Final Report: predictive modelling for default detection

Introduction

We are now surrounded by data everywhere all the time and want to extract useful and meaningful information from them. We have data representing applications of the loan to a loan agency and the applicants' historical transaction data from the agency itself, other credit card agency and credit bureau as well. The historical data might be an important barometer for the agency to learn the applicant's behaviour to predict the capability and prospect of their repayment in the future. Those data also could help the applicants who have no history of previous application to approve their credible attitude towards the future repayment scheme.

Data are stored in six files in comma separated values, and the total size of data reaches to 2.5GB for over 300000 applicants [1]. Explanatory data analysis (EDA) is focused on missing data management, abnormality detection and correlation of each pair of variables to learn about which predictors are highly correlated to the target variable (a binary variable indicating well-repay or difficult) and which are strongly correlated to each other. We prefer to retain predictors with a high correlation to target and remove some predictors with a strong correlation to each other as our goal is to build the simplest model under the condition of predicting target value accurately as possible as for unseen future data. We also aggregate each the historical data and join these with the application data. At the end of this exercise, we have a clean but high-dimensional data set.

Modelling process consists of encoding for categorical variables, standardisation of values, hyper-parameter tuning with cross-validation, feature engineering using several different algorithms and scoring using Area under Curve (AUC) of Receiver Operating Characteristic (ROC). To test the models from hyper-parameter tuning and feature engineering, we split data into validation and hold-on sets in advance.

Exploratory data analysis

There are a range of EDA activities in this project – missing data imputation for all datasets, abnormality detection and management for all, missing indicator variable implementation for the application data, encoding for the categorical variables, reviewing the trend with time of the credit card balance, extracting the correlation matrix for each dataset and aggregating the historical data by application ID. Keeping in mind that the EDA is purposed to prepare data for the modelling and find any relevant and interesting features, we simplify the findings focusing on the modelling. Table 1 represents the summary of the dimension of data, missingness and analytical process.

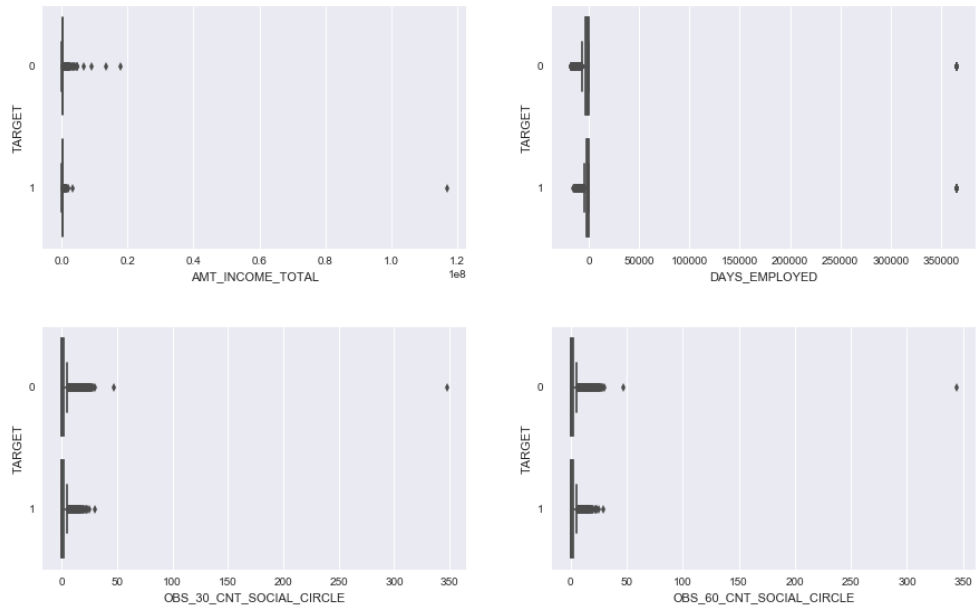
Table 1: Summary of EDA activities for each dataset

Dataset	Num. of rows	Num. of columns	Max. NA %	NA indicators	Outliers	Encoding	Aggregation
Application	307511	122	69.9	V	V	V	
CC balance	3840312	23	20.0				V
Bureau	1716428	17	71.5				V
Bureau balance	27299925	3	11.4			V	V
Instalments	13605401	8	0.0				V
POS balance	10001358	8	0.3				V

Application data

There are 8.1% of applications classified as a possible default, or TARGET = 1, 34% of male applicants and 69% of homeowners in the application dataset. We detect few erroneous outliers in variables of annual income, length of employment in days and the number of social circles – one data point represents about 1.2e8 as annual income, two data points over 360000 days (or 1000 years) of employment and 360 social circles in 30 days and in 60 days as shown in Figure 1.

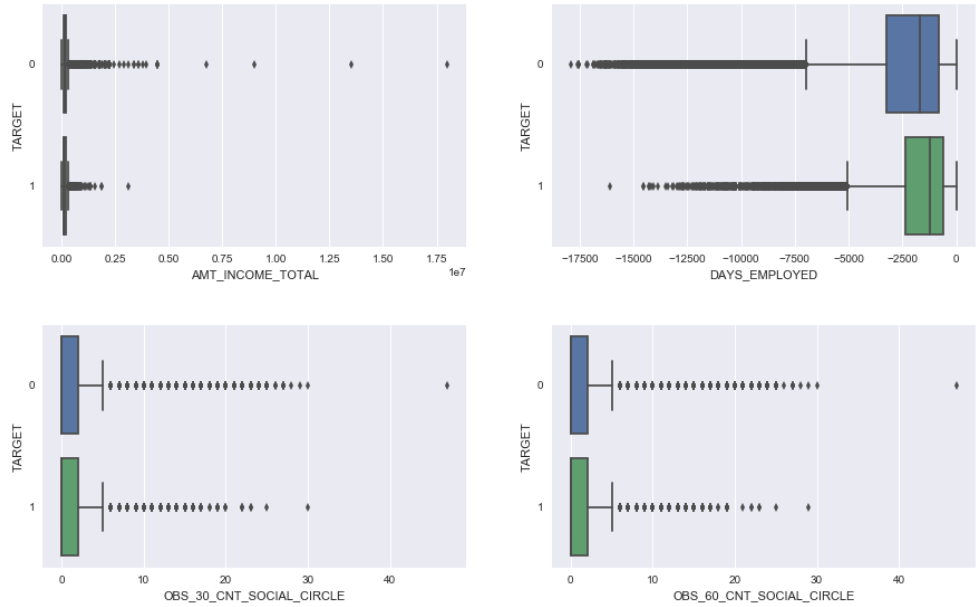
Figure 1: Outliers in annual income, employment length and social circles in 30 days and 60days



These values seem to be obvious errors based on logical judgement; it is impossible to be an employee for 100 years or to earn 120 billion dollars a year, and it does not likely exercise 360 social circles in 60 days comparing the other values. But we neither know the true values nor the reason for that. So, we address the abnormal values with replacing NA and figure the four variables after taking off those outliers in boxplots again as shown in It is interesting to see that how the feature selection results on these variables.

