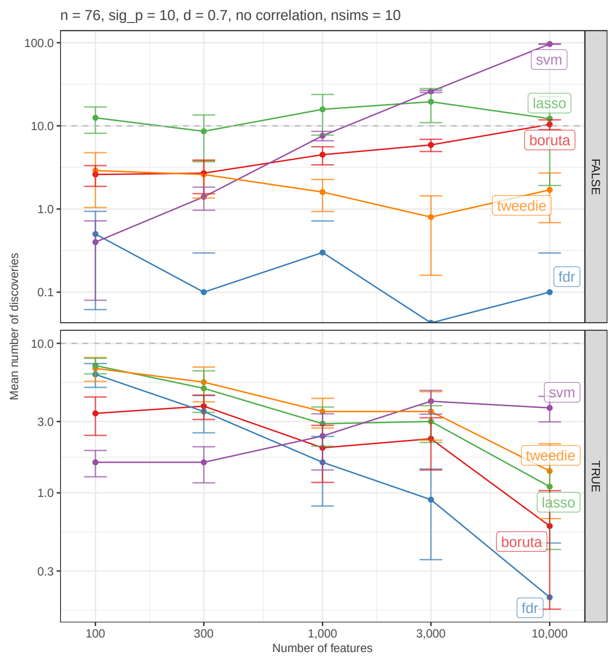


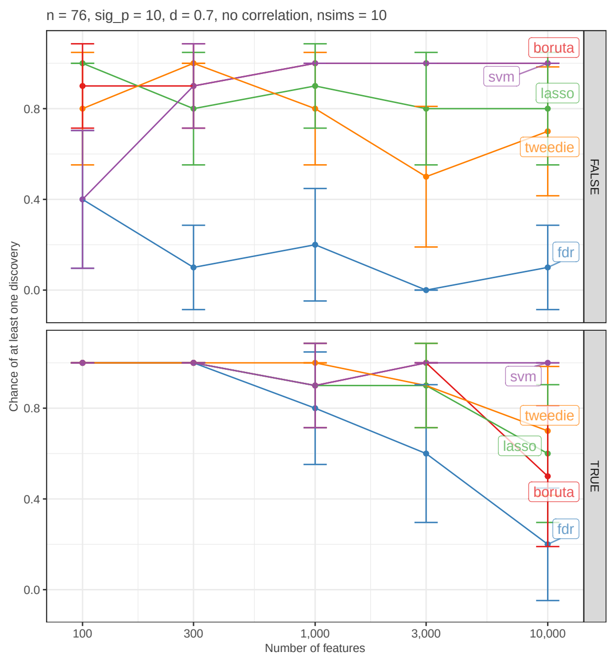


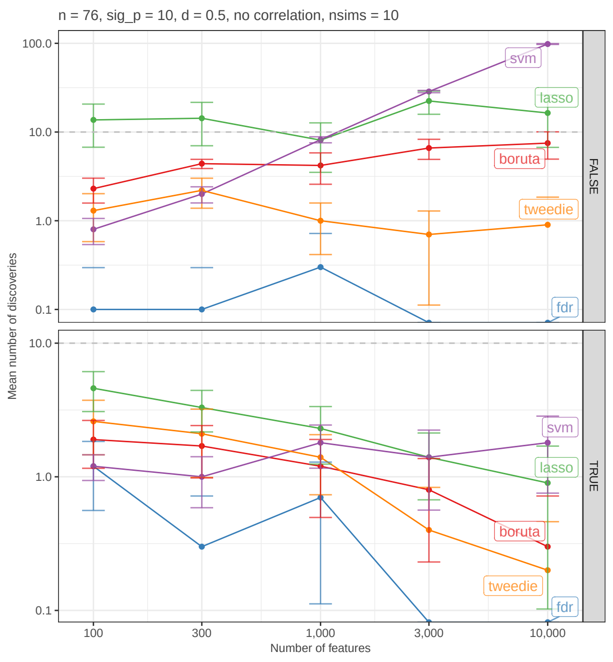


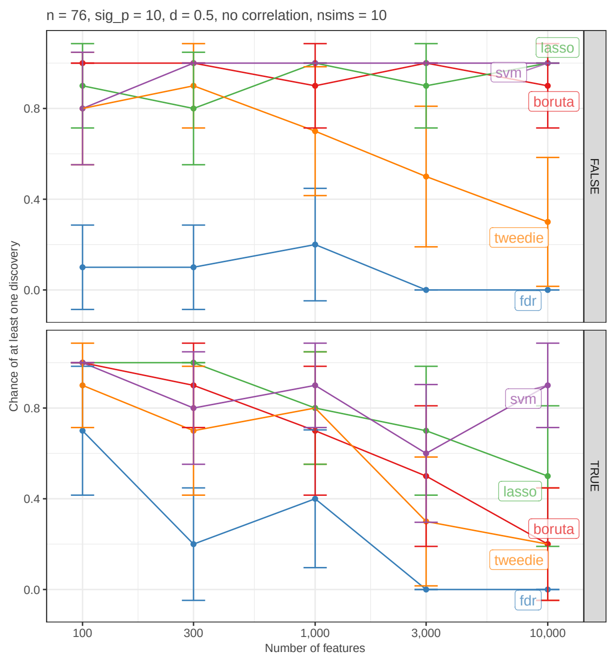

n = 76, sig_p = 10, d = 0.9, no correlation, nsims = 10
Mean number of discoveries
Number of features
100
300
1,000
3,000
10,000
FALSE
TRUE
svm
boruta
lasso
tweedie
fdr

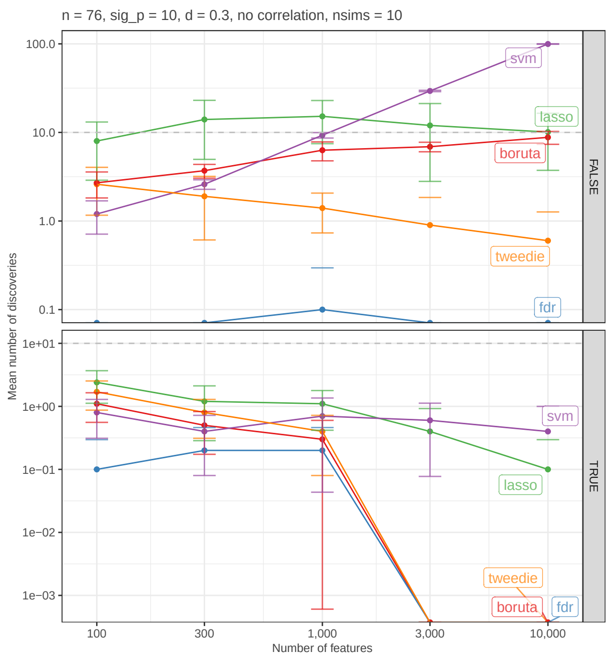

n = 76, sig_p = 10, d = 0.9, no correlation, nsims = 10
Chance of at least one discovery
Number of features
100
300
1,000
3,000
10,000
FALSE
TRUE
boruta
svm
tweedie
lasso
fdr

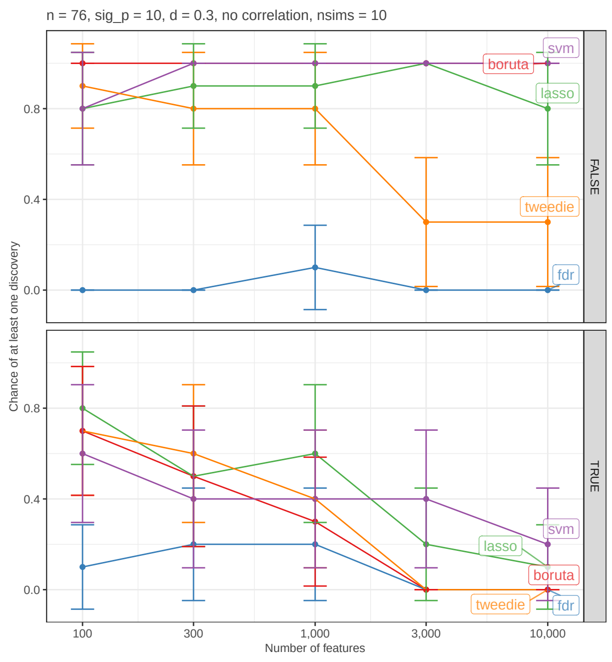

n = 76, sig_p = 10, d = 0.7, no correlation, nsims = 10
Mean number of discoveries
Number of features
100
300
1,000
3,000
10,000
FALSE
TRUE
svm
lasso
boruta
tweedie
fdr

