Mini-Project (ML for Time Series) - MVA 2021/2022

Eya Ghamgui eya.ghamgui@telecom-paris.fr
Siwar Mhadhbi siwar.mhadhbi@telecom-paris.fr

March 23, 2022

1 Introduction and contributions

Feature selection, as a data preprocessing strategy, has been shown to be effective and efficient in preparing data for various machine learning problems. The goals of feature selection include improving data mining performance and preparing clean and understandable data. In this project, we revisit advances in feature selection research by comparing selected algorithms. To highlight the differences and similarities of these algorithms, we choose to work on three different groups as categorized by [1]: sparse-learning-based, similarity-based, and information-theory-based methods. Throughout our study, we consider supervised and unsupervised learning scenarios for classification and regression tasks.

To conduct this project, each of us worked on a different task (classification or regression), while regularly discussing and sharing ideas with each other to ensure a complete contribution and a coherent work. As the article is a survey, it solely compares between the different feature selection algorithms without providing an experimental setup. Hence, we chose to draw our own experimental pipeline: we carried out our study on two datasets of our choice, on which we performed data enhancement by removing missing values and outliers by means of median filtering. We re-implemented six feature selection algorithms, with little use of the provided repository of about 30%. Added to that, we performed parameters tuning to study their impact. We chose three models (Random Forest and Gradient Boosting with grid search, and Neural Network), and a total of seven metrics for evaluation.

2 Feature Selection Methods

2.1 Sparsity-based methods

Feature selection is an optimization problem by nature. Methods based on sparse learning aim at minimizing the fitting errors with some sparse regularization term. The sparse regularizer forces many feature coefficients to be small, or exactly zero, and the corresponding features can then simply be eliminated. For the regularization term, several norms can be used. Regularization by the $l_{2,0}$-norm and $l_{2,\ 0<q<1}$-norm, makes the optimization problem NP-hard and discrete. Therefore, regularization by the $l_{2,1}$-norm is preferred and widely used. Moreover, it is more robust to noise. In our project, we have selected two methods using this regularization term.

- ***Robust and Efficient Feature Selection (REFS)***
  The aim of this method is to solve the following minimization problem:

$$\min_{W} \|XW - Y\|_{2,1} + \alpha \|W\|_{2,1}$$

  Since the $l_{2,1}$-norm is more efficient, the authors of [2], used this norm for the loss term plus a $l_{2,1}$-norm regularizer to achieve the sparsity of the group features. The solution of this problem is based on the calculation of the derivative of the Lagrangian function. The iterative algorithm 1 is proposed by the authors of [2] to obtain a global optimal solution to the previous problem.

- ***Unsupervised Discriminative Feature Selection (UDFS)***
  The aim of this algorithm [3] is to select the most discriminative features for data representation. It is based on the $L_{2,1}$ regularization approach to minimize the objective function. The optimization problem is therefore written as:

$$\min_{W^TW=I_d} \sum_{i=1}^{n} \{Tr(G_{(i)}^T H_{k+1} G_{(i)} - DS_i)\} + \gamma ||W||_{2,1}$$

$$\text{where: } \begin{cases} G_{(i)} = S_i^T X^T W \\ H_{k+1} = I - \frac{1}{k+1} 1_{k+1} 1_{k+1}^T \in \mathbb{R}^{k+1 \times k+1} \\ (S_i)_{pq} = 1 \quad \text{if } x_p \text{ is in the nearest neighbours of } x_q, \text{ 0 otherwise} \\ DS_i = Tr[W^T X S_i \tilde{X}_i^T (\tilde{X}_i \tilde{X}_i^T + \lambda I)^{-1} \tilde{X}_i S_i^T X^T X] \\ \tilde{X}_i = X_i H_{k+1}, \text{ where: } X_i = [x_i, x_{i_1}, \cdots, x_{i_k}] \quad \text{k are the k nearest neighbors} \end{cases}$$

  The solution of this problem is based on the algorithm 2. The latter provides feature weights by maximizing the local inter-class divergence and minimizing the local intra-class divergence simultaneously while minimizing the $l_{2,1}$-norm of the coefficient matrix. A high local discrimination score indicates that the instance can be well discriminated.

2.2 Similarity-based methods

Similarity based algorithms determine feature importance by looking at how well features preserve data similarity.

- ***Laplacian Score (LS)***
  Laplacian Score [4] is a widely used feature ranking algorithm used in unsupervised learning. The scores are computed based on three steps. First, we constructed the similarity matrix "S". Each element is the exponential of the distance between the corresponding two features. In a second step, we computed the Laplacian matrix "L" as the difference between the degree matrix "D" and the similarity matrix "S". Finally, for each feature, we compute a score using this formula:

$$laplacian - score(X_i) = \frac{\tilde{X}_i^T L \tilde{X}_i}{\tilde{X}_i^T D \tilde{X}_i}, \text{ where: } \tilde{X}_i = X_i - \frac{X_i^T D1}{1^T D1} 1$$

- ***Fisher Score (FS)***
  Fisher Score is a supervised feature selection algorithm. It selects features such that the feature values of samples within the same class are similar while the feature values of samples from different classes are dissimilar. The Fisher Score of feature X_i is evaluated as follows:

$$fisher - score(X_i) = \frac{\sum_{j=1}^{c} n_j (\mu_{i,j} - \mu_i)^2}{\sum_{j=1}^{c} n_j \sigma_{i,j}^2} \quad \text{where: } \begin{cases} c : \text{number of classes} \\ n_j : \text{number of data points in class } j \\ \mu_i : \text{mean value of feature } X_i \\ \mu_{i,j} : \text{mean value of feature } X_i \text{ for samples in class } j \\ \sigma_{i,j}^2 : \text{variance value of feature } X_i \text{ for samples in class } j \end{cases}$$

  In our work, we used a different expression of the Fisher Score that uses the previously defined Laplacian Score as proposed by [4]:

$$fisher - score(X_i) = \frac{1}{laplacian - score(X_i)} - 1$$

2.3 Information-based methods

- ***Mutual Information Maximization or Information Gain (MIM)***
  The Mutual Information measures the importance of a feature X_i by its correlation with the class labels. It assumes that when a feature has a strong correlation with the target, it can help achieve good classification performance. The MI score for feature X_i is defined as follows:

  $$J_{MIM}(X_i) = I(X_i; Y) \quad \text{where: } \begin{cases} I : \text{information gain - measure of shared information} \\ Y : \text{class labels} \end{cases}$$

- ***Mutual Information Feature Selection (MIFS)***
  Mutual Information Feature Selection considers both the feature relevance and feature redundancy in the feature selection phase. For a feature X_i, it is defined as follows:

  $$J_{MIFS}(X_i) = I(X_i; Y) - \beta \sum_{j \in S} I(X_j; X_i) \quad \text{where: } \begin{cases} \beta : \text{non-negative parameter between 0 and 1} \\ S : \text{current selected feature set} \end{cases}$$

3 Data

3.1 Stock Market Data Set

The data is a set of features collected on companies in order to study their contributions to the stock market [5]. Due to limited resources, we chose one firm from this data set to be processed. Figures 3 and 4 show the input features as a function of time. Part of the features are trade prices. Yesterday and Close prices features share the same aspect over time since both describe the closing price of the trade. The features Open, Last, Low and High are highly correlated with each other as illustrated in the heatmap of features 6. In addition, there are other features regarding the quantity traded and the value of the trade. As for the number of transactions, this feature shows an increase in values especially at the beginning and the end of the year or in summer. Indeed, these periods are considered as rush periods. All this information will be used as input to forecast the next day's return. The latter is a univariate time series signal. From Figure 5, we can notice that it is not stationary.

