Kaggle's Credit Card Fraud Detection Analysis

Ray Pan (yulinp3@illinois.edu)

5/4/2021

Contents

Abstract

Billions of dollars could be lost due to fraudulent transactions each year, and it's important for credit card companies to protect their customers from being victimized. I, in this analysis, fit 3 different machine learning algorithms to determine whether a transaction is legal by using a subset that makes it less biased. As result, logistic regression is the most accurate with an accuracy of 0.9698. Credit card companies should use caution when doing analysis based on this regression since there's still a small possibility of false prediction.

Introduction

Credit fraud is a common criminal nowadays, everybody who owns a credit card may be at risk of being the victim of credit fraud. It is extremely important for credit card companies to recognize whether a transaction is genuine or fraudulent to protect their customers from being charged for what they did not pay. The dataset contains transactions made by credit cards in September 2013 by European cardholders.

Methods

Data

Below are total of NA values in each column and percentage of each type of transaction presented in the dataset.

```
##   Time     V1     V2     V3     V4     V5     V6     V7     V8     V9    V10
##      0      0      0      0      0      0      0      0      0      0      0
##    V11    V12    V13    V14    V15    V16    V17    V18    V19    V20    V21
##      0      0      0      0      0      0      0      0      0      0      0
##    V22    V23    V24    V25    V26    V27    V28 Amount  Class
##      0      0      0      0      0      0      0      0      0
```

```
##
##       fraud     genuine
## 0.001727486 0.998272514
```

There's no NA values in the dataset. However, the response variable is very unbalanced. I created a subset named "cc_new" with fraud and genuine split evenly to analysis.

```
set.seed(42)
fraud=cc_change[which(cc_change$Class=="fraud"),]
genuine=cc_change[which(cc_change$Class=="genuine"),]
genuine=genuine[sample(nrow(genuine),500),]
quantile(genuine$Amount, seq(0, 1, by=.25))
```

```
##        0%       25%       50%       75%      100%
##    0.0000    4.9825   21.8200   79.0000 2468.2000
```

```
quantile(fraud$Amount, seq(0, 1, by=.25))
```

```
##      0%     25%     50%     75%    100%
##    0.00    1.00    9.25  105.89 2125.87
```

```
cc_new<-rbind(fraud, genuine)
```

It shows no matter how much money a transaction has, a genuine or fraudulent transaction could always happen.

We could see from the graph that fraud transactions are more likely to happen at night when comparing to genuine transactions. But fraud transactions, same as genuine transactions, are more likely to happen during the day when comparing to itself.

Modeling

Logistic Regression

I first do a test-train split Training (70%) and Testing (30%)

```
set.seed(100)
trn_index <- sample(nrow(cc_new), size = 0.7 * nrow(cc_new))
cc_trn = cc_new[trn_index, ]
cc_tst = cc_new[-trn_index, ]
```

Fit a logistic regression and print the confusion matrix to find the accuracy.

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction fraud genuine
##    fraud     132       0
##    genuine     9     157
##
```

```
##                Accuracy : 0.9698
##                  95% CI : (0.9434, 0.9861)
##     No Information Rate : 0.5268
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9392
##
##  Mcnemar's Test P-Value : 0.007661
##
##             Sensitivity : 0.9362
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.9458
##              Prevalence : 0.4732
##          Detection Rate : 0.4430
##    Detection Prevalence : 0.4430
##       Balanced Accuracy : 0.9681
##
##        'Positive' Class : fraud
##
```

The accuracy is rather high, so I'm interested in how the full model would perform. I did a test-train split Training (70%) and Testing (30%) on the full dataset