Figure 2; distributions of values are clearer – there seem to be differences between two groups of TARGET in the annual income and length of employment though they do not seem significant. It is interesting to see that how the feature selection results on these variables.

Figure 2: After removing extreme values in annual income, employment length and social circles



Another atrocious character of the application data is missingness; there are missing data in 65 variables and as high as 70 % in some variables as shown for the top ten missing variables in Table 2. In this table, the first column represents column name, the second the percentage of missing data, the third the TARGET percentage of missing data and the last the TARGET percentage of non-missing data. Recalling the population TARGET of 8.1%, there seems a gap between missing and non-missing groups in TARGET percentages.

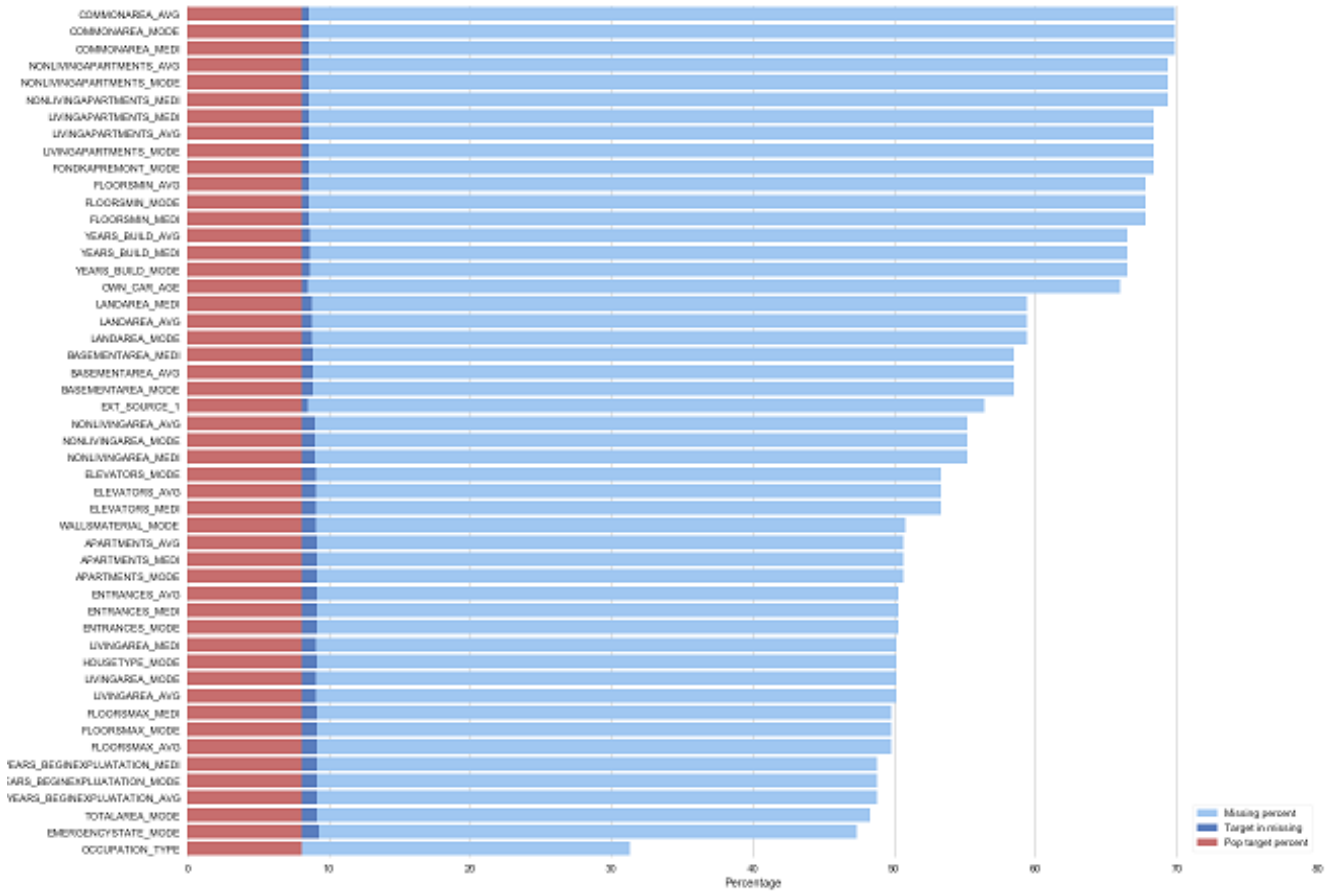
Table 2: Table of top ten missing variables

	variable	na_percent	na_target_percent	no_na_target_percent
61	COMMONAREA_MODE	69.9	8.6	6.9
75	COMMONAREA_MEDI	69.9	8.6	6.9
47	COMMONAREA_AVG	69.9	8.6	6.9
83	NONLIVINGAPARTMENTS_MEDI	69.4	8.6	6.9
55	NONLIVINGAPARTMENTS_AVG	69.4	8.6	6.9
69	NONLIVINGAPARTMENTS_MODE	69.4	8.6	6.9
53	LIVINGAPARTMENTS_AVG	68.4	8.6	6.9
67	LIVINGAPARTMENTS_MODE	68.4	8.6	6.9
81	LIVINGAPARTMENTS_MEDI	68.4	8.6	6.9
85	FONDKAPREMONT_MODE	68.4	8.6	6.9

Figure 3 shows the percentage of missing in light-blue coloured bars for the most missing 50 variables. Overlapping on the long light-blue bar, the dark-blue coloured bar indicates the TARGET percent of

