Capstone Project

Report

Machine Learning Nanodgree

Minxia Ji

-Project Overview-

Background:

When a customer accept a credit card, he or she needs to agree to certain terms, such as make minimum payment by the due date listed on theircredit card statement. If the customer missed the minimum credit card payment six months in a row, his or her credit card will be in default. The credit card issuer will likely close the customer's account and report the default to the credit bureaus. But to the credit card issuer(usually a bank), the lost might be irretrievable.Eventually, usually
after a period of 90 days of nonpayment, the loan/payment is written off. Banks are required by law to maintain an account for loan loss reserves to cover these losses. Banks could reduce credit risk by conducting a credit risk analysis on credit card
applicants/holders. Banks can substantially reduce their credit risk by lending to their customers, since they have much more information about them than about others, which helps to reduce adverse selection. Checking and savings accounts can reveal how well the customer handles money, their minimum income and monthly expenses, and the amount of their reserves to hold them over financially stressful times. Banks will also verify incomes and employment history, and get credit reports and credit scores from credit reporting agencies.

Reference:

Bank Risks
http://thismatter.com/money/banking/bank-risks.htm
Banks that make the most money, and the least, on credit card loans
http://www.creditcards.com/credit-card-news/bank-yields-loans-1276.php

Dataset and inputs

Data:default of credit card clients Data Set

Data Source:UCI https://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients

Overview of the dataset:

3000 instances and 23 features are included in the dataset. Most of features are numerical features, i.e. bill statement, and we need to check the skewness and may need to do some transformation. Some of features, which represent customer information, i.e.gender, are categorical. We need to transfer variables under these features to dummies variables.

Deal with class imbalance: try both random undersampling the majority class and oversampling minority class

Problem statement:

Inputs:

3000 instances with 23 features and one label are included in the dataset. Most of features are numerical features, i.e. bill statement. Some of features, which represent customer information, such as gender and marriage, are categorical.

Output:

a trained model which performed well on predicting whether a customer would default or not given his/her information. Learning task:

Binary classification: detecting the clients who are likely to default or not(default:1;not default:0)

Solution statement:

Data:

-Deal with class imbalance: try both random undersampling the majority class and oversampling minority class

-Splitting data: training set: 80% testing set 20%. In order to maintain class balance in training and testing sets, use train_test_split function in sklearn and specify a param ‘stratify’.

-Check the skewness of features and deal with it.(i.e.log transformation.)

Models:

-Classification models: employ RandomForestClassifier, GradientBoostingClassifier and LogisticRegressionClassifier.

