VARIABLE SELECTION FOR CATEGORICAL RESPONSE DATA WITH MULTIPLE CATEGORIES

PROJECT REPORT

Ramit Nandi
Roll No : MD2211
Course : M. Stat. (2nd Year)

Supervisor
Dr. Soham Sarkar (SMU,ISI-D)

May 25, 2024

1 Introduction

There exists a rich literature on variable selection, since it is a classical problem in statistics. The problem is more relevant in the recent years than it was before, because of the availability of a vast amount of data with large sets of variables. Even relaxing the need for computational resources, a prediction can behave like random guess in the presence of a large number of features in training data, simply due to high collinearity, spurious correlation, noise accumulation etc. With a proper selection method and under suitable conditions, it is possible to build consistent models which are easy to interpret, to avoid overfitting in prediction and estimation, and to identify relevant variables for applications or further study.

In this project we discuss variable selection as a two step procedure - at first we assign importance scores corresponding to features in a dataset and in the next step we set a threshold to select/reject a feature based on those scores.

NOTATION If not mentioned otherwise, we will consider

$$n \text{ observations} : \{\vec{\boldsymbol{x}}_i, \vec{\boldsymbol{y}}_i\}_{i=1}^n$$

where $\vec{\boldsymbol{x}}_i \in \mathbb{R}^m$ (i.e. #features = m) and $\vec{\boldsymbol{y}}_i \in \{0,1\}^c$ (i.e. #target classes = c). We denote the j-th feature column as $\mathbf{x}_j \in \mathbb{R}^n$ and also denote the one-hot encoding corresponding to k-th target class as $\mathbf{y}_k \in \{0,1\}$. Hence the observed data is of the form

$$dataset : \mathbf{X}^{n\times m}, \mathbf{y}^{n\times c}$$

So each row of **x** and **y** is an observation. Each column of **x** corresponds to a feature, while each column of **y** is indicator variable corresponding to a target class

Also, for any matrix **A**, we will denote the i-th row and j-th column respectively as $\mathbf{A}_{io}$ and $\mathbf{A}_{oj}$.

2 An analogue of t-test for classification setup : How things change beyond generalized linear model (GLM)

Consider first the linear model

$$\vec{\boldsymbol{y}}_i^{continuous}|\vec{\boldsymbol{x}}_i \sim \mathcal{N}(\sum_{j=1}^{m} \beta_j \mathbf{x}_{ij}, \sigma^2)$$

In a linear model, the basic way to determine whether a feature $\mathbf{x}_j$ is important or not is to test $\mathbf{H}_{0j} : \beta_j = 0$ vs. $\mathbf{H}_{1j} : \beta_j \neq 0$. Similar approach can be adopted for binary logistic regression model

$$\vec{\boldsymbol{y}}_i^{binary}|\vec{\boldsymbol{x}}_i \sim Bernoulli\big(p_i = \frac{1}{1+\exp(-\sum_{j=1}^{m} \beta_j \mathbf{x}_{ij})}\big)$$

For both linear regression model or binary classification model above, j-th feature is important iff $\mathbf{H}_{0j}$ is rejected. Smaller the p-value higher the importance of a feature.

But, the case of multiclass classification is quite different. For instance, consider the following multiclass model (we will stick to this particular parameterization to match **python** convention)

$$\vec{\boldsymbol{y}}_i|\vec{\boldsymbol{x}}_i \sim multinomial(1; p_{i1}, p_{i2}, ..., p_{ic})$$

with $p_{ik} = \frac{\exp(\boldsymbol{\beta}_{ko}\vec{\boldsymbol{x}}_i)}{\sum_{r=1}^{c}\exp(\boldsymbol{\beta}_{ro}\vec{\boldsymbol{x}}_i)}$, $\sum_{k=1}^{c} p_{ik} = 1$; $\boldsymbol{\beta} \in \mathbb{R}^{c\times m}$ but $(c-1)\times m$ coefficients are identifiable

[We denote the above model as $\vec{\boldsymbol{y}}_i|\vec{\boldsymbol{x}}_i \sim multinomial_{softmax}(\boldsymbol{\beta})$] The first difference we notice here is that the j-th feature now corresponds to more than one coefficients $\beta_{kj}, k = 1, 2, ..., c$. Feature $\mathbf{x}_j$ is important iff $\mathbf{H}_{0j} : \beta_{1j} = \beta_{2j} = ... = \beta_{cj}$ is rejected. But directly performing such a test is not so easy.

- When we use MLE as our $\hat{\boldsymbol{\beta}}$, we can exploit the asymptotic normality to perform wald test. For various penalized regression (e.g. LASSO) methods it is not valid anymore.
- Some classifiers (e.g. SVM) are inherently binary in nature, multiclass response is handled via a collection of One-vs-Rest binary classification. Corresponding to each binary subproblem we will have a testing $\mathbf{H}_{0j}^{k}$, $k = 1, 2, ..., c$ there. We can decide "The j-th feature is important iff $\mathbf{H}_{0j}^{k}$ is rejected for atleast one k". But for each feature $\mathbf{x}_j$ we have the set of p-values $\{p_{j,1}, p_{j,2}, ..., p_{j,c}\}$ now instead of a value p_j. It is not clear to us how to conclude "Which feature is more important than others?"
- Sometimes we can use nonparametric classifiers (e.g. Decision Tree) as well. No $\hat{\boldsymbol{\beta}}$ is there in the estimated model anymore, but measuring the importance of the features is still a good point of interest.

Based on the analogy with linear model we propose the importance of $\mathbf{x}_j$ in the population as

$$\mathbf{T}_j = \sum_{l=1}^{c}\sum_{k\neq l} |\beta_{kj} - \beta_{lj}|$$

In the next Section 3 we discuss -"How we can assign a scalar importance to each feature of a dataset" and in Section 4 we revisit the underlying question behind the testing problem - "Is the j-th feature is relevant?"

3 Different types of available methods considered in this project

3.1 Least Absolute Shrinkage and Selection Operator (LASSO)

LASSO [1] is arguably the most popular feature selection method in recent days. In the case of multiclass classification LASSO can be applied in two ways -

- **One-vs-Rest (OvR) :** We can formulate One-vs-Rest subproblems by considering one target class as positive at each time. Thus each subproblem becomes a binary Logistic-Lasso Regression. Let $\boldsymbol{b}_k \in \mathbb{R}^m; k = 1, 2, ..., c$

be the solution of $\min_{\boldsymbol{\beta}_k \in \mathbb{R}^m} \sum_{i=1}^{n} [\mathbf{y}_{ik} \log p_{ik} + (1 - \mathbf{y}_{ik}) \log(1 - p_{ik})] + \lambda_k \|\boldsymbol{\beta}_k\|_1$, where the predicted probabilites $p_{ik} = 1/(1 + \exp(-\boldsymbol{\beta}_k' \vec{\boldsymbol{x}}_i))$. The importance of feature $\mathbf{x}_j$ can be measured as $\sum_{k=1}^{c} |b_{kj}|$.

- **Multinomial :** One can directly solve the multinomial Logistic-Lasso Regression. Let $\boldsymbol{b}^{c \times m}$ be the solution of $\min_{\boldsymbol{\beta} \in \mathbb{R}^{c \times m}} \sum_{i=1}^{n} \sum_{k=1}^{c} [\mathbf{y}_{ik} \log p_{ik}] + \lambda \|\boldsymbol{\beta}\|_1$ where $p_{ik} = \exp(\boldsymbol{\beta}_{ko} \vec{\boldsymbol{x}}_i) / \sum_{r=1}^{c} \exp(\boldsymbol{\beta}_{ro} \vec{\boldsymbol{x}}_i)$ are the estimated softmax probabilities. Only $(c-1) \times m$ coefficients of $\boldsymbol{b}$ are identifiable in this case. For example if we consider the c-th target class as baseline, the importance of feature $\mathbf{x}_j$ can be measured as $\sum_{k=1}^{c-1} |b_{kj} - b_{cj}|$.

The regularization parameter λ is often determined using cross-validation. L_1-penalty has the effect of forcing some of the coefficient estimates to be exactly equal to zero when the tuning parameter λ is sufficiently large, thus generating a sparse solution, that leads to feature selection.

3.2 Discriminative Least Squares Regression (Discriminative LSR)

LSR is often applied to data when the target variable is continuous. To apply it in classification, We can use ϵ-dragging. With a slack variable $\epsilon_{ik} \geq 0$ we drag the output $\{0, 1\}$ to $\{-\epsilon_{ik}, 1 + \epsilon_{ik}\}$, to enlarge the distance between data points from different classes. Denote $\mathbf{y}_\epsilon = \mathbf{y} + \boldsymbol{B} \odot \boldsymbol{M}$[1], be the $\mathbf{y}$ after dragging. We solve [2]

	class	y	y after ε-dragging	constraint
$\mathbf{x}_1$	1	[1, 0, 0]	$[1+\varepsilon_{11}, -\varepsilon_{12}, -\varepsilon_{13}]$	$\varepsilon_{11}, \varepsilon_{12}, \varepsilon_{13} \geq 0$
$\mathbf{x}_2$	1	[1, 0, 0]	$[1+\varepsilon_{21}, -\varepsilon_{22}, -\varepsilon_{23}]$	$\varepsilon_{21}, \varepsilon_{22}, \varepsilon_{23} \geq 0$
$\mathbf{x}_3$	2	[0, 1, 0]	$[-\varepsilon_{31}, 1+\varepsilon_{32}, -\varepsilon_{33}]$	$\varepsilon_{31}, \varepsilon_{32}, \varepsilon_{33} \geq 0$
$\mathbf{x}_4$	2	[0, 1, 0]	$[-\varepsilon_{41}, 1+\varepsilon_{42}, -\varepsilon_{43}]$	$\varepsilon_{41}, \varepsilon_{42}, \varepsilon_{43} \geq 0$
$\mathbf{x}_5$	3	[0, 0, 1]	$[-\varepsilon_{51}, -\varepsilon_{52}, 1+\varepsilon_{53}]$	$\varepsilon_{51}, \varepsilon_{52}, \varepsilon_{53} \geq 0$
$\mathbf{x}_6$	3	[0, 0, 1]	$[-\varepsilon_{61}, -\varepsilon_{62}, 1+\varepsilon_{63}]$	$\varepsilon_{61}, \varepsilon_{62}, \varepsilon_{63} \geq 0$

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^{c \times m}, \boldsymbol{t} \in \mathbb{R}^c, \boldsymbol{M}} \|\mathbf{y}_\epsilon - (\mathbf{x}\boldsymbol{\beta}' + \mathbf{1}_n \boldsymbol{t}')\|_F^2 + \lambda \|\boldsymbol{\beta}'\|_F^2$$

$$\text{s.t. } \boldsymbol{M}^{n \times c} = ((\epsilon_{ik})) \geq 0$$

$$\boldsymbol{B}^{n \times c} = ((\boldsymbol{B}_{ik})); \boldsymbol{B}_{ik} = \begin{cases} +1 & \mathbf{y}_{ik} = 1 \\ -1 & \mathbf{y}_{ik} = 0 \end{cases}$$

The importance of feature $\mathbf{x}_j$ is measured as $\sum_{k=1}^{c} b_{kj}^2$.