While analyzing the data, we noticed that only the Value feature contains 4492 missing values. Thus, we thought to fill in these values with the **median value** of this firm. Plotting the pairwise plot of the features, we noticed that some of them contain outliers. So we applied the **median filter** on the Value and Quantity traded features. We first detected the outliers using the histogram. Then, we replaced the selected values using the median of a window of size 10. In addition, we performed a feature engineering step. We calculated the daily returns, the quantity traded divided by the number of trades, the value of current returns, dividing the open price by the daily returns and the square of the current returns (Figure 8). After that, we divided the data into training and test data, taking into consideration the temporal order between the two data sets. Finally, we scaled the data, as most supervised and unsupervised learning methods are sensitive to unscaled data.

3.2 Wine data set

For the classification task, we focused on the Wine Quality dataset [6, 7, 8] from UCI ML repository. This data contains information related to red wine. The features are based on physicochemical tests to estimate the wine quality based on a score. In our classification task, we want to estimate whether the wine is "good" or "bad" corresponding to a score above or below the mean value, respectively. Three of the features we are working on represent the amount of acid present in the wine: fixed acidity, volatile acidity and citric acid. Residual sugar measures the sweetness of the wine, while chlorides measures the saltiness. Free sulfur dioxide and Total sulfur dioxide are used to prevent oxidation and microbial spoilage. The pH is a measure of the strength and concentration of dissociated acids present in the medium, which explains its strong correlation with fixed acidity. Sulfates maintain the flavor and freshness of the wine. The density represents the specific gravity of the wine. And finally, alcohol, which is a key component of wine. That's a total of 11 features affecting wine quality.

This data, unlike the previous one, was found pre-processed in UCI Machine Learning repository. It is therefore clean, with no missing values or outliers.

4 Results

4.1 Models

For both tasks, we used 3 models. For the machine learning approach, we used Random Forest (RF) and Gradient Boosting (GB). We performed a grid search to tune their parameters with a random splitter for the classification task and a time series splitter for the regression task. The third model is a Neural Network (NN). It is based on **LSTM** and fully connected layers for the regression task while only fully connected layers for the classification task. In fact, the **LSTM** layer is important for forecasting as it takes into account the sequential aspect of the data which helps the model predict future values.

4.2 Regression Problem

4.2.1 Metrics

We used for evaluation three different regression metrics; Mean Absolute Error (MAE) which calculates the absolute difference between the actual and predicted signal, Mean Absolute Percentage Error (MAPE) which is MAE normalized by the actual observation, and Root Mean Squared Error (RMSE).

4.2.2 Results:

The REFS method is the fastest of all methods. It finishes the calculation of weights in 0.0236 seconds. Figure 9 illustrates the features importance. When varying the α parameter (Figure 19), we obtained fluctuating curves. Thus, we cannot conclude on a specific impact of α. The minimum value of both metrics are reached at $\alpha = 0.9$. Choosing a threshold value equal to the half of the maximum of scores, the algorithm selects the features: High, Last, and Price-returns; which reduced the data size by quarter-fifths. These features were found to have the closest reconstruction signal to the ground truth when solving the corresponding optimization algorithm. In consequence, this method provided better performance than without feature selection, with an exception for the RMSE metric in case of Random Forest and Gradient Boosting algorithms (Table 1).

The UDFS method takes about 0.3 seconds to finish and only gives importance to two features which are the Last and the square of the current returns (Figure 10) after tuning the parameter γ (Figure 20). All other scores are almost zero. Moreover, we can notice that this method outperforms all other methods when comparing the metrics. This can be interpreted as the Last feature represents price information and the squared current returns feature represents returns information. Thus, we can say that these features are the most relevant for predicting the next day's returns. We can conclude that this method simplified the data and reduced redundancy and correlated features.

The Laplacian score is the method with the highest computational time. We tuned the parameter of its similarity matrix, which turns out to be equal to $var = 70$, as shown in Figure 21. It selected current return, squared current return and price return with very high scores as illustrated in Figure 11. However, these features describe almost the same information. We can say that a potential drawback of this method is that it does not remove redundant features during the selection process. Therefore, including redundant features, even if they are relevant, does not provide any additional information, resulting in increased training time without improving performance.

We plotted the error graph for the predictions of the UDFS method. From the Figures 12, 13, and 14, we can notice that for all methods, the error curves are very close. These curves show a peculiar behavior at 3 dates (important peaks at November 8, November 23, and December 7). If we go back to the original signal, these points are those corresponding to outliers. Therefore, we can say that the models are able to learn the general behavior of the next day's performance, but when it comes to outliers, these algorithms are unable to predict their values. We can conclude that the error plot helps detect the outliers positions in a time series.

4.3 Classification Problem

4.3.1 Metrics

We used different classification metrics in order to obtain a fair evaluation of our models. We used, for instance, Accuracy, Precision, Recall, and F1-score metrics.

4.3.2 Results

Fisher Score method is the slowest of all methods. It takes around 13 seconds to compute the scores for all features. The result is shown in Figure 15. The most relevant ones appeared to be alcohol in the first place, then volatile acidity, total sulfure dioxide, and sulphates, respectively. The remaining features have very low scores. Thus, we chose to select only the first 4 features out of 11 having relatively higher scores.

Mutual Information Maximization method is the fastest method to calculate the scores of all features in only 0.023 seconds. However, the relative scores of this method are closer than those of the previous one. Thus, the selection of the most relevant features is a little more difficult. In our case, we selected the first 6 features out of 11. The selected features, as shown in Figure 16, are the same as those selected by the Fisher score method, to which are added density and chloride features. As explained in section 2.3, the MIM method sorts the features based on their correlation with the target attribute. We can say that our results confirm the theory as the features are indeed correlated with the quality target as shows the first column of the correlation plot in Figure 7. Furthermore, we notice that the MIM scores are proportional to the correlation values with wine quality; the higher the feature correlation, the higher its MIM score, except for the density feature .