-Train and find a best classifier and optimize it using grid search method

-Conduct predicting default based on new dataset using the classifier

A set of evaluation metrics

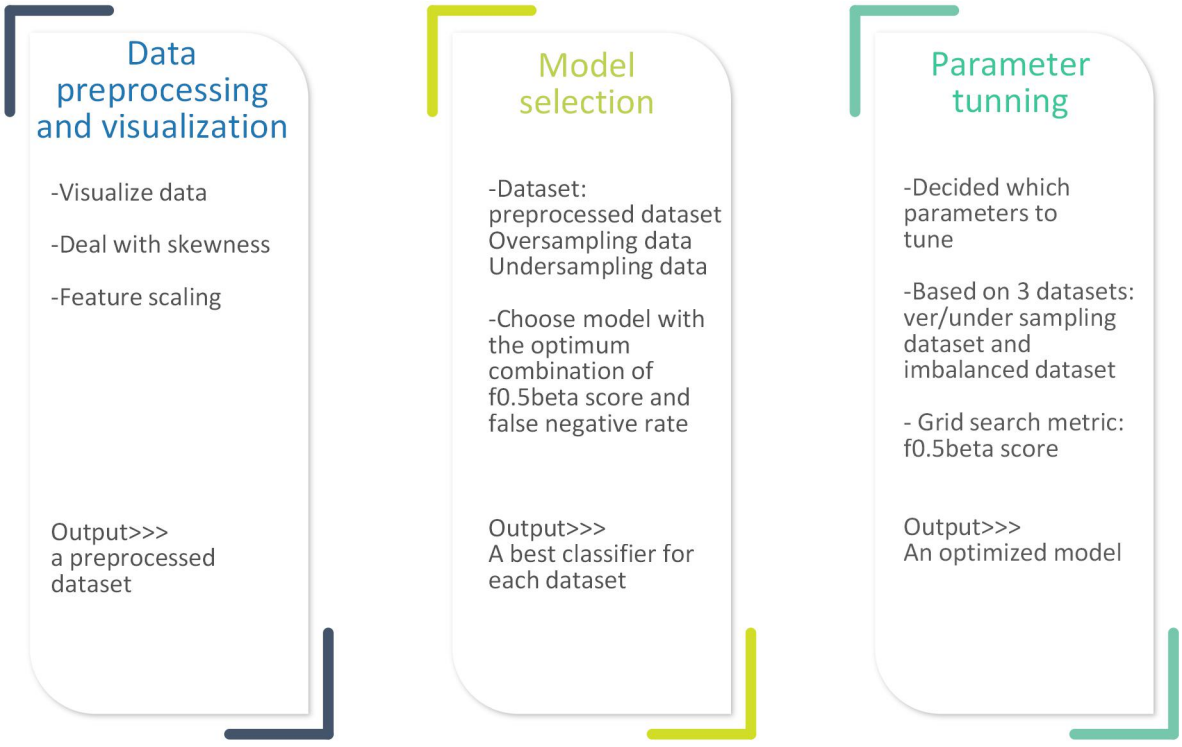
-F0.5score =$(1+0.5^2)$* (precision*recall/(0.5^2* precision)+recall)

Why F0.5score: our aim is to find out who will conduct default payment in the future.We hope that we could improve the precision(true positive/classified positive). Thus, we can introduce F-beta score, choose beta = 0.5 so that more emphasis is placed on precision.

-False negative rate = false negative/(true positive + false nagetive)

In our problem, if the false negative rate is low, then our model is good. Otherwise, if we have high false negative rate, which means among the clients who are going to default, we classified a lot of them as 'not going to default'. That is extremely bad and we don't want to see that happens.

An outline of the project design:



-Data exploration/visualization and preprocessing-

Features:

X1: Amount of the given credit (NT dollar): it includes both the individual consumer credit and his/her family (supplementary) credit.

X2: Gender (1 = male; 2 = female).

X3: Education (1 = graduate school; 2 = university; 3 = high school; 4 = others).

X4: Marital status (1 = married; 2 = single; 3 = others).

X5: Age (year).

X6 - X11: History of past payment. We tracked the past monthly payment records (from April to September, 2005) as follows: X6 = the repayment status in September, 2005; X7 = the repayment status in August, 2005; . . .;X11 = the repayment status in April, 2005. The measurement scale for the repayment status is: -1 = pay duly; 1 = payment delay for one month; 2 = payment delay for two months; . . .; 8 = payment delay for eight months; 9 = payment delay for nine months and above.

X12-X17: Amount of bill statement (NT dollar). X12 = amount of bill statement in September, 2005; X13 = amount of bill statement in August, 2005; . . .; X17 = amount of bill statement in April, 2005.

X18-X23: Amount of previous payment (NT dollar). X18 = amount paid in September, 2005; X19 = amount paid in August, 2005; . . .;X23 = amount paid in April, 2005.