Sometimes $\|\cdot\|_F^2$[2] is replaced by $\|\cdot\|_{2,1}$-norm[3] for better performance, though no closed form solution is available in the later case. [$\mathbf{y}_\epsilon - (\mathbf{x}\boldsymbol{\beta}' + \mathbf{1}_n \boldsymbol{t}')$ is the collection of residuals, where each row corresponds to a observation and each column corresponds to a target class. Substituting $\|\cdot\|_F^2$ with $\|\cdot\|_{2,1}$ implies we are using 'absolute residual' instead of 'squared residual' over the observations. So the method becomes robust to the outliers. On the other hand $\|\boldsymbol{\beta}'\|_F^2 = \sum_j \sum_k \beta_{kj}^2$ is replaced by $\|\boldsymbol{\beta}'\|_{2,1} = \sum_j \|\boldsymbol{\beta}_{oj}\|_2$ which induces some sparsity over the feature columns.]

- **pros :** Unlike c subproblems in OvR setup, here we have one model to train. So it is much faster, specially when large number of classes are there.

3.3 Support Vector Machines (SVM)

SVM is inherently a binary classifier. The key idea behind SVM is to select the hyperplane in feature space that maximizes the distance between the hyperplane and the nearest data point of each class, making this method robust and less sensitive to outliers. Let $y_i^{\pm} = \pm 1$ when the target for i-th observation is $True/False$ respectively. Then SVM is defined as the solution of $\boldsymbol{b}_k = \arg\min_{\boldsymbol{\beta}} \sum_{i=1}^{n} [1 - y_i^{\pm}(\boldsymbol{\beta}' \vec{\boldsymbol{x}}_i)]_+ + \lambda \|\boldsymbol{\beta}\|_2$. For multiclass classification, SVM can be adapted using OvR scheme.

While standard SVM shrinks the model coefficients towards zero using L_2 regularization, for variable selection it can be replaced by L_1 regularization [3] to generate sparse solution. Instead of hinge loss, squared-hinge loss is used for easier optimization. That is, for $\mathbf{y}_{ik}^{\pm} = \pm 1$ when target class for i-th observation is k/not k respectively, we solve

$$\min_{\boldsymbol{\beta}_k} \sum_{i=1}^{n} ([1 - \mathbf{y}_{ik}^{\pm}(\boldsymbol{\beta}_k' \vec{\boldsymbol{x}}_i)]_+)^2 + \lambda_k \|\boldsymbol{\beta}_k\|_1; k = 1, 2, ..., c$$

[1] Hadamard product: $(\boldsymbol{A} \odot \boldsymbol{B})_{ij} = (\boldsymbol{A})_{ij} (\boldsymbol{B})_{ij}$

[2] Frobenius norm: $\|\boldsymbol{A}\|_F = \sqrt{\sum_i \sum_j (\boldsymbol{A})_{ij}^2}$

[3] $\mathbf{L}_{2,1}$ norm : $\|\boldsymbol{A}\|_{2,1} = \sum_j \sqrt{\sum_i (\boldsymbol{A})_{ij}^2}$

The importance of feature $\mathbf{x}_j$ can be measured as $\sum_{k=1}^{c} |b_{kj}|$. Also $|b_{kj}|$ can be considered as class specific importance of $\mathbf{x}_j$.

Note: SVM can handle non-linear decision boundaries via kernel-trick. But, to access coefficient based feature importances, no non-linear kernel can be used.

3.4 Sure Independence Screening (SIS)

The idea of Independence Screening relies on the fact that - an unimportant feature should take quite similar values irrespective of the target class. So the conditional distributions $\mathbf{X}_j|(\mathbf{Y}^{cat} = k)\ \forall k = 1, 2, ..., c$ are identically distributed as the marginal distribution of $\mathbf{X}_j$. Here $\mathbf{Y}^{cat}$ is the categorical response with c categories.

Setup- all features continuous [4] : $\mathbf{E}_{\mathbf{X}_j}[Var_{\mathbf{Y}^{cat}}(F(\mathbf{X}_j|\mathbf{Y}^{cat}))] \geq 0\ \forall j = 1, 2, ..., m$. The more it deviates from 0, the higher the importance of j-th feature. The importance is estimated as

$$\widehat{MV}_j = \frac{1}{n}\sum_{k=1}^{c}\sum_{i=1}^{n} \hat{p}_k[\hat{F}_{jk}(\mathbf{x}_{ij}) - \hat{F}_j(\mathbf{x}_{ij})]^2$$

where $\hat{p}_k = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}\{\mathbf{y}_i^{cat} = k\}$, $\hat{F}_j(x) = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}\{\mathbf{x}_{ij} \leq x\}$, $\hat{F}_{jk}(x) = \sum_{i=1}^{n} \mathbf{1}\{\mathbf{x}_{ij} \leq x, \mathbf{y}_i^{cat} = k\}/\sum_{i=1}^{n} \mathbf{1}\{\mathbf{y}_i^{cat} = k\}$ are respectively empirical PMF for $\mathbf{Y}^{cat}$, CDF for $\mathbf{X}_j$ and conditional CDF for $\mathbf{X}_j|(\mathbf{Y}^{cat} = k)$. The importances $\widehat{MV}_j, j = 1, 2, ..., m$ are ordered and the top-ranked features are selected.

- **pros :** This method does not require any model assumption on $\mathbf{Y}|\mathbf{X}$.
- **cons :** It only utilizes marginal relationship of $(\mathbf{Y}, \mathbf{X}_j)$ and any kind of interactions between the features are overlooked.

4 Threshold for Selection/Rejection

Once we have obtained the importance of individual features, our goal is to find a suitable cutoff value for selection/rejection. Even if some methods have natural cutoff (e.g. 0 for LASSO), sometimes they just end up including many features with negligible importance. So here we search for some data-driven cutoff values.

4.1 Cumulative Score based Cutoff

(i) Order the features based on their importances, in decreasing manner, say the importances are $f_{(1)} \geq f_{(2)} \geq ... \geq f_{(m)}$; denote $\sum_{j=1}^{m} f_j = \boldsymbol{f}$

(ii) Assign cumulative scores to the ordered features such that $\mathbf{score}_{(j)} = \frac{100}{\boldsymbol{f}} \times \sum_{j'=j}^{m} f_{(j')}$; $j = 1, 2, ..., m$

(iii) Select those features above some threshold, e.g. select if $\mathbf{score}_{(j)} \geq 5\%$. Elbow Hunting method can also be applied here.

- **pros:** This cutoff method can be applied on any existing feature importance scores. No access to the original data is required (Unlike method 4.2 below).
- **cons:** The method fails when all features are unimportant (though it is a rare scenario in practice).

4.2 Permutation Importance (PIMP) Threshold

(i) Permute $\mathbf{y}$ several times, keeping $\mathbf{x}$ fixed. Say $\mathbf{y} \xrightarrow{permutation} \mathbf{y}^{(r)}; r = 1, 2, ..., T$.

(ii) Compute feature importances $[f_1^{(r)}, f_2^{(r)}, ..., f_m^{(r)}]$ for each $(\mathbf{x}, \mathbf{y}^{(r)})$. Collectively these will emulate the null distribution [5] of feature importances.

(iii) Compute observed importances $[f_1, f_2, ..., f_m]$ based on the data $(\mathbf{x}, \mathbf{y})$.

(iv) Select those features where observed importances exceeds some fixed quantile of null distribution corresponding to that feature. Say, ε_j is some chosen quantile of $\{f_j^{(r)} | r = 1, 2, ..., T\}$. Then select $\{j | f_j > \varepsilon_j\}$

- **pros :** (a) We have separate threshold accounting for scale and bias of each feature importances. (b) This method generates empirical p-values corresponding to feature importances.
- **cons :** Simulating the null distribution is computationally expensive.

5 Various performance evaluation criteria in use

Recall that, we have m features in total. A feature is 'Important' if $\mathbf{H}_{0j}$ is not true in the population, 'Unimportant' otherwise. We say a feature is 'selected' if its importance exceeds the threshold, 'Rejected' otherwise. All the features can be classified into four disjoint groups as follows

- TP (true +ve): instances that an 'Important' feature is 'Selected'.
- FP (false +ve): instances that an 'Unimportant' feature is 'Selected'.
- TN (true -ve): instances that an 'Unimportant' feature is 'Rejected'.
- FN (false -ve): instances that an 'Important' feature is 'Rejected'.

features	Selected	Rejected
Important	TP	FN
Unimportant	FP	TN

Now we describe briefly the various metrices used in this project for performance evaluation of the variable selection methods.

- $PCER$: $100 \times \#\{FP\}/m$; lower value implies better performance
- $PFER$: $\#\{FP\}$; lower value implies better performance
- FDR : $100 \times \#\{FP\}/\#\{Selected\}$; lower value implies better performance
- TPR : $100 \times \#\{TP\}/\#\{Important\}$; higher value implies better performance
- $minModel\_size\_ratio$: $\#\{$features to be selected to include all important features$\}/\#\{Important\}$; ≥ 1, 1 is the ideal case
- $selection\_proportion$: $\#\{Selected\}/m$; should be near the proportion $\#\{Important\}/m$
- $selection\_Youden\text{-}J$: $(\#\{TP\}/\#\{Important\}) - (\#\{FP\}/\#\{Unimportant\}); \in [-1, 1]$, best case is 1 and worst case is -1

For each iteration of the simulations, we compute one or more of this diagnostic measures and report their median values over the iterations along with their semi-interquartile range.