MIFS method adds a weighted penalty term to the MIM expression that decreases the scores of features that are more correlated with the rest. The weight β was adjusted to maximize the F1-score metric. The found value, $\beta = 0.65$, also maximizes the Precision and Accuracy metrics, but not the Recall metric, as shown in Figure 18. However, we can notice that the corresponding Recall value is still relatively high. We cannot conclude a specific influence of the weight because the curves of the metrics do not show a distinctive pattern, except that they reach their maximum for a higher value of β. The result of the scoring is shown in Figure 17, with the scores gradually increasing as the feature correlation decreases, confirming the theoretical interpretation of the MIFS expression. For example, the most relevant feature according to MIFS is total sulfur dioxide, which is the least correlated feature with all the others, except with the free sulfur dioxide feature, which moreover has a very low MIFS score and is therefore eliminated by the algorithm. We chose for this method the first eight features out of 11 in order to obtain better classification results.

According to the four selected evaluation metrics, represented in Tables 2-3, the MIFS selection method is the best and gives promising results for the classification task. It performs better than MIM, the other information-based method. This may be related to the fact that, in reality, good features should not only be highly correlated with the class labels, but also not be highly correlated with each other. We also note that all tested feature selection methods improved the classification results by about 2% up to 7% over the results without feature selection, except for the neural network classifier. This can be explained by the fact that the field of deep learning in general needs larger data and many features to learn better.

References

[1] J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, and H. Liu, "Feature selection: A data perspective," *ACM computing surveys (CSUR)*, vol. 50, no. 6, pp. 1–45, 2017.

[2] F. Nie, H. Huang, X. Cai, and C. Ding, "Efficient and robust feature selection via joint 2, 1-norms minimization," *Advances in neural information processing systems*, vol. 23, 2010.

[3] Y. Yang, H. T. Shen, Z. Ma, Z. Huang, and X. Zhou, "L2, 1-norm regularized discriminative feature selection for unsupervised," in *Twenty-second international joint conference on artificial intelligence*, 2011.

[4] X. He, D. Cai, and P. Niyogi, "Laplacian score for feature selection," *Advances in neural information processing systems*, vol. 18, 2005.

[5] https://www.kaggle.com/t/14f956bf170e463dadbcb58bb3f36540

[6] P. Cortez, A. Cerdeira, F. Almeida, T. Matos, and J. Reis, "Modeling wine preferences by data mining from physicochemical properties," *Decision support systems*, vol. 47, no. 4, pp. 547–553, 2009.

[7] https://archive.ics.uci.edu/ml/datasets/wine+quality

[8] https://www.kaggle.com/datasets/nareshbhat/wine-quality-binary-classification

Appendix

	MAE				RMSE				MAPE			
	w/o FS	REFS	UDFS	LS	w/o FS	REFS	UDFS	LS	w/o FS	REFS	UDFS	LS
RF	74.661	69.676	**60.343**	68.483	101.685	108.248	**94.967**	101.23	5.312	3.993	**3.398**	3.766
GB	73.782	69.336	**59.409**	67.827	98.985	111.701	**93.601**	98.919	6.125	4.237	**3.223**	4.318
NN	82.606	62.066	**54.892**	64.792	111.791	97.09	**95.486**	99.557	4.411	3.951	**2.341**	3.056

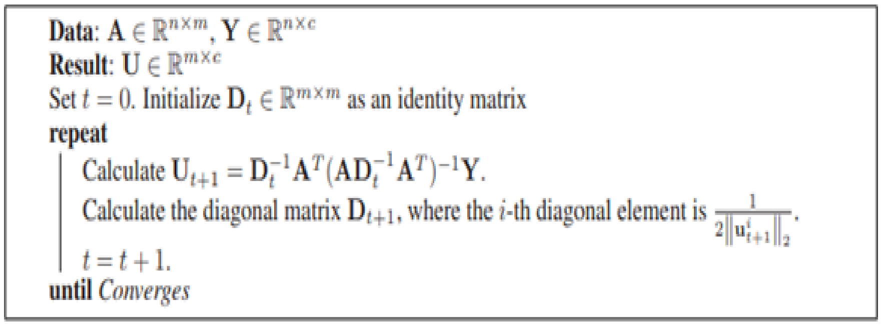
Table 1: MAE, RMSE, MAPE metrics for the regression problem

	Accuracy				Precision			
	w/o FS	Fisher Score	MIM	MIFS	w/o FS	Fisher Score	MIM	MIFS
RF	0.792	0.804	0.802	**0.812**	0.817	**0.842**	0.823	0.839
GB	0.792	0.794	0.796	**0.815**	0.813	0.813	0.814	**0.845**
NN	0.752	0.721	0.723	**0.769**	0.772	0.801	0.782	**0.842**

Table 2: Accuracy, Precision metrics for the classification problem

	Recall				F1-score			
	w/o FS	Fisher Score	MIM	MIFS	w/o FS	Fisher Score	MIM	MIFS
RF	0.805	0.798	**0.82**	**0.82**	0.811	0.819	0.822	**0.83**
GB	0.813	0.816	**0.82**	0.816	0.813	0.815	0.817	**0.83**
NN	**0.787**	0.663	0.697	0.719	**0.779**	0.725	0.737	0.776

Table 3: Recall, F1-score metrics for the classification problem



```
Data: $\mathbf{A} \in \mathbb{R}^{n\times m}$, $\mathbf{Y} \in \mathbb{R}^{n\times c}$
Result: $\mathbf{U} \in \mathbb{R}^{m\times c}$
Set $t = 0$. Initialize $\mathbf{D}_t \in \mathbb{R}^{m\times m}$ as an identity matrix
repeat
    Calculate $\mathbf{U}_{t+1} = \mathbf{D}_t^{-1}\mathbf{A}^T(\mathbf{A}\mathbf{D}_t^{-1}\mathbf{A}^T)^{-1}\mathbf{Y}$.
    Calculate the diagonal matrix $\mathbf{D}_{t+1}$, where the $i$-th diagonal element is $\frac{1}{2\|\mathbf{u}_{t+1}^i\|_2}$.
    $t = t + 1$.
until Converges
```

Figure 1: REFS algorithm

```
Algorithm 1: The UDFS algorithm.
1  for $i = 1$ to $n$ do
2      $B_i = (\tilde{X}_i^T \tilde{X}_i + \lambda I)^{-1}$
3      $M_i = S_i H_{k+1} B_i H_{k+1} S_i^T$;
4  $M = X \left( \sum_{i=1}^{n} M_i \right) X^T$;
5  Set $t = 0$ and initialize $D_0 \in \mathbb{R}^{d \times d}$ as an identity matrix;
6  repeat
7      $P_t = M + \gamma D_t$;
8      $W_t = [p_1, ..., p_c]$ where $p_1, ..., p_c$ are the
       eigenvectors of $P_t$ corresponding to the first $c$
       smallest eigenvalues;
9      Update the diagonal matrix $D_{t+1}$ as
       $D_{t+1} = \begin{bmatrix} \frac{1}{2\|w_t^1\|_2} & & \\ & \cdots & \\ & & \frac{1}{2\|w_t^d\|_2} \end{bmatrix}$;
10     $t = t + 1$;
11 until Convergence;
12 Sort each feature $f_i |_{i=1}^{d}$ according to $\|w_t^i\|_2$ in
   descending order and select the top ranked ones.
```