```
set.seed(100)
cc$Class <- as.numeric(cc$Class)
train_index <- sample(nrow(cc), size = 0.7 * nrow(cc))
train <- cc[train_index,]
test <- cc[-train_index,]
```

Fit a logistic regression on the full dataset and print the confusion matrix to find the accuracy.

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction     0     1
##          0 85274    54
##          1    22    93
##
##                Accuracy : 0.9991
##                  95% CI : (0.9989, 0.9993)
##     No Information Rate : 0.9983
##     P-Value [Acc > NIR] : 7.876e-11
##
##                   Kappa : 0.7095
##
##  Mcnemar's Test P-Value : 0.0003766
##
##             Sensitivity : 0.9997
##             Specificity : 0.6327
##          Pos Pred Value : 0.9994
##          Neg Pred Value : 0.8087
##              Prevalence : 0.9983
```

```
##          Detection Rate : 0.9980
##    Detection Prevalence : 0.9987
##       Balanced Accuracy : 0.8162
##
##        'Positive' Class : 0
##
```

A logistic regression model achieved a 0.9698 accuracy for the subset model and a 0.9993 accuracy for the full model, with 0.9362 and 0.9997 sensitivity, which is good. However, for the full model, the accuracy might not be so reliable since the dataset is so biased that most of the transactions are labeled as genuine, which may cause the accuracy to be relatively high.

K-Nearest Neighbors

Print the confusion matrix to find the accuracy.

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction fraud genuine
##    fraud      76      39
##    genuine    65     118
##
##                Accuracy : 0.651
##                  95% CI : (0.5939, 0.7051)
##     No Information Rate : 0.5268
##     P-Value [Acc > NIR] : 9.61e-06
##
##                   Kappa : 0.2934
##
##  Mcnemar's Test P-Value : 0.01423
##
##             Sensitivity : 0.5390
##             Specificity : 0.7516
##          Pos Pred Value : 0.6609
##          Neg Pred Value : 0.6448
##              Prevalence : 0.4732
##          Detection Rate : 0.2550
##    Detection Prevalence : 0.3859
##       Balanced Accuracy : 0.6453
##
##        'Positive' Class : fraud
##
```

The results for K-Nearest Neighbors are not very ideal as it only has accuracy of 0.651 with sensitivity of 0.539.

Random Forest

Fit a random forest model with ntree=2000 (Number of branches will grow after each time split). Print the confusion matrix to find the accuracy.

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction fraud genuine
##    fraud     130       2
##    genuine    11     155
##
##                Accuracy : 0.9564
##                  95% CI : (0.9266, 0.9766)
##     No Information Rate : 0.5268
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9122
##
##  Mcnemar's Test P-Value : 0.0265
##
##             Sensitivity : 0.9220
##             Specificity : 0.9873
##          Pos Pred Value : 0.9848
##          Neg Pred Value : 0.9337
##              Prevalence : 0.4732
##          Detection Rate : 0.4362
##    Detection Prevalence : 0.4430
##       Balanced Accuracy : 0.9546
##
##        'Positive' Class : fraud
##
```

The accuracy for this model is 0.9597 with a sensitivity of 0.922.

The cross-validated accuracy is also calculated here

```
## [1] "Cross-validated accuracy:0.938035658429778"
```

A Random Forest model achieved a 0.9698 accuracy for the subset model, with a 0.9362 sensitivity, which is very good. After cross validation, the accuracy is 0.9337

Results

	Logistic	Logistic full model	K-Nearest Neighbors	Random Forest
Accuracy	0.9697987	0.9991105	0.6510067	0.9563758
Sensitivity	0.9361702	0.9997421	0.5390071	0.9219858

Both the logistic model and random forest regression show reasonable results with relatively high accuracy. The logistic model would be considered the best model here with accuracy around 0.9698 and sensitivity around 0.9362. The K-Nearest model doesn't seem to perform well with accuracy only at 0.651.