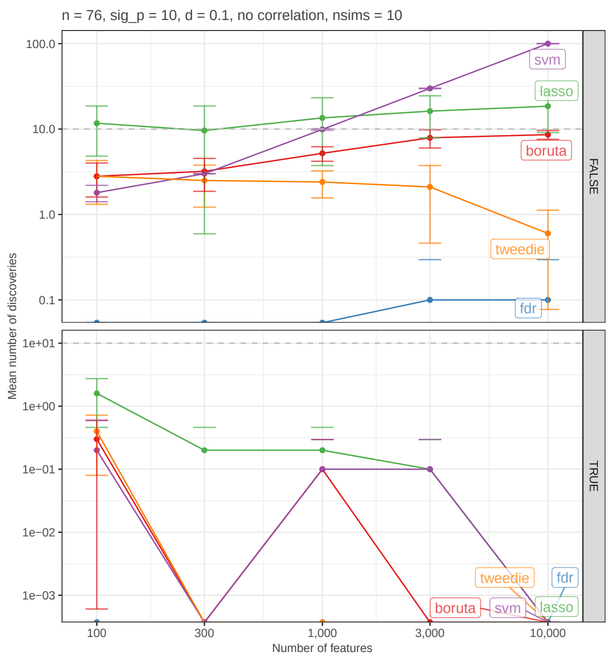

n = 76, sig_p = 10, d = 0.7, no correlation, nsims = 10
Chance of at least one discovery
Number of features
100
300
1,000
3,000
10,000
FALSE
TRUE
boruta
svm
lasso
tweedie
fdr

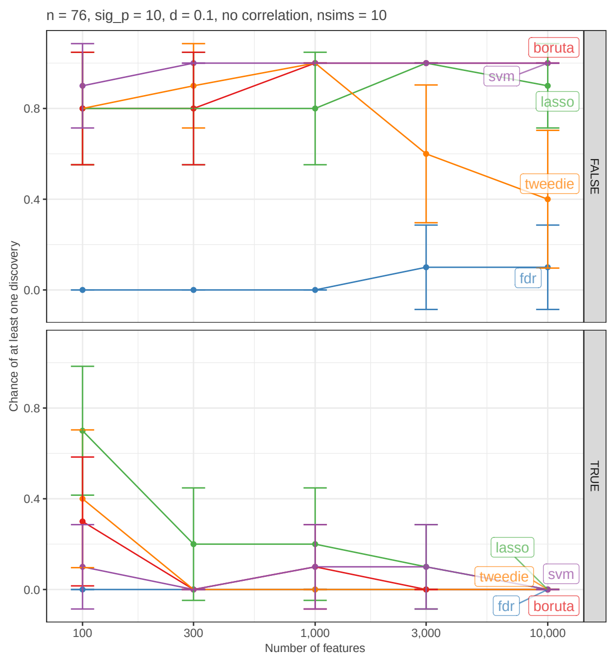

n = 76, sig_p = 10, d = 0.5, no correlation, nsims = 10
Mean number of discoveries
100.0
10.0
1.0
0.1
svm
lasso
boruta
tweedie
fdr
FALSE
TRUE
100
300
1,000
3,000
10,000
Number of features


n = 76, sig_p = 10, d = 0.5, no correlation, nsims = 10
Chance of at least one discovery
0.8
0.4
0.0
lasso
svm
boruta
tweedie
fdr
FALSE
TRUE
100
300
1,000
3,000
10,000
Number of features

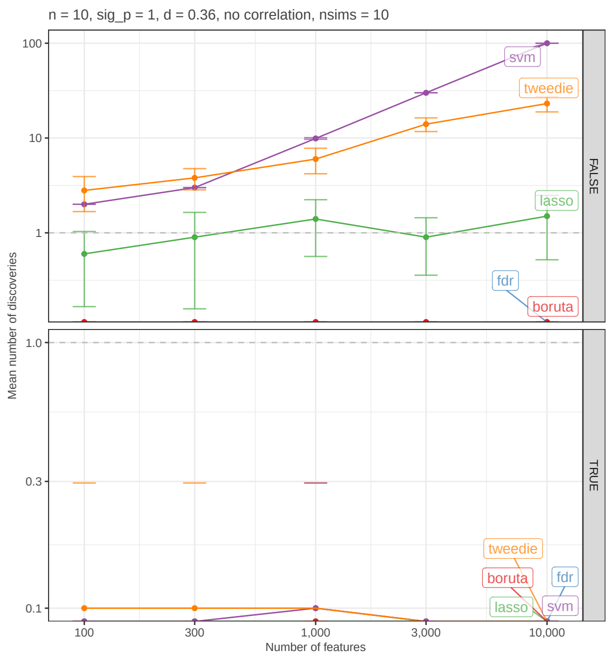

n = 76, sig_p = 10, d = 0.3, no correlation, nsims = 10
Mean number of discoveries
100.0
10.0
1.0
0.1
1e+01
1e+00
1e−01
1e−02
1e−03
svm
lasso
boruta
tweedie
fdr
FALSE
TRUE
100
300
1,000
3,000
10,000
Number of features

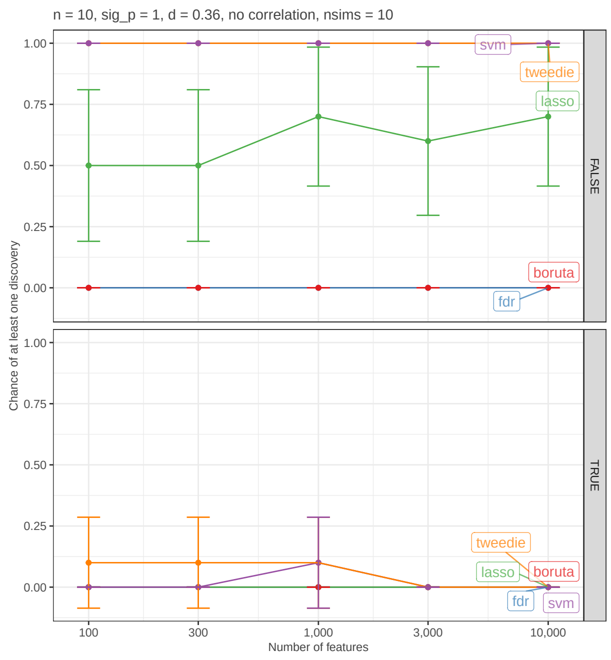

n = 76, sig_p = 10, d = 0.3, no correlation, nsims = 10
Chance of at least one discovery
0.8
0.4
0.0
svm
boruta
lasso
tweedie
fdr
FALSE
TRUE
100
300
1,000
3,000
10,000
Number of features

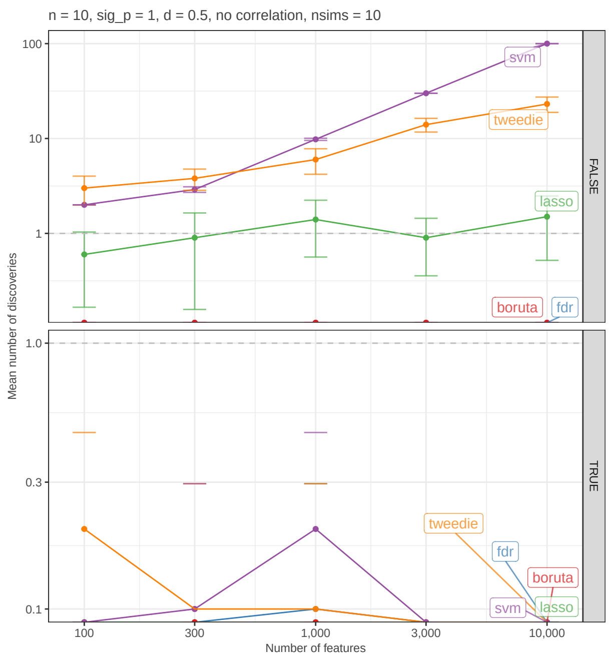
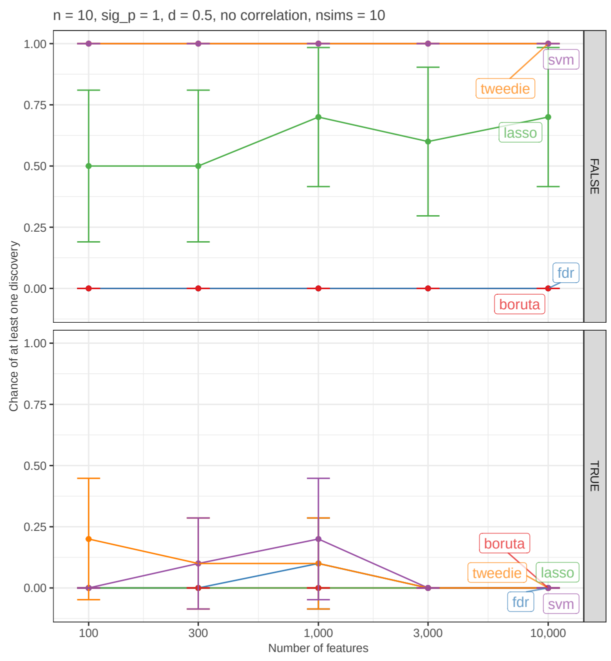
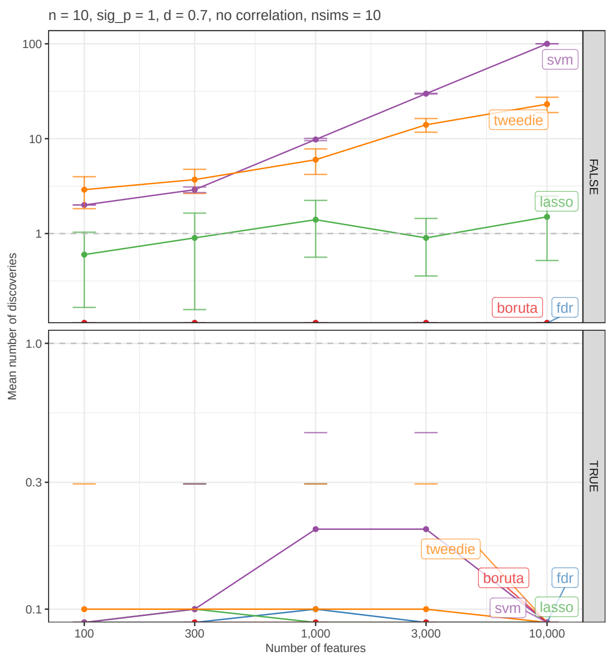
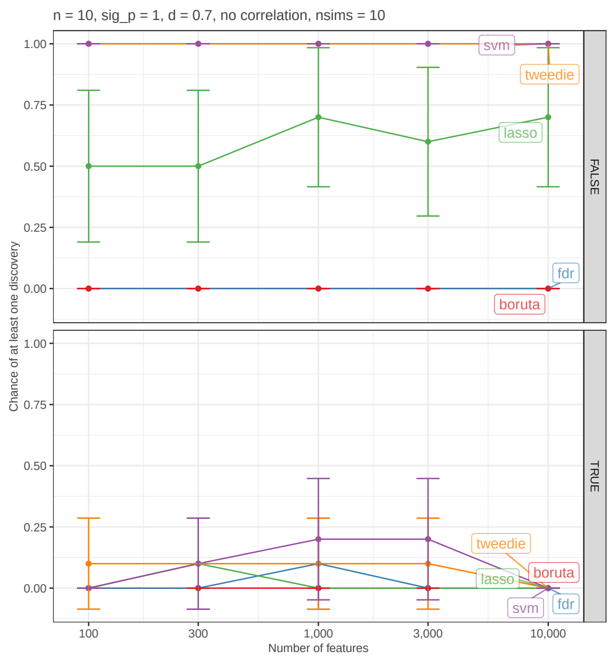


