HarvardX Data Science Program

Credit Card Fraud Detection project

Do Quang Anh

2022-04-08

Contents

1 Introduction

Billions of dollars of loss are caused every year due to fraudulent credit card transactions. The design of efficient fraud detection algorithms is key to reducing these losses, and more algorithms rely on advanced machine learning techniques to assist fraud investigators. The design of fraud detection algorithms is however particularly challenging due to non-stationary distribution of the data, highly imbalanced classes distributions and continuous streams of transactions. At the same time public data are scarcely available for confidentiality issues, leaving unanswered many questions about which is the best strategy to handle this issue.

1.1 About Dataset

The dataset contains transactions made by credit cards in September 2013 by European cardholders.This dataset from Kaggle is available here: https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud. This dataset presents transactions that occurred in two days, where 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, 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-sensitive learning. Feature 'Class' is the response variable and it takes value 1 in case of fraud and 0 otherwise.

Given the class imbalance ratio, recommend measuring the accuracy using the Area Under the Precision- Recall Curve (AUPRC). We will also use different sampling techniques (details below) on the train dataset in order to address the issue of imbalanced classes while training our models.

Dataset	Number of Rows	Number of Columns
Credit card	284,807	31

2 Exploratory data analysis and data cleaning

All the features, apart from "time" and "amount" are anonymised. Let's see whether there is any missing data.

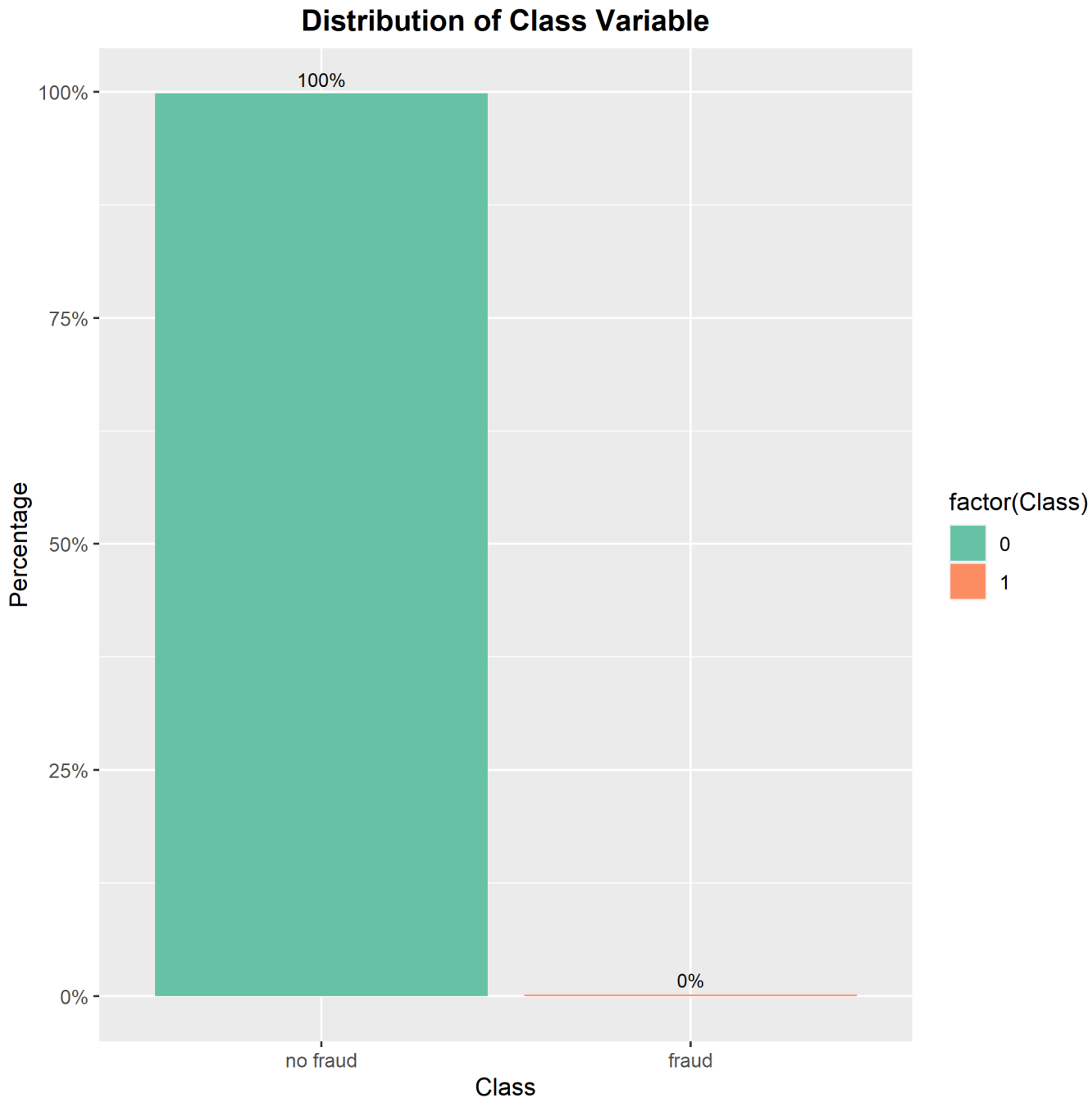
	x
Time	0
V1	0
V2	0
V3	0
V4	0
V5	0
V6	0
V7	0
V8	0
V9	0
V10	0
V11	0
V12	0
V13	0
V14	0
V15	0
V16	0
V17	0
V18	0
V19	0
V20	0
V21	0
V22	0
V23	0
V24	0
V25	0
V26	0
V27	0
V28	0
Amount	0
Class	0

