Faculty of Mathematics and Information Science
Warsaw University of Technology

Project 2 documentation

Feature selection methods in classification task

Subject: Advanced Machine Learning

Author:
Patryk Wrona

Warsaw
2022

1. Feature selection methods in classification task

1.1. Goal

The goal of this report is to present the performance of feature selection methods on given 2 datasets – *artificial* and *digits*. The code relating to chosen feature selection techniques, chosen classification models, as well as hyperparameter tuning of the most efficient model across the best subsets of features can be found in : *aml-notebook.jpynb*. Plots were created in *making-plots.ipynb* using obtained results from *aml-notebook.jpynb*. The final score on the validation dataset achieved by the best model is known uniquely by project supervisor. All details could be found in attached project scope – *Project 2.pdf*. This project's aim was to achieve a balance between the number of selected features and the value of *Balanced Accuracy* metric – the definition of used *Score* was defined in *Project 2.pdf*

1.2. Description of used feature selection methods

Following 4 feature selection methods were used in given order, but with different parameters for each case:

— **Variance Threshold** – features of 0 variance were deleted at the beginning, a function **removing** up to given number of lowest variance features was implemented in *aml-notebook.jpynb*
— **Normality Tests** – a function **removing** up to given number of features having the lowest p values for D'Agostino and Pearson's test was implemented in *aml-notebook.jpynb* (it removes features coming from normal distribution)
— **Boruta algorithm** – *BorutaPy()* from *boruta* Python package, **selecting** confirmed significant features, with the use of Random Forest classifier from *sklearn.ensemble*
— **VIF** – an iterative function **leaving** given number of features of the lowest variance inflation factor was implemented in *aml-notebook.jpynb* (it removes multicollinear features left by previous methods). Only this method decided on the final number of selected features (as the Boruta algorithm's confirmed features were left without interference).

1.3. Description of used classifiers

In case of *artificial* dataset, the feature selection methods were compared along with 18 machine learning models for classification task, being models of different hyperparameters or ensembles of these 4 classifiers:

— **RandomForestClassifier** – from *sklearn.ensemble* package
— **GradientBoostingClassifier** – from *sklearn.ensemble* package
— **LogisticRegression** – from *sklearn.linear_model* package
— **LinearSVC** – from *sklearn.svm* package

It was noticed that **GradientBoostingClassifier** outperformed other classifiers and due to limited computational power and time, other models were skipped in case of (larger) *digits* dataset.

1.4. Description of conducted experiments

After the split of train dataset into 2 subsets (train_train and train_valid) in ratio 75%/25% and 80%/20% respectively in case of *artificial* and *digits* datasets, the procedure of choosing the best pair of a tuned model and selected features' set looked the same for both datasets:

— **Step 1** – Calculating scores for each parameter set of feature selection methods (including omitting of low variance removal and normality tests) and each classifier in use
— **Step 2** – Basing on the scores, choosing the most efficient model and the best subsets of features, but also counting the occurrences of features and creating additional subsets of the most frequently occurring features (trying to create better sets of features than the best ones from previous step)
— **Step 3** – Hyperparameter tuning of given 1 model (GradientBoostingClassifier) across the best sets of features (the most frequently occurring and those from Step 1 and 2)

To alleviate the influence of randomness, train-test split and model training were repeated and the *Scores* were averaged (for given pair of hyperparameters' and features' sets). The above procedure was also checked on standardized data (the number of model fits was doubled).

After the above steps, for each dataset, the features' set and the hyperparameter's set obtaining the highest value of *Score* was selected.

Besides, after getting above model and features, the influence of train dataset's size was checked in order to estimate if it is worth training the final model on the whole training data.

There was also an attempt to make better predictions based on **unsupervised learning outlier detection method** – *Isolation Forest*. Outliers detected in *digits* train dataset were belonging in majority to the class *-1* and there was similar percentage of outliers in both train and validation datasets. Nevertheless, the model's balanced accuracy was relatively high (almost 1.0) and the ratio of classes across validation dataset outliers were kept (*-1* classes were majority) – that is why it was not used to improve ultimate predictions.

1.5. Results of conducted experiments

Firstly, as it was stated above, the chosen best model for classification was **GradientBoostingClassifier**.

Secondly, final models were trained on the **whole training dataset** (because the results on validation dataset increased with the size of training dataset).

The final model was trained on standardized data in case of *artificial* data, and on unstandardized *digits* data.

The impact on *Score* of different hyperparameters and number of features selected are presented in the next 2 figures (for *artificial* 1.1 and *digits* 1.2 respectively).

Final models and selected features were chosen independently on the plots (basing not on number of features but rather on a given subset of features), but still the chosen best features are in local maximum of these plots – they have 10 and 180 elements for *artificial* and *digits* data respectively.

