Machine Learning: Exercise Set VI

Georgios Manos
csd4333

8 December 2022

1 Part A

1.1 Exploratory Data Analysis

To choose the 5 features, I chose the following criterions. First, I calculated Pearson's correlation for each variable to the target variable Y and chose all variables with medium correlation and above (2 variables with threshold $|x| \geq 0.29$, where x the Pearson's correlation score). Then, I split the dataset into categorical and continuous variables by counting the number of different values per variable (categorical values threshold < 14, the rest are classified as continuous). For continuous variables, I calculated the coefficient of variation ($CV = \frac{mu}{std}$, where mu is the mean value of that variable and std is the standard deviation respectively) and chose all variables with $CV \geq 1.0$ (essentially expecting sparse continuous variables, resulted 1 as such). For categorical ones respectively, I calculated the category percentage vector for each variable and calculated the coefficient of variation for that vector. I chose all variables with $CV \geq 1.5$, as all categorical variables were quite sparse.

The result statistics are as follows:

Variable	Score	Metric
5	-0.407340	Pearson's Correlation
4	0.299083	Pearson's Correlation
13	2.199363	Continuous High CV
1	1.840119	Categorical High CV
9	1.842367	Categorical High CV

The resulting histogram are as follows. I hand-checked the variable percentages for categorical variables by hand and saw that indeed my metric picked variables with really high imbalance (not sure how ”highest” would be defined, but I sorted the calculations on descending order and saw that as the CV score descended, the categories were spread out more evenly). I also used absolute value on Pearson’s correlation as we don’t care about the type of relationship (negative or positive) as long as they are highly related. Maybe Pearson’s metric was not the best one since its a linear metric and assumes variable independence, but I went for it eitherways.

The resulting plots are following. For the High Correlation to target variable features, we don’t extract much information about them from the histogram. As for the rest, we indeed see that they are sparsely distributed.

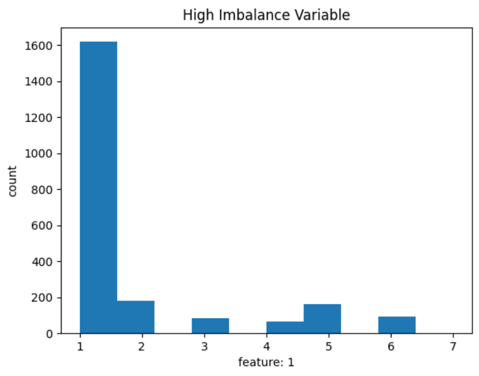

Figure 1: High Imbalance Variable histogram.

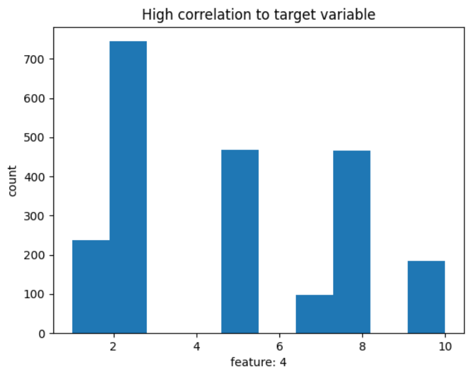

Figure 2: High correlation to target variable histogram.

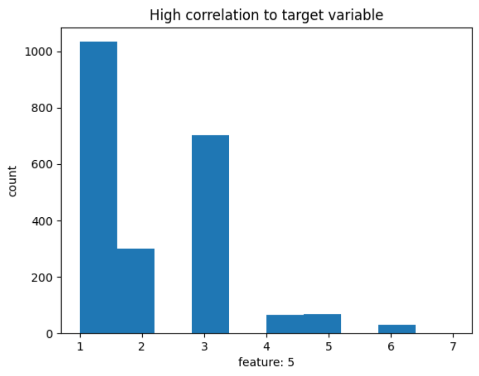

Figure 3: High correlation to target variable histogram.

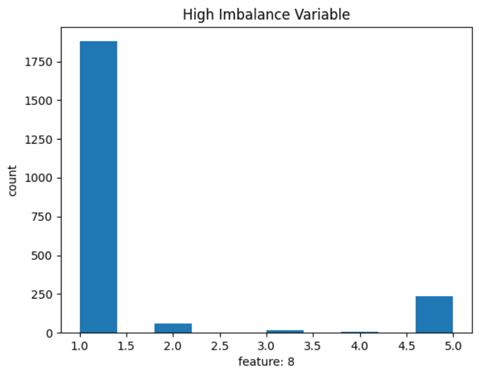

Figure 4: High Imbalance Variable histogram.