6 Simulations and Findings

6.1 Simulation-1 : [3/5 -active features, all independent]

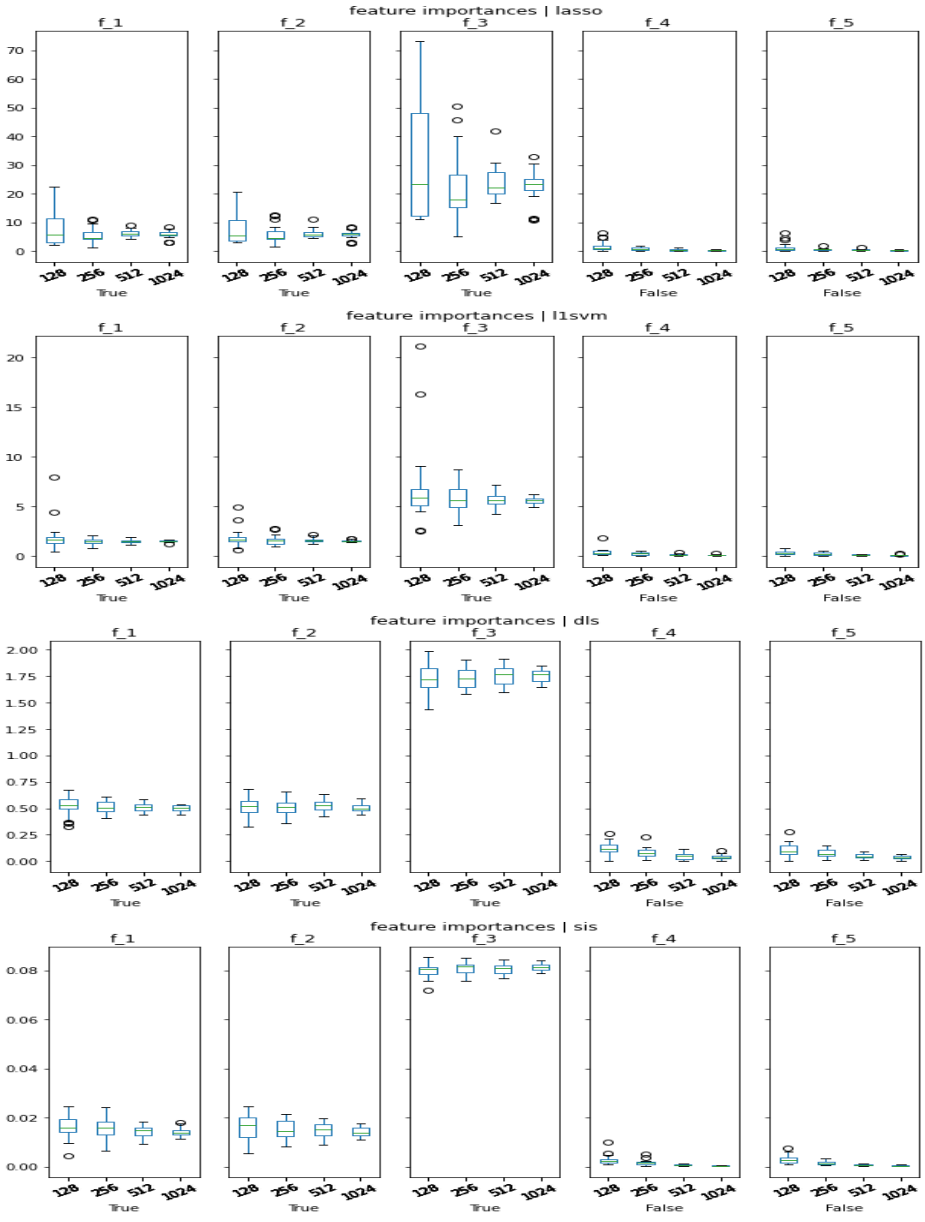
- Feature columns $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_5$ sampled iid from $\mathcal{N}(0, 1)$.
- Target $\mathbf{y}$ has 3-classes. $\vec{\boldsymbol{y}}_i | \vec{\boldsymbol{x}}_i \sim multinomial_{softmax}(\boldsymbol{\beta})$ [see notations].

  The coefficients $\boldsymbol{\beta}^{3\times 5}$ is given as $\begin{bmatrix} 5 & 7 & 10 & 100 & 0 \\ 10 & 10 & 0 & 100 & 0 \\ 7 & 5 & -10 & 100 & 0 \end{bmatrix}$
- #iterations = 24

6.1.1 Feature Importances over the iterations

Each time simulate data $(\mathbf{x}, \mathbf{y})$ and compute feature importances based on various methods. So we have #iterations = 24 importances corresponding to each base method discussed in section 3. The boxplots corresponding to feature importances are shown below in Figure 1.

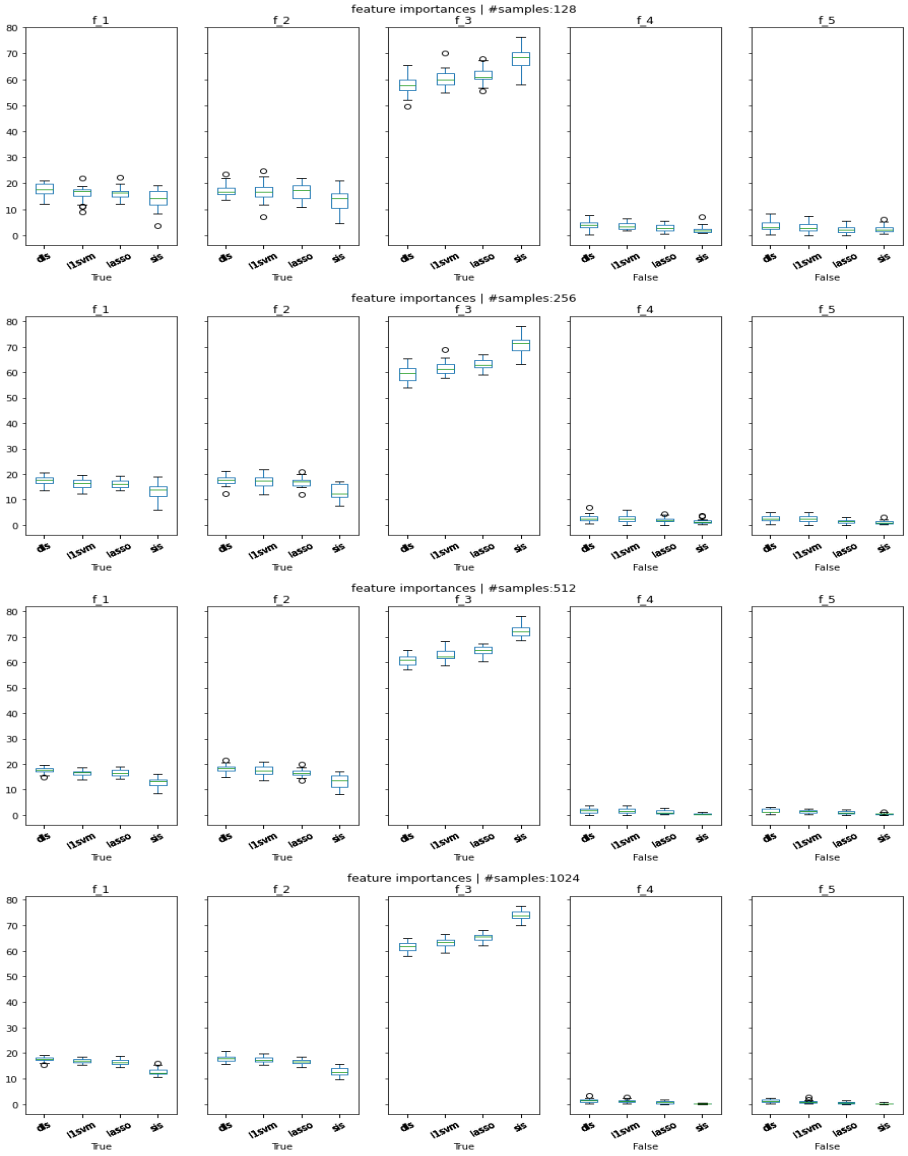
Figure 1: Raw feature importances | Simulation1



Comment: We notice that the dispersion of the feature importances decreases with increasing sample sizes (see the span of the boxplots gradually decreases for $n = 128, 256, 512, 1024$).

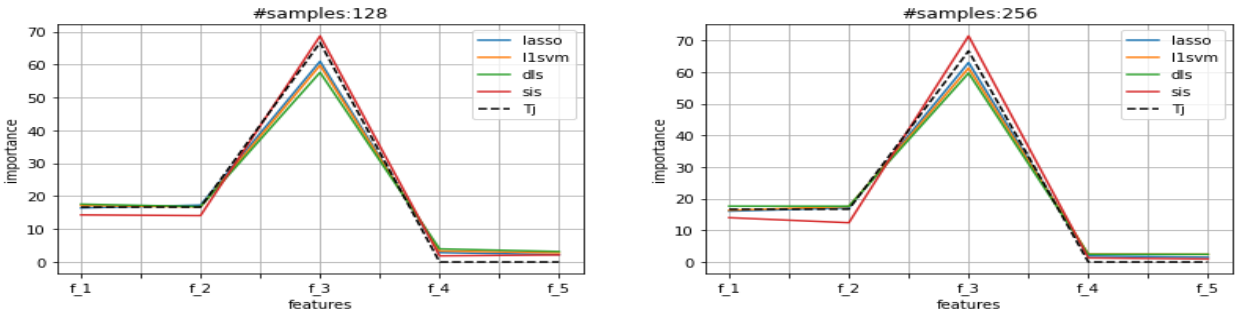
But the raw importances generated by different methods are on widely different scales. Since we are interested in relative comparison, in Figure 2 and later on we will report scaled $[f_1, f_2, ..., f_m]$ for better comparison, such that for each method we get $\sum_{j=1}^{m} f_j = 100$ on any particular iteration.

