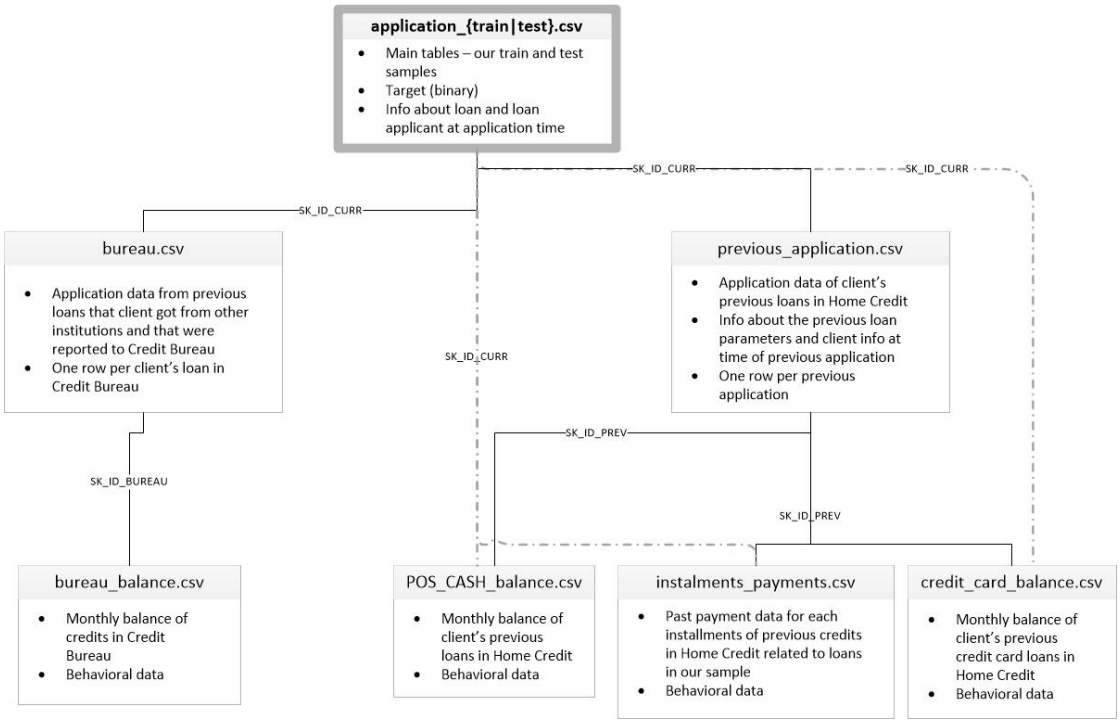
Home Credit Default Risk Analysis

1. Introduction

This report focusses on the Kaggle competition “Home Credit Default Risk”. The data is provided by Home Credit to predict whether or not a client will repay a loan or have difficulty. There are 7 different sources of data: application_train/application_test, bureau, bureau_balance, previous_application, POS_CASH_BALANCE, credit_card_balance, installments_payment. We first do the pre-processing for the data, including transforming the dummy variables and dealing with missing values. Then we apply three methods to select the most important features for further study. Next, we use four algorithms, which are Logistic Regression, Random Forest, Gradient Boosting and Ada Boost, to fit models and use cross validation to evaluate their performance by calculating their average AUC. Finally, We choose the algorithm with the largest AUC to predict probability for the TARGET variable.

2. Data processing



This is a standard supervised classification task: The labels are included in the training data and the goal is to train a model to learn to predict the labels from the features.The label is a binary variable, 0 (will repay loan on time), 1 (will have difficulty repaying loan).

The graph above shows the relationship among the data sources. There are totally 221 different variables in all the files. The main data of this competition is given by application_{train|test} file.

After merging the bureau_balance.csv to the bureau.csv, there is an ID column in each sub csv files showing the relationship to the main data. Most of them have several records for a single ID and different persons have different numbers of records which is a hard work for us to merge the data to the main one. Then, we decided to take the average of the records for each person and generate a new column to save the number of records. For category variables, we generate dummy variables and then get the average by each column for each person. After doing so, the dummy variables will become the rate of the original dummy variables for different person.The weakness of this treatment is that we do not take all the relationship among variables into consideration. Finally, we get 382 variables, which are enough for the rest of the work in this project.

3. Important Features

3.1 Feature Selection

Before doing feature selection, we remove any columns built on the SK_ID_CURR, because SK_ID_CURR is a unique identifier for each client, we would not want to build a model trained on this "feature".

For original data, we have 382 features. More features might seem like a good thing, and they may help our model learn. However, irrelevant features, highly correlated features, and missing values can prevent the model from learning and decrease generalization performance on the testing data. Therefore, we prefer to keep the most relevant variables. We will use three methods for feature selection: Remove collinear features; Remove features with greater than a threshold percentage of missing values; Keep only the most relevant features using feature importances from a model.