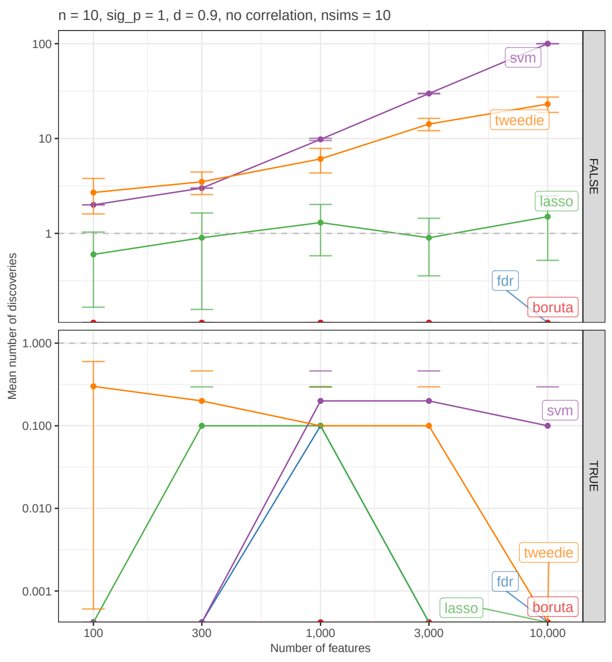


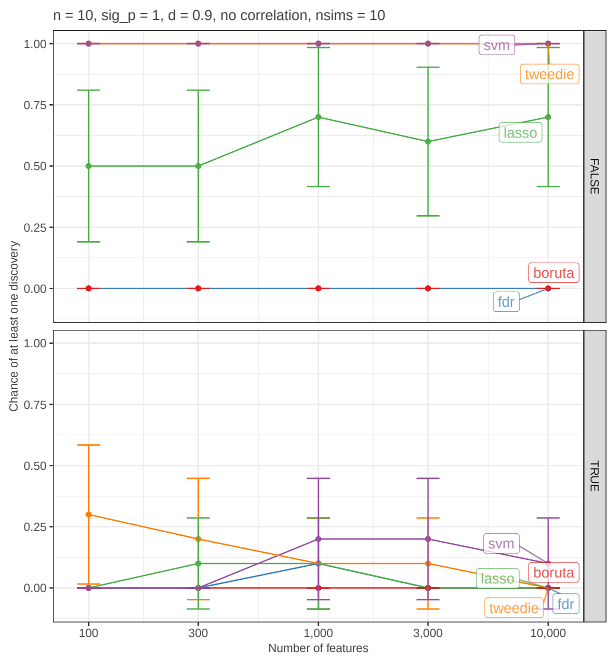
A.2 n = 10, sig_p = 1, uncorrelated, d = varying

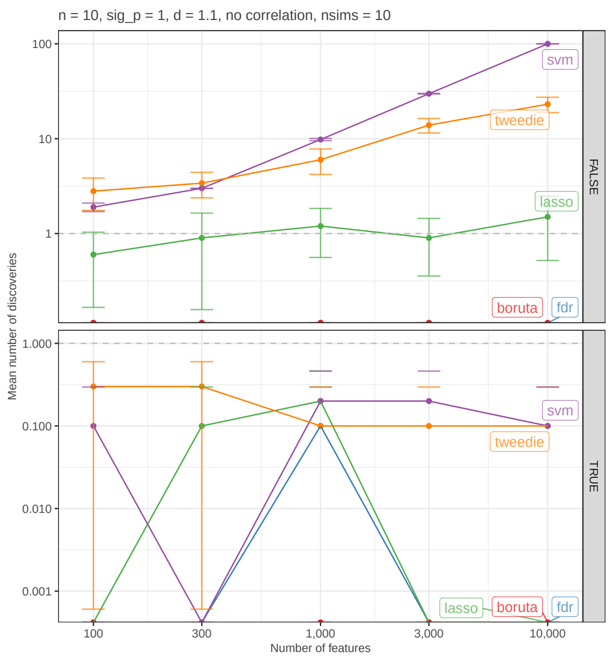


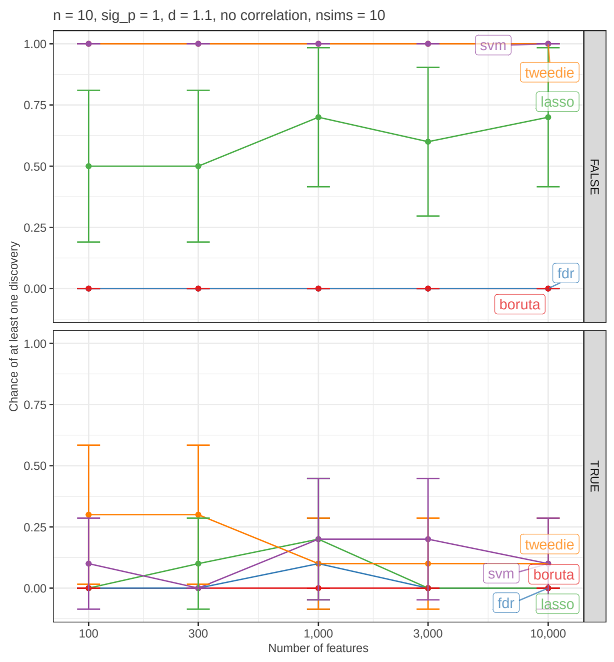


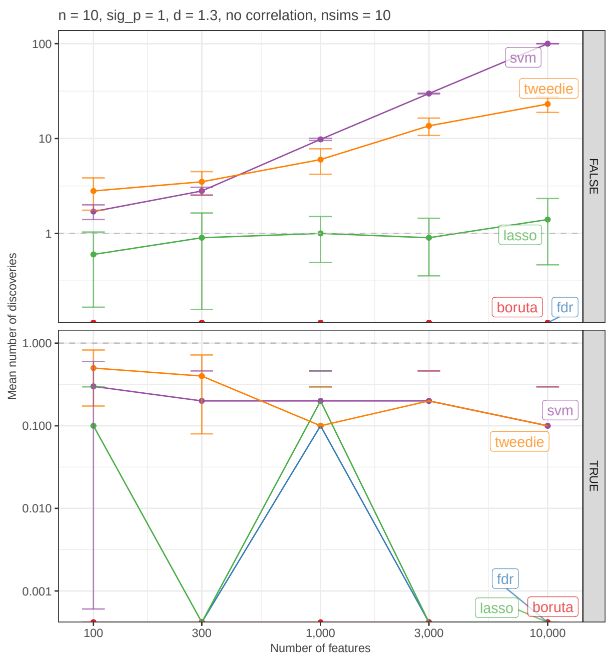

n = 10, sig_p = 1, d = 0.5, no correlation, nsims = 10
Mean number of discoveries
100
10
1
1.0
0.3
0.1
svm
tweedie
lasso
boruta
fdr
FALSE
TRUE
100
300
1,000
3,000
10,000
Number of features

n = 10, sig_p = 1, d = 0.5, no correlation, nsims = 10
Chance of at least one discovery
1.00
0.75
0.50
0.25
0.00
svm
tweedie
lasso
fdr
boruta
FALSE
TRUE
100
300
1,000
3,000
10,000
Number of features

n = 10, sig_p = 1, d = 0.7, no correlation, nsims = 10
Mean number of discoveries
100
10
1
1.0
0.3
0.1
svm
tweedie
lasso
boruta
fdr
FALSE
TRUE
100
300
1,000
3,000
10,000
Number of features

n = 10, sig_p = 1, d = 0.7, no correlation, nsims = 10
Chance of at least one discovery
1.00
0.75
0.50
0.25
0.00
svm
tweedie
lasso
fdr
boruta
FALSE
TRUE
100
300
1,000
3,000
10,000
Number of features