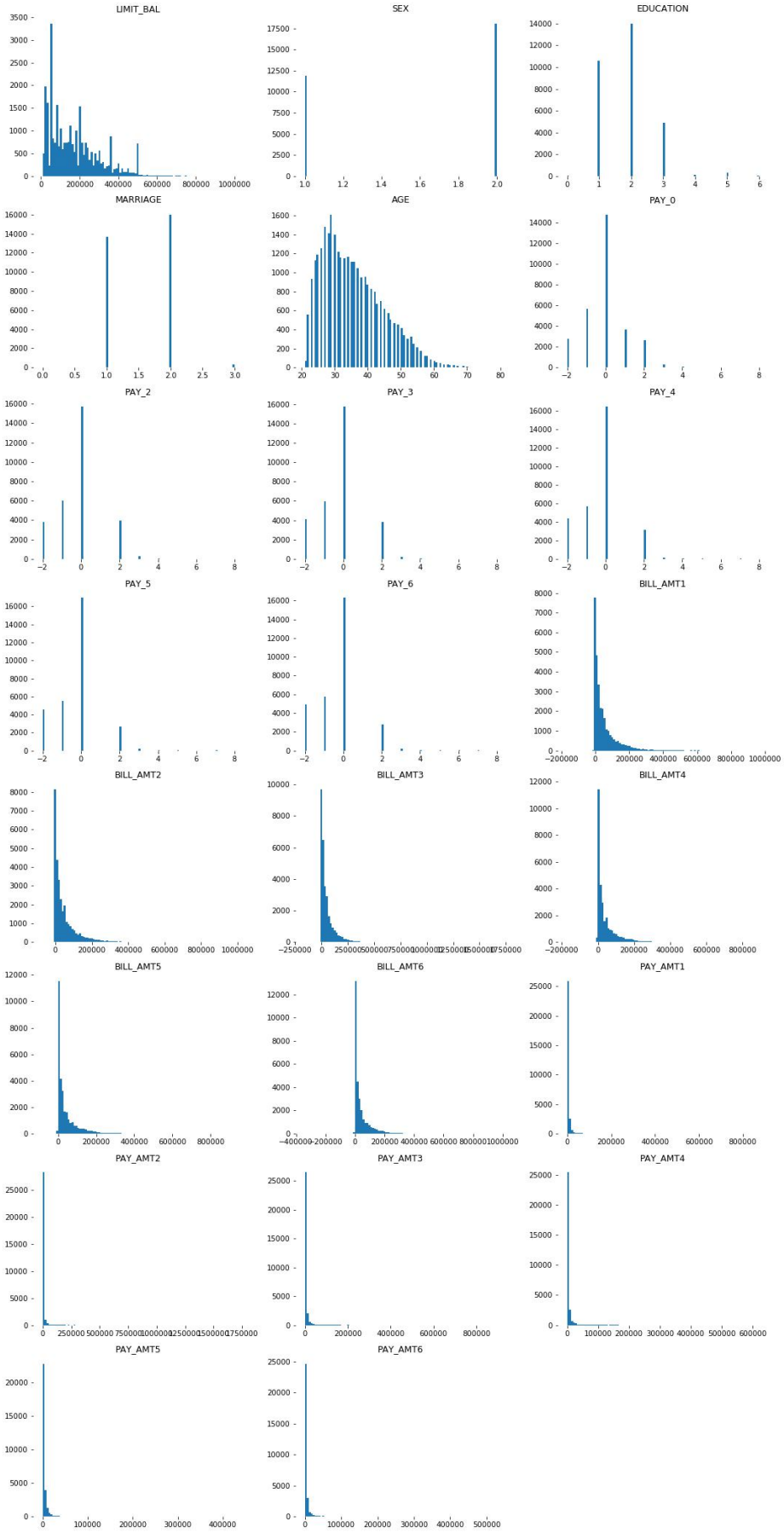
Classes:

default payment next month:

1 stands for the customer who will conduct default payment next month.

0 stands for customer who will not conduct default payment next month.