Collinear variables are those which are highly correlated with one another. Removing collinear variables is a useful step to increase the accuracy of modeling. We plan to remove features by the pearson correlation coefficient is greater than 0.8. Then, a relatively simple choice of feature selection is removing missing values. In this implementation, if any columns have greater than 75% missing values, they will be removed. Also, most models cannot handle missing values, so we will use median to replace missing values in remain features. The last method we can employ for feature selection is to use the feature importances of a model. We want to remove all zero importance features from the model. If this leaves too many features, then we can

consider removing the features with the lowest importance. We will use Random Forest to assess feature importances.

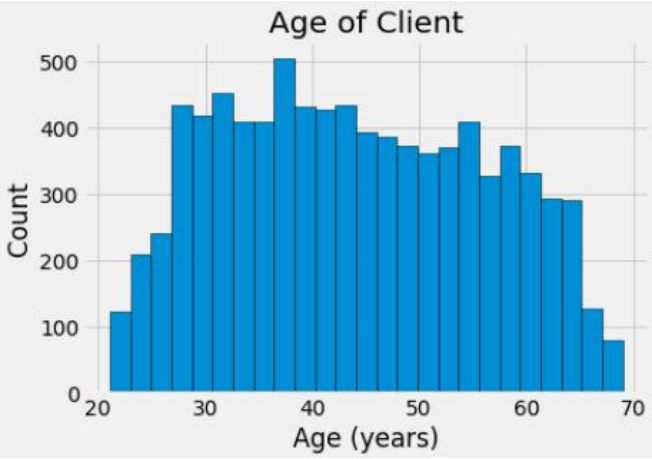
Finally, by comparing these three methods, we select a subset of the features that can fit the tree-based models well. We decrease the number of features from 382 to 238. And the top most important features are : 'EXT_SOURCE_1', 'EXT_SOURCE_2', 'EXT_SOURCE_3', 'DAYS_BIRTH'.

3.2 Data Exploration

From above, we can find that the important features are 'EXT_SOURCE_1', 'EXT_SOURCE_2', 'EXT_SOURCE_3', 'DAYS_BIRTH'. Then, we want to analyze those features' relationship with our TARGET y.

Effect of Age

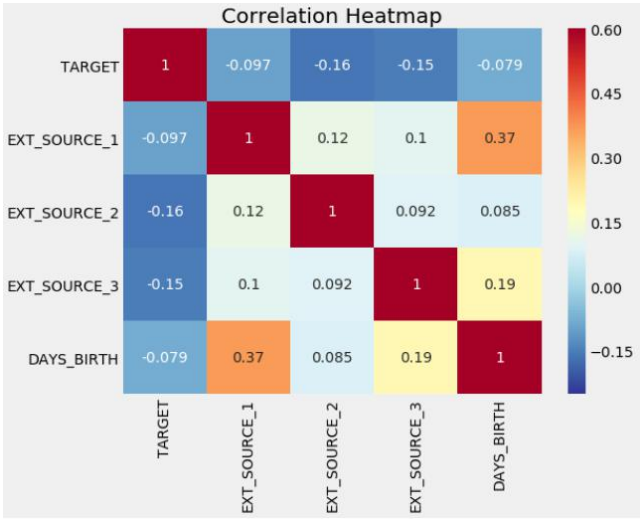
We can observe: as the client gets older, there is a negative linear relationship with the target meaning that as clients get older, they tend to repay their loans on time more often. Then, from the histogram of the age, the distribution of age tells us that there are no outliers as all the ages are reasonable. To visualize the effect of the age on the target, we will next make a kernel density estimation plot (KDE) colored by the value of the target. The" target = 1" curve skews towards the younger end of the range. Although this is not a significant correlation (-0.082 correlation coefficient), this variable is likely going to be useful in a machine learning model because it does affect the target.