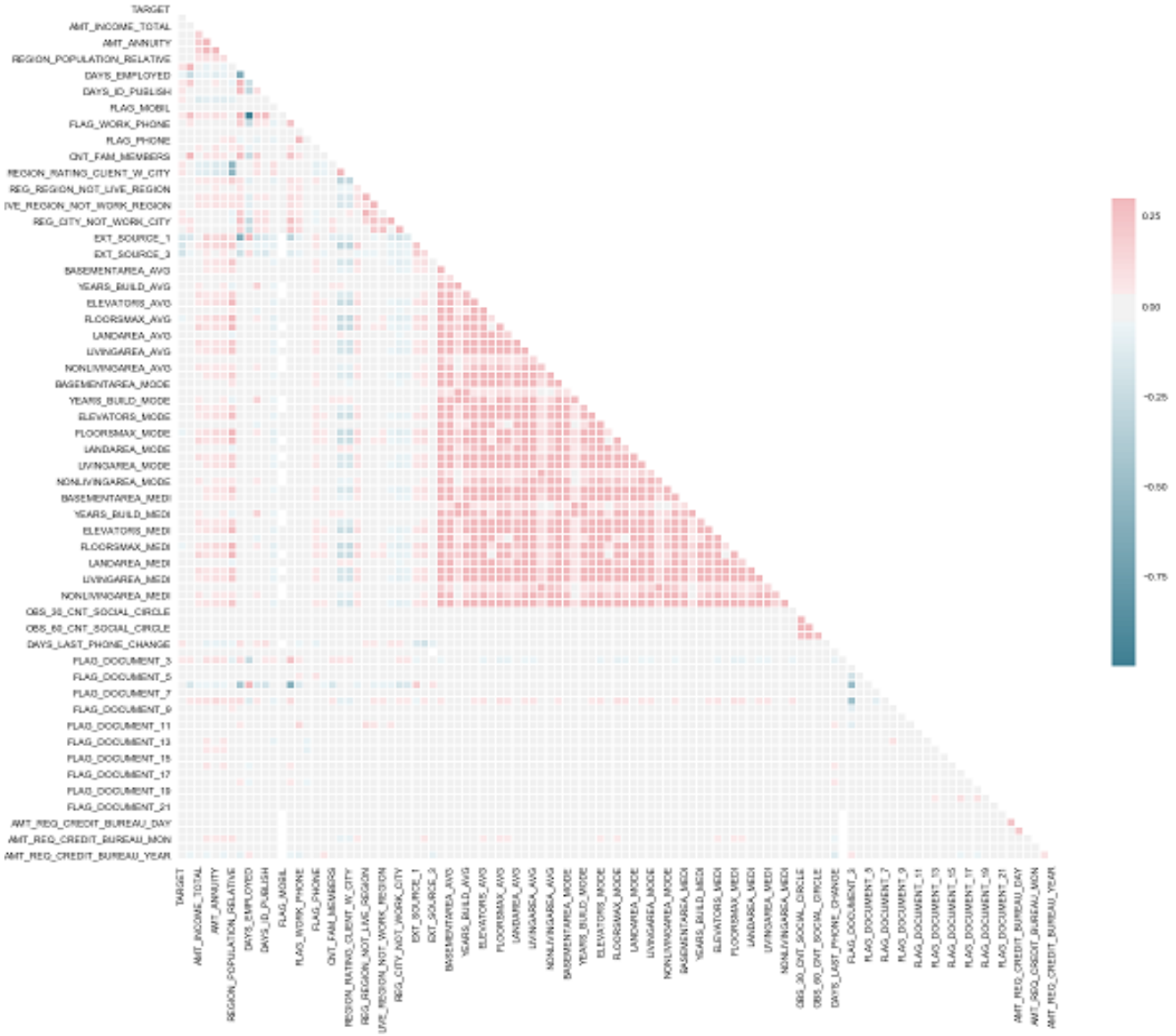
the missing group and the brown coloured bar the TARGET percent of the non-missing group for each variable. From the visual assessment, we want to perform Chi-square tests to confirm the significance of the differences. The result of the tests informs us that there are compelling gaps between the missing and non-missing groups in TARGET percentages for 63 variables among 65 variables with missing. We implement indicator variable for these 63 variables for the modelling. We also implement missing values with their median value for each column.

Figure 3: Missing percentages with TARGET percentages in missing and on-missing groups



We assume that predictors having a strong correlation with TARGET would help to build a reliable model, and adversely, any predictors having a strong correlation to each other would not be helpful. As our target variable is binary it is hard to expect a strong correlation with any predictor – for example, the target variable has the strongest correlation coefficient with the variable of Document_6 as strong as -0.18. On the other hand, the most robust correlation between two predictors values 0.99 for the Credit-amount and Goods-price. It is interesting to see that whether both of these variables would be selected, or one or both of them would be eliminated from the feature engineering process – let we check later.

Figure 4: Heatmap of the correlation coefficient matrix



Credit card balance data

This data set is quite clean with the maximum of missingness as 20% - it is reasonable to interpret the missing values as no histories, and so we impute these as zeros. We aggregate the data by application ID and also by backwards-time of the records in months; aggregation by ID help to understand characteristics of data in the aspect of individuals, while the timewise aggregation help to find any trends in mean balance, mean draw counts and so on. Overlooked features of columns are displayed in Figure 5; the diagonal shows the univariate distributions in histograms and off-diagonal the bivariate distributions in scatter plots. The histograms imply that the values are heavy at one bin and very sparse at other bins, but there are a couple of interesting scatter plots showing some correlation between two variables. From the close look at Figure 6, we interestingly discover that the people who have more records actually show less mean balance, and ones with larger mean balance have the bigger amount of total receivables.

