CSC 529
Winter 2017
Sarah Cummings

Case Study 1: Using Ensemble Classifiers to Detect Credit Card Fraud

Introduction:

In the bank and credit card industry, one of most common and important applications of data science is detecting credit card fraud. Catching a fraudulent credit card transaction quickly can enable the credit card company to shut of the card and prevent future fraudulent transactions. This protects both the company and the user from unlawful transactions affecting the owner's credit. In this type of problem, accuracy is not the most important feature of the model— certainly it is better to to have false positives for fraud than have false negatives; the cost of a falsely labeling a transaction as fraud is simply a phone call or a bit of concern on the credit card owners part. The cost of a false negative, not catching a fraudulent charge, is unlimited— equal to the amount of money spent in the transaction itself. in other words, sensitivity outranks specificity in its value for this case.

The idea of identifying a transaction as fraudulent or not is a classic classification problem. Within the realm of data science, methods for classification are robust and plentiful. Decision trees, K-nearest neighbors algorithms, naive babes models, and random forests are just a few of many algorithms that can be used to classify data. In this paper, we will explore these methods to classify credit card transaction data as genuine or fraudulent. Our focus will be on random forrest, as it is a bagging ensemble classification method. The algorithm uses bootstrap aggregation with decision trees, training a classifier on each bootstrap sample and using majority voting to determine the class label of the of ensemble classifier.This method is often more successful than simpler methods, and will likely give us the best results in our attempt to classify our transactions as fraudulent or genuine.

Data Description and Data Cleaning:

For this case study, the data was taken from the Kaggle page titled “Credit Card Fraud Detection”. It can be found at the following link: https://www.kaggle.com/dalpozz/creditcardfraud. The dataset contains information on credit card transactions over the course of two days in 2013, made be European credit card holders. It details 284,807 transactions, of which 492 are fraudulent. Note than this is thus a highly unbalanced dataset, further highlighting that accuracy will not be the best gage of success for our model. We could label all transactions as genuine and have a very high accuracy, but this would miss the goal of the project.

For these transactions, we have the following features:

amount: the amount of money used in the transaction

time: the amount of seconds that have passed between the first transaction in the data set and the current transaction

class: a binary variable with 0 being indicative of a genuine transaction, and 1 being fraudulent

V1-V28: scores from a principal component analysis that had been conducted in advance.

Since this data is confidential, we do not have any information on the original features that were used for PCA. Note that PCA takes a set of variables performs orthogonal transformation to create a set of linearly uncorrelated variables. This eliminates any issues that may arise from correlated variables, and also, perhaps more importantly in this case, erases much of the meaning from the variables thus encoding any confidential or personally identifiable information from the data.

The ambiguous nature of this data set is a limitation: despite how successful our model may be, we won’t be able to make any hard and fast conclusions about which variables are important or any trends in the features found in fraudulent transactions. The unbalanced nature of this data is also a limitation, as is the fact that it comes from Kaggle and we don’t know anything more than how the data is described on the website.

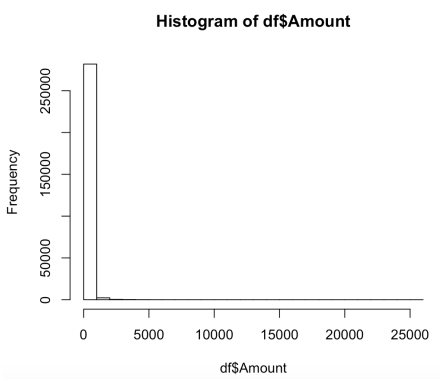
In this data set, we have no missing values or potential for inconsistencies. The data is already transformed via PCA , thus values are already standardized and normalized. There is no need for further cleaning.

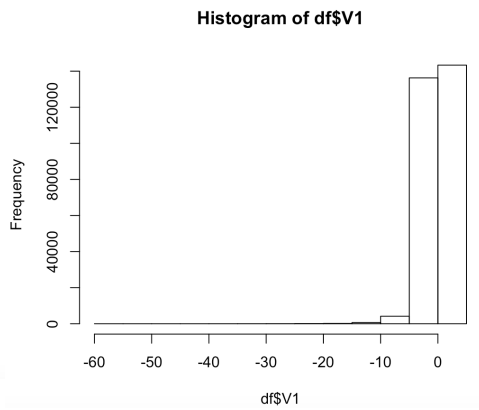
Data Analysis:

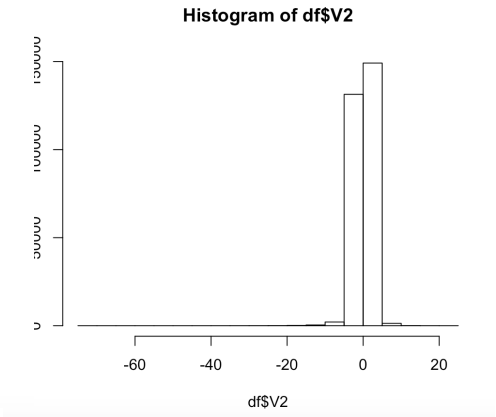
In examining our data, there is not much we can tell from the PCA component variables. They are all standardized with a mean of zero. Our other known variables have a distribution as shown below. Our smallest transaction is $0, and the largest is $25691.

```
      Time                Amount             Class
 Min.   :     0   Min.   :    0.00   0:284315
 1st Qu.: 54202   1st Qu.:    5.60   1:   492
 Median : 84692   Median :   22.00
 Mean   : 94814   Mean   :   88.35
 3rd Qu.:139320   3rd Qu.:   77.17
 Max.   :172792   Max.   :25691.16
```

By creating histograms of select variables, we find that these variables are generally not normally distributed.