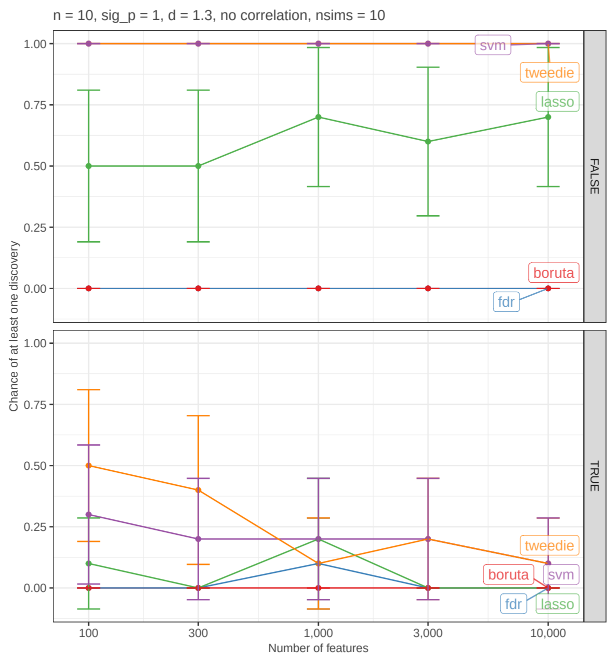

n = 10, sig_p = 1, d = 0.9, no correlation, nsims = 10
Mean number of discoveries
Number of features
100
300
1,000
3,000
10,000
FALSE
TRUE
svm
tweedie
lasso
fdr
boruta

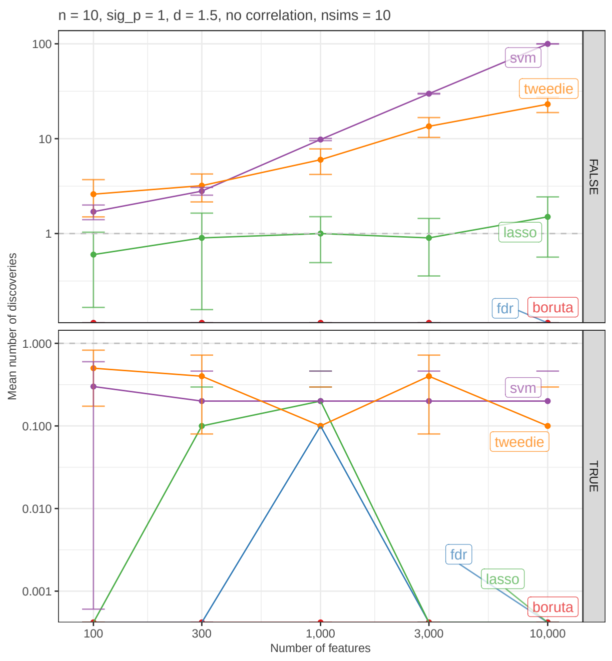

n = 10, sig_p = 1, d = 0.9, no correlation, nsims = 10
Chance of at least one discovery
Number of features
100
300
1,000
3,000
10,000
FALSE
TRUE
svm
tweedie
lasso
boruta
fdr

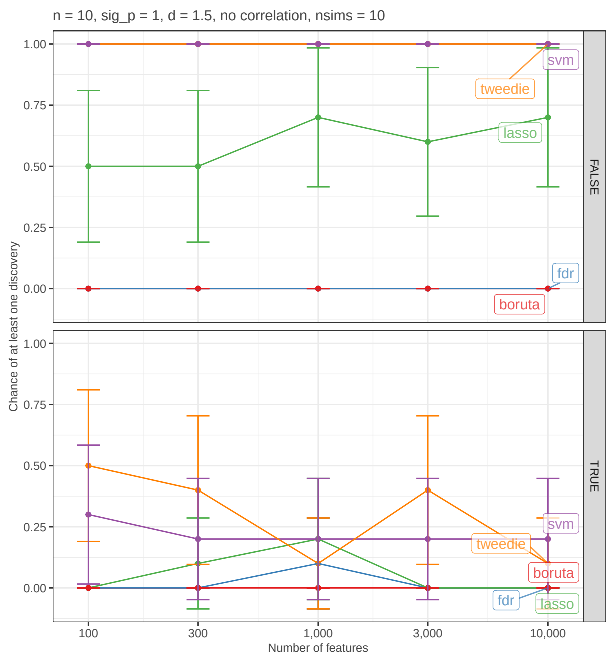

n = 10, sig_p = 1, d = 1.1, no correlation, nsims = 10
Mean number of discoveries
Number of features
100
300
1,000
3,000
10,000
FALSE
TRUE
svm
tweedie
lasso
boruta
fdr

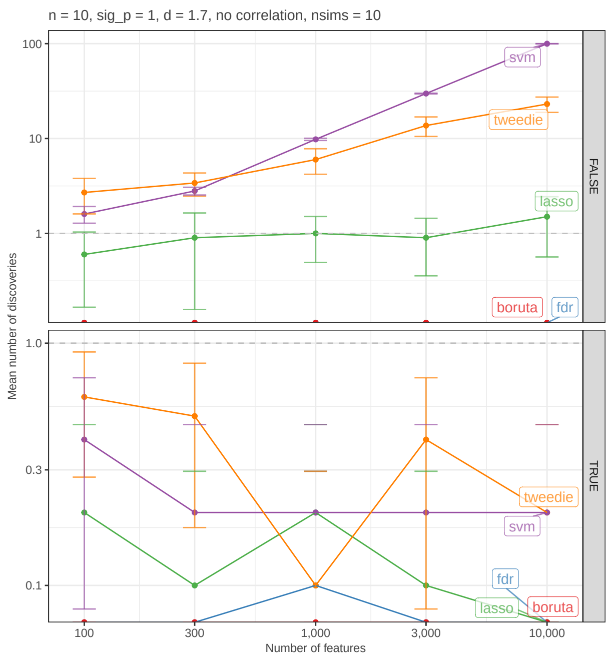

n = 10, sig_p = 1, d = 1.1, no correlation, nsims = 10
Chance of at least one discovery
Number of features
100
300
1,000
3,000
10,000
FALSE
TRUE
svm
tweedie
lasso
fdr
boruta

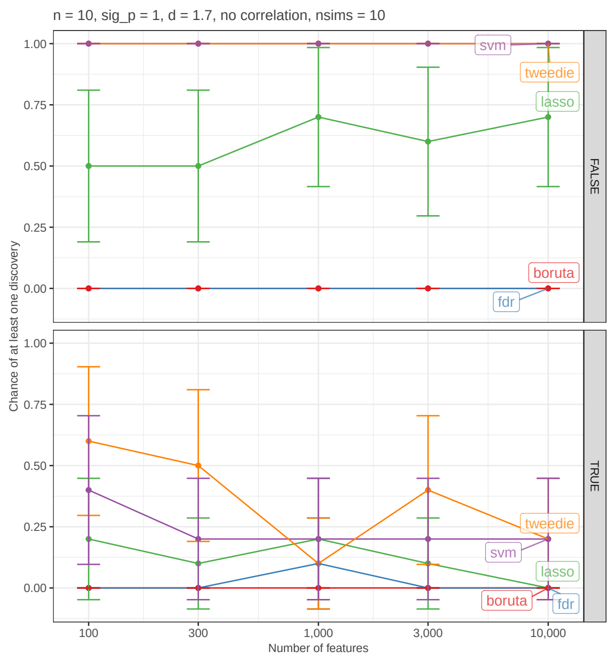

n = 10, sig_p = 1, d = 1.3, no correlation, nsims = 10
Mean number of discoveries
svm
tweedie
lasso
boruta
fdr
FALSE
TRUE
Number of features
100
300
1,000
3,000
10,000