Figure 5: Pair plots of aggregated data of credit card balance by ID

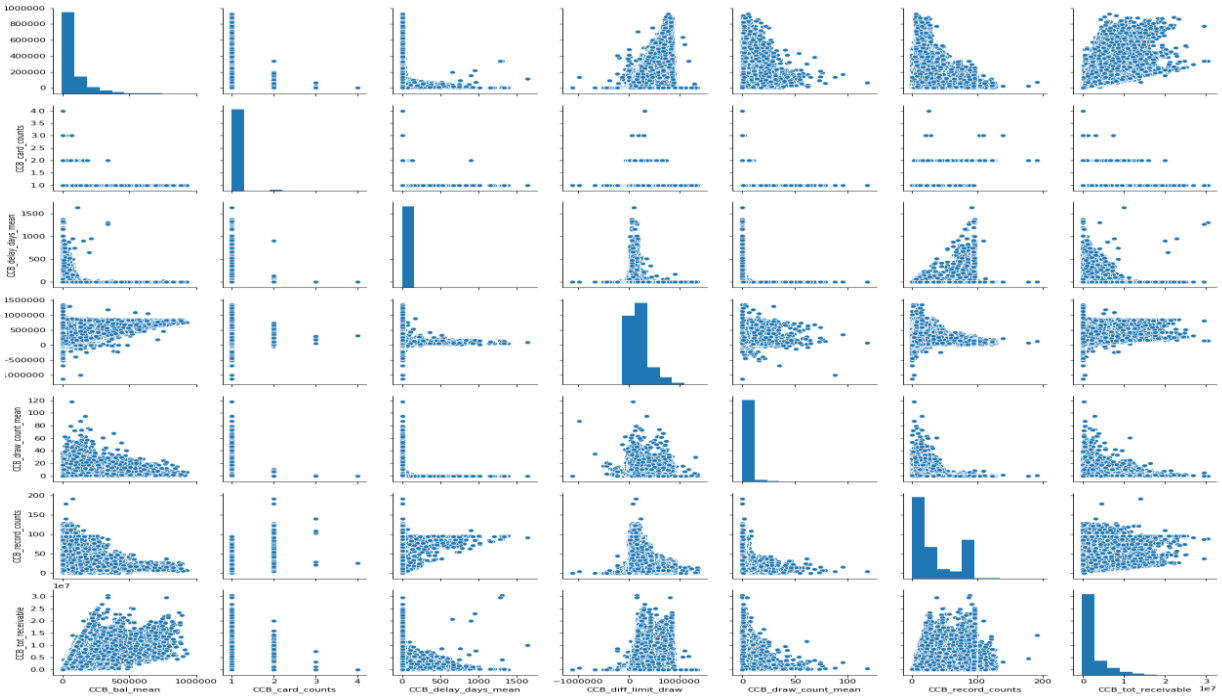
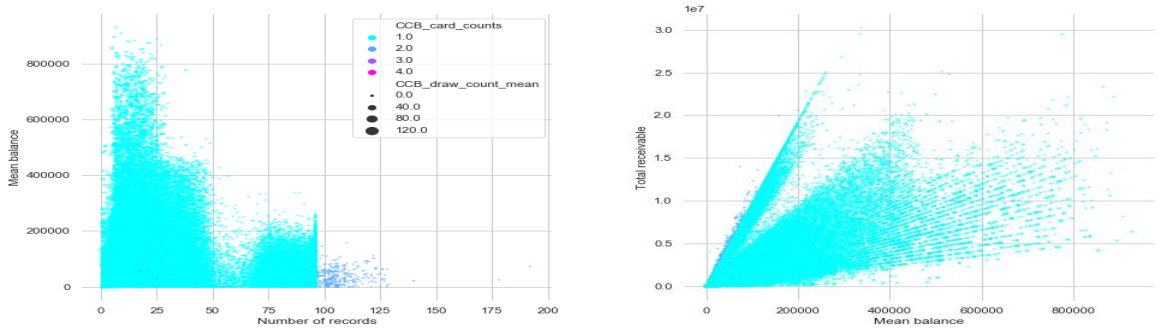
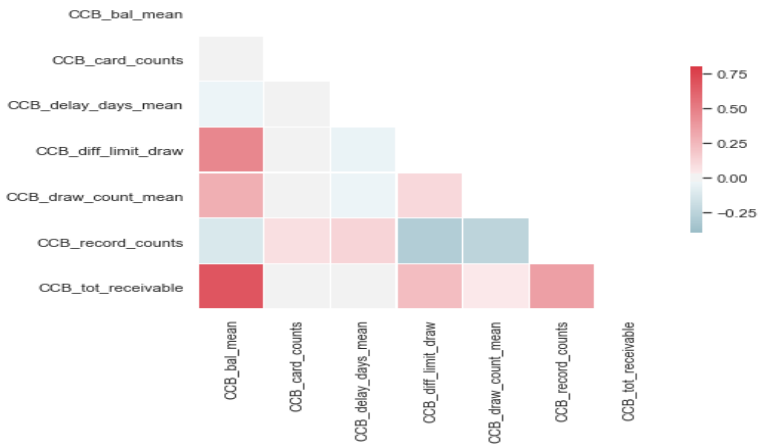


Figure 6: Scatter plots of (1) mean balance vs. record counts and (2) total receivable vs. mean balance



