Imbalanced Classification with Credit Card Fraud Detection

Kevin Vu Duc

06/02/2022

1. Introduction

Fraud is a major problem for credit card companies, because the large volume of transactions that are completed each day and many fraudulent transactions look a lot like normal transactions. The design of efficient fraudulent detection algorithms is key to decrease these losses. The design of fraud detection algorithms is however particularly challenging due to non-stationary distribution of the data, highly imbalanced classes distributions. This project will find some fraudulent detection algorithms and methods to deal with high imbalanced classes distributions.

The dataset used for this project is 'creditcard', which contains many transactions in September, 2013 by European cardholders. This dataset has 492 frauds among 284807 transactions or only 0.172% of transactions are fraudulent, which is very high imbalanced.

Due to confidential issue, we do not have access to the original information of dataset, only generic variables are available which are principle components of a principle component analysis. These generic features are V1, V2, ..., V28. Beside that we have two other variables that is 'Time' and 'Amount'. Feature 'Time' contains the seconds elapsed between each transaction and the first transaction in the dataset. 'Amount' is the amount of transaction. 'Class' is response variable, which is equal to 0 if transaction is normal otherwise it gets value of 1(fraud).

Due to the imbalance of the dataset, we will use the Area Under Curve(AUC) as the performance metric and then introduce some sampling techniques to deal with the imbalanced issue.

2. Data Exploration

2.1 Data summary

Dimension of dataset

```
284807      31
```

Below are some first few lines of dataset

Time	V1	V2	V3	V4	V5	V6	V7
0	-1.35981	-0.07278	2.53635	1.37816	-0.33832	0.46239	0.23960
0	1.19186	0.26615	0.16648	0.44815	0.06002	-0.08236	-0.07880
1	-1.35835	-1.34016	1.77321	0.37978	-0.50320	1.80050	0.79146
1	-0.96627	-0.18523	1.79299	-0.86329	-0.01031	1.24720	0.23761

V8	V9	V10	V11	V12	V13	V14	V15
0.09870	0.36379	0.09079	-0.55160	-0.61780	-0.99139	-0.31117	1.46818
0.08510	-0.25543	-0.16697	1.61273	1.06524	0.48910	-0.14377	0.63556
0.24768	-1.51465	0.20764	0.62450	0.06608	0.71729	-0.16595	2.34586
0.37744	-1.38702	-0.05495	-0.22649	0.17823	0.50776	-0.28792	-0.63142

V16	V17	V18	V19	V20	V21	V22	V23
-0.47040	0.20797	0.02579	0.40399	0.25141	-0.01831	0.27784	-0.11047
0.46392	-0.11480	-0.18336	-0.14578	-0.06908	-0.22578	-0.63867	0.10129
-2.89008	1.10997	-0.12136	-2.26186	0.52498	0.24800	0.77168	0.90941
-1.05965	-0.68409	1.96578	-1.23262	-0.20804	-0.10830	0.00527	-0.19032

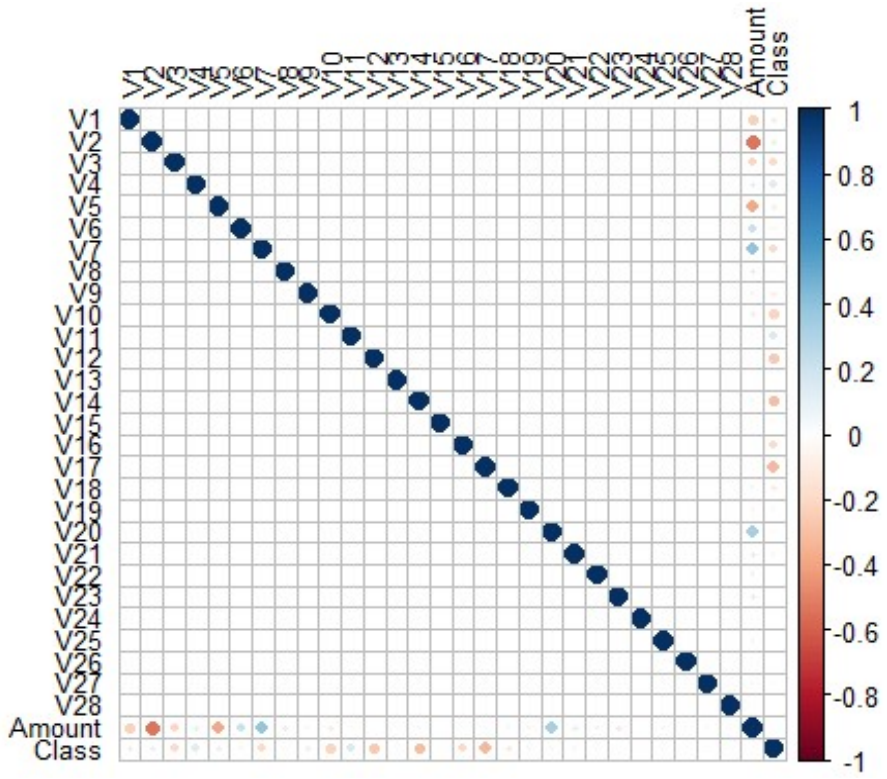
V24	V25	V26	V27	V28	Amount	Class
0.06693	0.12854	-0.18911	0.13356	-0.02105	149.62	0
-0.33985	0.16717	0.12589	-0.00898	0.01472	2.69	0
-0.68928	-0.32764	-0.13910	-0.05535	-0.05975	378.66	0
-1.17558	0.64738	-0.22193	0.06272	0.06146	123.50	0

Now, we can verify the imbalanced issue of the dataset by looking at the proportion of fraudulent cases. It is obvious that only 0.173% of cases is fraudulent, and it is confirmed that the distribution of 'Class' is highly imbalanced.

```
Number of observations in each class
     0      1
  284315    492

Proportion of each class
       0         1
  0.9982725 0.0017275
```

The correlation table below shows that there is no or very little correlation between variables. This is because of V1, V2…V28 are principle components of PCA.