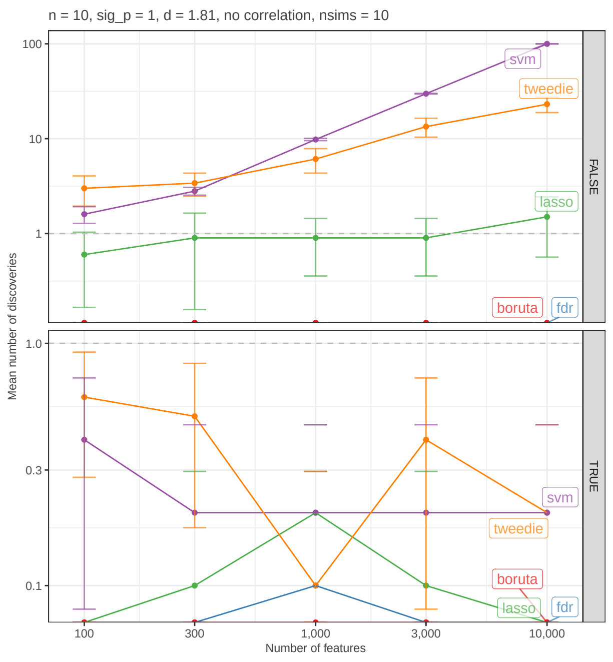

n = 10, sig_p = 1, d = 1.3, no correlation, nsims = 10
Chance of at least one discovery
svm
tweedie
lasso
boruta
fdr
FALSE
TRUE
Number of features

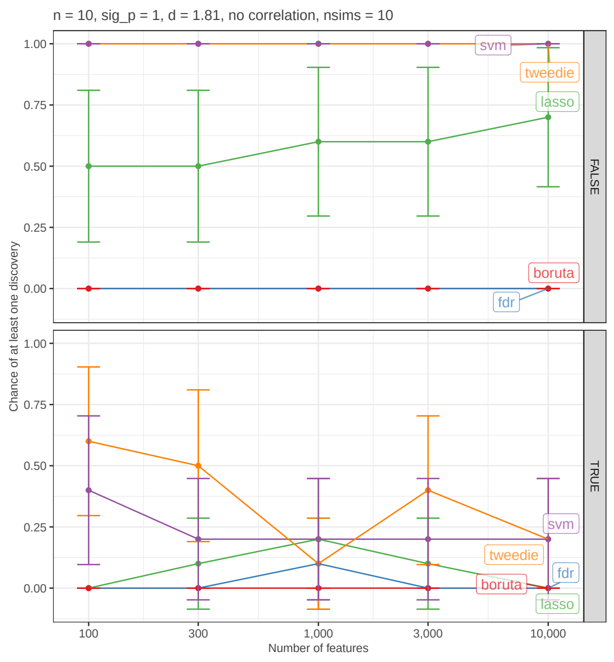

n = 10, sig_p = 1, d = 1.5, no correlation, nsims = 10
Mean number of discoveries
svm
tweedie
lasso
fdr
boruta
FALSE
TRUE
Number of features


n = 10, sig_p = 1, d = 1.5, no correlation, nsims = 10
Chance of at least one discovery
svm
tweedie
lasso
fdr
boruta
FALSE
TRUE
Number of features

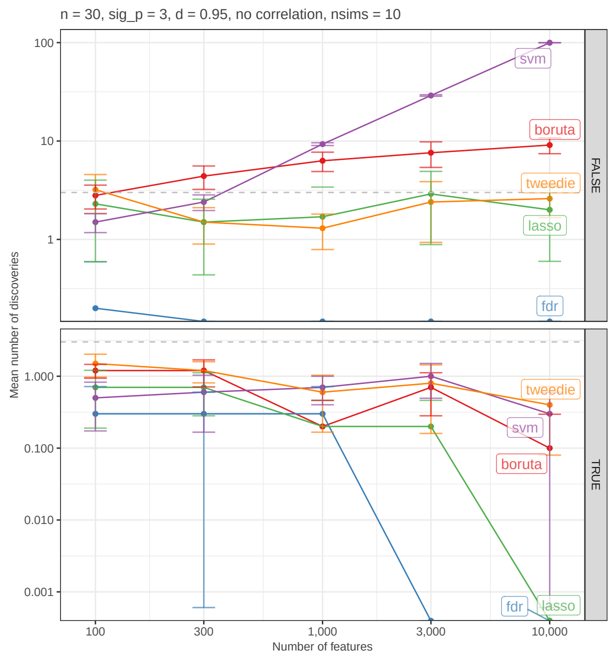

n = 10, sig_p = 1, d = 1.7, no correlation, nsims = 10
Mean number of discoveries
Number of features
100
300
1,000
3,000
10,000
FALSE
TRUE
svm
tweedie
lasso
boruta
fdr

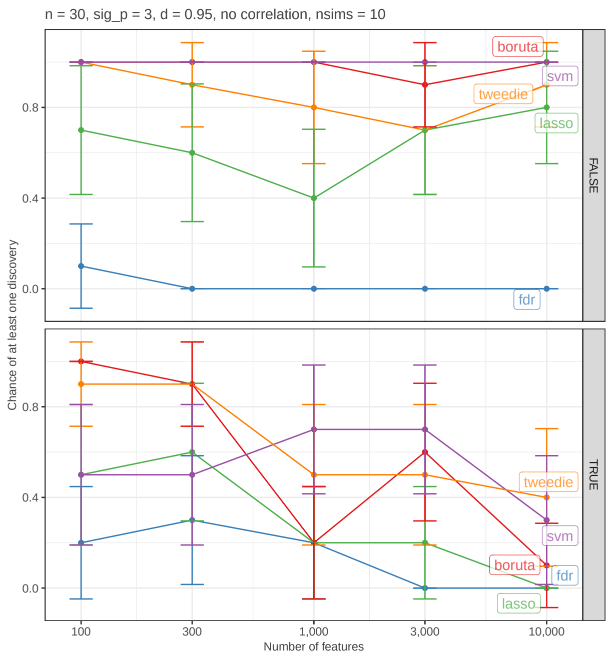

n = 10, sig_p = 1, d = 1.7, no correlation, nsims = 10
Chance of at least one discovery
Number of features
100
300
1,000
3,000
10,000
FALSE
TRUE
svm
tweedie
lasso
boruta
fdr

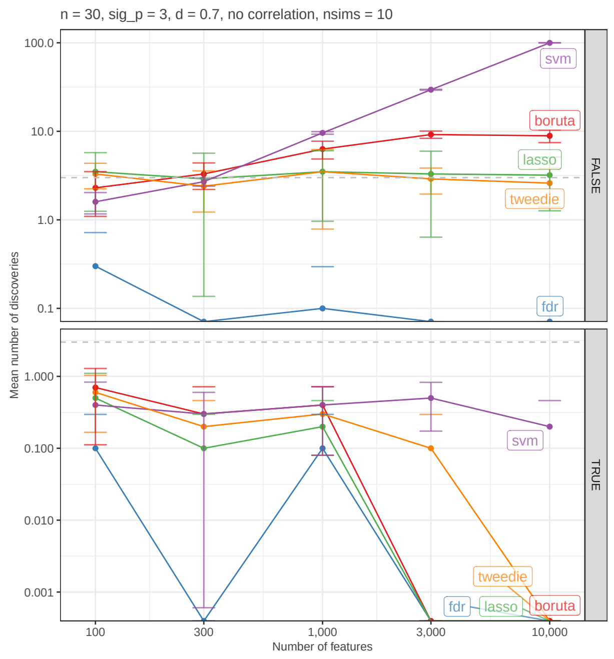

n = 10, sig_p = 1, d = 1.81, no correlation, nsims = 10
Mean number of discoveries
Number of features
100
300
1,000
3,000
10,000
FALSE
TRUE
svm
tweedie
lasso
boruta
fdr

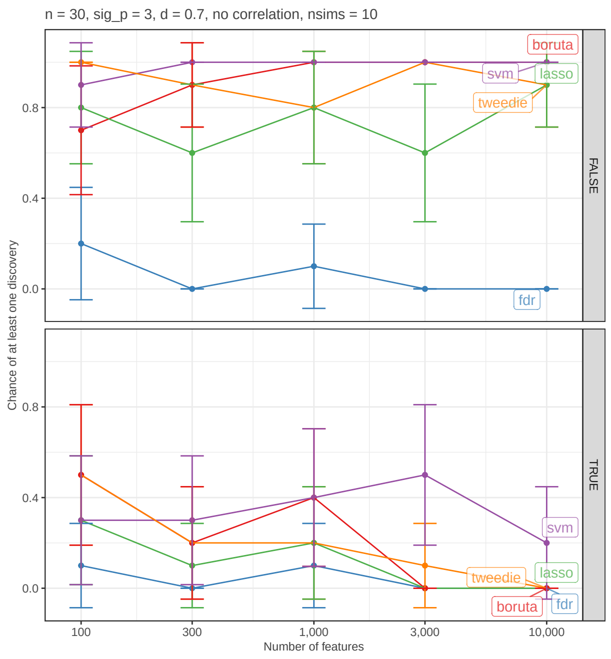

n = 10, sig_p = 1, d = 1.81, no correlation, nsims = 10
Chance of at least one discovery
Number of features
100
300
1,000
3,000
10,000
FALSE
TRUE
svm
tweedie
lasso
boruta
fdr