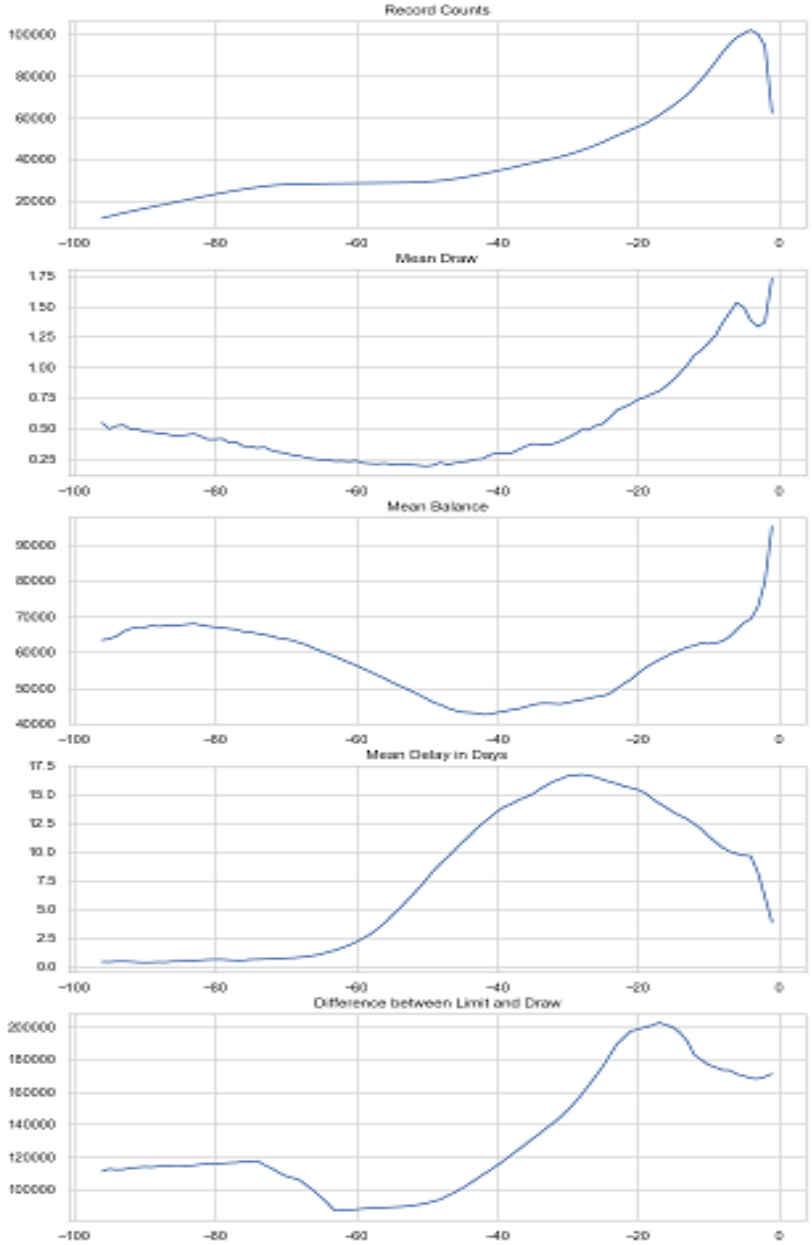
The variables of total receivables and mean balance have a moderately strong correlation as shown in Figure 7, with the coefficient of 0.67.

Figure 7: Heatmap of the correlation matrix



From the timewise aggregation, we display time series plots for few variables as shown in Figure 8: The first plot shows that there is an increasing trend of the number of records until about five months before the applying a loan. The plot of the mean of monthly draws represents a hyperbolic curve until the recent time. Interestingly, the mean of monthly balances has a similar trend with the second plot of the mean monthly draws, except a bit of wavy curve at the early time with having a peak at the time of -85. Comparing these, the plot of the difference between the limit and draw has a curvier shape - this one starts to rise earlier at about -50 and starts to drop from just after -20. On the other hand, the mean of delays in days flats until -60, then shapes parabolic, which has a peak at about -30. From these plots, we can conclude that mean balance seem to dramatically increase from the time of a year before applying for a loan, and the mean of delays seem to drop from the time of two years before applying.

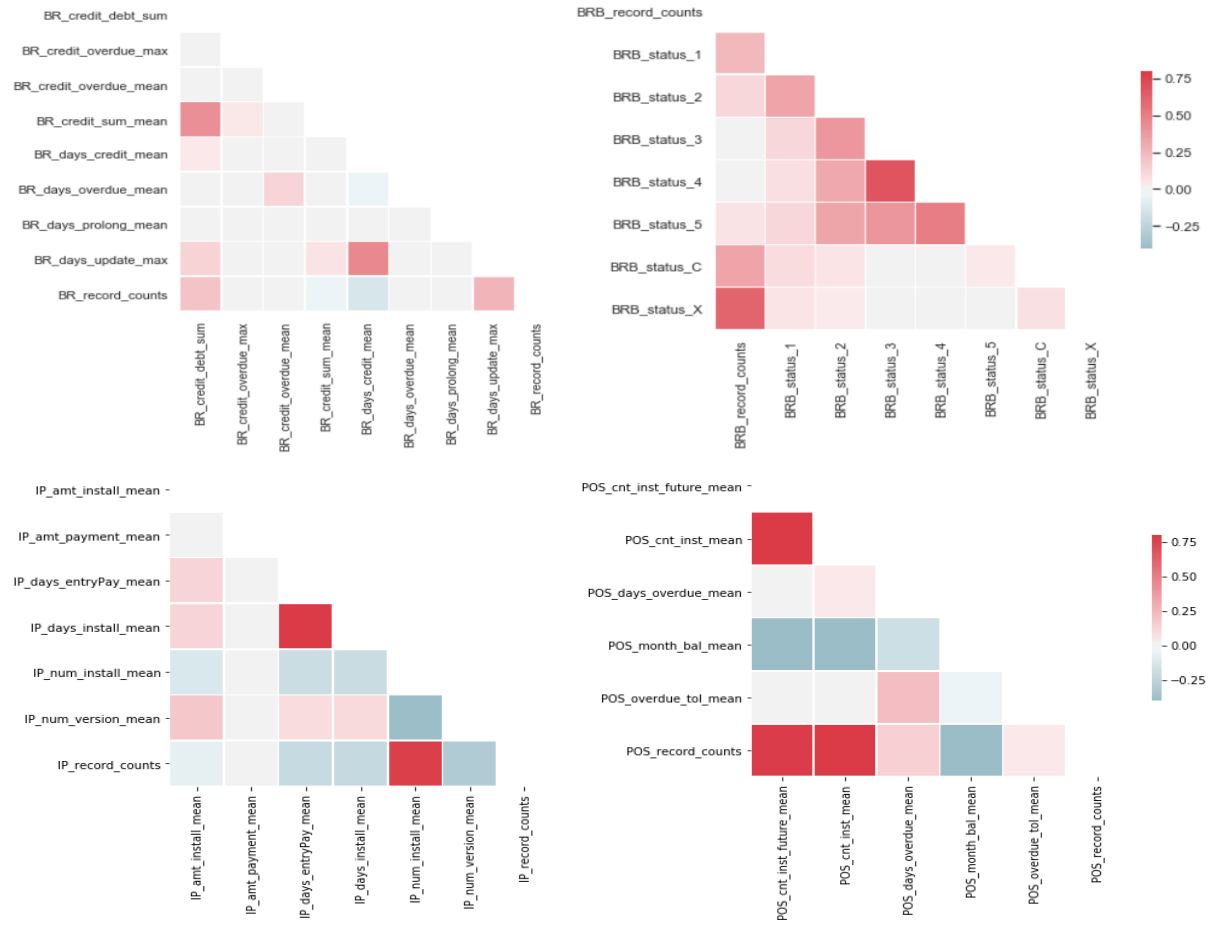
Figure 8: Plot of record counts, draw counts, balance, delays and difference between limit an draw by time