2.2 Data transformation

Since 'Time' feature do not indicate the actual time of transaction, it just a list of transaction in order of time and it has little or no significant correlation with our analysis then we can remove it from the dataset. We also need to change 'Class' variable to factor type, and scale numerical variables so the new features have the same range.

```
dat <- dat[,-1]
#change Class variable to factor
dat$Class <- as.factor(dat$Class)
levels(dat$Class) <- c("Not_Fraud","Fraud")
#scale numeric variables
dat[,-30] <- scale(dat[,-30])
```

3 Modelling Methods:

3.1 Sampling methods

Standard machine learning algorithms face difficulty on imbalanced data because of the unequal distribution in dependent variable. This causes the performance of existing classifiers to get biased towards majority class. The methods to deal with this problem are widely known as 'Sampling Methods'. It modifies an imbalanced data into balanced distribution using some mechanism.

This section introduces three sampling techniques to deal with imbalanced data issue, they are 'under sampling', 'over sampling' and 'random over-sampling examples'

Under sampling

Under sampling will reduce the number of observations from majority class to make the data set balanced. This method is the best use when the data set is huge and it can decrease the running time as well as storage space.

There are two types of 'under sampling': Random and Informative.'Random under sampling' randomly chooses observations from majority class which are eliminated until the data set gets balanced. 'Informative under sampling' follows a pre-specified selection criterion to remove the observations from majority class.

A possible problem with 'under sampling' is that we may lose important information pertaining to majority class because of removing so many observations from the dataset.

Over sampling

Opposite to 'under sampling', this method works with minority class. It replicates the observations from minority class to balance the data. It is also known as 'up sampling'. This method also have two categories: Random 'up sampling' and Informative 'up sampling'. Random oversampling balances the data by randomly oversampling the minority class. Informative oversampling uses a pre-specified criterion and synthetically generates minority class observations.

The advantage of this method is that it keeps all original information. And the disadvantage is it can lead to overfitting because of many observations replicated.

Random over-sampling examples(ROSE)

This method generates artificial data instead of replicating and adding observations from minority class. It is also a type of oversampling method. ROSE (random over-sampling examples) uses smoothed bootstrapping to draw artificial samples from the feature space neighbourhood around the minority class.

3.2 Classification methods

There are various classified algorithms, here are some of them: - Logistic regression - Decision Trees - Random Forest - Support Vector Machine - K-nearest neighbor - Neural network

In this project, we will use three algorithms: Logistic regression, Decision Trees and Random Forest to apply on different sampling data set and then compare performances to find the best models.

4. Simulation and Results

4.1 Generating sampling dataset

We first need to divide creditcard dataset into train and test sets, then apply different sampling methods on train dataset in order to generate down_train, up_train, rose_train dataset which correspond to 'under sampling', 'up-sampling', 'rose sampling' methods.

```
Dimension of train dataset

 213605     30

Dimension of test dataset

 71202    30

Class Distribution in train dataset

Not_Fraud     Fraud
  213236       369

Class Distribution in down_train
Not_Fraud     Fraud
   369          369

Class Distribution in up_train dataset
Not_Fraud     Fraud
  213236     213236

Class Distribution in rose_train dataset
Not_Fraud     Fraud
  107024     106581
```

4.2 Training and evaluation models

We do not fit and evaluate models in this file as it takes long time and make difficulty to generate the report. The full code for this is put in the script 'Credit Card Fraud Detection.R'. We just summary the results from that file.

Decision Trees Method

Applying Decision tree algorithm on train, down_train, up_train and rose_train then make the prediction on test dataset. Below is the performance for each case(AUC)

Algorithm	origin_data	down_sampling	up_sampling	rose_sampling

Decision Tree	0.927	0.962	0.957	0.941

Logistic Regression Method

Logistic regression method only converges on train and rose_train dataset. Below is the summary of performance:

Algorithm	origin_data	down_sampling	up_sampling	rose_sampling
Logistic Regression	0.976	NA	NA	0.982

Random Forest Method

The random forest method gives quite similar performance compare with logistic regression and it work on all four samplings. Below are the results:

Algorithm	origin_data	down_sampling	up_sampling	rose_sampling
Random Forest	0.971	0.977	0.978	0.962

We can combine all above results to get summary table for all models and cases as below. We can see that Logistic regression and Random forest algorithms outperform Decision tree but random forest can work diversified samples than Logistic Regression, so we can go with Random forest algorithm and up sampling method.

Algorithms	origin_data	down_sampling	up_sampling	rose_sampling
Decision Tree	0.927	0.962	0.957	0.941
Logistic Regression	0.976	NA	NA	0.982
Random Forest	0.971	0.977	0.978	0.962

5. Conclusion

We have implemented different classified algorithms and use different sampling methods to identify fraudulent transactions. The best algorithms are logistic regression and random forest. Although performances on different sampling methods are not much significant in these two algorithm, it shows how we can deal with imbalanced dataset issue. There are several other classification methods, which potentially give higher performance such as neural network, XGBoost, support vector machine that we may explore in the future.