There are no NA values in the data.

2.1 Check for class imbalance

Unbalanced data refers to unequal instances of different classes. The visualization shown below further reflects the imbalance of non-fraud and fraud transactions in the dataset. We have class (0 — No fraud, 1 — fraud) on the X-axis and the percentage of instances plotted on Y-axis. We see that our dataset is highly unbalanced with respect to the class of interest(Fraud).

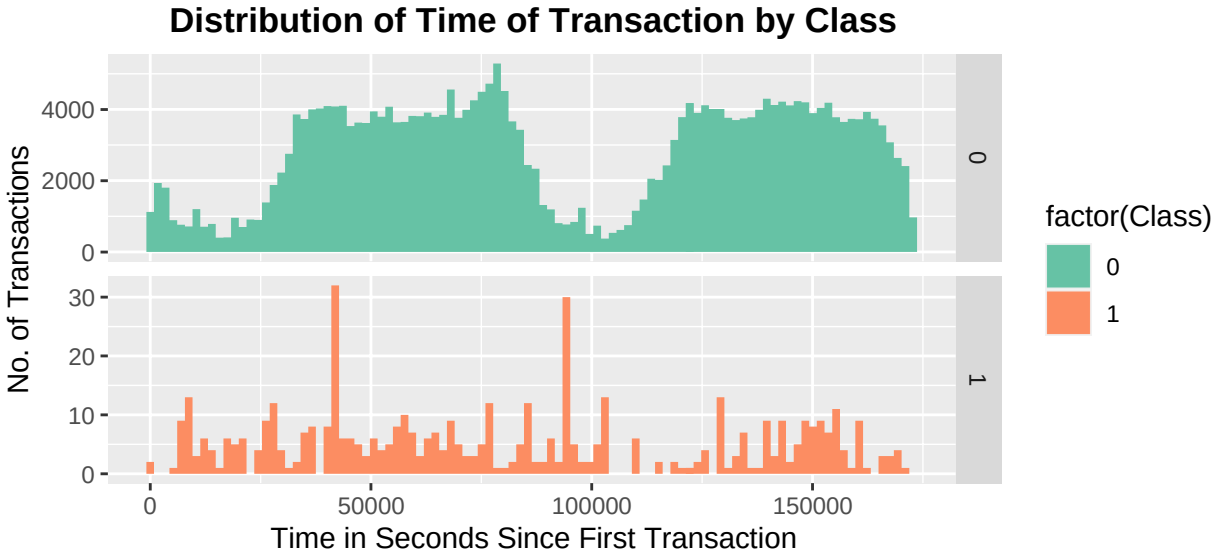
Class	Count
0	284,315
1	492


Distribution of Class Variable
100%
75%
50%
25%
0%
Percentage
100%
0%
no fraud
fraud
Class
factor(Class)
0
1