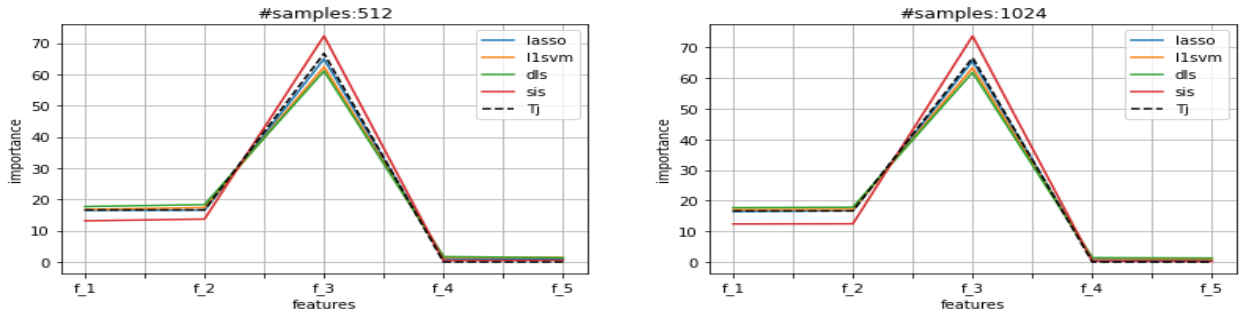
Figure 2: Scaled importances | Simulation1



Alternatively in Figure 3 we also compare the median of these boxplots with $[\mathbf{T}_1, \mathbf{T}_2, \mathbf{T}_3, \mathbf{T}_4, \mathbf{T}_5]$, after scaling $\mathbf{T}_j$'s in a similar manner that $\sum_{j=1}^{m} \mathbf{T}_j = 100$.

Figure 3: Observed vs Proposed importances (after scaling) | Simulation1



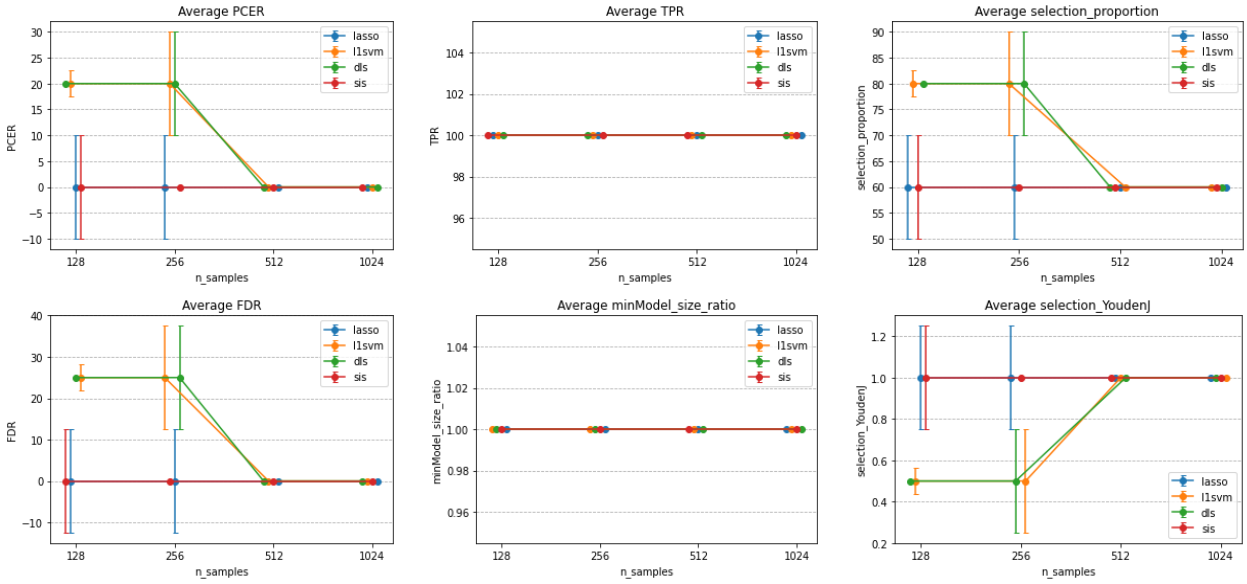


Comment: The order and ranking of feature importances are correctly identified by every method. Each time $\mathbf{x}_2$ gets the highest importance, while $\mathbf{x}_0$ and $\mathbf{x}_1$ gets nearly equal importance. $\mathbf{x}_3$ and $\mathbf{x}_4$ have nearly zero observed importances since they satisfy $\mathbf{H}_{0j} : \beta_{1j} = \beta_{2j} = ... = \beta_{cj}$.

6.1.2 Cutoff by $\geq$5% cumulative score

Using method 4.1 try to select those features where $\mathbf{score}_{(j)} \geq 5\%$, i.e. the features contributing top 95% cumulative scores are selected. The median values of various performance evaluation criteria from section 5 and their semi-interquartile ranges are reported in Figure 4 below.

Figure 4: Cumulative Score Cutoff | Simulation1

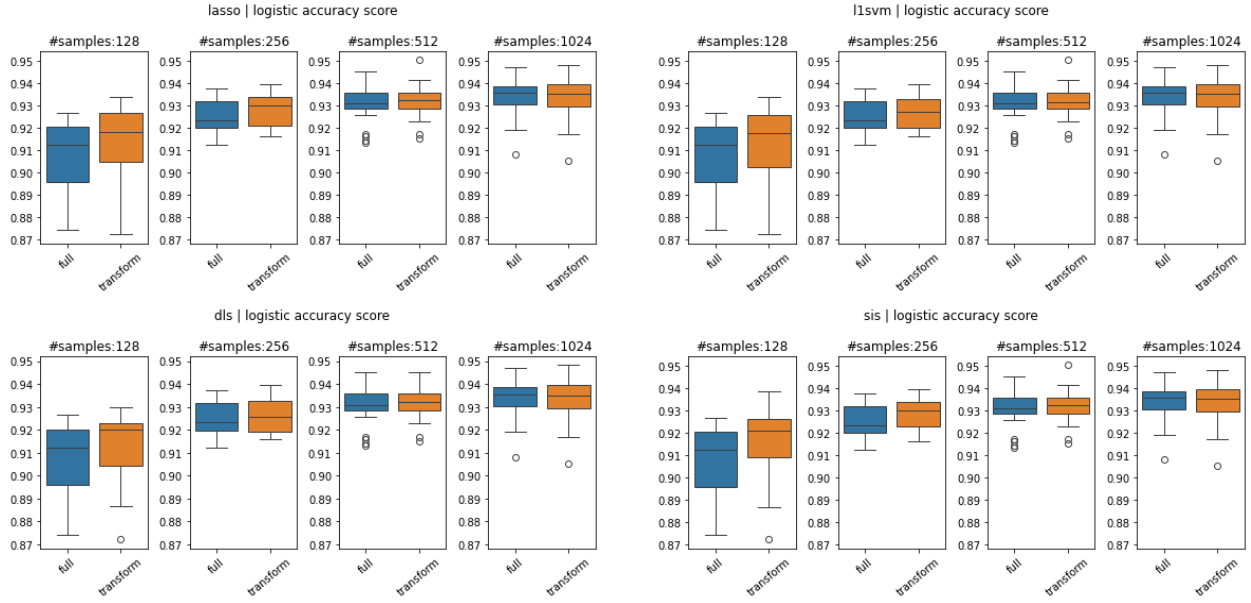


Comment: As the sample-size increases PCER & FDR $\rightarrow 0$, selection Youden-J $\rightarrow 1$, selection proportion $\rightarrow 3/5$, minModel size ratio $= 1$, **TPR** $= 1$. So each method converges to ideal case.

6.1.3 Accuracy Comparison

At each iteration, we fit a logistic regression once with the full set of features and once with the set of selected features. The prediction accuracies are reported here.

Figure 5: Accuracy | Simulation1



Comment: On average the accuracy based on the selected features are only a little bit higher than the accuracy based on all possible features.

NOTE: In the simulation above, the results from Figure 4 seems to be over optimistic. When we have have 5 features in the true model, out of which 3 are 'Important' maybe there is not enough scope of error for the variable selection methods.

From an practical point of view also, a data set with 5-6 features is not appealing for feature selection. In Figure 5 we can not see any significant improvement in terms of accuracy, after performing variable selection.

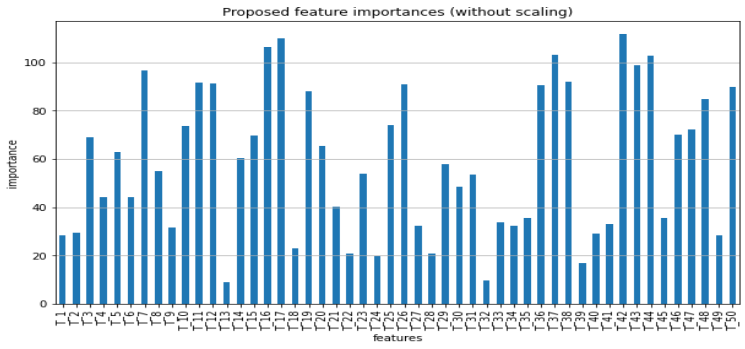
So we increase the size of the data before further simulations.

- Target **y** has 3-classes. $\vec{\boldsymbol{y}}_i|\vec{\boldsymbol{x}}_i \sim multinomial_{softmax}(\boldsymbol{\beta}_{(a,m)})$. The coefficients $\boldsymbol{\beta}_{(a,m)}$ is a matrix of shape $c \times m$ with only first (m-a) non-null columns, say

$$\begin{bmatrix} 9.051 & -4.347 & 13.982 & -10.170 & 5.039 & 4.314 & \dots & 0 & 0 \\ 1.928 & -6.068 & -3.242 & -5.041 & 11.612 & 7.765 & \dots & 0 & 0 \\ 4.537 & -11.687 & 8.17 & 0.864 & -4.136 & -3.321 & \dots & 0 & 0 \end{bmatrix}$$

  Instead of writing the whole matrix here we list out the values of $\mathbf{T}_j$ in Figure 6 for better comprehension.

Figure 6: Possible values of $\mathbf{T}_j$



For the setup $\beta_{(a,m)}$ the unscaled [$\mathbf{T}_1$,$\mathbf{T}_2$,...,$\mathbf{T}_a$] will be first a values of the barplot above, remaining $\mathbf{T}_{a+1}, \mathbf{T}_{a+2}, ..., \mathbf{T}_m$ will be 0.

- Feature columns $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_m$ are sampled iid from $\mathcal{N}(0, 1)$ [same as earlier].
- #iterations = 24 [same as earlier].