Other datasets

For other datasets, we perform the similar procedure to the credit card balance data – missing data management, derive correlation matrix, encoding and aggregation. Here we display their correlation matrix heat maps to learn some interesting relationships of any pairs. The first map of Bureau data shows very plain hues in almost all cells implying that any pair hardly has an association with each other. The second plot of bureau balance represents moderate correlations between some pairs such as the pair of Status 3 and Status 4 as well as one of Record count and Status X. On the other hand, instalment payments data has shown negative correlation though they look quite weak, and also strong positive correlations for the pair of the entry pay mean days and Install mean days with correlation coefficient of 0.99 - of course, they do. The last figure of POS balance shows vivid and opposite colours: Strong positive correlations are for the pairs of (instalment count, future instalment count), (instalment count, record count) and (record count, future instalment count), and strong negative correlations are for (record count, monthly balance). We are interested in the feature engineering stage to acknowledge how they are fit in the model, so we now gather these all aggregated variables.

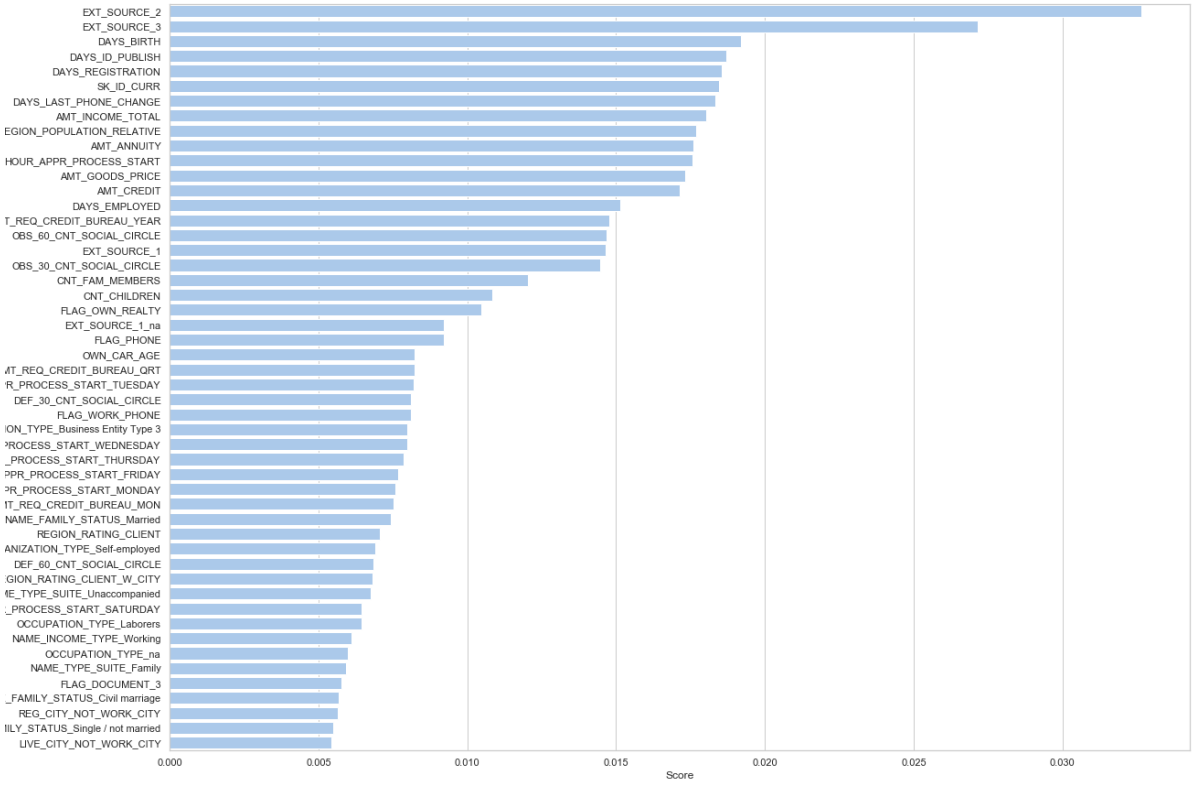
Figure 9: Correlation matrix heatmaps for (1) bureau, (2) bureau balance, (3) instalment and (4) POS balance



Feature engineering

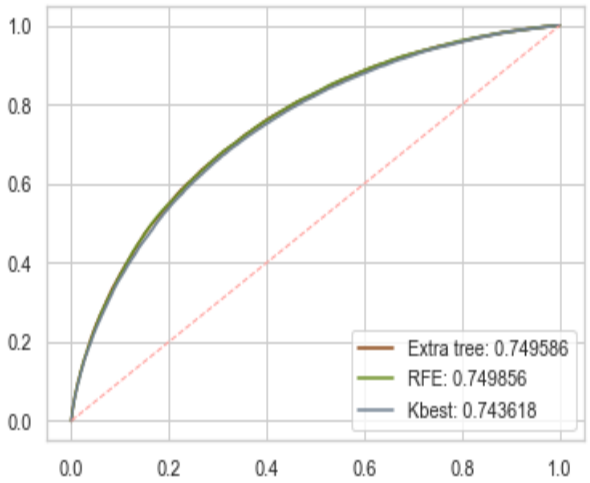
We are interested in a number of different algorithms for feature engineering to simplify the model under the criteria of getting the best ROC AUC score for the unseen test data – we hold 30% of data for the test and use 70% of data for the modelling and validation. We consider Extra Trees Classifier, Recursive Feature Elimination (RFE) and Select K Best. As we do not know that how many features are appropriate to test these three methods, we use Extra Trees Classifier first that are not required to set the number of features in advance. Then make a decision on the number of features satisfying the score of equal to or greater than 0.002. We use only application dataset to save computing time here and apply the algorithm selected from this exercise to the whole datasets in the next chapter. We have the top score of 0.032 for EXT_SOURCE_2 and the second of 0.028 for EXT_SOURCE_3 from Extra Trees Classifier and select 137 predictors having the score of over 0.002 - Figure 10 shows the top 50 scores from this algorithm. We set 137 as the number of features for the other two algorithms to see any differences in selection and to compare the AUC scores from the logistic regression algorithm as well.