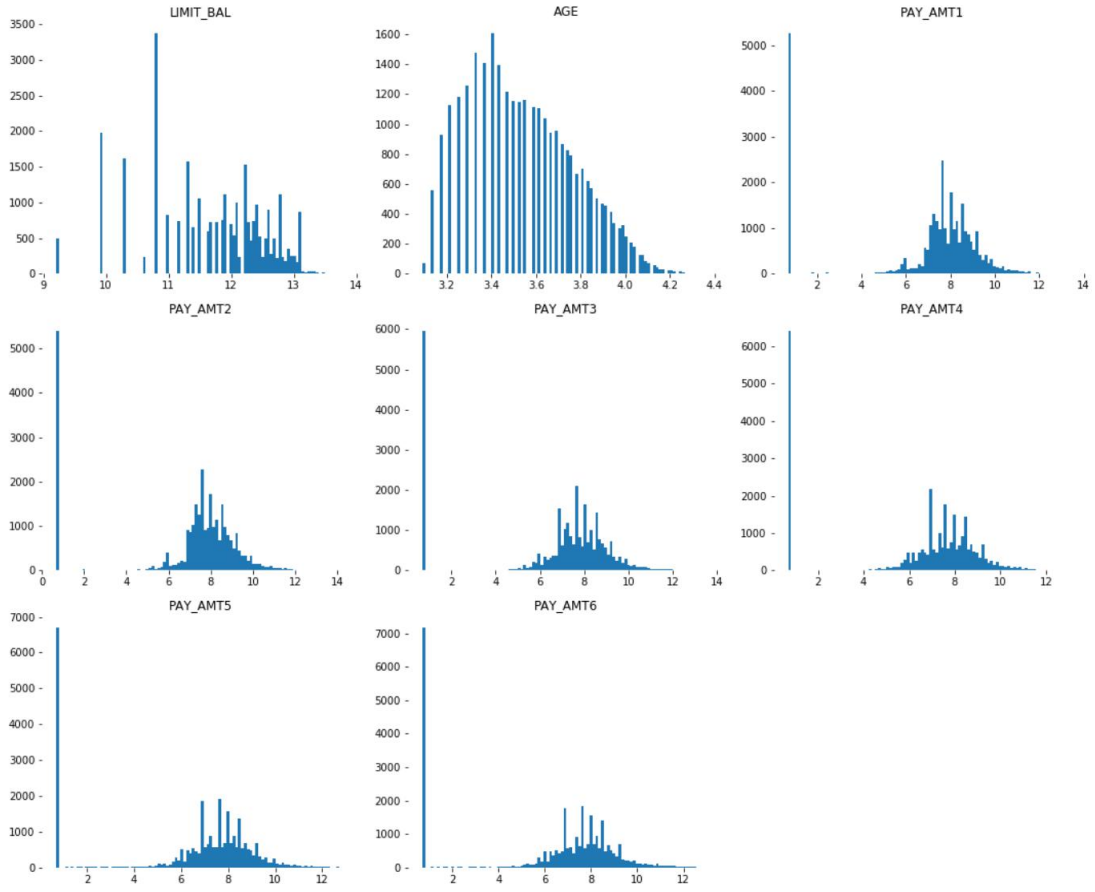
Distribution check:


LIMIT_BAL
SEX
EDUCATION
MARRIAGE
AGE
PAY_0
PAY_2
PAY_3
PAY_4
PAY_5
PAY_6
BILL_AMT1
BILL_AMT2
BILL_AMT3
BILL_AMT4
BILL_AMT5
BILL_AMT6
PAY_AMT1
PAY_AMT2
PAY_AMT3
PAY_AMT4
PAY_AMT5
PAY_AMT6

From the plot above, we found these features are **left-skewed**: 'LIMIT_BAL','AGE','PAY_AMT1','PAY_AMT2','PAY_AMT3','PAY_AMT4','PAY_AMT5','PAY_AMT6'.

Thus, we need to perform log transformation on them.Noticed that the min of some features I want to transform is 0, in order to conduct log transformation, I **add 1** to those features of each data point.

Skewed feature after log transformation:



From the plot above, we can see that after log transformation, the distribution of these skewed features became **normally distributed**.

Feature scaling:

Employed MinMaxScaler to do Feature scaling on numerical features.

-Model Selection-

-3 datasets-

	imbalanced	oversampling	undersampling
training	x_train, y_train	Oversampled data from x_train, y_train	Undersampled data from x_train, y_train
testing	x_test, y_test	x_test, y_test	x_test, y_test

Because we want to know whether oversampling or undersampling would have influence on the result, we need to control test dataset and only conduct oversampling or undersampling on training dataset.