Figure 2: UDFS algorithm

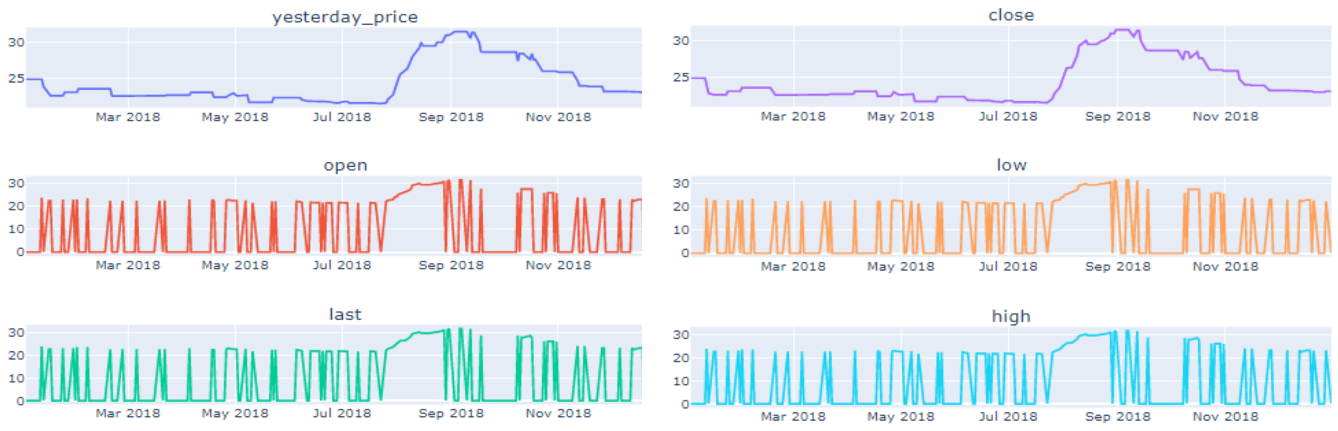


Figure 3: Prices features in function of time

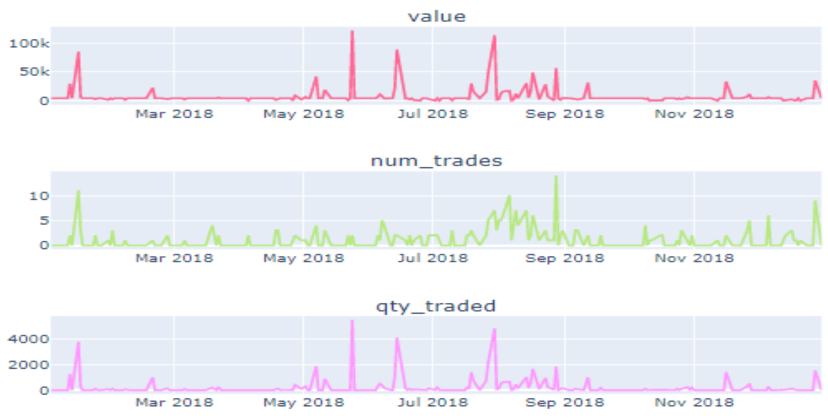


Figure 4: Features about trade's information

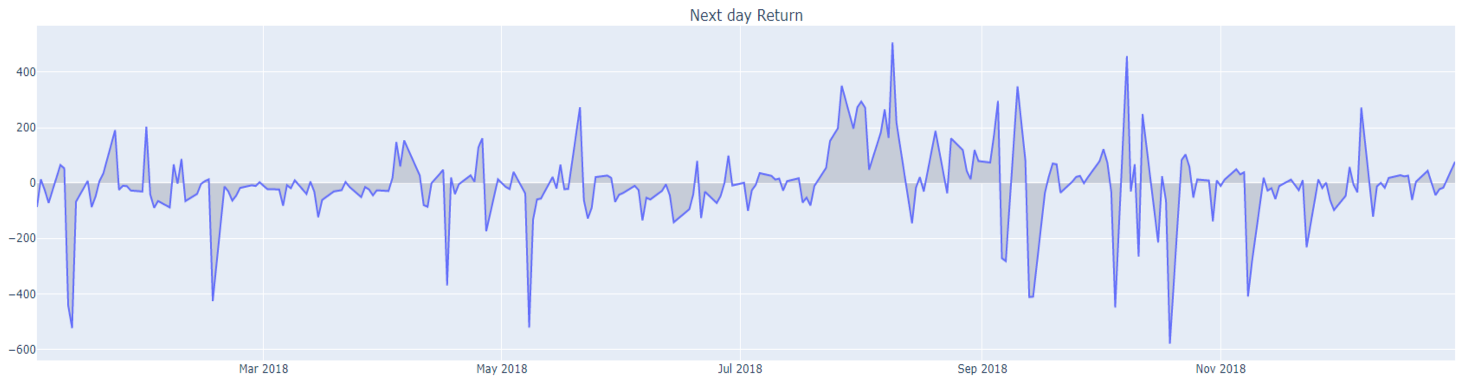


Figure 5: Target feature: next day's returns

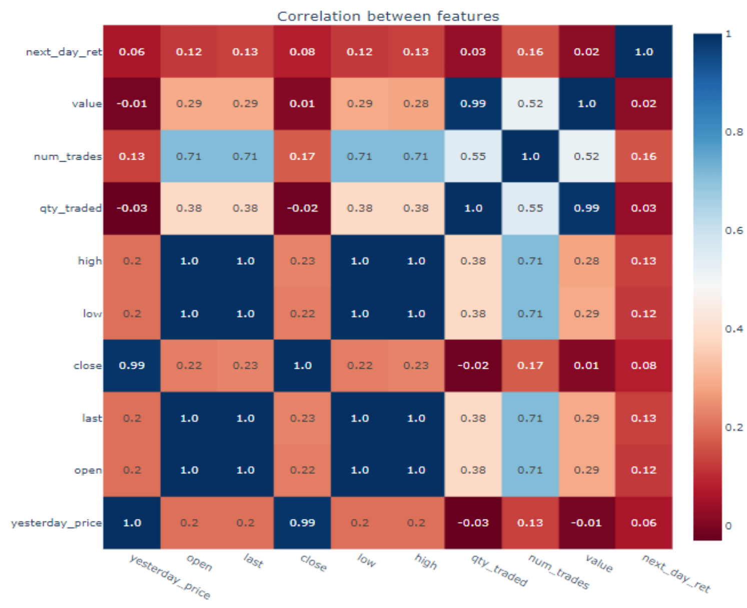