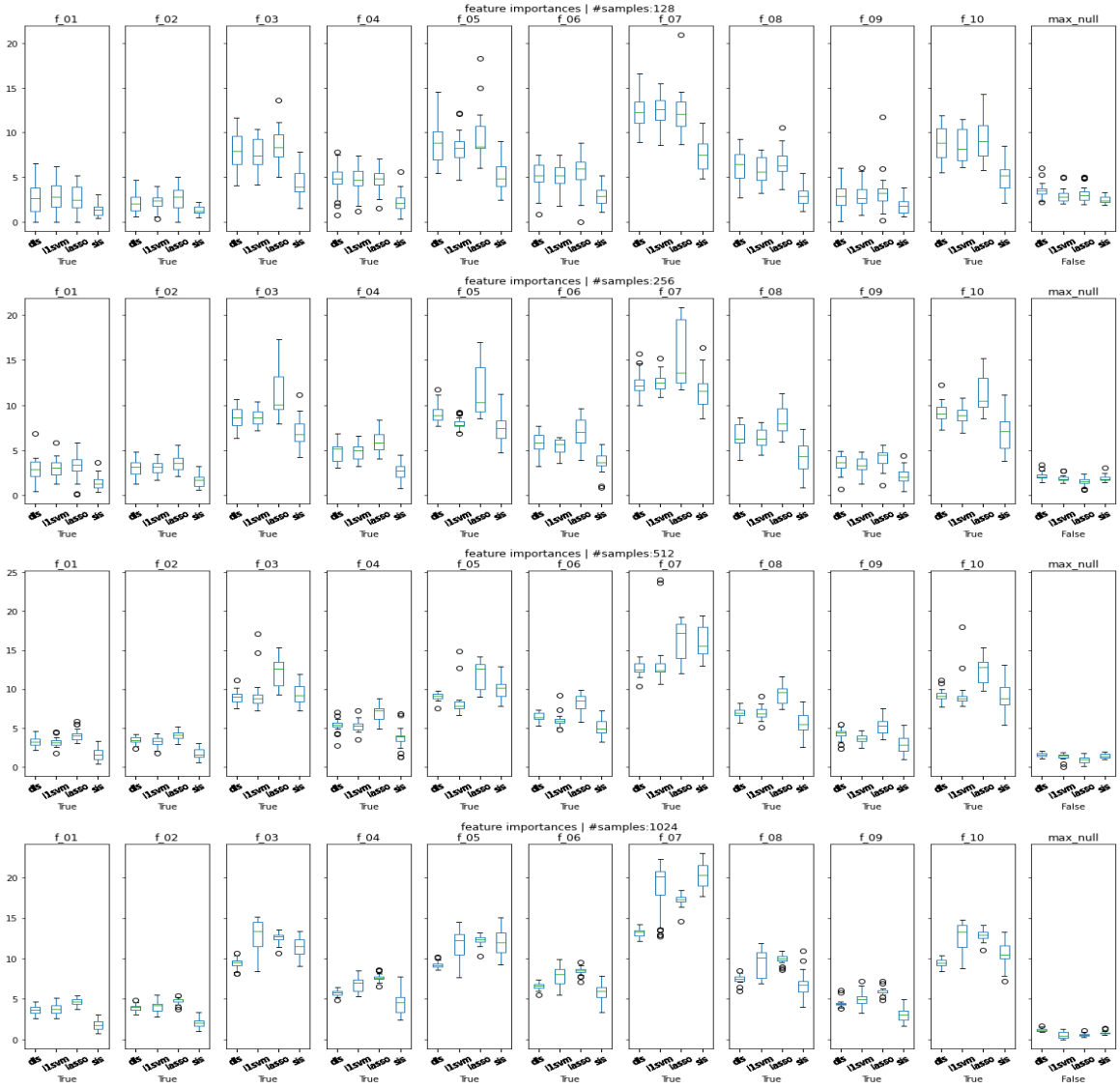
6.2 Simulation-2 : [Effect of Sample Size]

Fix $m = 100, a = 10$. At each iteration, generate data ($\mathbf{x}^{1024\times100}, \mathbf{y}^{1024\times3}$). then apply the methods based on first $n = 128, 256, 512, 1024$ observations only.

6.2.1 Feature Importances over the iterations

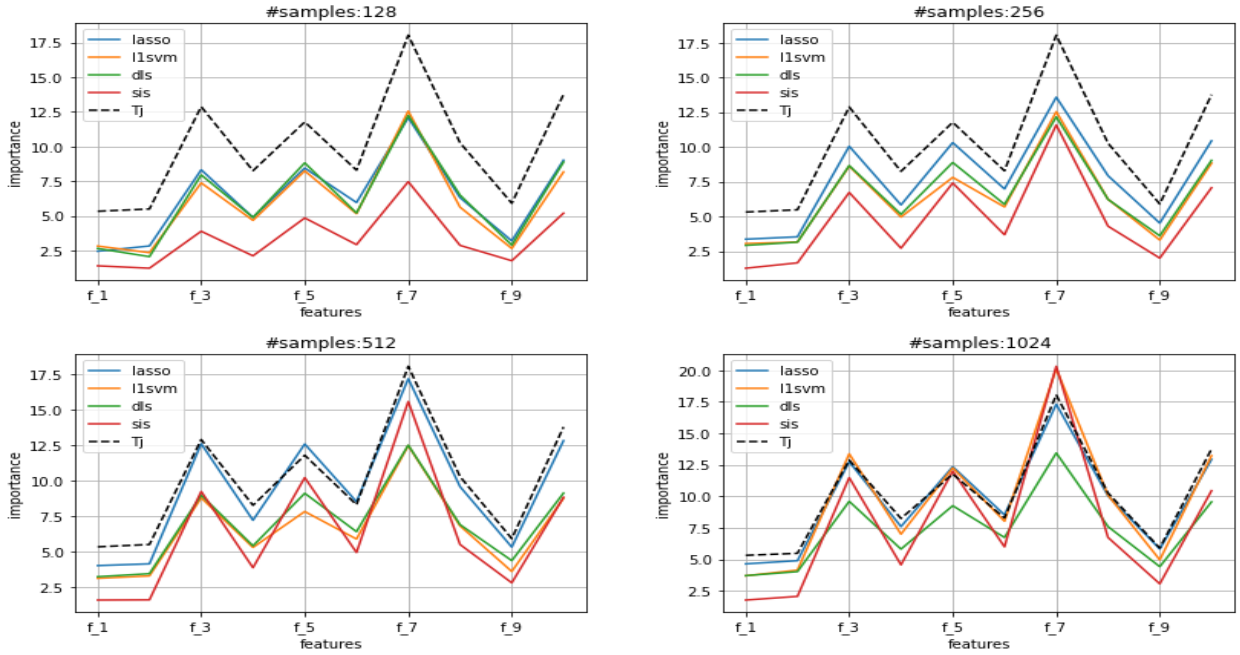
We start with the boxplots Figure 7 of feature importances. Here first $a = 10$ features are known to be active, remaining feature importances should be near zero. So we compare [$f_1, f_2, ..., f_{10}$] with the maximum observed importances of the null features $\max\{f_j | j > 10\}$.

Figure 7: Scaled feature importances | Simulation2



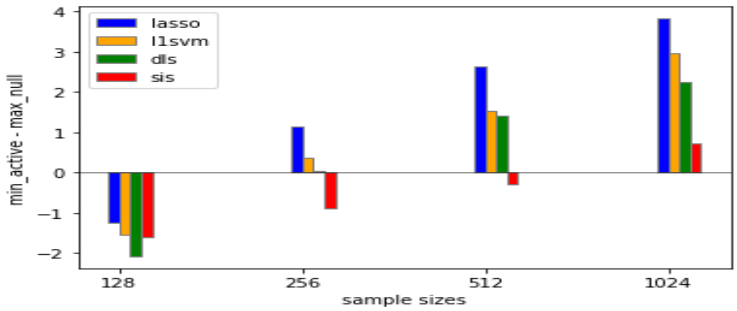
Alternatively in Figure 8 we also compare the median of these boxplots with $[\mathbf{T}_1, \mathbf{T}_2, \mathbf{T}_3, ..., \mathbf{T}_{10}]$, after scaling $\mathbf{T}_j$'s in a manner that $\sum_{j=1}^{m} \mathbf{T}_j = 100$.

Figure 8: Observed vs Proposed importances (after scaling) | Simulation2



Comment: It is desired to have the minimum importance among active features higher than the maximum importance among null features. For lower sample sizes, the observed importance of a null feature sometimes wrongly exceeds the importance of a non-null feature in Figure 7. It can also be better visualized in Figure 9 if we compare the median of the difference $\min\{f_j|j \leq 10\} - \max\{f_j|j > 10\}$ (desired to be +ve). As sample size increases, this issue tends to be resolved. Both from Figure 7,Figure 8 we also say that the relative ordering of importances (f_7 is the highest, f_1 and f_2 nearly equal etc.) remains quite stable.

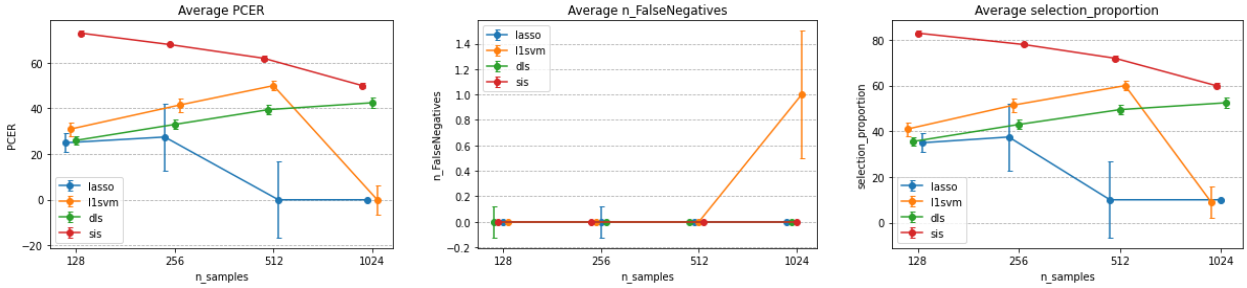
Figure 9: Active vs Null importances (after scaling) | Simulation2



6.2.2 Cutoff by ≥5% cumulative score

Using method 4.1 we select those features where $\textbf{score}_{(j)} \geq 5\%$, i.e. the features contributing top 95% cumulative scores are selected. The median values of various performance evaluation criteria from section 5 and their semi-interquartile ranges are reported in Figure 10 below.