Discussion

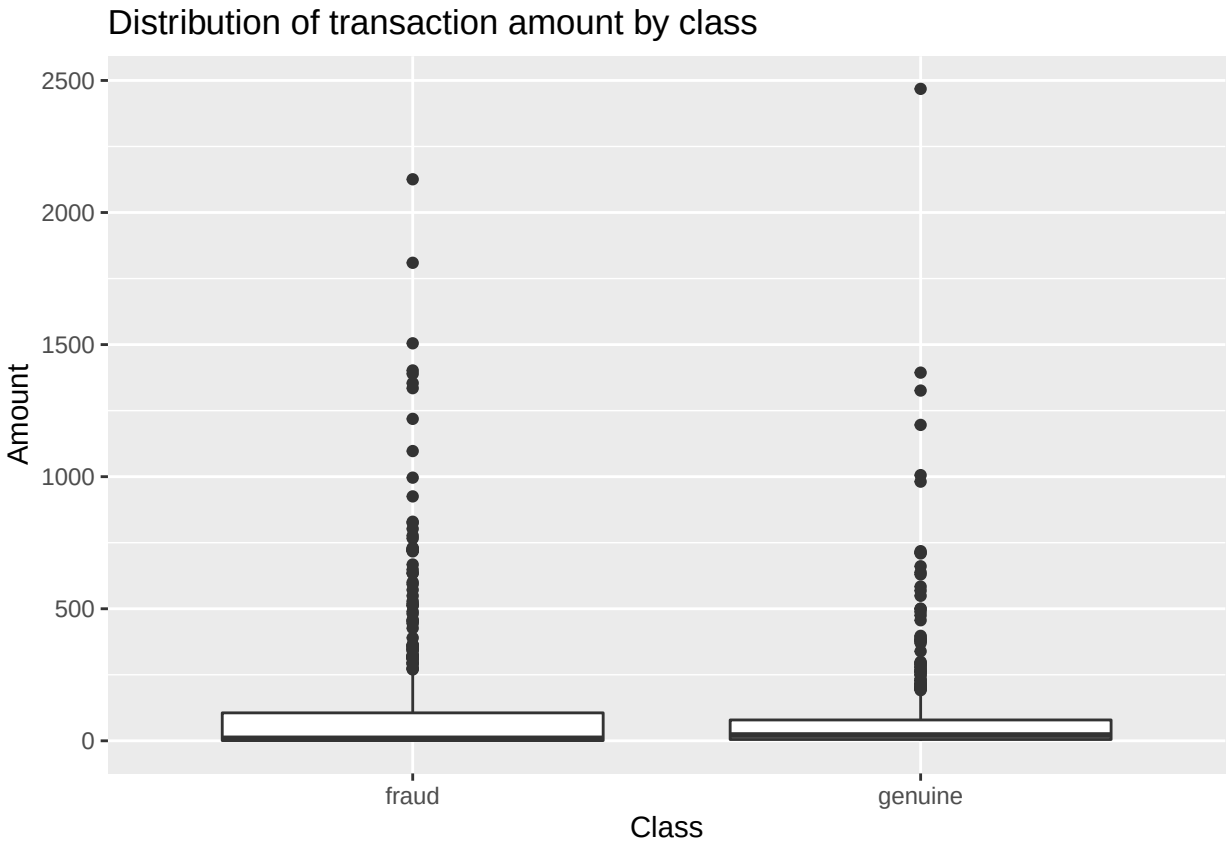
The logistic model in this analysis is rather reliable in identifying the "true positive" results(credit transaction labeled as fraud). High accuracy is important for credit card companies since they don't want to wrongly report a genuine transaction nor they don't want to let go of any fraudulent transaction to protect their valued customers. Even though there's a difference in the barplot, from the distribution plot and quartile table we could see that a genuine or fraudulent transaction could happen at any time and any amount. Further investigation might be needed when using this model since there's still a small possibility of false prediction and those companies don't want to let go of any fraudulent transactions.

Appendix

Data Dictionary

- Time - the seconds elapsed between each transaction and the first transaction in the dataset
- Amount - transaction Amount
- V1-V28 - principal components obtained with PCA. Due to confidentiality issues, original features and more background information about the data cannot be provided.
- Class - transaction type "Genuine" or "Fraud"

Boxplot of Amount vs. Class



Distribution of time of transaction