Università
Ca'Foscari
Venezia

Artificial Intelligence and Data Analytics 1

Home Credit Default Risk Analysis - Report

Kanan Mammadli 888195

Introduction

The Analysis focuses on predicting whether the client will pay back the loan or not and what factors impact the capability of paying back loans. Data related to Credit Default Risk, and there are 10 data files. As already mentioned above, there are 10 CSV data files where 2 of them are train and test data. This Analysis concentrated mainly on application_train and application_test data, which are considered as primary data. TARGET variable indicates whether or not the client can repay the loan. However, our prediction will not show precisely 0 or 1; instead, it will show probability between 0 and 1.

Main decisions and Methodology

During Analysis, the model to "predict.proba" method is used to predict the chances of not repaying a loan, which is the primary goal. The first column represents the probability of the target being 0, and the second column represents the probability of the target is 1. In Analysis, first of all, the "TARGET" variable was separated from the train set. The principal methodologies that are used are Logistic Regression, Decision Tree, and Random Forest classifier. In order to reach the best result, GridSearchCV hyperparameter tuning was applied. After understanding the data set with some graphs, cleaning of data from missing values was applied. Before that, the main libraries used during Analysis were imported and loaded "application_train" and "application_test" files. Furthermore, there was used a simple loop to detect missing values and then drop missing values both in test and train data which has more than 30% of missing values.

Moreover, in order to be more precise, during handling missing values, the dataset is divided into three parts: categorical, integer and float parts. First checked missing values in each new dataset, then different methodologies applied in each part in order to handle missing more accurately. Precisely, categorical values changed with their median, integer with their mean, and float changed with 0. (float, it was not easy to change with mean because it requires too much time). Finally, after checking for missing values for the last time to be sure, we concatenated three parts together.

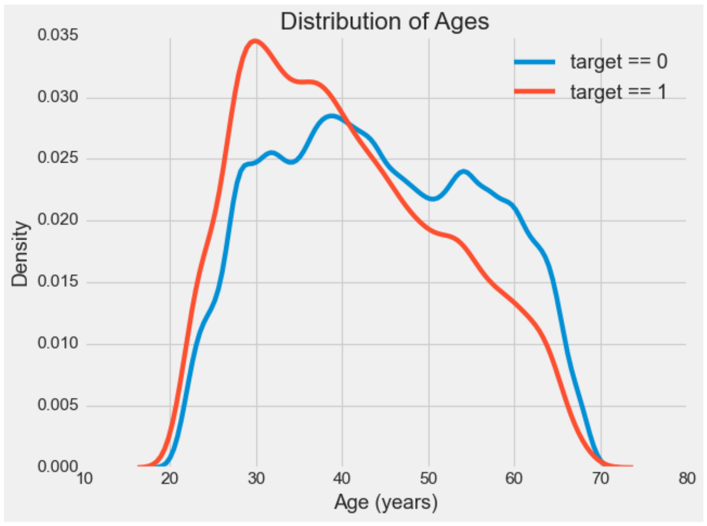
After checking categorical variables, it seems that there is one variable that has more than 50 features. Therefore, this variable's features decreased to 10 and replaced the 'OTHERS' feature, and only nine most important features stayed. I want to mention that these processes applied both train and test set at the same time.

Additionally, new custom features are generated to check the future, whether they have some impact on our prediction. First is "LOAN_RATE," which is equal to "AMT_OF_ANNUITY" divided by "AMNT_OF_CREDIT," second "CREDIT_INCOME_RATIO," which represent the percentage of credit given to total income. ("AMOUNT_CREDIT" divided by "TOTAL_INCOME"). Moreover, the last one is "CHILDEREN_RATIO," which represents the percentage of children in the family and counts like "CNT_CHILDEREN" divided by "CNT_FAMILY_MEMBERS."

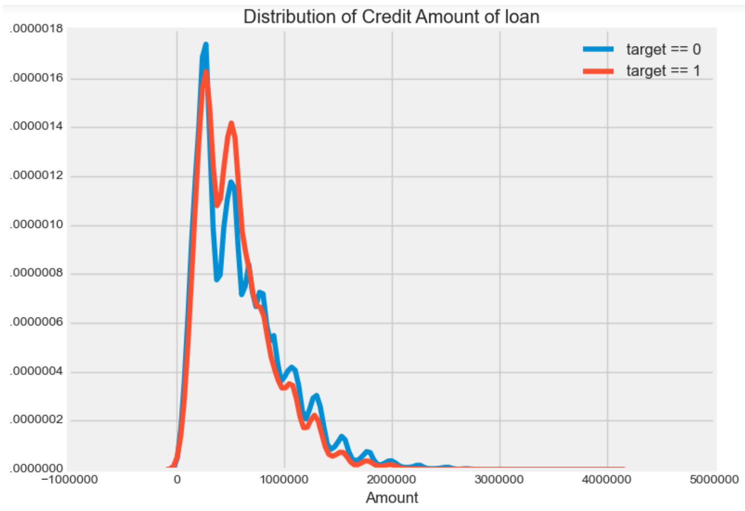
After that encoding process was applied to categorical variables; for the categorical variables with only two features, LabelEncoding from sklearn function applied whether variables more than two categories, OneHotEncoding function was applied. After Encoding, original variables were dropped and aligned train and test sets to avoid mismatch problems. Following, there applied MinMaxScaling to the datasets. It is essential to mention that the main train, test sets copied and named train and test sets, and second train set without new custom features is called train1 and test1 accordingly.

Graphs

The graph shows that people aged between 25-40 and have slight difficulties paying back loans.



The second graph shows that credit holders with credit amounts between 0-100,000 USD have some difficulties paying back the money.



DATA MODELLING

First of all, GridSearhCV was used for Hyperparameter Tuning for Logistic regression for both new features and without new features. After tuning, we got an answer: Inverse of regularization strength should be 0.01, and as a penalty, we have to use L2. First tuning applied to a set that has all features (include new ones). Tuning for the second training set answered differently. When we train set without new features, it indicates that best C (Inverse of regularization strength) should be 1.0 and L2 penalty for the best penalty.
Furthermore, the next GridSearchCV applied for Random Forest Classifier. First applied with new features a get result like this: 'max_depth': 25, 'n_estimators': 200
It indicates that the number of trees should be 200, and the max depth of trees would not be more than 25, and this tuning gives 91.04 % accuracy.
During hyper tuning for training set without new features gives same results as above:
: ‘max_depth': 25, 'n_estimators': 200,but it gave slightly more accuracy for tuning- 91.05%

Finally, for experience purposes, the Decision Tree Classifier used in GridSearchCV and gave these results.
With all features: Best parameters {'max_depth': 100} and accuracy 86%
Without new custom features: Best parameters {'max_depth': 100} and accuracy 86%

Results

To conclude, considering our three best models, we have found out that the first two models' results are pretty similar. Considering all features, our best model is **Random Forest Classifier** with a max depth of trees = 25 and number of trees = 200, which giving us a score = 0,71403; when new custom features dropped, accuracy decreases and reached 0,70529.
The second-best model is **Logistic Regression** with 0.70082 accuracies when used first tuned model with all features. Nevertheless, when we drop new custom features and use the second parameters that we obtain during tuning, it dramatically decreases to about 0.61.

The Decision Tree is the worse model for our prediction. When we consider all features, it gives 61% with the first tuned model and about 53% with the second tuned model, without new custom features. That is why the Decision tree Classifier is not suitable for prediction.