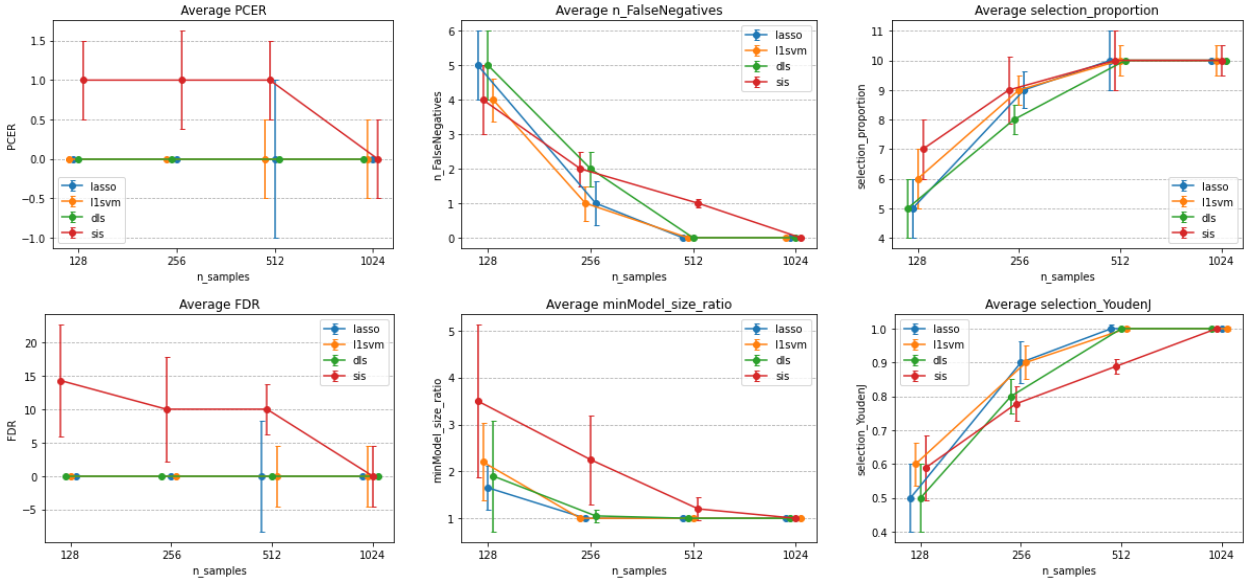
Figure 10: Cumulative Score Cutoff | Simulation2



Comment: This method selects many null features, type-I error rate is too much inflated here. The issue with this method is that, the threshold 5% is chosen arbitarily. It does not carry any information about "How much deviation from zero can be safely neglected as null-importance?". So we discard this method from further simulations.

6.2.3 Cutoff by PIMP, 5% FDR control with Benjamini–Yekutieli (BY) correction

Figure 11: PIMP Cutoff | Simulation2



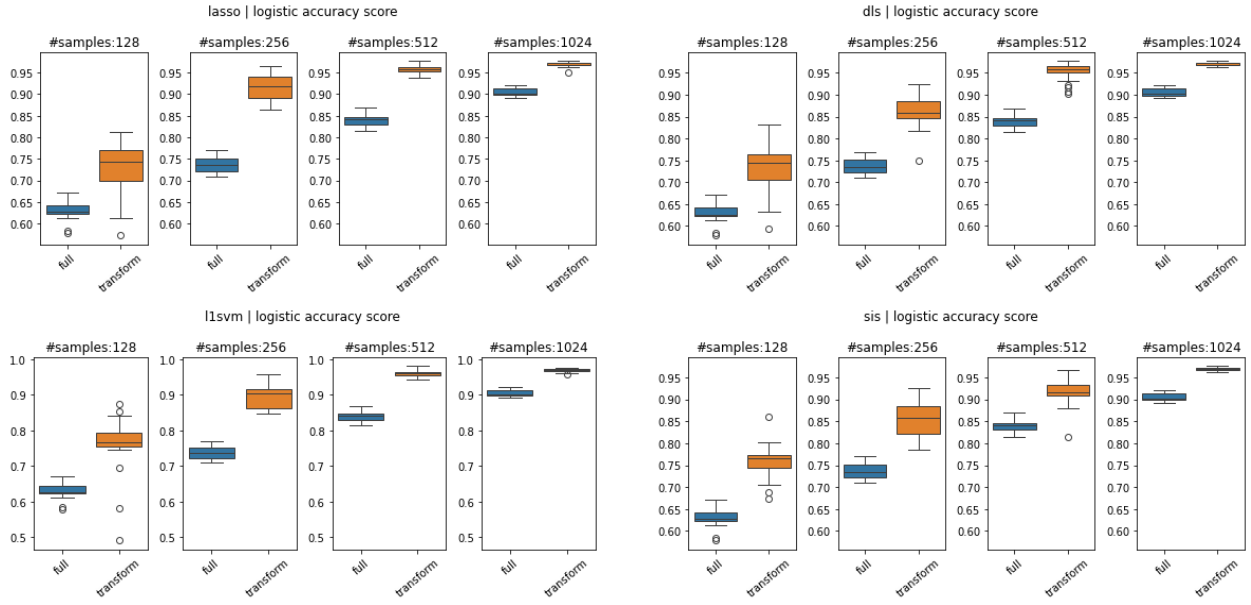
The mehod 4.2 is followed here to obtain empirical p-values corresponding to each features, based on $T = 100$ permutations. We know smaller the p-value, higher the evidence that corresponding feature is not null feature. Let us arrange the p-values in increasing order $p_{(1)} \leq p_{(2)} \leq ... \leq p_{(m)}$. It is desired to drop a few features with least importances to keep our model simple, at the same time not affecting the performance too much. The Benjamini–Yekutieli procedure [6] is applied for selection/rejection at level $\alpha = 0.05$. i.e. the feature corresponding to $p_{(j)}$ is selected iff $min\{p_{(l)} \times \frac{m}{l} \sum_{t=1}^{m} \frac{1}{t} : l = j, j+1, ..., m\} \wedge 1 \leq \alpha$. The resulting plots for various metrices are attached above in Figure 11.

Comment: We see that type-I error (false +ve) is under control here. But for smaller sample sizes there is high type-II error (false -ve). It can be justified by the fact we noticed earlier that some non-null features there get observed importances even smaller than null features. With increase in sample size, the issue tends to be resolved.

6.2.4 Accuracy Comparison

At each iteration, we fit a logistic regression once with the full set of features and once with the set of selected features. Their prediction accuracies are reported as boxplots below in .

Figure 12: Accuracy | Simulation2



Comment: Higher the number of available samples, greater the prediction accuracy. Overall the simpler model with only selected features performs better than the full model in terms of accuracy.

6.3 Simulation-3 : [Effect of #features]

Fix $n = 256, a = 10$. At each iteration,

- Generate $\mathbf{x}^{256\times 150}$,
- Based on first $m = 30, 60, 100, 150$ columns of $\mathbf{x}$, say $\mathbf{x}(m)^{256\times m}$ and the coefficients $\boldsymbol{\beta}_{(a=10,m)}$ generate $\mathbf{y}(m)^{256\times 3}$.
- Run the methods on ($\mathbf{x}$(m),$\mathbf{y}$(m)).

6.3.1 Feature Importances over the iterations

Here the first $a = 10$ features are known to be active, remaining feature importances should be near zero. So we compare $[f_1, f_2, ..., f_{10}]$ with the maximum observed importances of the null features $\max\{f_j | j > 10\}$ in Figure 13.

Also we can further be interested to know what is the contribution of active features $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_{10}$ in the total feature importance $\sum_{j=1}^{m} f_j$. So we plot scaled $[f_1, f_2, ..., f_{10}]$ along with scaled $[\mathbf{T}_1, \mathbf{T}_2, \mathbf{T}_3, ..., \mathbf{T}_{10}]$ in Figure 14 such that $\sum_{j=1}^{m} \mathbf{T}_j = 100$, $\sum_{j=1}^{m} f_j = 100$

Comment: From Figure 13,Figure 14 we see that the overall pattern (f_7 is the highest, f_1 and f_2 nearly equal etc.) looks quite stable over the change of m. In Figure 14 we increase the total number of features m, keeping the number of active features fixed $a = 10$, the contibution of [f_1,f_2,...,f_{10}] in $\sum_{j=1}^{m} f_j = 100$ decreases (based on the observation that for a fix feature blue line $m = 30$ is above the red line $m = 150$, the $m = 30, 60, 100, 150$ lines are arranged in

decreasing order). So with increasing m, the remaining portion of $\sum_{j=1}^{m} f_j = 100$ is contributed by the additional noise features included in the data.