Figure 10: Top 50 features selected from Extra Trees Classifier



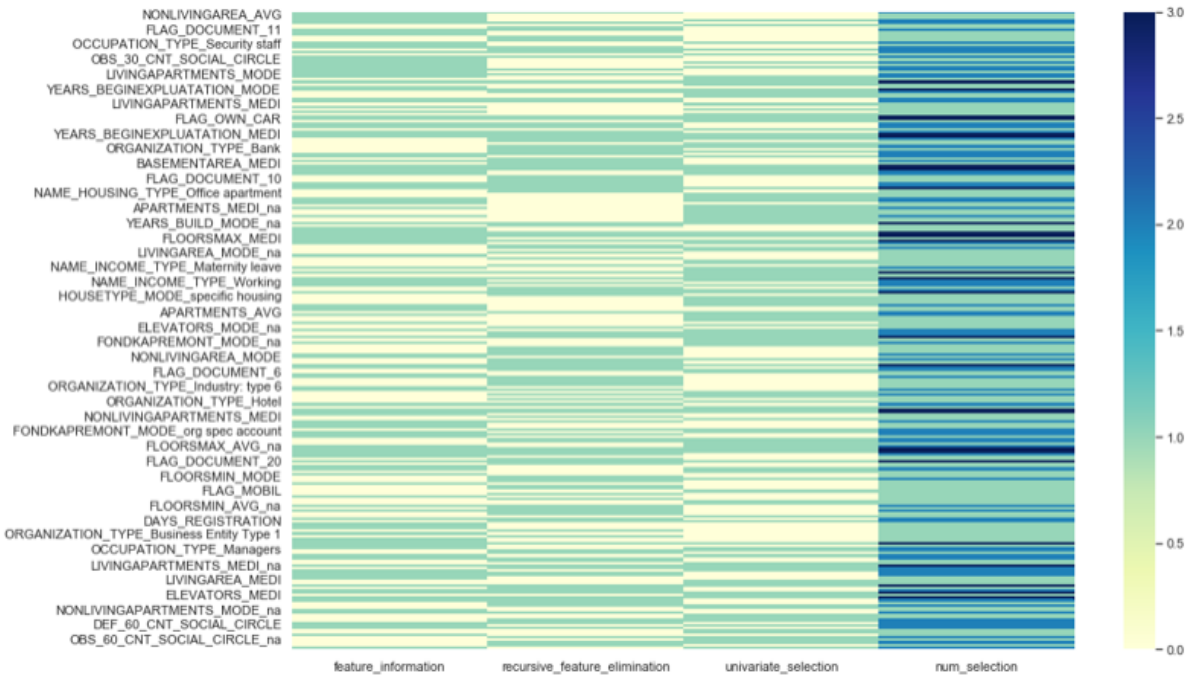
From this exercise, we have AUC scores of 0.74972, 0.74998 and 0.74363 from consider Extra Trees Classifier, RFE and Select K Best, respectively as shown in Figure 11 –REF has the best score, but it is computationally expensive and would be more for the whole data, regarding the tiny bit of gain in score - so we use Extra Trees Classifier as feature engineering tool to model the prediction of TARGET for the combined data of application and historical records in the next chapter.

Figure 11: ROC curves from three featuring engineering algorithms



There are differences among those three methods in selecting features as shown in Figure 12: Though all the labels are not shown on the left side of the plot, there is the total number of 257 features that are chosen from at least one algorithm and 31 features chosen commonly in all three methods.

Figure 12: Features selected from three algorithms



Interestingly, the list of 31 commonly selected features includes gender, some occupation types, region rating, number of external sources and a couple of NA indicators.

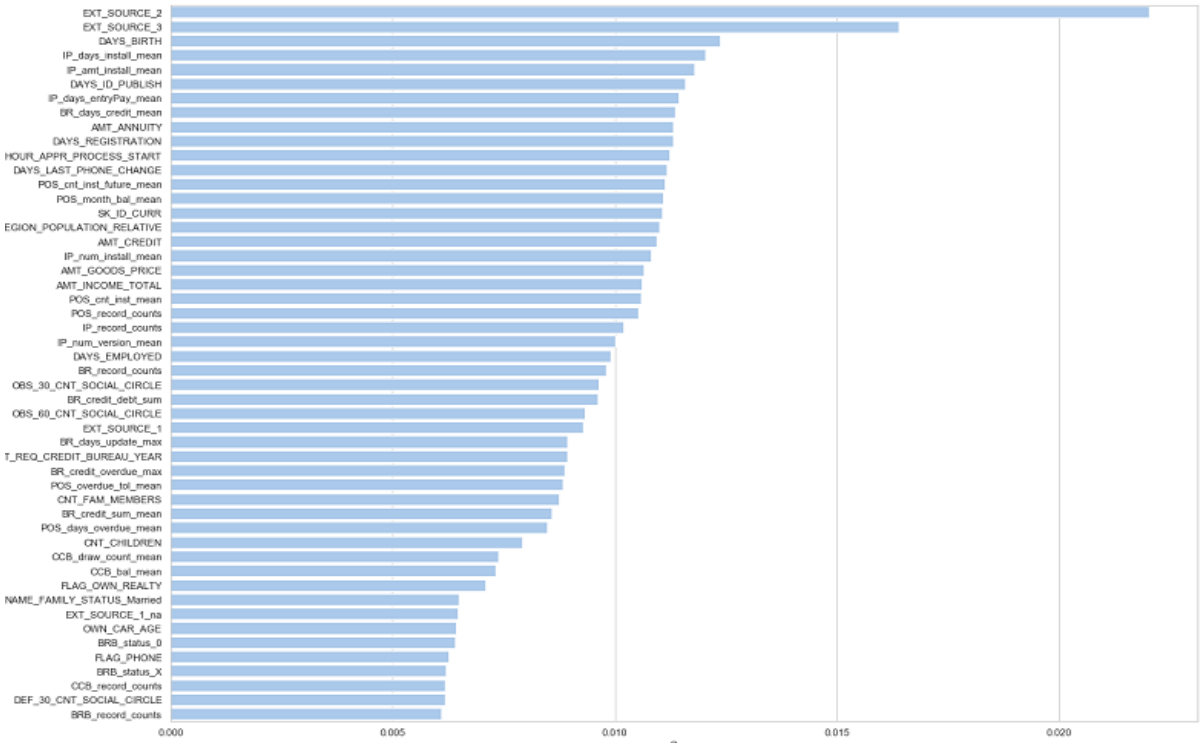
Parameter tuning for logistic regression