Figure 6: Correlation heatmap between features & target for regression task

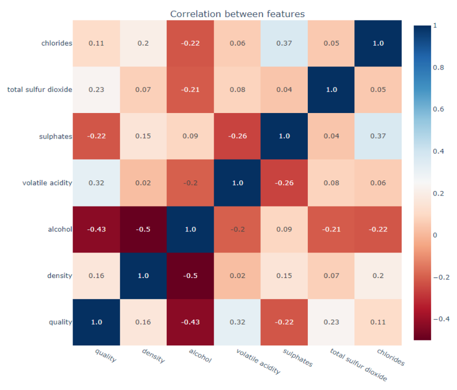


Figure 7: Correlation heatmap between features & target for classification task

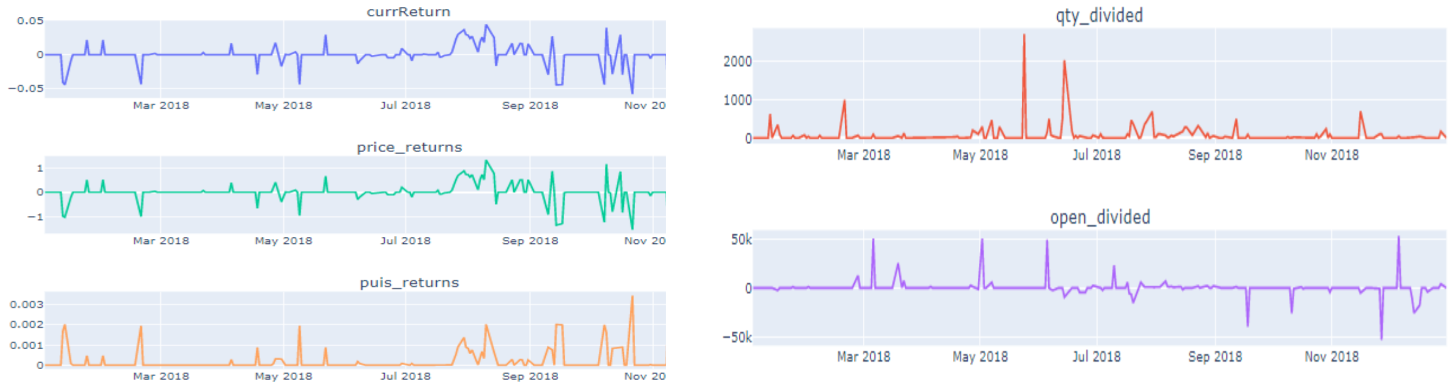


Figure 8: New features for regression task

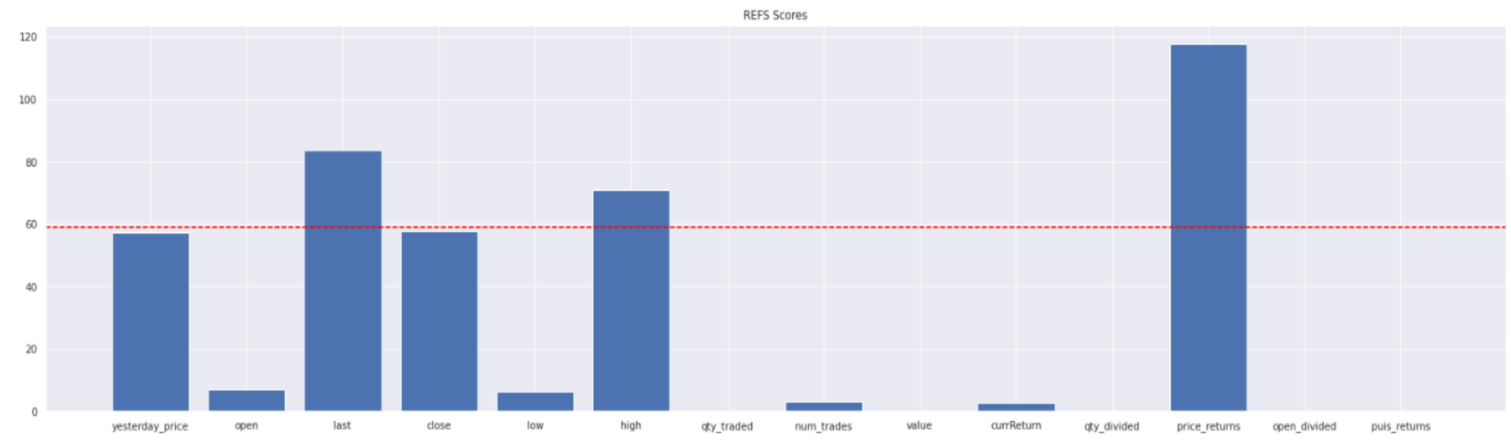


Figure 9: REFS Scores

