KING KHALID UNIVERSITY

COLLEGE OF COMPUTER SCIENCE

Detect Credit Card Fraud

By

Ahlam Mohammed Saad 441800053

Haifa Ibrahim al-mana 441813442

Supervised by:

MS.Mona Mushabab

Table of contents

1.1 Problem Statement

The aim of this R project is to build a classifier that can detect credit card fraudulent transactions. We will use a variety of machine learning algorithms that will be able to discern fraudulent from non-fraudulent one.

1.2 The Data Set

The datasets contains transactions made by credit cards in September 2013 by european cardholders.
This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions. The dataset is highly unbalanced, the positive class (frauds) account for 0.172% of all transactions.

It contains only numerical input variables which are the result of a PCA transformation. Unfortunately, due to confidentiality issues, we cannot provide the original features and more background information about the data. Features V1, V2, … V28 are the principal components obtained with PCA, the only features which have not been transformed with PCA are 'Time' and 'Amount'. Feature 'Time' contains the seconds elapsed between each transaction and the first transaction in the dataset. The feature 'Amount' is the transaction Amount, this feature can be used for example-dependant cost-senstive learning. Feature 'Class' is the response variable and it takes value 1 in case of fraud and 0 otherwise.

1.3 The Analysis

3.1 Importing the Datasets

We are importing the datasets that contain transactions made by credit cards

The dataset used in this project is available here – Fraud Detection Dataset

3.2 Data Exploration

In this section of the fraud detection project, we explore the data that is contained in the creditcard_data dataframe. We will proceed by displaying the creditcard_data using the head() function. We will then proceed to explore the other components of this dataframe like summary(data$Amount) it show us the "MIN, 1ST OU, MEDINE , MEAN, 3RD QU, MAX".

```
head(data)
A tibble: 6 x 31
 Time     V1      V2    V3     V4      V5      V6      V7      V8     V9     V10    V11    V12
<dbl>  <dbl>   <dbl> <dbl>  <dbl>   <dbl>   <dbl>   <dbl>   <dbl>  <dbl>   <dbl>  <dbl>  <dbl>
    0 -1.36  -0.0728 2.54   1.38  -0.338   0.462   0.240   0.0987  0.364  0.0908 -0.552 -0.618
    0  1.19   0.266  0.166  0.448  0.0600 -0.0824 -0.0788  0.0851 -0.255 -0.167   1.61   1.07
    1 -1.36  -1.34   1.77   0.380 -0.503   1.80    0.791   0.248  -1.51   0.208   0.625  0.0661
    1 -0.966 -0.185  1.79  -0.863 -0.0103  1.25    0.238   0.377  -1.39  -0.0550 -0.226  0.178
    2 -1.16   0.878  1.55   0.403 -0.407   0.0959  0.593  -0.271   0.818  0.753  -0.823  0.538
    2 -0.426  0.961  1.14  -0.168  0.421  -0.0297  0.476   0.260  -0.569 -0.371   1.34   0.360
... with 18 more variables: V13 <dbl>, V14 <dbl>, V15 <dbl>, V16 <dbl>, V17 <dbl>, V18 <dbl>,
  V19 <dbl>, V20 <dbl>, V21 <dbl>, V22 <dbl>, V23 <dbl>, V24 <dbl>, V25 <dbl>, V26 <dbl>,
  V27 <dbl>, V28 <dbl>, Amount <dbl>, Class <dbl>
apply(data, 2, function(x) sum(is.na(x)))
Time     V1     V2     V3     V4     V5     V6     V7     V8     V9    V10    V11    V12    V13
   0      0      0      0      0      0      0      0      0      0      0      0      0      0
 V14    V15    V16    V17    V18    V19    V20    V21    V22    V23    V24    V25    V26    V27
   0      0      0      0      0      0      0      0      0      0      0      0      0      0
 V28 Amount  Class
   0      0      0
```

Figure 1

As showin in figure 1 All the features, apart from “time” and “amount” are anonymised. Let’s see whether there is any missing data. and There are no NA values in the data.

```
summary(data$Amount)
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
    0.00     5.60    22.00    88.35    77.17 25691.16
```

Figure 2

As showin in figure 2 we can see the different statistics for our dataset

Visualization for the data set

By visualizing the data, we tried to find the range of amounts for the balances that were defrauded in our data set, and we found that the amounts range from (20000 to 60,000)