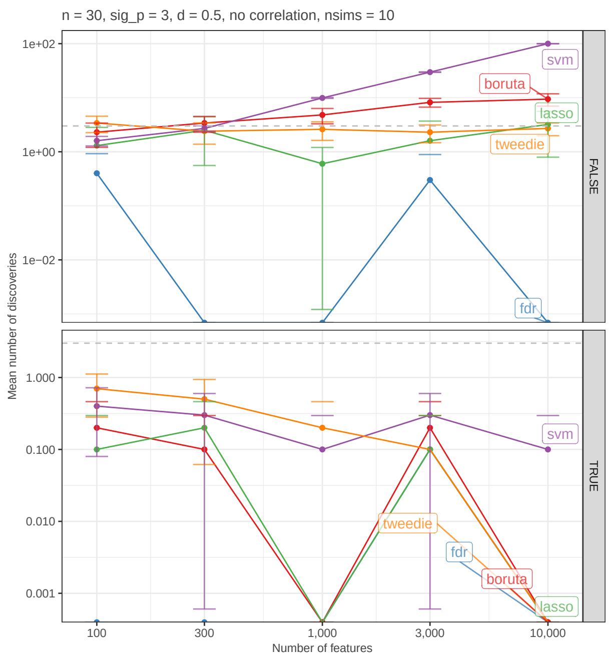
A.3 n = 30, sig_p = 3, uncorrelated, d = varying

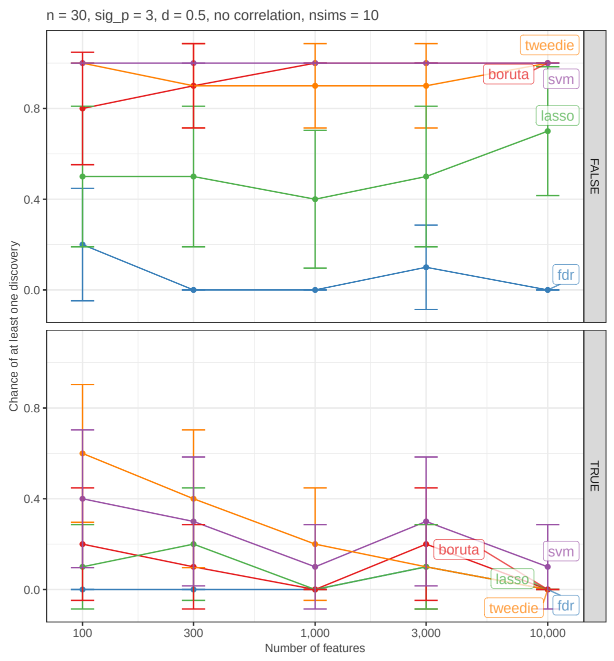


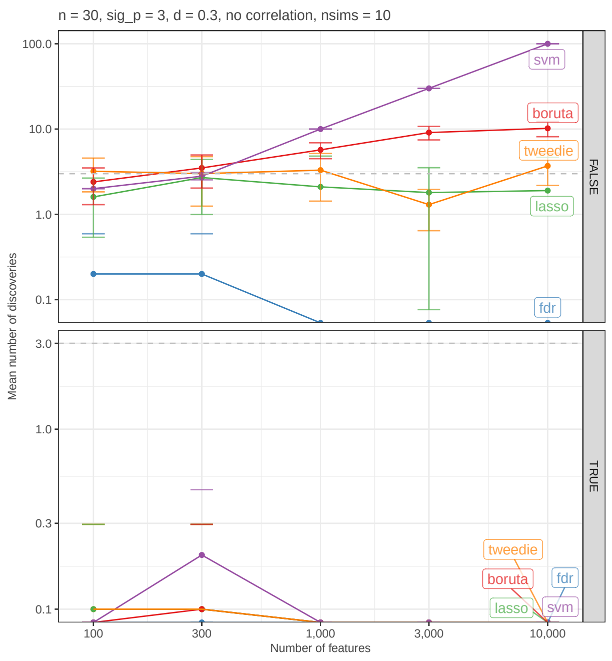


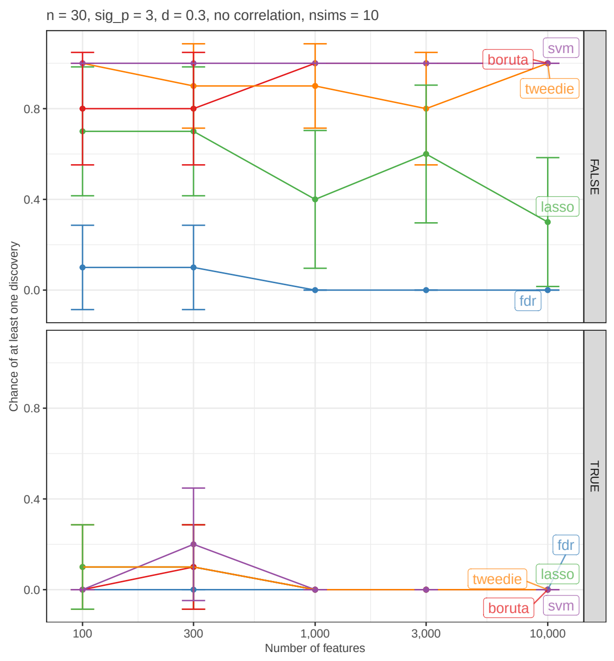


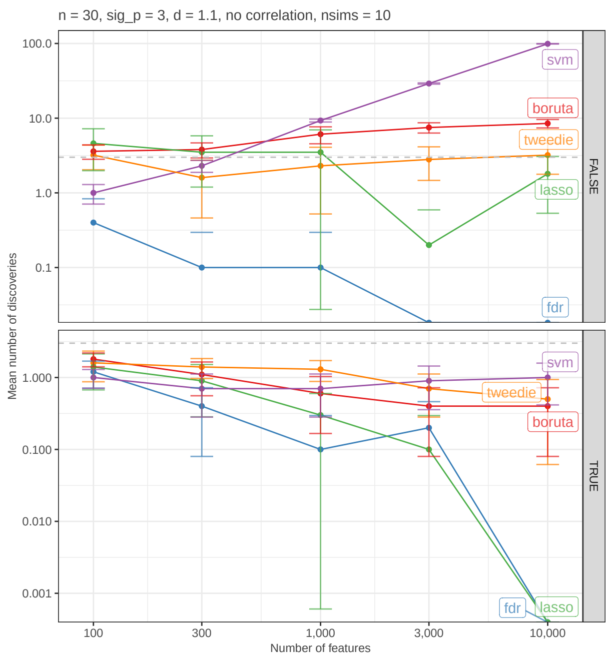


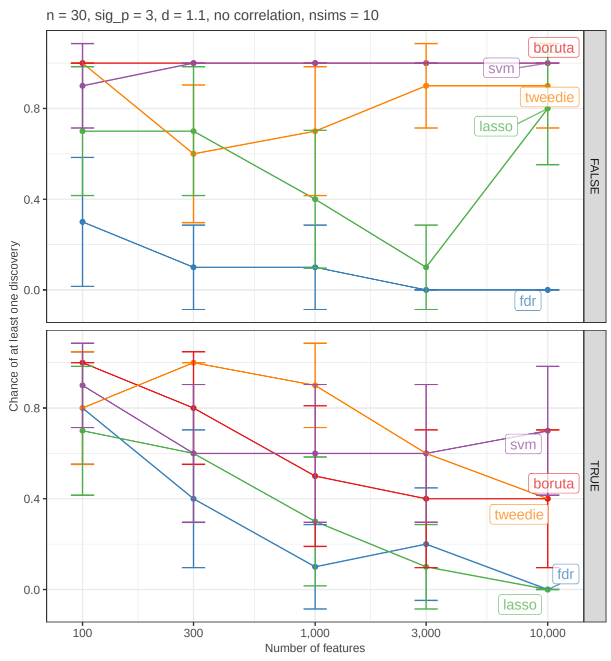

n = 30, sig_p = 3, d = 0.5, no correlation, nsims = 10
Mean number of discoveries
Number of features
FALSE
TRUE
svm
boruta
lasso
tweedie
fdr

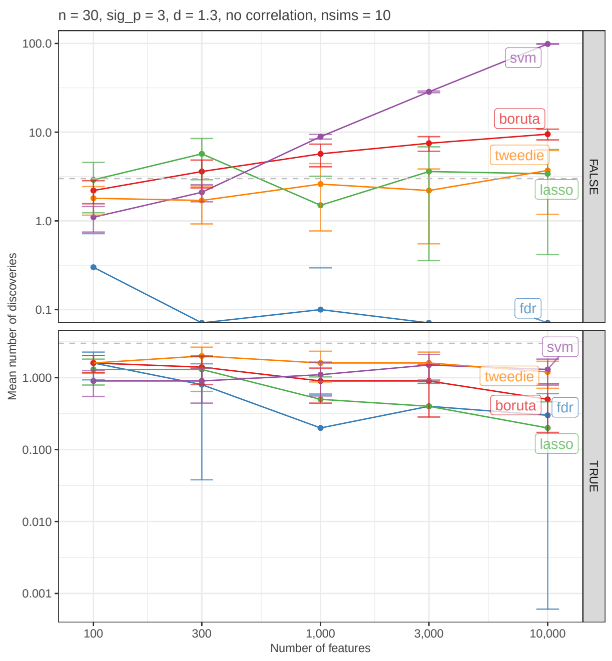

n = 30, sig_p = 3, d = 0.5, no correlation, nsims = 10
Chance of at least one discovery
Number of features
FALSE
TRUE
tweedie
boruta
svm
lasso
fdr