2.2 Time variable - Frauds over Time Distribution

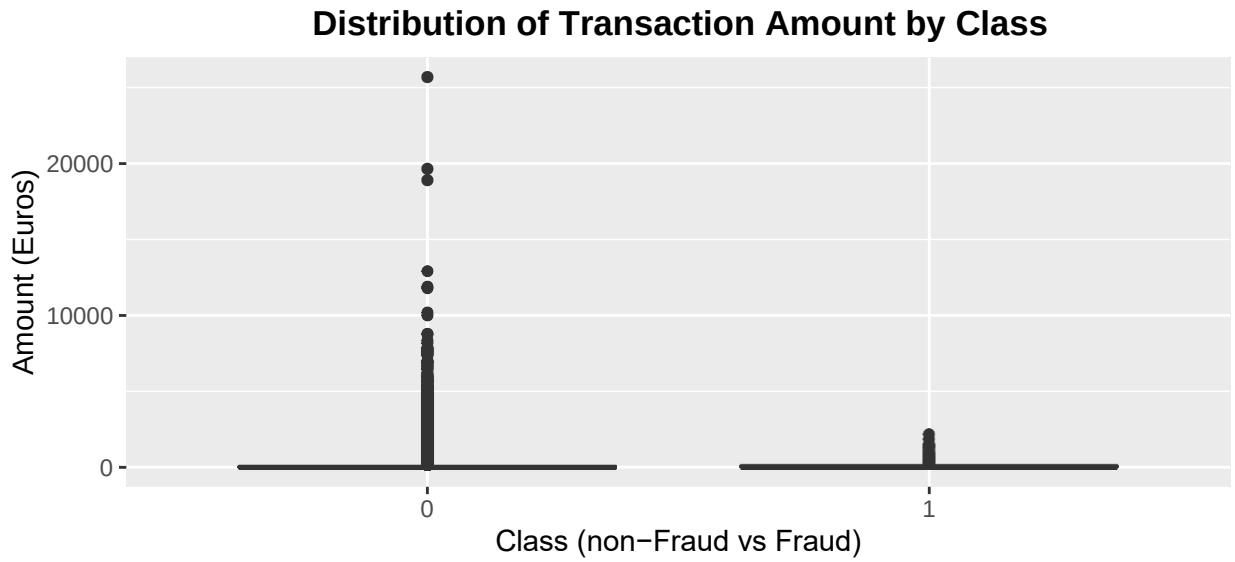
In the graph below, notice that the number of regular transactions drops sharply around the 90,000th-second mark, to surge again around the 110,000th-second mark. It wouldn't be absurd to assume that this period is during the night when individuals naturally perform fewer purchases and transactions than during the daytime.