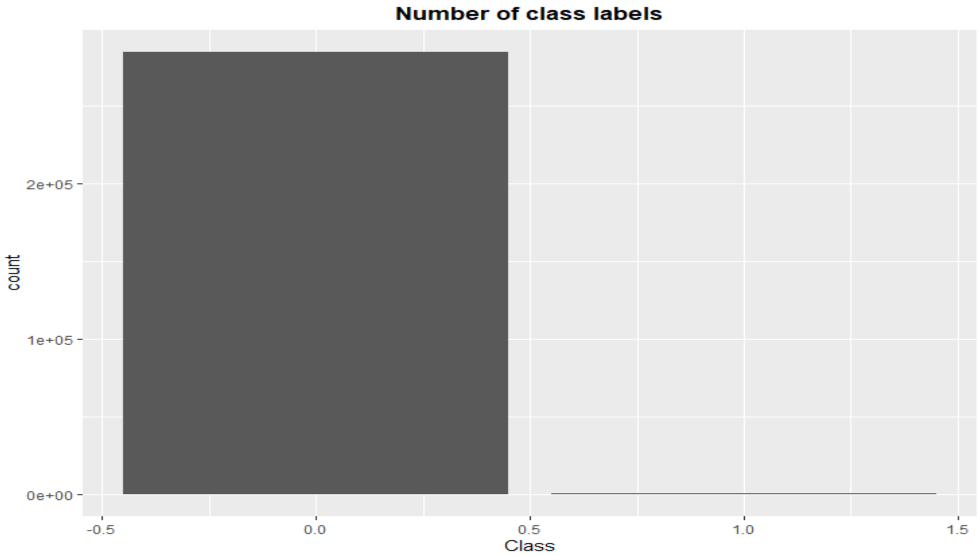

Figure 3

As showin in figure 3 Clearly, the dataset is extremely unbalanced. Even a “null” classifier which always predicts class=0 would obtain over 99% accuracy on this task.

```
> print(p)
> summary(data)
      Time              V1                  V2                  V3
 Min.   :     0   Min.   :-56.40751   Min.   :-72.71573   Min.   :-48.3256
 1st Qu.: 54202   1st Qu.: -0.92037   1st Qu.: -0.59855   1st Qu.: -0.8904
 Median : 84692   Median :  0.01811   Median :  0.06549   Median :  0.1799
 Mean   : 94814   Mean   :  0.00000   Mean   :  0.00000   Mean   :  0.0000
 3rd Qu.:139321   3rd Qu.:  1.31564   3rd Qu.:  0.80372   3rd Qu.:  1.0272
 Max.   :172792   Max.   :  2.45493   Max.   : 22.05773   Max.   :  9.3826
       V4                  V5                   V6                  V7
 Min.   :-5.68317   Min.   :-113.74331   Min.   :-26.1605   Min.   :-43.5572
 1st Qu.:-0.84864   1st Qu.:  -0.69160   1st Qu.: -0.7683   1st Qu.: -0.5541
 Median :-0.01985   Median :  -0.05434   Median : -0.2742   Median :  0.0401
 Mean   : 0.00000   Mean   :   0.00000   Mean   :  0.0000   Mean   :  0.0000
 3rd Qu.: 0.74334   3rd Qu.:   0.61193   3rd Qu.:  0.3986   3rd Qu.:  0.5704
 Max.   :16.87534   Max.   :  34.80167   Max.   : 73.3016   Max.   :120.5895
       V8                   V9                  V10                  V11
 Min.   :-73.21672   Min.   :-13.43407   Min.   :-24.58826   Min.   :-4.79747
 1st Qu.: -0.20863   1st Qu.: -0.64310   1st Qu.: -0.53543   1st Qu.:-0.76249
 Median :  0.02236   Median : -0.05143   Median : -0.09292   Median :-0.03276
 Mean   :  0.00000   Mean   :  0.00000   Mean   :  0.00000   Mean   : 0.00000
 3rd Qu.:  0.32735   3rd Qu.:  0.59714   3rd Qu.:  0.45392   3rd Qu.: 0.73959
 Max.   : 20.00721   Max.   : 15.59500   Max.   : 23.74514   Max.   :12.01891
      V12                 V13                 V14                 V15
 Min.   :-18.6837   Min.   :-5.79188   Min.   :-19.2143   Min.   :-4.49894
 1st Qu.: -0.4056   1st Qu.:-0.64854   1st Qu.: -0.4256   1st Qu.:-0.58288
 Median :  0.1400   Median :-0.01357   Median :  0.0506   Median : 0.04807
 Mean   :  0.0000   Mean   : 0.00000   Mean   :  0.0000   Mean   : 0.00000
 3rd Qu.:  0.6182   3rd Qu.: 0.66251   3rd Qu.:  0.4931   3rd Qu.: 0.64882
 Max.   :  7.8484   Max.   : 7.12688   Max.   : 10.5268   Max.   : 8.87774
      V16                  V17                  V18                  V19
 Min.   :-14.12985   Min.   :-25.16280   Min.   :-9.498746   Min.   :-7.213527
 1st Qu.: -0.46804   1st Qu.: -0.48375   1st Qu.:-0.498850   1st Qu.:-0.456299
 Median :  0.06641   Median : -0.06568   Median :-0.003636   Median : 0.003735
 Mean   :  0.00000   Mean   :  0.00000   Mean   : 0.000000   Mean   : 0.000000
 3rd Qu.:  0.52330   3rd Qu.:  0.39968   3rd Qu.: 0.500807   3rd Qu.: 0.458949
 Max.   : 17.31511   Max.   :  9.25353   Max.   : 5.041069   Max.   : 5.591971
      V20                  V21                  V22                  V23
```