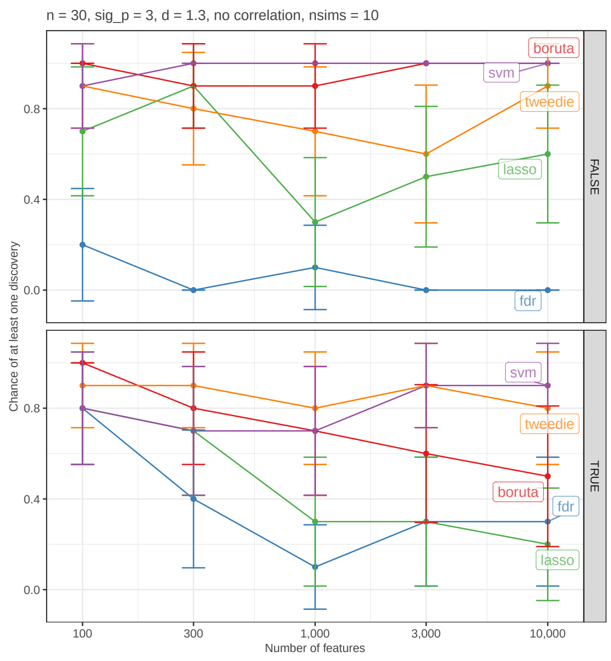

n = 30, sig_p = 3, d = 0.3, no correlation, nsims = 10
Mean number of discoveries
Number of features
FALSE
TRUE
svm
boruta
tweedie
lasso
fdr

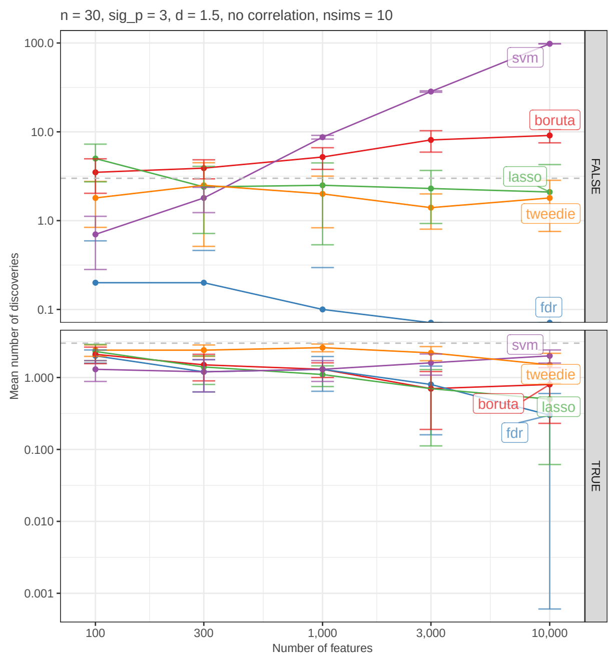

n = 30, sig_p = 3, d = 0.3, no correlation, nsims = 10
Chance of at least one discovery
Number of features
FALSE
TRUE
svm
boruta
tweedie
lasso
fdr

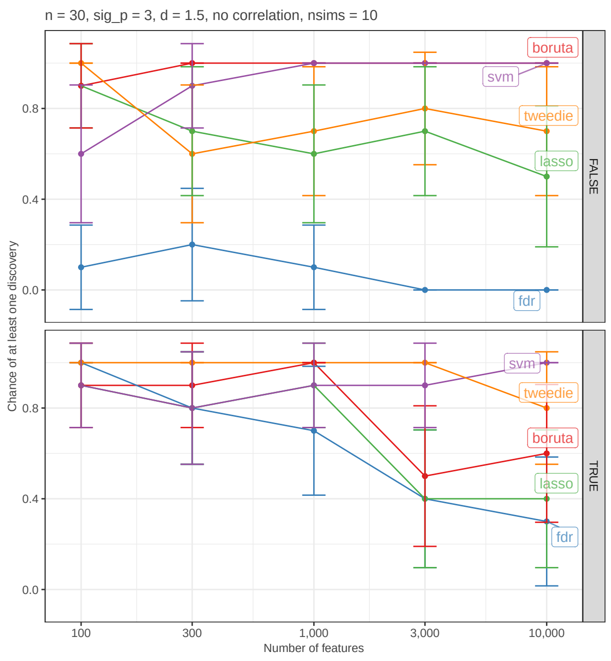

n = 30, sig_p = 3, d = 1.1, no correlation, nsims = 10
Mean number of discoveries
Number of features
FALSE
TRUE
svm
boruta
tweedie
lasso
fdr

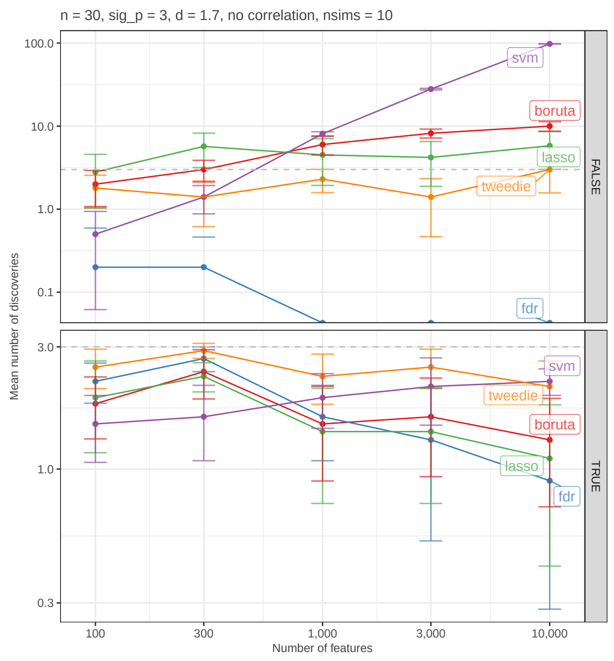

n = 30, sig_p = 3, d = 1.1, no correlation, nsims = 10
Chance of at least one discovery
Number of features
FALSE
TRUE
boruta
svm
tweedie
lasso
fdr

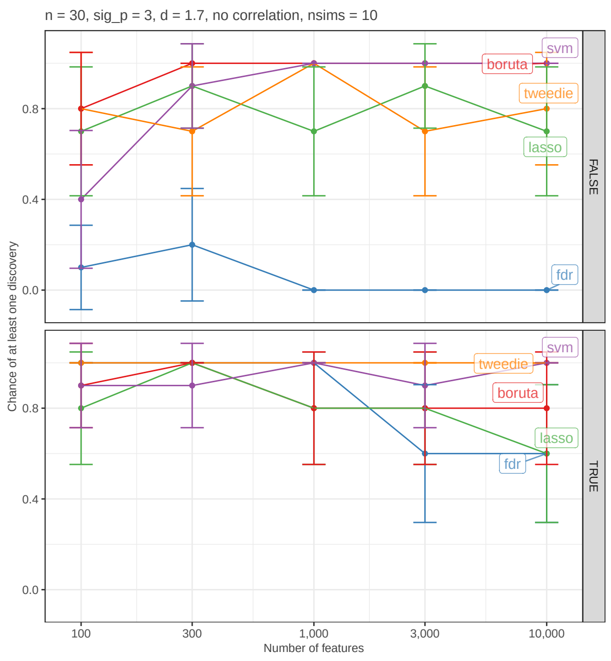

n = 30, sig_p = 3, d = 1.3, no correlation, nsims = 10
Mean number of discoveries
Number of features
FALSE
TRUE
svm
boruta
tweedie
lasso
fdr

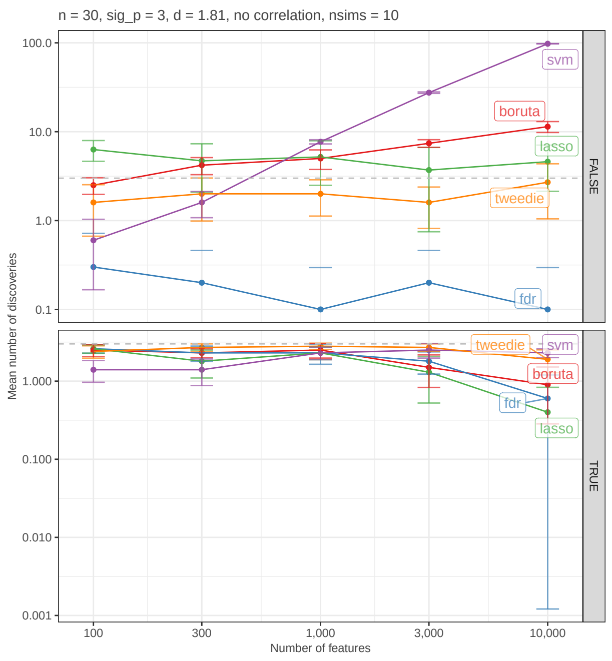

n = 30, sig_p = 3, d = 1.3, no correlation, nsims = 10
Chance of at least one discovery
Number of features
FALSE
TRUE
boruta
svm
tweedie
lasso
fdr