Figure 13: Scaled feature importances | Simulation3

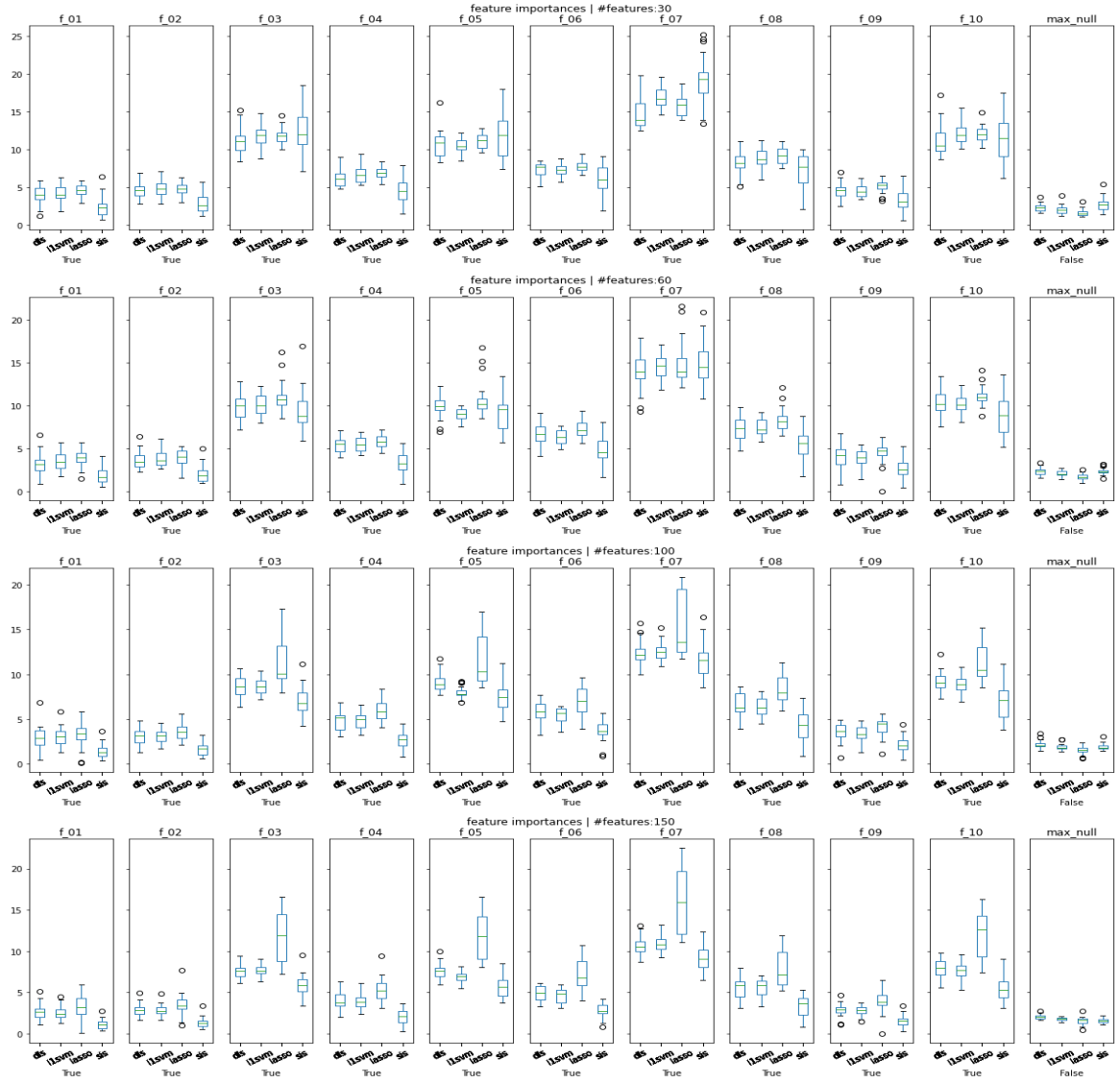
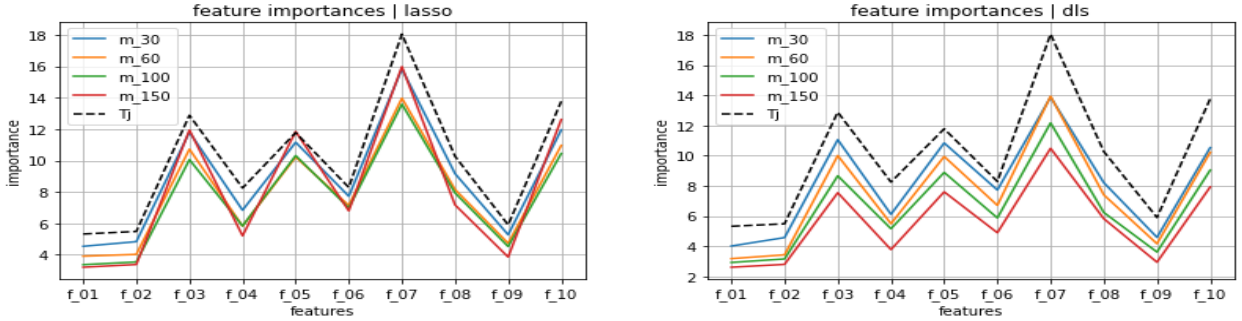
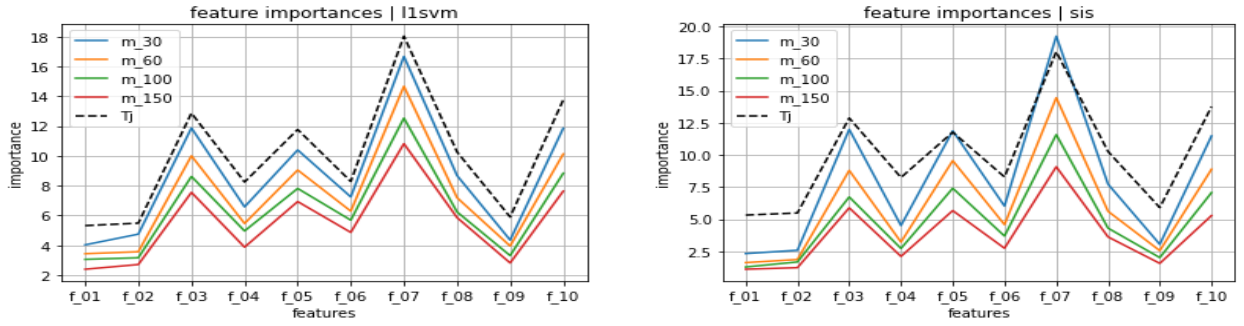


Figure 14: Contribution of active features (after scaling) | Simulation3

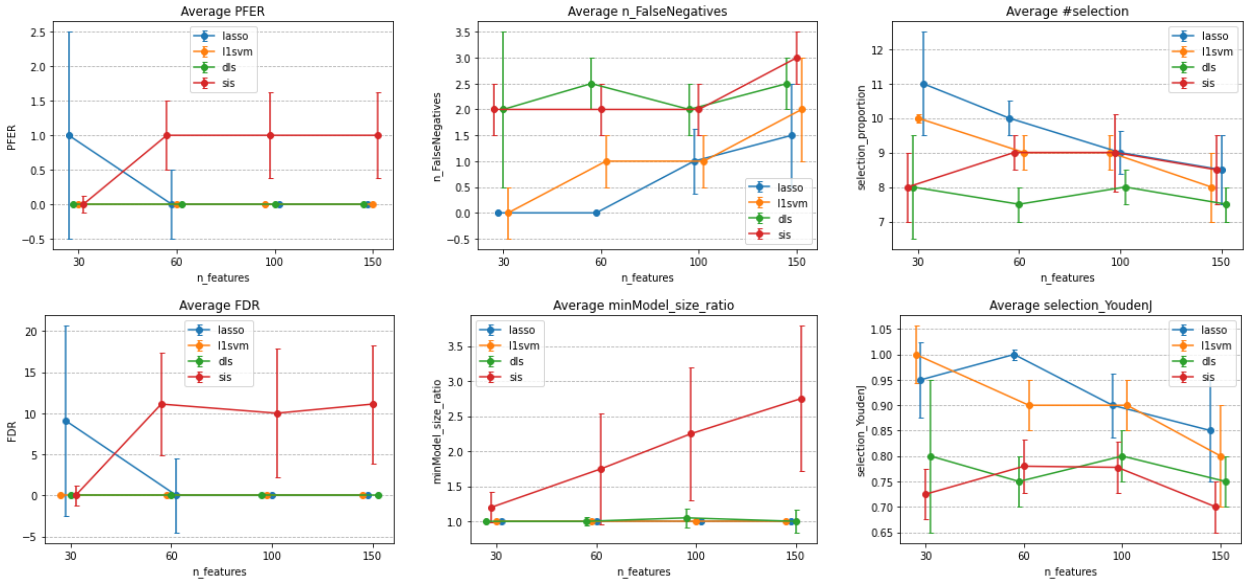




6.3.2 Cutoff by PIMP, 5% FDR control with Benjamini–Yekutieli (BY) correction

Using the mehod 4.2 empirical p-values are computed based on $T = 100$ permutations. The Benjamini–Yekutieli procedure is applied over it for selection/rejection at level $\alpha = 0.05$. The resulting plots of variour evaluation metrices are shown below in Figure 15

Figure 15: PIMP Cutoff | Simulation3

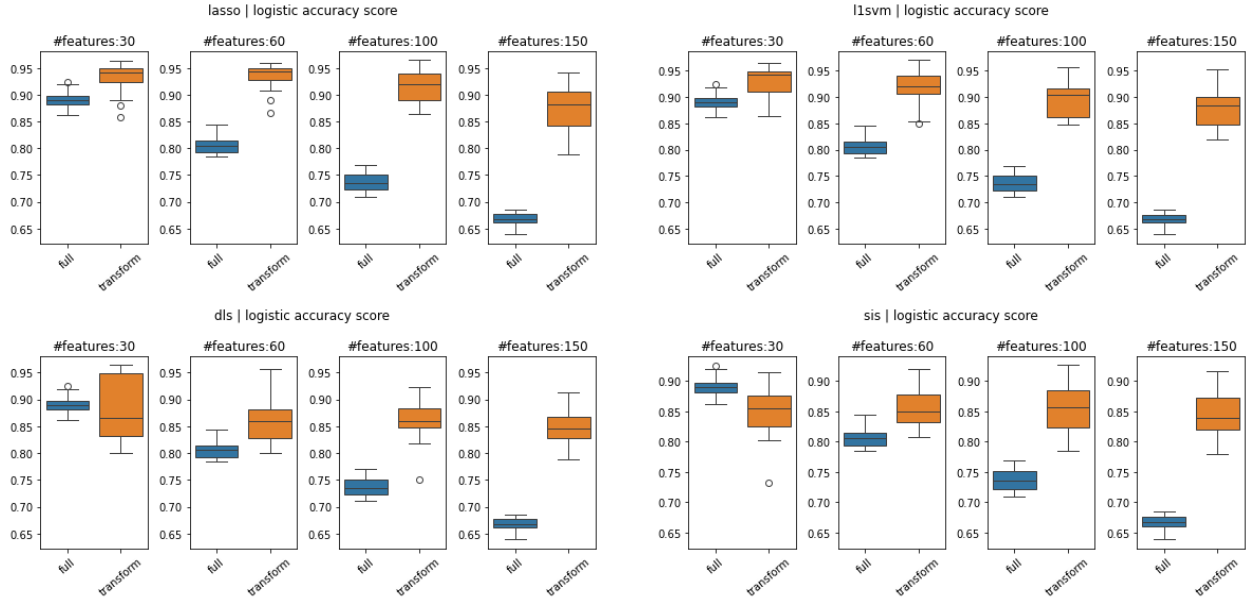


Comment: Here we observe the number of selection remains more or less constant (it is desired, since the number of active features is fixed, even though the number of total features inceares). With increasing m, there is a tendency of increase in type-II error (false -ve).

6.3.3 Accuracy Comparison

At each iteration, we fit a logistic regression once with the full set of features and once with the set of selected features. The prediction accuracies are reported below in Figure 16.

Figure 16: Accuracy | Simulation3



Comment: Overall the simpler model with only selected features performs better than the full model in terms of accuracy. (Except for the methods dls ans sis at $m = 30$. Here the number of null features is not too much. "The desired increase in accuracy after removing those few null features" is possibly overshadowed by "the desired decerease in accuracy by false -ve" here.) The difference in performance between full model and reduced model becomes more and more noticable when higher number of null features are present there in full model.