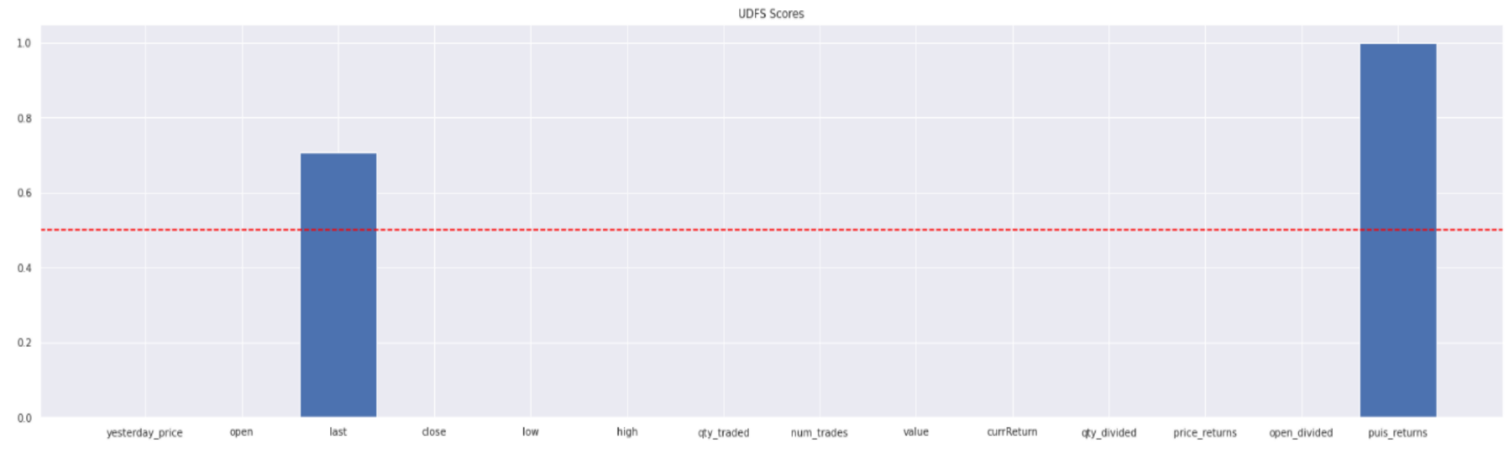


Figure 10: UDFS Scores

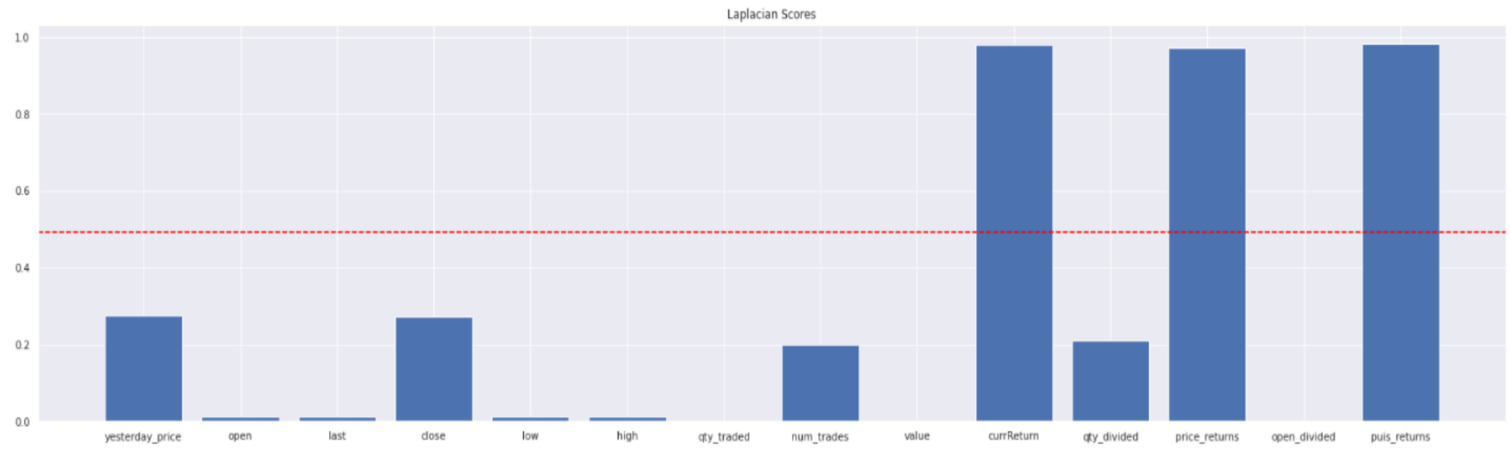


Figure 11: Laplacian Scores

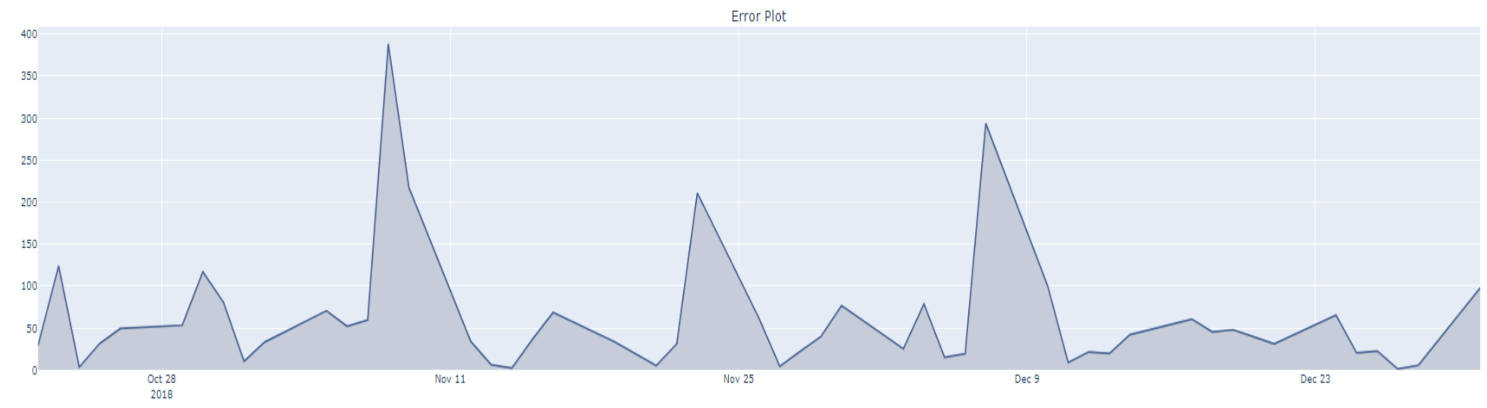


Figure 12: Random Forest error plot with UDFS method

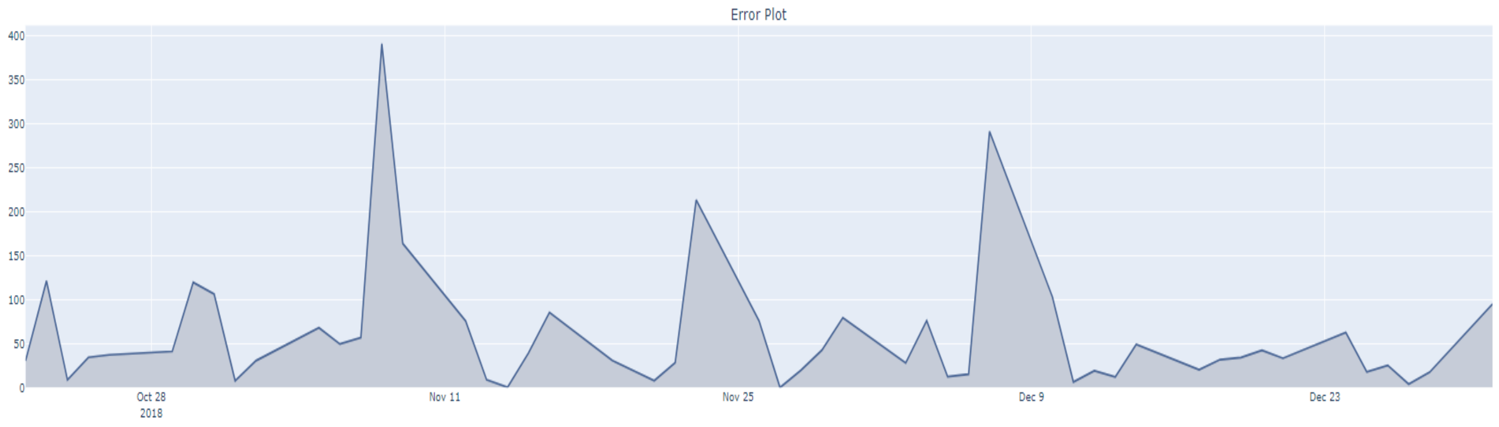


Figure 13: Gradient Boosting error plot with UDFS method

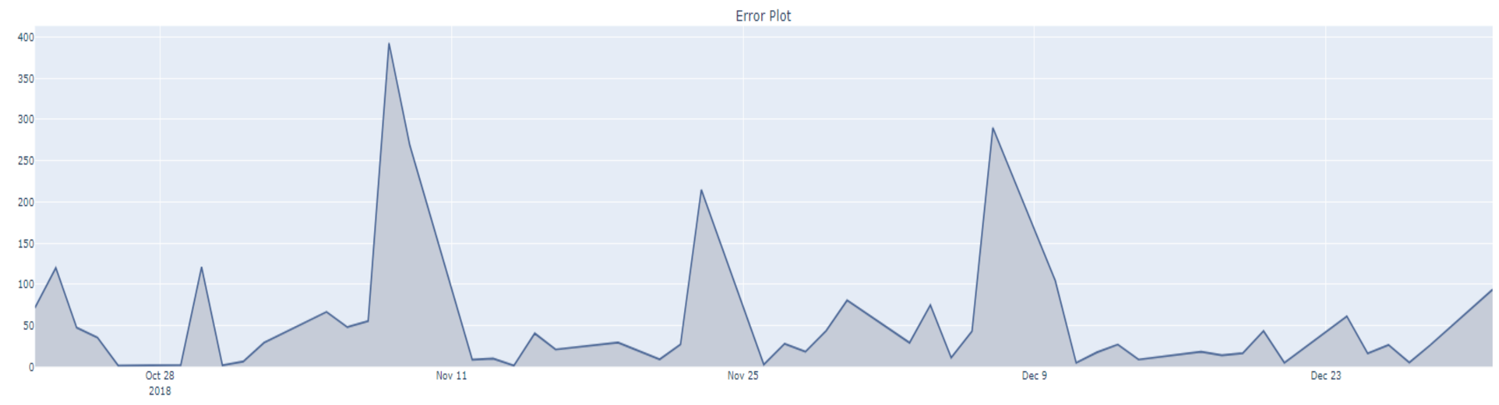