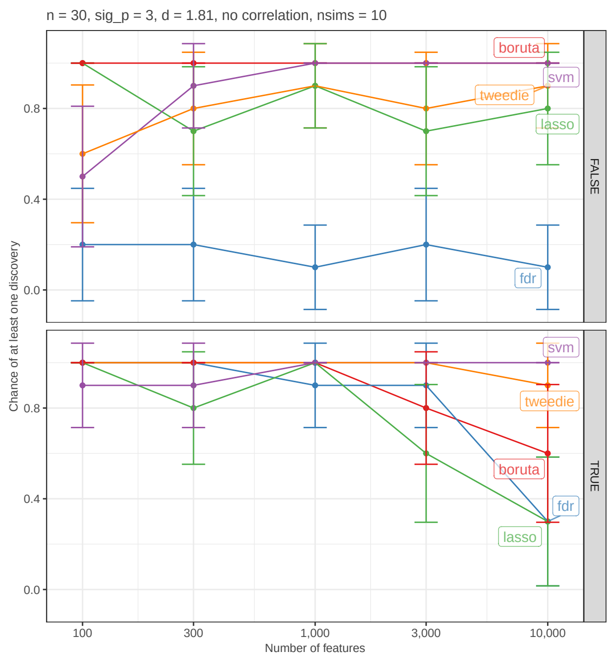

n = 30, sig_p = 3, d = 1.5, no correlation, nsims = 10
Mean number of discoveries
Number of features
100
300
1,000
3,000
10,000
FALSE
TRUE
svm
boruta
lasso
tweedie
fdr

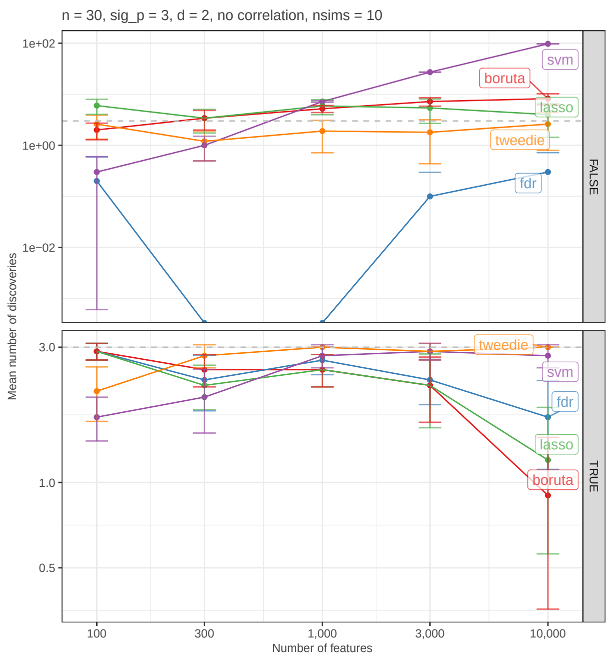

n = 30, sig_p = 3, d = 1.5, no correlation, nsims = 10
Chance of at least one discovery
Number of features
100
300
1,000
3,000
10,000
FALSE
TRUE
boruta
svm
tweedie
lasso
fdr

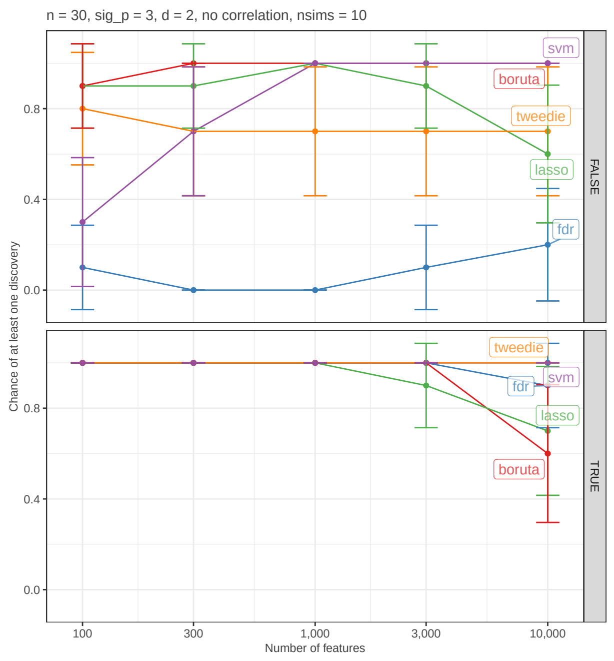

n = 30, sig_p = 3, d = 1.7, no correlation, nsims = 10
Mean number of discoveries
Number of features
100
300
1,000
3,000
10,000
FALSE
TRUE
svm
boruta
lasso
tweedie
fdr


n = 30, sig_p = 3, d = 1.7, no correlation, nsims = 10
Chance of at least one discovery
Number of features
100
300
1,000
3,000
10,000
FALSE
TRUE
svm
boruta
tweedie
lasso
fdr









B Session info

```
R version 4.1.1 (2021-08-10)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 18.04.5 LTS

Matrix products: default
```

```
BLAS:   /usr/lib/x86_64-linux-gnu/blas/libblas.so.3.7.1
LAPACK: /usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.7.1

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
 [3] LC_TIME=en_GB.UTF-8        LC_COLLATE=en_US.UTF-8
 [5] LC_MONETARY=en_GB.UTF-8    LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_GB.UTF-8       LC_NAME=C
 [9] LC_ADDRESS=C               LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_GB.UTF-8 LC_IDENTIFICATION=C

attached base packages:
[1] parallel  stats     graphics  grDevices utils     datasets  methods
[8] base

other attached packages:
 [1] ecolicdf_2.18.0     affy_1.70.0         Biobase_2.52.0
 [4] BiocGenerics_0.38.0 limma_3.48.3        ggrepel_0.9.1
 [7] kableExtra_1.3.4    ggcorrplot_0.1.3    forcats_0.5.1
[10] stringr_1.4.0       dplyr_1.0.6         purrr_0.3.4
[13] readr_1.4.0         tidyr_1.1.3         tibble_3.1.2
[16] ggplot2_3.3.3       tidyverse_1.3.1

loaded via a namespace (and not attached):
 [1] bitops_1.0-7           fs_1.5.0               lubridate_1.7.10
 [4] bit64_4.0.5            webshot_0.5.2          RColorBrewer_1.1-2
 [7] httr_1.4.2             GenomeInfoDb_1.28.2    tools_4.1.1
[10] backports_1.2.1        utf8_1.2.1             R6_2.5.0
[13] affyio_1.62.0          DBI_1.1.1              colorspace_2.0-1
[16] withr_2.4.2            tidyselect_1.1.1       bit_4.0.4
[19] compiler_4.1.1         preprocessCore_1.54.0  cli_2.5.0
[22] rvest_1.0.0            xml2_1.3.2             labeling_0.4.2
[25] bookdown_0.22          scales_1.1.1           mvtnorm_1.1-2
[28] systemfonts_1.0.2      digest_0.6.27          rmarkdown_2.8
[31] svglite_2.0.0          XVector_0.32.0         pkgconfig_2.0.3
[34] htmltools_0.5.1.1      fastmap_1.1.0          dbplyr_2.1.1
[37] rlang_0.4.11           readxl_1.3.1           rstudioapi_0.13
[40] RSQLite_2.2.8          farver_2.1.0           generics_0.1.0
[43] jsonlite_1.7.2         RCurl_1.98-1.4         magrittr_2.0.1
[46] GenomeInfoDbData_1.2.6 S4Vectors_0.30.0       Rcpp_1.0.7
[49] munsell_0.5.0          fansi_0.5.0            lifecycle_1.0.0
[52] stringi_1.6.2          yaml_2.2.1             zlibbioc_1.38.0
[55] plyr_1.8.6             grid_4.1.1             blob_1.2.1
```

```
[58] crayon_1.4.1          Biostrings_2.60.2      haven_2.4.1
[61] KEGGREST_1.32.0       hms_1.1.0              knitr_1.33
[64] pillar_1.6.1          reshape2_1.4.4         stats4_4.1.1
[67] reprex_2.0.0          glue_1.4.2             evaluate_0.14
[70] BiocManager_1.30.16   modelr_0.1.8           png_0.1-7
[73] vctrs_0.3.8           cellranger_1.1.0       gtable_0.3.0
[76] assertthat_0.2.1      cachem_1.0.5           xfun_0.23
[79] broom_0.7.6           viridisLite_0.4.0      IRanges_2.26.0
[82] AnnotationDbi_1.54.1  memoise_2.0.0          ellipsis_0.3.2
```