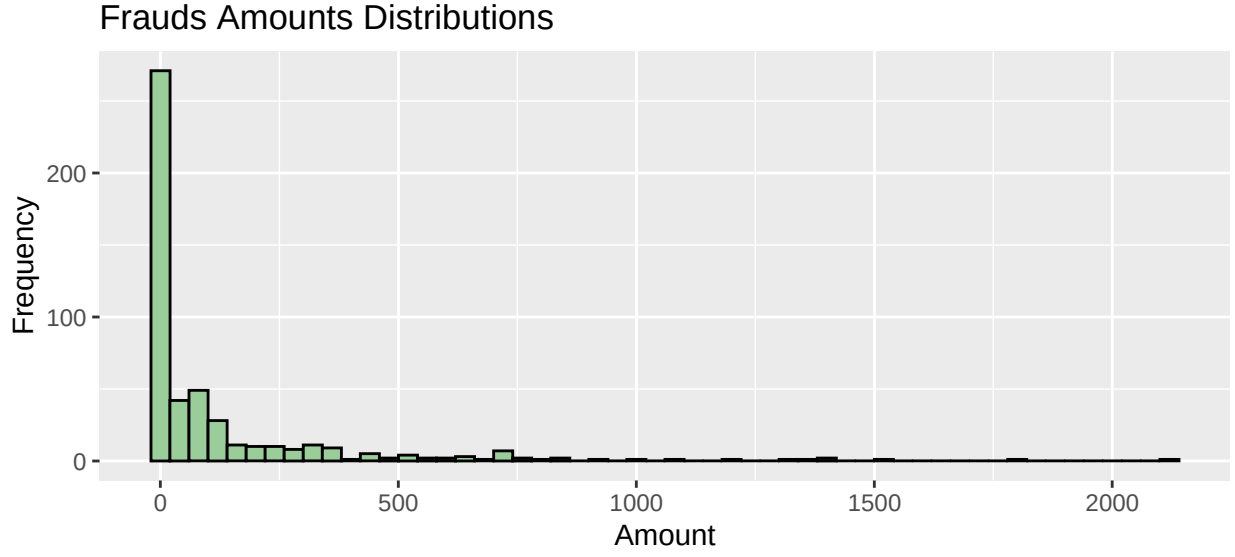
On the other hand, a great number of fraudulent transactions occurred around the 100,000 mark, which could confirm the previous assumption, considering that criminals should prefer to commit fraud late at night, assuming there would be less surveillance and victims would not realize they were being scammed soon enough.



2.3 Amount Variable

The boxplot below demonstrates the Amount of each transaction is more variable with the non-fraud trans- actions than with the fraud transactions given the number of outliers. Most transactions, both regular and fraudulent, were of "small" values. Small amount of money, less or equal of one dollar are scammed more frequently.