Figure 14: Neural Network error plot with UDFS method

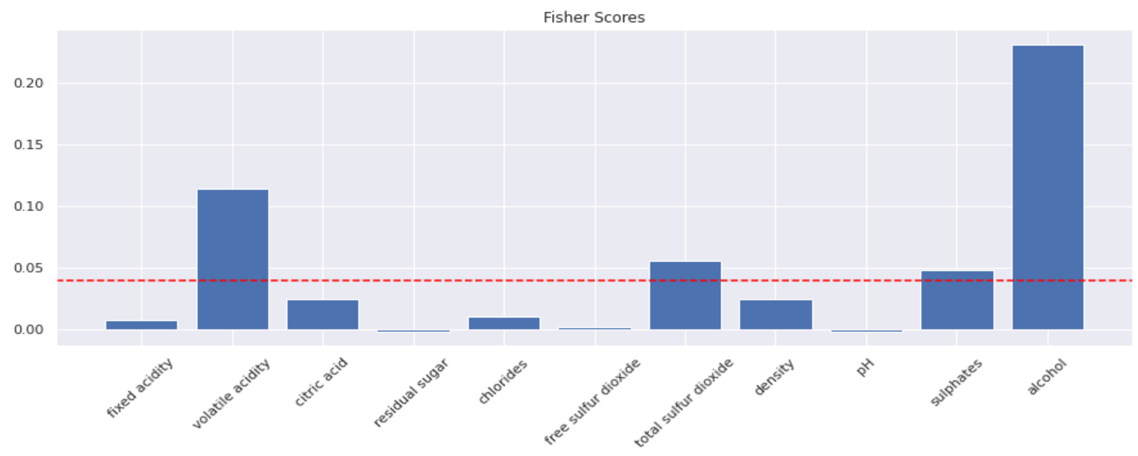


Figure 15: Fisher Scores

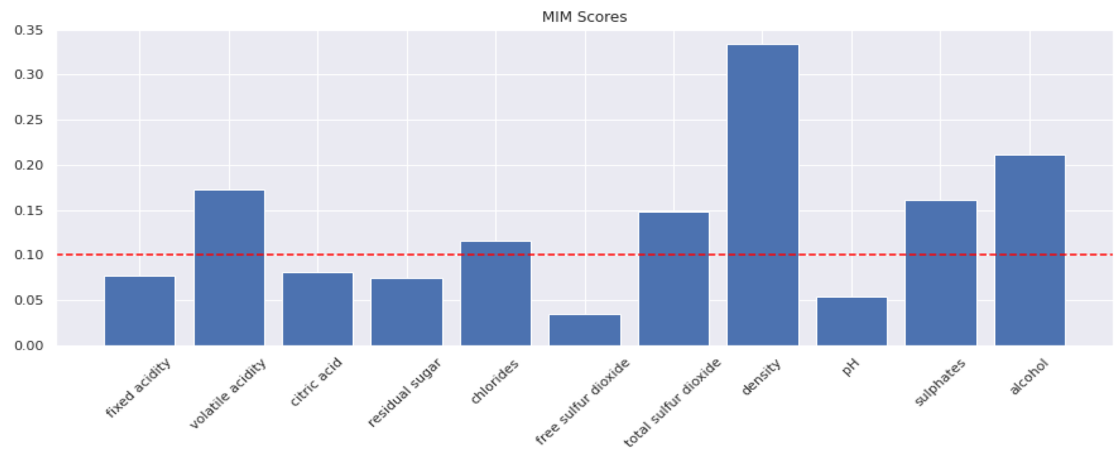


Figure 16: Mutual Information Maximization Scores

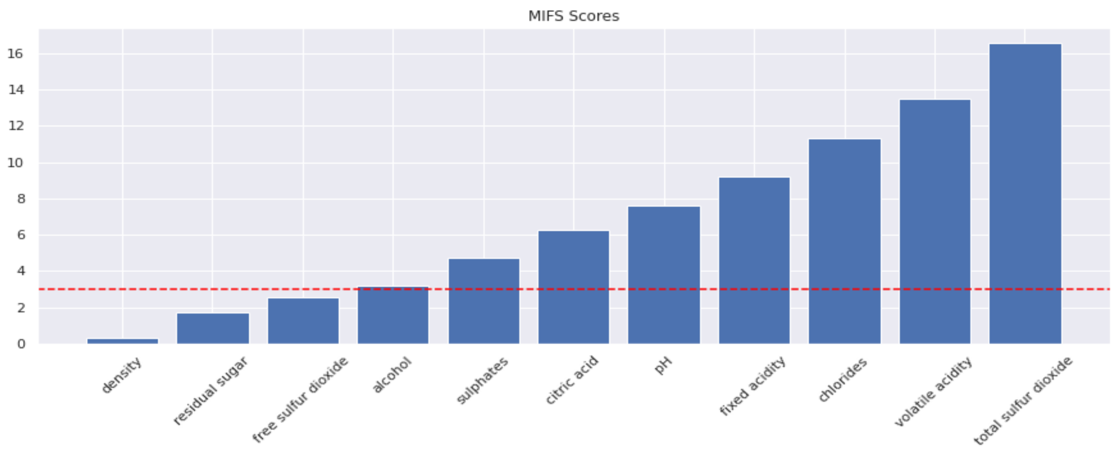


Figure 17: Mutual Information Feature Selection Scores

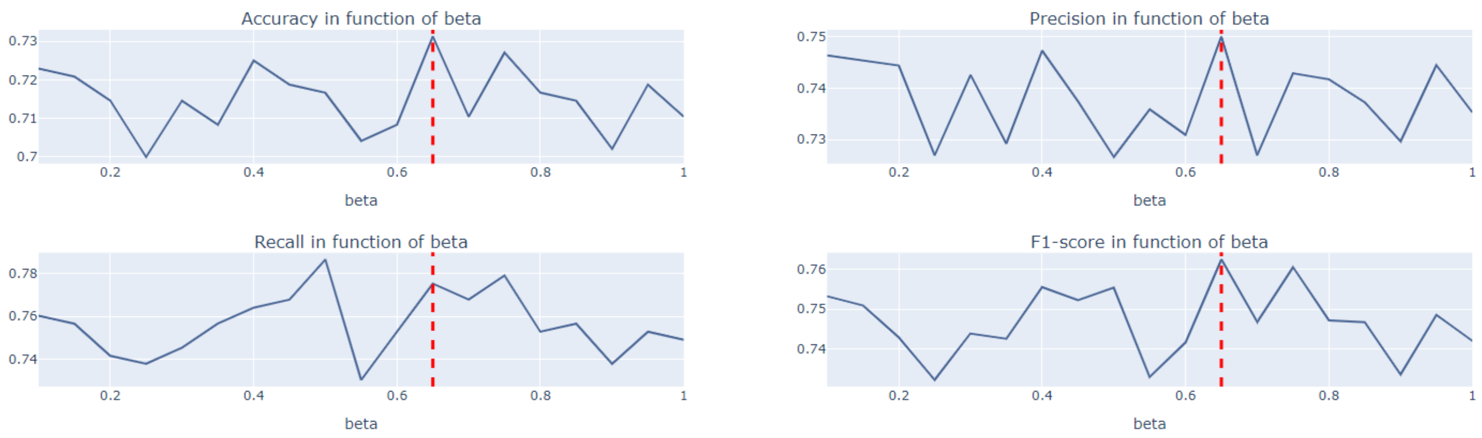