The most performing model's hyperparameters are:

— **learning_rate** = 0.15 (artificial) and 0.3 (digits)
— **n_estimators** = 70 (artificial) and 60 (digits)
— **max_depth** = 7 (artificial) and 5 (digits)

1.6. Conclusions

Removing the multicollinearity using VIF (variance inflation factor) and performing Boruta algorithm seemed to have the highest effect on the efficiency of feature selection. Somehow

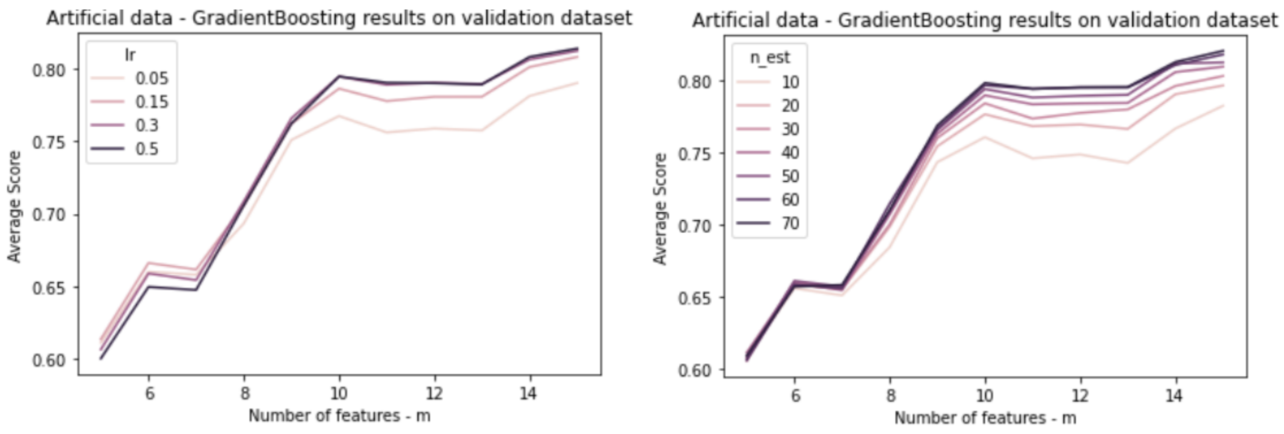

Figure 1.1: Influence of number of features on obtained average Score on artificial validation data – categorized by learning rate and number of estimators of GradientBoostingClassifier

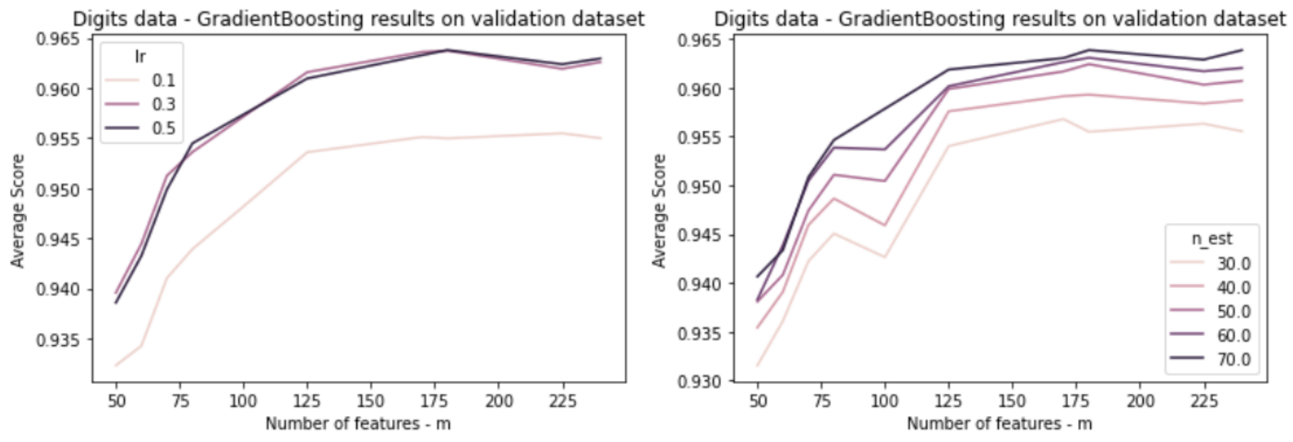

Figure 1.2: Influence of number of features on obtained average Score on digits validation data – categorized by learning rate and number of estimators of GradientBoostingClassifier

essential method was to reject features with very low variance. Unfortunately, removing normal variables seemed to have the lowest impact on final features – the most probably, strictly normal variables were further removed by Boruta algorithm.

The final value of *Score* should be calculated by the project supervisor, so there is no mean of calculating it beforehand. However, my expectations for *artificial* dataset is $Score = \mathbf{0.85}$ and for *digits* dataset $Score = \mathbf{0.95}$.

The resulting probabilities of assigning observations to the class *1* were saved in files:

— PATWRO_artificial_prediction.txt
— PATWRO_digits_prediction.txt

The indexes of selected features were saved in files:

— PATWRO_artificial_features.txt
— PATWRO_digits_features.txt

Files containing the results were also attached to this project – in *results-artificial.csv* there is model and feature selection method search (**Step 1**). The feature selection search results for *digits* data are in *results-digits.csv*.

Finally, results of hyperparameter tuning of **GradientBoostingClassifier** across chosen feature set cases are in *results5_2-artificial.csv* and *results2-digits.csv* files for *artificial* and *digits* data respectively.