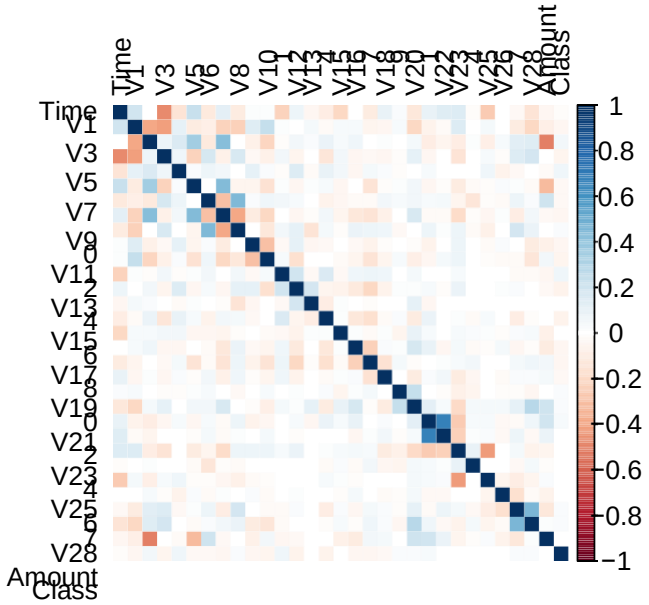


2.4 Correlations between each variables

##	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11
## Time	1.00	0.19	-0.01	-0.47	-0.14	0.23	-0.11	0.12	-0.10	0.02	0.01	-0.23
## V1	0.19	1.00	-0.38	-0.41	0.10	-0.11	-0.09	-0.23	-0.24	0.14	0.23	0.01
## V2	-0.01	-0.38	1.00	0.07	0.16	0.34	-0.11	0.43	0.01	-0.08	-0.21	-0.01
## V3	-0.47	-0.41	0.07	1.00	0.06	-0.22	0.17	-0.07	0.16	-0.03	-0.07	0.06
## V4	-0.14	0.10	0.16	0.06	1.00	0.02	0.08	0.06	-0.01	0.11	0.03	-0.01
## V5	0.23	-0.11	0.34	-0.22	0.02	1.00	-0.01	0.43	-0.06	-0.06	-0.15	0.01
## V6	-0.11	-0.09	-0.11	0.17	0.08	-0.01	1.00	-0.29	0.44	0.03	0.06	0.06
## V7	0.12	-0.23	0.43	-0.07	0.06	0.43	-0.29	1.00	-0.39	-0.10	-0.21	0.00
## V8	-0.10	-0.24	0.01	0.16	-0.01	-0.06	0.44	-0.39	1.00	0.00	-0.12	0.05
## V9	0.02	0.14	-0.08	-0.03	0.11	-0.06	0.03	-0.10	0.00	1.00	-0.29	-0.03
## V10	0.01	0.23	-0.21	-0.07	0.03	-0.15	0.06	-0.21	-0.12	-0.29	1.00	0.03
## V11	-0.23	0.01	-0.01	0.06	-0.01	0.01	0.06	0.00	0.05	-0.03	0.03	1.00

```
##  V12     0.07  0.00  0.05 -0.05  0.15 -0.05  0.04 -0.04  0.11  0.10 -0.16  0.20
##  V13    -0.05  0.05  0.03  0.01 -0.02  0.02  0.00 -0.02 -0.14 -0.05  0.00 -0.04
##  V14    -0.09 -0.02  0.09 -0.13  0.10  0.04 -0.07  0.06  0.04 -0.03  0.00  0.15
##  V15    -0.21  0.03  0.04  0.05  0.05 -0.04 -0.05 -0.05 -0.02  0.00 -0.03 -0.04
##  V16     0.00  0.04  0.07 -0.05 -0.04 -0.01 -0.02 -0.12  0.09 -0.03 -0.06  0.07
##  V17    -0.10 -0.06  0.00 -0.04  0.05 -0.15 -0.03 -0.15  0.07 -0.07 -0.15  0.07
##  V18     0.09 -0.04  0.01 -0.05  0.02 -0.02  0.05 -0.07  0.07 -0.01 -0.05  0.07
##  V19     0.03  0.03  0.00 -0.02 -0.02  0.01  0.03  0.00 -0.03  0.00  0.04  0.01
##  V20    -0.12 -0.21  0.04  0.11 -0.01  0.06  0.05  0.16  0.01 -0.02 -0.09  0.04
##  V21     0.10 -0.06 -0.10 -0.04  0.04 -0.04  0.03 -0.02  0.08 -0.07 -0.05  0.00
##  V22     0.13 -0.03 -0.06  0.00  0.00 -0.01  0.04 -0.02  0.02  0.00  0.04  0.00
##  V23     0.15  0.15 -0.09 -0.14  0.01 -0.11 -0.05 -0.19  0.08  0.07  0.08  0.03
##  V24    -0.02  0.00  0.01  0.00 -0.01 -0.02 -0.14 -0.01 -0.02  0.00 -0.01 -0.01
##  V25    -0.26  0.05 -0.09  0.03  0.06 -0.02 -0.01 -0.05 -0.07 -0.02 -0.02 -0.01
##  V26     0.00  0.01  0.03 -0.02 -0.06  0.03 -0.02  0.02 -0.01  0.03 -0.03 -0.01
##  V27    -0.04 -0.10  0.13  0.14 -0.03  0.01  0.07 -0.09  0.19  0.00 -0.02 -0.03
##  V28    -0.14 -0.21  0.14  0.18  0.00 -0.03  0.02  0.06  0.09 -0.10 -0.13 -0.03
##  Amount -0.04 -0.09 -0.50  0.00 -0.02 -0.31  0.21 -0.03  0.00 -0.08  0.05 -0.04
##  Class  -0.01 -0.04  0.05 -0.06  0.06 -0.03 -0.04 -0.05  0.02 -0.05 -0.06  0.06
##            V12   V13   V14   V15   V16   V17   V18   V19   V20   V21   V22   V23
##  Time   0.07 -0.05 -0.09 -0.21  0.00 -0.10  0.09  0.03 -0.12  0.10  0.13  0.15
##  V1     0.00  0.05 -0.02  0.03  0.04 -0.06 -0.04  0.03 -0.21 -0.06 -0.03  0.15
##  V2     0.05  0.03  0.09  0.04  0.07  0.00  0.01  0.00  0.04 -0.10 -0.06 -0.09
##  V3    -0.05  0.01 -0.13  0.05 -0.05 -0.04 -0.05 -0.02  0.11 -0.04  0.00 -0.14
##  V4     0.15 -0.02  0.10  0.05 -0.04  0.05  0.02 -0.02 -0.01  0.04  0.00  0.01
##  V5    -0.05  0.02  0.04 -0.04 -0.01 -0.15 -0.02  0.01  0.06 -0.04 -0.01 -0.11
##  V6     0.04  0.00 -0.07 -0.05 -0.02 -0.03  0.05  0.03  0.05  0.03  0.04 -0.05
##  V7    -0.04 -0.02  0.06 -0.05 -0.12 -0.15 -0.07  0.00  0.16 -0.02 -0.02 -0.19
##  V8     0.11 -0.14  0.04 -0.02  0.09  0.07  0.07 -0.03  0.01  0.08  0.02  0.08
##  V9     0.10 -0.05 -0.03  0.00 -0.03 -0.07 -0.01  0.00 -0.02 -0.07  0.00  0.07
##  V10   -0.16  0.00  0.00 -0.03 -0.06 -0.15 -0.05  0.04 -0.09 -0.05  0.04  0.08
##  V11    0.20 -0.04  0.15 -0.04  0.07  0.07  0.07  0.01  0.04  0.00  0.00  0.03
##  V12    1.00  0.19 -0.05 -0.07 -0.12 -0.20 -0.07  0.02  0.01  0.03 -0.01  0.08
##  V13    0.19  1.00 -0.09  0.01  0.00 -0.02 -0.01 -0.01  0.10 -0.05  0.01  0.00
##  V14   -0.05 -0.09  1.00  0.02 -0.13 -0.17 -0.03  0.02 -0.11  0.08 -0.01  0.02
##  V15   -0.07  0.01  0.02  1.00  0.01 -0.02  0.01 -0.05  0.09  0.03  0.00  0.03
##  V16   -0.12  0.00 -0.13  0.01  1.00 -0.24  0.02  0.02  0.18  0.09 -0.05  0.04
##  V17   -0.20 -0.02 -0.17 -0.02 -0.24  1.00 -0.16  0.05 -0.02 -0.01  0.01  0.04
##  V18   -0.07 -0.01 -0.03  0.01  0.02 -0.16  1.00  0.00 -0.11  0.03  0.06 -0.11
##  V19    0.02 -0.01  0.02 -0.05  0.02  0.05  0.00  1.00  0.23 -0.01 -0.01 -0.08
##  V20    0.01  0.10 -0.11  0.09  0.18 -0.02 -0.11  0.23  1.00  0.11  0.03 -0.19
##  V21    0.03 -0.05  0.08  0.03  0.09 -0.01  0.03 -0.01  0.11  1.00  0.68 -0.25
##  V22   -0.01  0.01 -0.01  0.00 -0.05  0.01  0.06 -0.01  0.03  0.68  1.00 -0.24
##  V23    0.08  0.00  0.02  0.03  0.04  0.04 -0.11 -0.08 -0.19 -0.25 -0.24  1.00
##  V24    0.00  0.00 -0.02  0.02 -0.03  0.05 -0.05 -0.06  0.04  0.03  0.01  0.10
##  V25    0.00  0.00  0.04  0.01 -0.02  0.00 -0.01  0.00  0.04  0.04  0.03 -0.43
##  V26    0.01  0.00  0.00 -0.02  0.04 -0.01 -0.03 -0.02  0.05 -0.04 -0.04  0.00
##  V27    0.03  0.00 -0.09  0.03 -0.05  0.05  0.02 -0.02  0.10 -0.03  0.07  0.05
##  V28   -0.01 -0.01 -0.02  0.04 -0.01  0.06  0.02 -0.04  0.28  0.07 -0.04 -0.07
##  Amount -0.04 -0.01  0.00 -0.07 -0.11  0.05  0.05 -0.01  0.20  0.19  0.08 -0.06
##  Class  -0.06  0.00 -0.06  0.00 -0.05 -0.04 -0.03  0.02  0.02  0.04  0.00 -0.01
##            V24   V25   V26   V27   V28 Amount Class
##  Time   -0.02 -0.26  0.00 -0.04 -0.14  -0.04 -0.01
##  V1      0.00  0.05  0.01 -0.10 -0.21  -0.09 -0.04
```