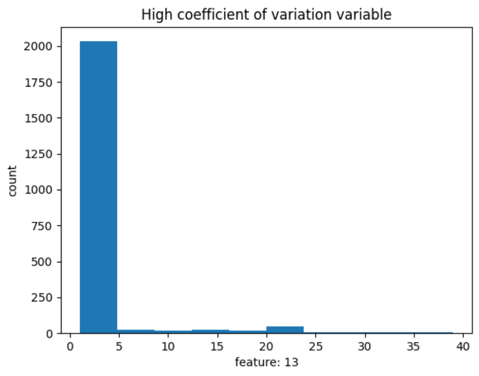

Figure 5: High Coefficient of Variation Variable histogram.

1.2 Model Selection

The models I will be comparing are Logistic Regression and Random Forest, with and without standardization (StandardScaler) and with various possible configurations.

I implemented CV protocol from scratch. However, I used the sklearn StratifiedKFold implemented split function for the Stratified K-Fold split. If I were to implement that from scratch too, I would remove the minority class features from the dataset, shuffle the rest and k-fold split them, and finally evenly distribute the minority class on each fold, trying to retain the original class ratio on each fold.

The code is pretty much explained by my comments too. The CV function is totally agnostic of the configurations provided, and what it does is essentially calculate the AUC score for each configuration and fold, then calculate the mean AUC score for each configuration, pick the configuration with the maximum mean AUC score and using it, train a classifier on all available data and return that model. The only hardcoded part of this function is the Standardization part as it only supports StandardScaler function for simplicity purposes.

Notes about the scaler, on each fold we fit the scaler on the training set and apply it to both training and test set before moving to our model evaluation. This way, we are not violating the golden rule and peeking into the test data. Also, the function returns a scaler (if applicable) that was fit and applied to all available data, prior to the training of the final model.

The model that is usually picked is Logistic Regression with standardization and C=1, with a resulting AUC score of approximately 0.865. On every run I made, Logistic Regression with standardization was picked, but perhaps with different C value, as all those models result in similar performance.

2 Part B

2.1 Computing the out-of-sample performance

The performance obtained by the Cross Validation is (AUC score) 0.8656. The out-of-sample performance however is 0.8728. We expect the out-of-sample performance to be higher than the performance obtained by the CV protocol as the latter is a conservative approximation of the first. This is due to that the final model is trained on all available data to the performance estimations.

2.2 The ROC Curve

The trivial (naive) classifier is the classifier that always classifies with the majority class of the training data. To produce a ROC curve, we need a score for each sample in the test set. The scores are all going to be equal to the percentage of the majority class on the training set. This results to an AUC of 0.50, which is much less to our chosen classifier (same pretty much as the random classifier).

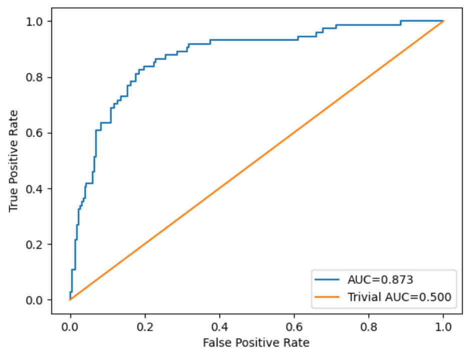

Figure 6: ROC Curves for our best classifier versus the Trivial Classifier.

2.3 Calculating the 95% Confidence Intervals

I applied 1000 bootstraps on the hold-out set and gained an AUC score for each bootstrapped dataset. The resulting histogram is:

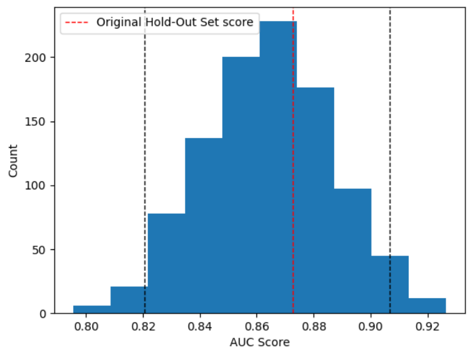

Figure 7: Model AUC scores for 1000 bootstraps on the hold-out set. The black dashed lines represent the 0.025 and 0.975 confidence intervals respectively.

The confidence interval lines show us how accurate the hold-out set AUC score is. The further they are, the more likely is to get a different performance when the model is taken into production. For example, if the 0.025 CI line is far from the original hold-out set, we would expect the model to get a lower AUC score to the hold-out set score in practice. Also note that the hold-out set AUC score is quite near the mean value of the (quite normal ☺) distribution