Figure 18: Tuning the parameter β in MIFS method

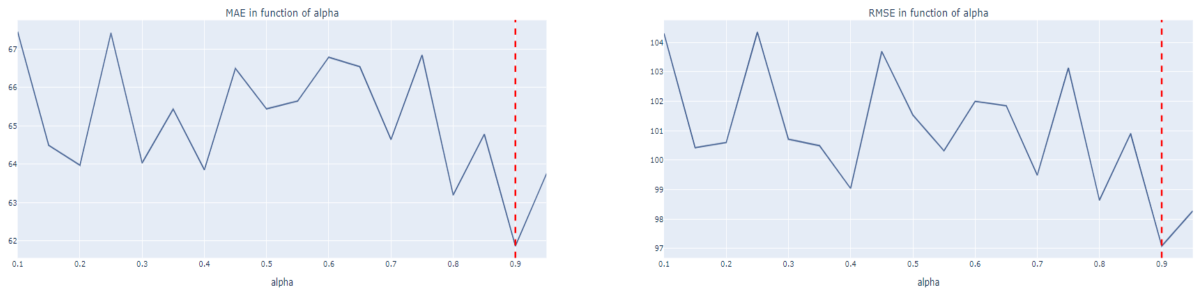


Figure 19: Tuning the parameter α in REFS method

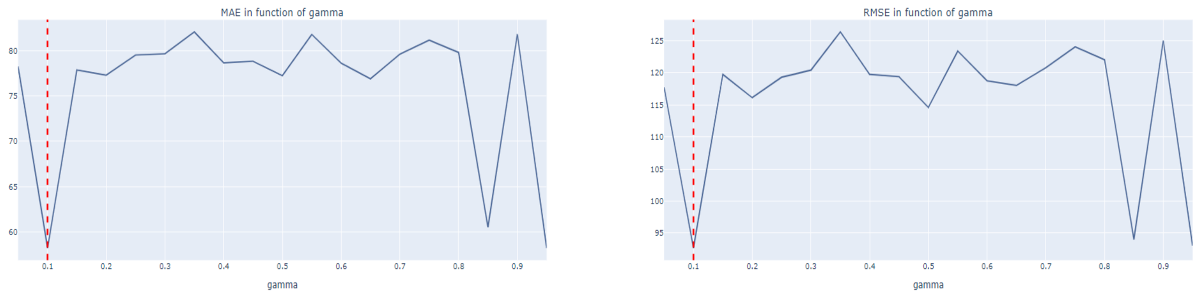


Figure 20: Tuning the parameter γ in UDFS method

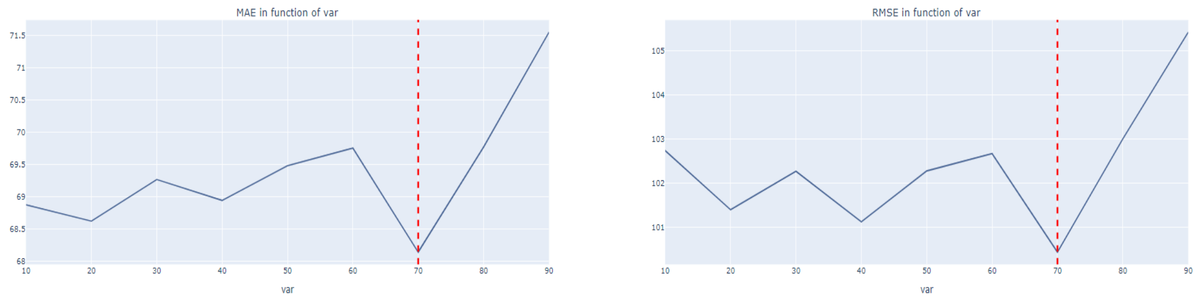


Figure 21: Tuning the parameter t (var) in Laplacian Score method