Figure 4

As showin in figure 4 All the anonymised features seem to have been be normalised with mean 0. We will apply that transformation to the “Amount” column.

Having normalized the “Amount” column, it is important to see how informative that feature would be in predicting whether a transaction was fraudulent. Hence, let’s plot the amount against the class of transaction.

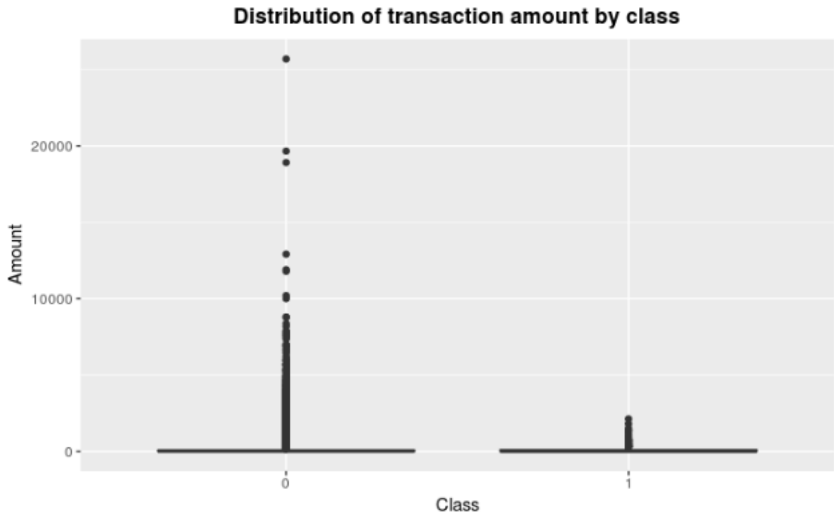

Figure 5

There is clearly a lot more variability in the transaction values for non-fraudulent transactions.To get a fuller picture, let’s compute the mean and median values for each class.in figure 5

	Class	`mean(Amount)`	`median(Amount)`
	<dbl>	<dbl>	<dbl>
1	0	88.3	22
2	1	122.	9.25

Figure 6

fraudulent transactions seem to have higher mean value than non-fraudulent ones, meaning that this feature would likely be useful to use in the predictive model. However, the median is higher for the legitimate ones, meaning the distribution of values for class “0” is left-skewed (also seen on the figure 6).

3.3 Data Manipulation

In this section of the R data science project, we normalize our data. We will apply this to the amount component of data amount. **Normalization** is used to compensate for the differences in sample quantity (concentration and/or path length) between the database spectrum and the unknown spectrum

```
normalization
normalize <- function(x){
  return((x - mean(x, na.rm = TRUE))/sd(x, na.rm = TRUE))
}
data$Amount <- normalize(data$Amount)
```