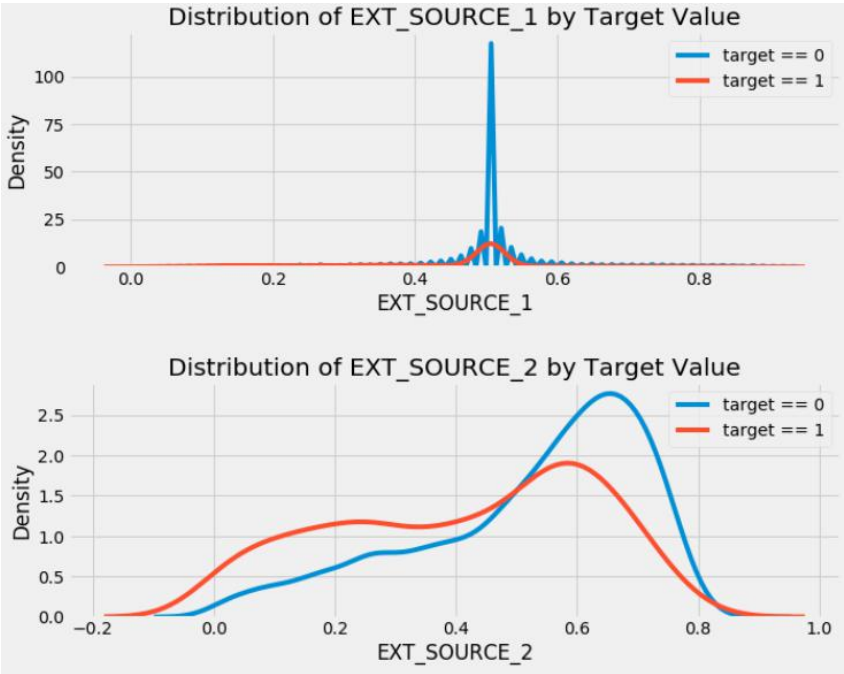
Exterior Sources

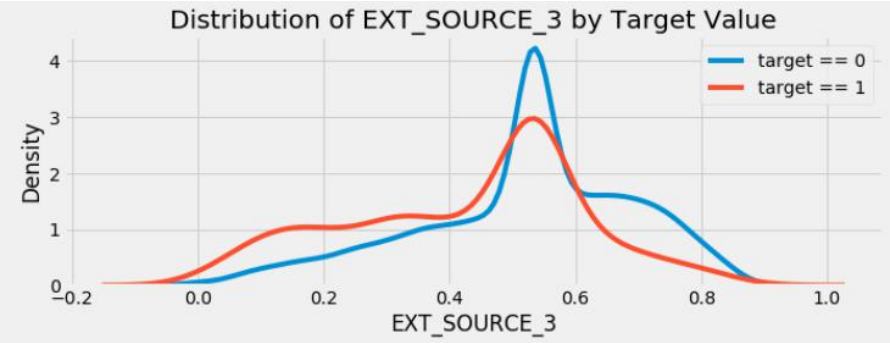
The 3 variables with the strongest correlations with the target are EXT_SOURCE_1, EXT_SOURCE_2, and EXT_SOURCE_3. According to the original data documentation, these features represent a "normalized score from external data source". By the correlations of the EXT_SOURCE features with the target and with each other, we find that all three EXT_SOURCE features have negative correlations with the target,

indicating that as the value of the EXT_SOURCE increases, the client is more likely to repay the loan. We can also see that DAYS_BIRTH is positively correlated with EXT_SOURCE_1 indicating that maybe one of the factors in this score is the client age.



Next, we can look at the distribution of each of these features colored by the value of the target. This will let us visualize the effect of this variable on the target. We can clearly see that this feature has some relationship to the likelihood of an applicant to repay a loan. The relationship is not very strong, but these variables will still be useful for a machine learning model to predict whether an applicant will repay a loan on time.





4. Model fitting and parameter choosing

Here we use four algorithms to fit the data. They are Logistic Regression, Random Forest, Gradient Boosting and Ada Boost. We use cross validation to do parameter selection. Specifically, the scoring we use in cross validation is the mean AUC and we expect it to be as large as possible. Because some defaults of the parameters are good enough, we only do the selections for necessary ones.

For Logistic Regression, we do not use the sklearn function to select the best parameter because Logistic is relatively easy to compute and we use our own code to get the best parameter. The parameter we need to choose is “inverse of regularization strength”, the result is 0.421875. Because Logistic is a previous courses method and it is a linear classifier, the result from the logistic regression is the base for the next part, the comparison of these four models.

For Random Forest, the parameter we need to choose is “the minimum number of samples required to split an internal node” and “number of trees in the forest”. However, the optimal results are not stable, here we choose the most frequently appeared ones, they are 80 and 100 respectively.

For Gradient Boosting, the parameter we need to choose are “the number of boosting stages to perform”, “maximum depth of the individual regression estimators”, “minimum samples split” and “minimum samples leaf”. As we need to select many parameters, we need to deal with them step by step. More specifically, we start with selecting “the number of boosting stages to perform” and set other parameters to default. When we find the optimal value, we set this parameter and choose next parameter. Finally, we find “the number of boosting stages to perform” is 180. As for max_depth, we just choose 7 as the best choice. Also, the best min_samples_split is 1000,which is the boundary of our setting.Thus, we still need to boarder our range. We also take the min_samples_leaf into consideration. And 'min_samples_leaf'= 100, 'min_samples_split'= 1400 is the best.

For Ada Boost, the running time take so long that we only choose “the maximum number of estimators”. The result is 140.

5. Model performance comparison

After parameter selection, we compare the performance of the prediction of four methods by the area under the curve which is suggested by the competition organizer. The conclusion is that AdaBoost is the best (AUC: 0.7125), then comes Random Forest (AUC: 0.7070) and Gradient Boosting (AUC: 0.7057), and logistic (AUC: 0.6670) is the worst. The result is not surprising, because Logistic Regression is a linear classifier and the rest three are non-linear classifiers. The AUCs of the prediction result for four algorithms are shown is the code part.

6. Further improvement

As time is limited, we only use part of the total data set to fit the model. As the original plan, we will use the whole data set to fit. Done is better than perfect in this situation so that we make a concession to sample the data. We also do not analyze the influence of changing the number of the data.

The span of the parameter choosing part is large. More precise the parameter we choose may lead to a better performance. It is also a time issue here.

According to discussion in the competition forum, the methods in this project do not seem to have much potential to improve. We need the improvement on the algorithm instead of finding the best parameters.