We extend the logistic regression algorithm to the merged data with hyper-parameter tuning as well as the feature engineering. As the preparation of the modelling, application data and aggregated data are combined by application ID and fill missing with zeros, standardised and split the data into train

and test sets in advance. We apply the hyper-parameter tuning and feature selection in multi-steps together; cross-validation for parameter tuning first, then feature engineering, do parameter tuning again, and so forth. The reason for doing so is to learn about whether the multiple hyper-parameter tunings produce different results for the reduced number of features in modelling. The first model is a logistic regression for the merged data without hyper-parameter tuning or feature engineering – the AUC score of this model is 0.7520 which is a better score than any of ones from the models for only the application data. We use RandomRearchCV module from sklearn.model_selection with five cross-validations to tune hyper-parameters of penalty (l1, l2), tol (1e-3, 1e-4, 1e-5) and C (0.01, 1, 100) for the logistic regression model. From the first cross-validation before feature engineering, the parameter tuned the penalty as l2, tol as 1e-5 and C as 0.01, and the AUC score of this model is 0.75365 for the test set.

To reduce the number of predictors without much loss of information, we apply ExtraTreesClassifiers module from sklearn.ensemble that chosen from the previous chapter, and its outcome is shown in Figure 13: We have the top score of 0.023 for EXT_SOURCE_2 and the second of 0.017 for EXT_SOURCE_3 from Extra Trees Classifier, which are the same top two as the feature engineering for the application data. However, there are a large number of aggregated data included in the top 50 of feature importance score for the whole dataset.

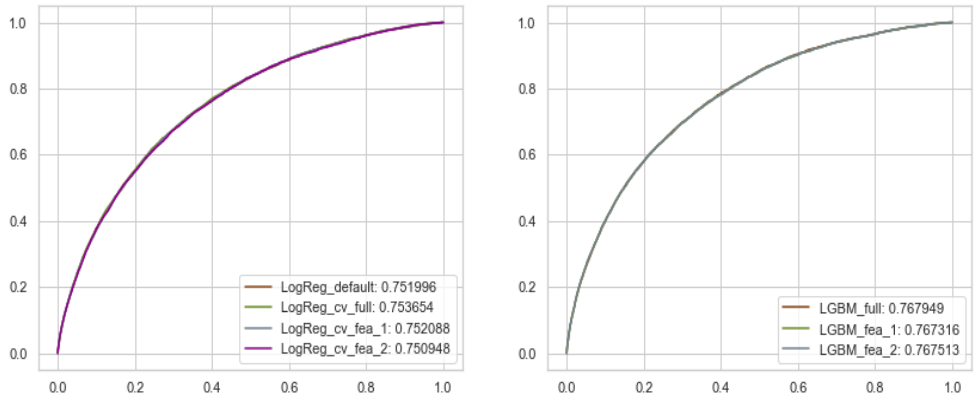
Figure 13: Top 50 high scored features using ExtraTreesClassifier



We select 206 features with the importance score of over 0.001 and apply Grid Search CV again – it extracts the same parameter tuning result as the first one. The AUC score of the logistic regression for the test data is worse as 0.75201. When we choose 152 features with the score greater than 0.002,

the grid search cross-validation gives us a different value of parameter penalty as l1, but the AUC score of the model for the test data decreases to 0.75095. As a result, we can conclude that the best logistic regression model with the hyper-parameters chosen from the grid search cross-validation for all predictors have the highest score of 0.75365. The first plot of Figure 14 shows the ROC curves for these four different logistic regression models.

Figure 14: ROC curves of logistic regression models with different numbers of features



Light gradient boosting machine models

Data scientists with experience recommend that the Light Gradient Boosting Machine (LGBM) is a fast and powerful algorithm, especially for data with a sparse target variable like a rare disease, fraud, spam-email or loan default detection [2]. We would like to apply this algorithm to the data with the default setting of hyper-parameters, but with different numbers of predictors – using the whole features (328 features), features form Extra Trees Classifier score over 0.001 (206) and ones with over 0.002 (152) as the same set of feature engineering as the earlier models of logistic regression. The outcome confirms that the LGBM works better than the logistic regression for our data, and the best model seems to consist of all features with AUC score of 0.76795 as shown in the second plot of Figure 14 and Table 3.

Table 3: Logistic Regression and LGBM model scores

Model	Number of features	Score
Logistic: default	328	0.751996
Logistic: cv5 for all features	328	0.753654
Logistic: cv5 for 206 features	206	0.752088
Logistic: cv5 for 152 features	152	0.750948
LGBM: all features	328	0.767949
LGBM: 206 features	206	0.767316
LFBM: 152 features	152	0.767513

Interestingly, all the pairs of features with strong correlations such (entry pay mean days, Install mean days) from instalment payment data and (instalment count, future instalment count), (instalment count, record count), (record count, future instalment count) and (record count, monthly balance) from POS balance data are selected together from feature engineering process with the threshold of 0.002 as well as 0.001.

Discussion

We drilled a variety of exercises in this project of the predictive modelling for the loan application data: There was a serious issue in missing values in the application dataset – we addressed this problem using imputation and implementation of the missingness indicators for the variables with the significant difference in target values between missing and non-missing groups. We aggregated historical data by application ID, merged with the application data and encoded categorical variables – as a result, we had the data with 328 features in total. We applied three different feature engineering algorithms for the logistic regression on the application data; the extra trees classifier gave us the competitive outcome in computation and score. With this featuring engineering tool, we exercised hyper-parameter tuning of logistic regression for the whole data, then performed LGBM. All of the modelling processes used 70% of data for train and 30% for the test. From these activities, we acquired the best AUC score of 0.76795 at the LGBM model with the full features.

There were some limitations on this project: As a lack of expertise on loan business, we could not apply more sophisticated aggregation methods such as timewise weight or specified missing imputation methods. We also did not apply the hyper-parameter tuning for the LGBM as there are a large number of parameters and so computationally expensive. We are quite sure that we gain better outcomes if we improve these deficiencies in another project in the future.

References

[1] https://www.kaggle.com/c/home-credit-default-risk/data

[2] https://lightgbm.readthedocs.io/en/latest/Parameters-Tuning.html