Figure 7

3.4 Data Modeling

After we have normalize our entire dataset, we use logistic model. In this section of credit card fraud detection project, we will fit our model. A logistic regression is used for modeling the outcome probability of a class such as pass/fail, positive/negative and in our case – fraud/not fraud. We proceed to implement this model on our test data. We use Logistic Regression because it tries to minimize cost of how wrong a prediction is

```
#Logistic regression
log_mod <- glm(Class ~ ., family = "binomial", data = X_train)
summary(log_mod)
```

```
Call:
glm(formula = Class ~ ., family = "binomial", data = X_train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-4.8020  -0.0298  -0.0194  -0.0123   4.6025

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -8.274e+00  2.702e-01 -30.618  < 2e-16 ***
Time        -4.086e-06  2.485e-06  -1.645 0.100068
V1           9.452e-02  4.686e-02   2.017 0.043683 *
V2           1.804e-02  6.488e-02   0.278 0.780999
V3          -2.502e-02  5.889e-02  -0.425 0.671000
V4           6.845e-01  8.277e-02   8.271  < 2e-16 ***
V5           1.380e-01  7.473e-02   1.846 0.064821 .
V6          -1.070e-01  8.162e-02  -1.310 0.190035
V7          -9.144e-02  7.410e-02  -1.234 0.217190
V8          -1.699e-01  3.336e-02  -5.095 3.50e-07 ***
V9          -2.817e-01  1.249e-01  -2.255 0.024104 *
V10         -8.052e-01  1.054e-01  -7.640 2.16e-14 ***
V11         -1.338e-01  9.018e-02  -1.483 0.137993
V12          1.414e-01  1.013e-01   1.395 0.162906
V13         -3.470e-01  9.289e-02  -3.735 0.000188 ***
V14         -5.972e-01  7.105e-02  -8.406  < 2e-16 ***
V15         -6.793e-02  9.646e-02  -0.704 0.481283
```

Figure 8

As showin in figure 8 The model can be fitted using gradient descent on the parameter vector. Equipped with some basic information, let’s see how the model performs.

3.5 Model Evaluation

We can evaluate a our model by doing the following:

The three main metrics used **to evaluate** a classification **model** are accuracy, precision, and recall.

```
# Use a threshold of 0.5 to transform predictions to binary
conf_mat <- confusionMatrix(y_test, as.numeric(predict(log_mod, X_test, type = "response") > 0.5))
print(conf_mat)
fourfoldplot(conf_mat$table)
```

```
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 56861     8
         1    34    58

               Accuracy : 0.9993
                 95% CI : (0.999, 0.9995)
    No Information Rate : 0.9988
    P-Value [Acc > NIR] : 0.0010494

                  Kappa : 0.7338
 Mcnemar's Test P-Value : 0.0001145

            Sensitivity : 0.9994
            Specificity : 0.8788
         Pos Pred Value : 0.9999
         Neg Pred Value : 0.6304
             Prevalence : 0.9988
         Detection Rate : 0.9982
   Detection Prevalence : 0.9984
      Balanced Accuracy : 0.9391

       'Positive' Class : 0
```

Figure 9

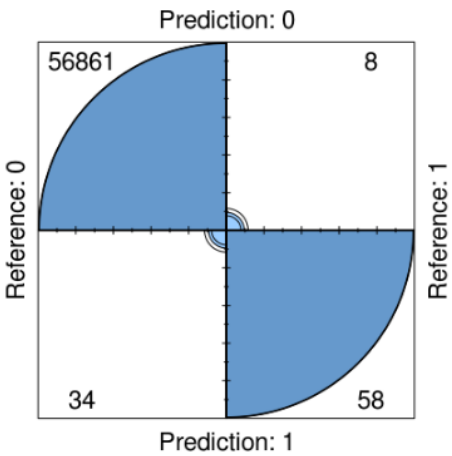

Figure 10

As you can see in figure 10 that the end result of our project So the model predict 56861 true positive, 8 false positive, 34 false negatives, 58 true negatives.

Receiver Operating Characteristic (ROC) curve

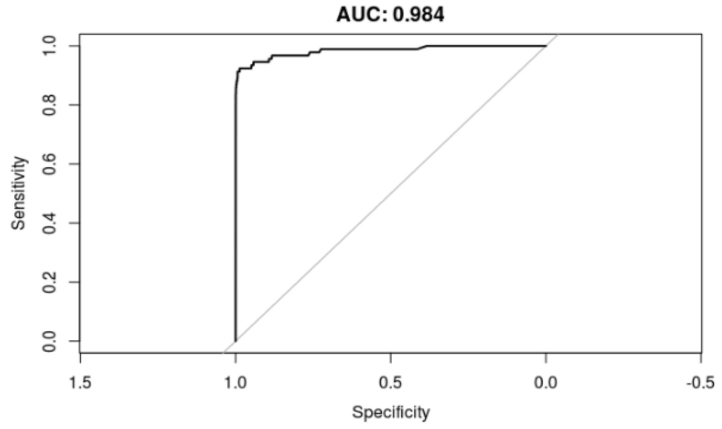

Figure 11

As you can see in figure 11 the performance is very good with area under the curve is 0.984.

1.5 Summary

Concluding our R Data Science project, we learnt how to develop our credit card fraud detection model using machine learning. We used logistic model. We learnt how data can be analyzed and visualized to discern fraudulent transactions from other types of data.

1.6 Reference

1-https://www.kaggle.com/mlg-ulb/creditcardfraud

2- For the Source code on github click the link https://github.com/Ahlam840/Credit-card