6.4 Simulation-4 : [Effect of #active features]

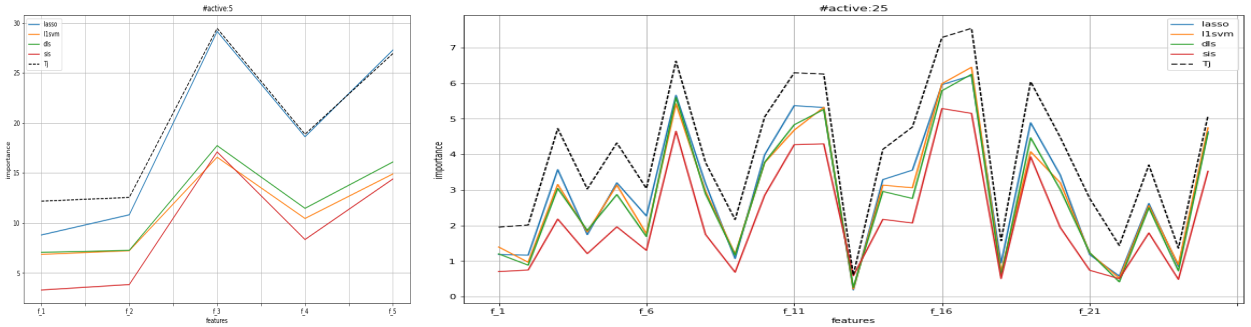
Fix $n = 256, m = 100$. At each iteration,

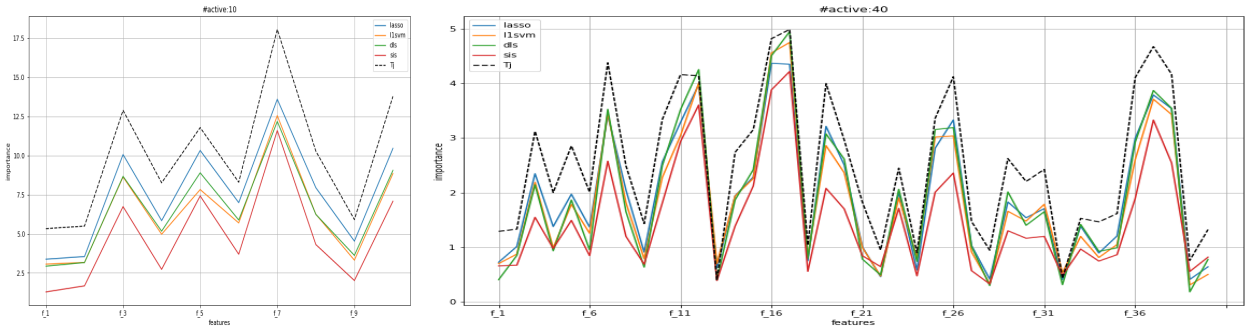
- Generate $\mathbf{x}^{256\times 150}$,
- Based on $\mathbf{x}$ and the coefficients $\boldsymbol{\beta}_{(a,m=100)}$ for $a = 5, 10, 25, 40$ generate $\mathbf{y}(a)^{256\times 3}$.
- Run the methods on ($\mathbf{x}$,$\mathbf{y}$(a)).

Here we start with scaled feature importances such that $\sum_{j=1}^{m} f_j = 100$. we compare [f_1,f_2,...,f_a] with [$\mathbf{T}_1$,$\mathbf{T}_2$,...,$\mathbf{T}_a$] in Figure 17.

Comment: The pattern of f_j and $\mathbf{T}_j$ matches quite well.

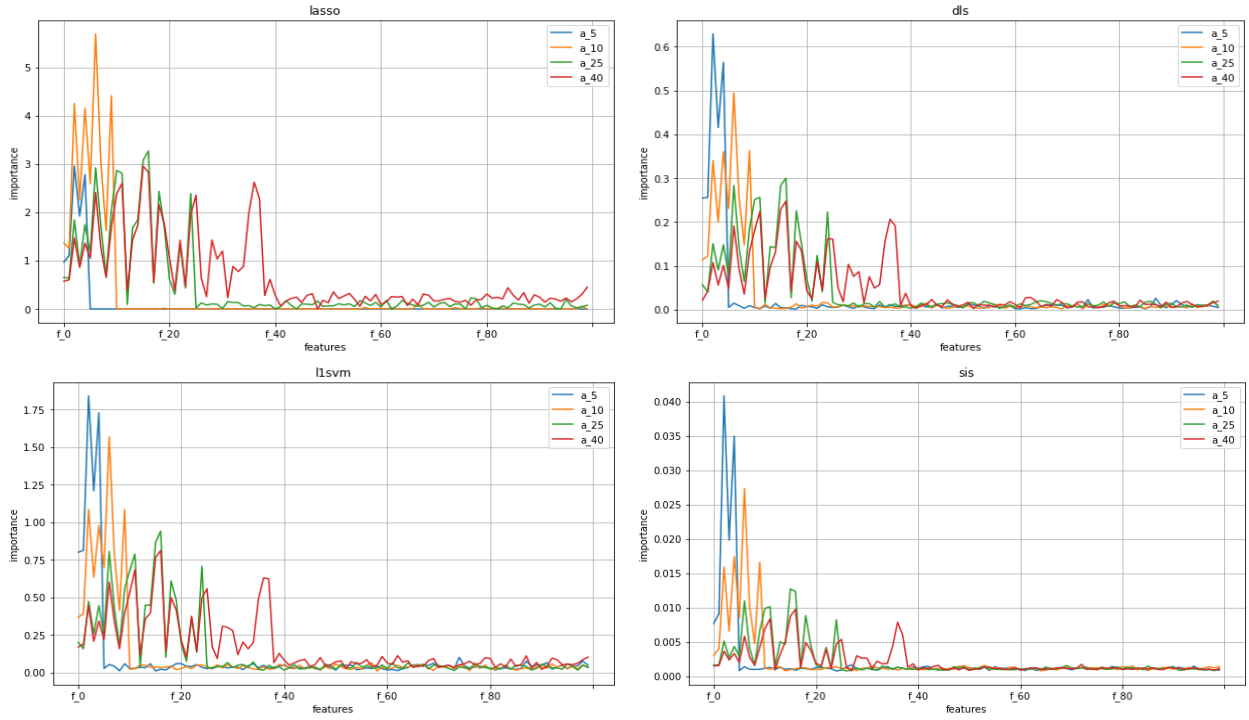
Figure 17: f_j vs T_j (after scaling) | Simulation4





But consider the cases, say $\boldsymbol{\beta}_{(a=5,m=100)}$ and $\boldsymbol{\beta}_{(a=10,m=100)}$. The first 5-columns are common in both the matrices, so can expect the values of $[f_1,f_2,f_3,f_4,f_5]$ to be same from both the cases before any scaling. Similarly for the other cases $a = 25, 40$ also. But in the simulation we see from Figure 18 that the values of the raw importances tend to decrease as we increase $a = 5, 10, 25, 40$.

Figure 18: Feature importances (without scaling) | Simulation4



The reason behind such behaviour is not yet clear to us.

7 Future Work

For time constrain, we could have only explored the setup where **x** is non-random, all features are continuous, features are independent. Some immediate possible extensions could be as follows

- **Features are correlated:** We can try to replace 'LASSO penalty' with 'elastic net penalty' [7] in various methods when the features are highly correlated. How to handle the underlying optimization problems, that are yet to be explored.

- **Some categorical features are there:** In all the methods discussed above, a categorical feature can be included in the one-hot-encoded format. Then, for a feature with u categories, we will have $u-1$ identifiable columns in our data, there is no specific way to combine those $u-1$ importances in a single value. Instead some 'target encoding' methods can be tried out. e.g. Let $\mathcal{D} = \{\mathbf{x}_{ij}\}_{i=1}^{n}$for some categorical feature $\mathbf{x}_j$, $\boldsymbol{\sigma}$ some random permutation of $\{1, 2, ..., n\}$ and $\{\mathbf{y}_i^{cat}\}_{i=1}^{n}$ are observed class labels. $\mathbf{x}_{ij}$ is encoded as [8]

$$\hat{\mathbb{E}}[\mathbf{Y}^{cat}|\mathbf{x}_{ij}] = \frac{\sum_{x\in\mathcal{D}_i} \mathbf{1}_{\{x=\mathbf{x}_{ij}\}}.\mathbf{y}_i^{cat} + a.p}{\sum_{x\in\mathcal{D}_i} \mathbf{1}_{\{x=\mathbf{x}_{ij}\}} + a}$$

  where $a > 0$ tuning parameter, $p = \sum_i \mathbf{y}_i^{cat}/n$ and $\mathcal{D}_i = \{i'|\boldsymbol{\sigma}(i') < \boldsymbol{\sigma}(i)\}$.

- **Random effect models: X** can be random variable as well. How to define feature importances in such setup, that are yet to be explored.

Also, since we have already obtained methods to access importances of individual features, we can apply modern methods, like knockoff [9], upon them to determine cutoff for selection/rejection with false discovery rate control, bypassing difficulties of multiple testing.

References

[1] M. Y. Park and T. Hastie, "L 1-regularization path algorithm for generalized linear models," *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 69, no. 4, pp. 659–677, 2007.

[2] S. Xiang, F. Nie, G. Meng, C. Pan, and C. Zhang, "Discriminative least squares regression for multiclass classification and feature selection," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 11, pp. 1738–1754, 2012.

[3] J. Zhu, S. Rosset, R. Tibshirani, and T. Hastie, "1-norm support vector machines," *Advances in Neural Information Processing Systems*, vol. 16, 2003.

[4] H. Cui, R. Li, and W. Zhong, "Model-free feature screening for ultrahigh dimensional discriminant analysis," *Journal of the American Statistical Association*, vol. 110, no. 510, pp. 630–641, 2015.

[5] A. Altmann, L. Toloşi, O. Sander, and T. Lengauer, "Permutation importance: a corrected feature importance measure," *Bioinformatics*, vol. 26, no. 10, pp. 1340–1347, 2010.

[6] Y. Benjamini and D. Yekutieli, "The control of the false discovery rate in multiple testing under dependency," *Annals of Statistics*, pp. 1165–1188, 2001.

[7] Z. Y. Algamal and M. H. Lee, "Applying penalized binary logistic regression with correlation based elastic net for variables selection," *Journal of Modern Applied Statistical Methods*, vol. 14, no. 1, p. 15, 2015.

[8] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "Catboost: unbiased boosting with categorical features," *Advances in neural information processing systems*, vol. 31, 2018.

[9] E. Candes, Y. Fan, L. Janson, and J. Lv, "Panning for gold:'model-x'knockoffs for high dimensional controlled variable selection," *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 80, no. 3, pp. 551–577, 2018.