We also find that there are no significant correlations among variables. To satisfy my own curiosity, I also performed a t test for difference in means of transaction amount among the two groups. I hypothesized that the fraudulent shoppers would have higher cost transactions, which proved to be correct. The mean for genuine transactions was $88.29 and the mean for fraudulent transactions was $122.21. The t test statistic was -2.98 (df= 492.61) which had fa p value p value 0.004. This allows us to conclude that the true difference in means is significantly different from zero.

To see if this data could benefit from an ensemble classification approach, I began by testing to see if a simple logistic regression would be sufficient. The model created had thirteen variables significant at the 0.01 level or greater, and correctly identified 308 of our 492 fraudulent transactions, seen to the right. Because this model missed 184 fraudulent transactions, which is 37%, I think an ensemble classification model might have a better fit.

	FALSE	TRUE
0	284273	42
1	184	308

To proceed we will first split the data into a 70-30 training testing split. The resulting subsets have the following classification distributions:

Original:

```
> table(df$Class)

     0      1
284315    492
```

Training:

```
> table(train$Class)

     0      1
199145    342
```

Testing:

```
> table(test$Class)

    0     1
85170   150
```

Experimental Analysis:

As an initial classification model, I created a decision tree for the data. The model had an accuracy of 0.9995 (with 95% confidence interval of (0.9993, 0.9996)). Note this is an impressive accuracy, but with such a small number of fraudulent transactions, it is important we examine the confusion matrix. The confusion matrix was as follows:

	Reference	
Prediction	0	1
0	199114	31
1	77	265

Looking at the positive cases (fraudulent), only 265 of the 342 fraudulent cases are correctly classified. That means our model would miss 77, or 22% of, fraudulent transactions. This is better than the logistic model, but not quite good enough. Using this model on the test set also gave a similar level of accuracy and fraudulent cases detected, meaning we don't have an overfitting issue.

In a second attempt at classifying the fraudulent credit card transactions, I created a random forest model. This model was just a little more successful, with an accuracy of 0.9996 (with 95% confidence interval of (0.9995, 0.9997)). This model is computationally much more expensive than the decision tree, and only correctly classifies five more fraudulent cases than the decision tree model. It still misses 72 fraudulent cases, which is 21% of all fraud cases. Note that using this model on the test set provides a similar accuracy and percentage of fraudulent cases detected, allowing us to conclude there is no overfitting.

	Reference	
Prediction	0	1
0	199137	8
1	72	270

Finally, I created a random forest model with 1000 trees. The training and testing confusion matrices were as follows:

Training:

	Reference	
Prediction	0	1
0	199137	8
1	69	273

Testing:

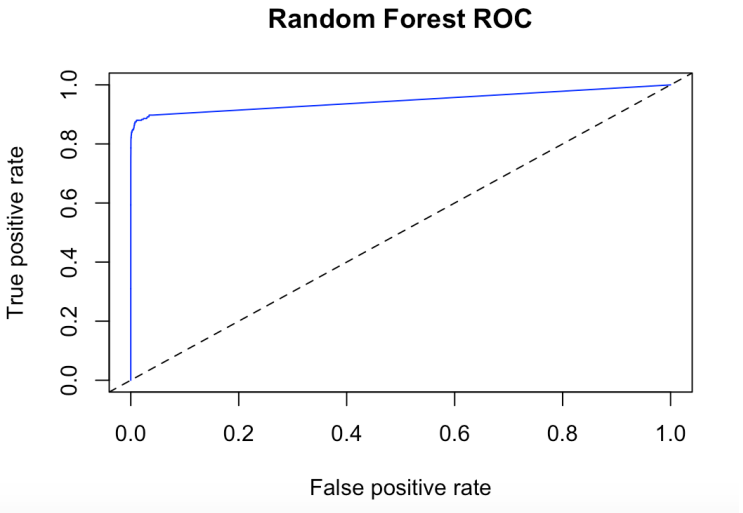
	Reference	
Prediction	0	1
0	85159	11
1	28	122

This final model has an accuracy of 0.9996, with confidence interval of (0.9995, 0.9997). It correctly identifies 273 of the 342 fraudulent cases in the training set, and performs similarly for the test set.