Choose for imbalanced data:

-Imbalanced dataset-

	F0.5beta score	False Negative Rate
GradientBoostingClassifier	0.565627266135	0.337110481586
RandomForestClassifier	0.506154232605	0.395802098951
LogisticRegression	0.00757002271007	0.0

Based on the results above, we should choose GradientBoostingClassifier as our learner since the f0.5beta score is the highest among 3 classifiers, and the false negative rate is also relatively low. However the false negative rate, 0.34, is still very high. It means out of 100 customers who are going to default, we will misclassified approximately 34 customers to be not going to default, which is not good.
The false negative rate for LogisticRegression is 0.0 is because that logistic regressor classified all the labels as '0'.

Therefore, we should choose **GradientBoostingClassifier** for imbalanced dataset.

Choose for undersampling data:

-Undersampling-

	F0.5beta score	False Negative Rate
GradientBoostingClassifier	0.485217391304	0.542122538293
RandomForestClassifier	0.448680351906	0.575707154742
LogisticRegression	0.456847322487	0.574297188755

we should choose **GradientBoostingClassifier** for undersampling.

Choose for oversampling data:

-Oversampling-

	F0.5beta score	False Negative Rate
GradientBoostingClassifier	0.558247526751	0.392307692308
RandomForestClassifier	0.506178192066	0.433939393939
LogisticRegression	0.466704610131	0.561027837259

we should choose **GradientBoostingClassifier** for undersampling.

-Parameter tuning-

GBM combines a set of weak learners to improve prediction accuracy. The outcomes are weighed based on the outcomes of previous models. For the outcomes predicted correctly, GBM would give a lower weight on it and give a higher weight on the ones miss-classified.

Combine the theory of GBM , the data itself and plot above,

Here are 4 parameters I choosed to tune:

Parameter	Reason
max_depth	Max_depth is used to control over-fitting. Higher depth will allow model to learn relations very specific to a particular sample.
min_samples_leaf	Defines the minimum samples (or observations) required in a terminal node or leaf. Used to control over-fitting. Generally lower values should be chosen for imbalanced class problems. In our problem, the class is slightly imbalanced so by adjusting this parameter, we could prevent overfitting hopefully.
learning_rate	Learning_rate determines the impact of each tree on the final outcome. GBM works by starting with an initial estimate which is updated using the output of each tree. The learning parameter controls the magnitude of

	this change in the estimates. Lower values are generally preferred as they make the model robust to the specific characteristics of tree and thus allowing it to generalize well. Lower values would require higher number of trees to model all the relations and will be computationally expensive.
n_estimators	The number of sequential trees to be modeled .Usually GBM is fairly robust at higher number of trees.

Fit GBC models on 3 different datasets: imbalanced dataset, oversampled dataset and undersampled dataset. For each dataset, split it into 75% for training. **Control testing data**: knowing that if we want to evaluate wether using a undersampled/oversampled dataset is good or not, we should control: test data. So we should check the performance on x_test we've already splitted from imbalanced dataset.

Performance on testing data

	imbalanced	oversampling	undersampling
F0.5beta score	0.582329317269	0.546504433904	0.471395374571
False negative rate	0.324657534247	0.400452488688	0.558777835318

From the numbers above, we can conclude that the **optimized model feed with imbalanced data** has the best performance.

-Conclusion-

Oversampling or undersampling doesn't improve the f0.5beta score or reduce the false negative rate.

The GBC model feed with imbalanced dataset has the f0.5beta score: 0.565627266135 and after parameter tunning, it improved to 0.582329317269; has the false negative rate: 0.337110481586 and after parameter tunning, it slightly reduced to 0.324657534247. Financial institutions can utilize this credit card default detection model to predict the probability of being default given certain historical information of a customer.

As a future work, other algorithms such as Convolutional Neural Networks (CNN) can be used to build new classification models on the same real world dataset. Also, oversamling or undersampling doesn't work well and that means if we would like to improve performance, we need to gather more real world data.