Experimental Results:

With our final model, we were able to correctly identify 273 fraudulent credit card transactions. We obtained a high accuracy level, though we could have done better. Unfortunately, since our variables are encoded, we don't gain any insights into what exactly is useful in predicting credit card fraud. Regardless, we have a few variables that are more. According to our results, the following variables were the most important in classifying the transactions as genuine or fraudulent: V17 (mean decrease gini 106.413927), V12 (mean decrease gini 96.172426), V14 (64.583149), and V16 (46.278997). Most of our variables have a mean decrease hovering somewhere in the 2-20 range, so the aforementioned variables are significantly more important.

Our ROC graph, shown right, confirms our high accuracy. Though our 1000 tree random forest model was best, its important to note that it was only slightly better than the decision tree and logistic models. If it is not worth the higher computation cost, using a simpler model may be better. In any case, we have shown that one can use a random forest model to identify fraudulent transaction.



Conclusion:

Credit card companies aim first to prevent, and second to detect, fraudulent credit card transactions. Fraudulent charges can be an expensive cost to the credit card company, and create a lot of undue stress for the credit card owner. If a fraudulent charge is detected quickly, the card can be shut off before another fraudulent transaction is made. In this paper, we examined three methods for detecting fraudulent credit card transactions, a logistic model, a decision tree, and random forests with two different tree sizes. The study found that the models were all highly accurate, with the large 1000 tree random forest model being the most successful. Unfortunately, few conclusions can be made about exactly what is the most

important in identifying credit card fraud, since the variables are encoded and transformed with PCA. The model is not perfect, but it would certainly help credit card agencies detect fraudulent charges.

Appendix:

###Output for Decision Tree Confusion Matrix:

Confusion Matrix and Statistics

```
          Reference
Prediction      0      1
         0 199114     31
         1     77    265

               Accuracy : 0.9995
                 95% CI : (0.9993, 0.9996)
    No Information Rate : 0.9985
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.8305
 Mcnemar's Test P-Value : 1.49e-05

            Sensitivity : 0.9996
            Specificity : 0.8953
         Pos Pred Value : 0.9998
         Neg Pred Value : 0.7749
             Prevalence : 0.9985
         Detection Rate : 0.9981
   Detection Prevalence : 0.9983
      Balanced Accuracy : 0.9474

       'Positive' Class : 0
```

###Output for 500 Tree Random Forest Confusion Matrix:

Confusion Matrix and Statistics

```
          Reference
Prediction      0      1
         0 199137      8
         1     72    270

               Accuracy : 0.9996
                 95% CI : (0.9995, 0.9997)
    No Information Rate : 0.9986
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.8708
 Mcnemar's Test P-Value : 1.873e-12
```

```
            Sensitivity : 0.9996
            Specificity : 0.9712
         Pos Pred Value : 1.0000
         Neg Pred Value : 0.7895
             Prevalence : 0.9986
         Detection Rate : 0.9982
   Detection Prevalence : 0.9983
      Balanced Accuracy : 0.9854

       'Positive' Class : 0
```

####Output for 1000 Tree Random Forest Confusion Matrix:

Confusion Matrix and Statistics

```
          Reference
Prediction      0      1
         0 199137      8
         1     69    273

               Accuracy : 0.9996
                 95% CI : (0.9995, 0.9997)
    No Information Rate : 0.9986
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.8762
 Mcnemar's Test P-Value : 8.051e-12

            Sensitivity : 0.9997
            Specificity : 0.9715
         Pos Pred Value : 1.0000
         Neg Pred Value : 0.7982
             Prevalence : 0.9986
         Detection Rate : 0.9982
   Detection Prevalence : 0.9983
      Balanced Accuracy : 0.9856

       'Positive' Class : 0
```

###Output for 500 Tree Random Forest Confusion Matrix— test set:

Confusion Matrix and Statistics

```
          Reference
Prediction     0     1
         0 85159    11
         1    28   122
```

```
               Accuracy : 0.9995
                 95% CI : (0.9994, 0.9997)
    No Information Rate : 0.9984
    P-Value [Acc > NIR] : < 2e-16

                  Kappa : 0.862
 Mcnemar's Test P-Value : 0.01041

            Sensitivity : 0.9997
            Specificity : 0.9173
         Pos Pred Value : 0.9999
         Neg Pred Value : 0.8133
             Prevalence : 0.9984
         Detection Rate : 0.9981
   Detection Prevalence : 0.9982
      Balanced Accuracy : 0.9585

       'Positive' Class : 0
```

###Gini index for RF model variables

	MeanDecreaseGini
Time	4.790046
V1	5.526913
V2	5.849657
V3	7.288483
V4	15.480556
V5	5.149714
V6	5.652694
V7	15.365524
V8	5.039168
V9	19.362883
V10	49.377395
V11	41.406915
V12	96.172426
V13	4.532585
V14	64.583149
V15	4.544277
V16	46.278997
V17	106.413927
V18	18.937555
V19	5.025100
V20	5.945517
V21	8.576637
V22	3.616869
V23	2.410227
V24	3.594924
V25	3.246839
V26	7.943432
V27	4.416237

V28	2.791673
Amount	4.580875