##	V2	0.01	-0.09	0.03	0.13	0.14	-0.50	0.05
##	V3	0.00	0.03	-0.02	0.14	0.18	0.00	-0.06
##	V4	-0.01	0.06	-0.06	-0.03	0.00	-0.02	0.06
##	V5	-0.02	-0.02	0.03	0.01	-0.03	-0.31	-0.03
##	V6	-0.14	-0.01	-0.02	0.07	0.02	0.21	-0.04
##	V7	-0.01	-0.05	0.02	-0.09	0.06	-0.03	-0.05
##	V8	-0.02	-0.07	-0.01	0.19	0.09	0.00	0.02
##	V9	0.00	-0.02	0.03	0.00	-0.10	-0.08	-0.05
##	V10	-0.01	-0.02	-0.03	-0.02	-0.13	0.05	-0.06
##	V11	-0.01	-0.01	-0.01	-0.03	-0.03	-0.04	0.06
##	V12	0.00	0.00	0.01	0.03	-0.01	-0.04	-0.06
##	V13	0.00	0.00	0.00	0.00	-0.01	-0.01	0.00
##	V14	-0.02	0.04	0.00	-0.09	-0.02	0.00	-0.06
##	V15	0.02	0.01	-0.02	0.03	0.04	-0.07	0.00
##	V16	-0.03	-0.02	0.04	-0.05	-0.01	-0.11	-0.05
##	V17	0.05	0.00	-0.01	0.05	0.06	0.05	-0.04
##	V18	-0.05	-0.01	-0.03	0.02	0.02	0.05	-0.03
##	V19	-0.06	0.00	-0.02	-0.02	-0.04	-0.01	0.02
##	V20	0.04	0.04	0.05	0.10	0.28	0.20	0.02
##	V21	0.03	0.04	-0.04	-0.03	0.07	0.19	0.04
##	V22	0.01	0.03	-0.04	0.07	-0.04	0.08	0.00
##	V23	0.10	-0.43	0.00	0.05	-0.07	-0.06	-0.01
##	V24	1.00	0.00	0.01	-0.04	0.05	-0.01	-0.01
##	V25	0.00	1.00	-0.05	-0.09	-0.05	0.02	0.00
##	V26	0.01	-0.05	1.00	-0.16	-0.02	-0.07	0.01
##	V27	-0.04	-0.09	-0.16	1.00	0.46	-0.13	0.03
##	V28	0.05	-0.05	-0.02	0.46	1.00	0.01	0.02
##	Amount	-0.01	0.02	-0.07	-0.13	0.01	1.00	-0.01
##	Class	-0.01	0.00	0.01	0.03	0.02	-0.01	1.00



3 Analysis - Models Building and Comparison

3.1 Data Pre Processing

1. Remove the "Time" column and Change 'Class' variable to factor from the dataset

```
# Set seed for reproducibility
set.seed(1234)
# Remove the "Time" column from the dataset
df <- df %>% select(-Time)
#Change 'Class' variable to factor df$Class <- as.factor(df$Class)
```

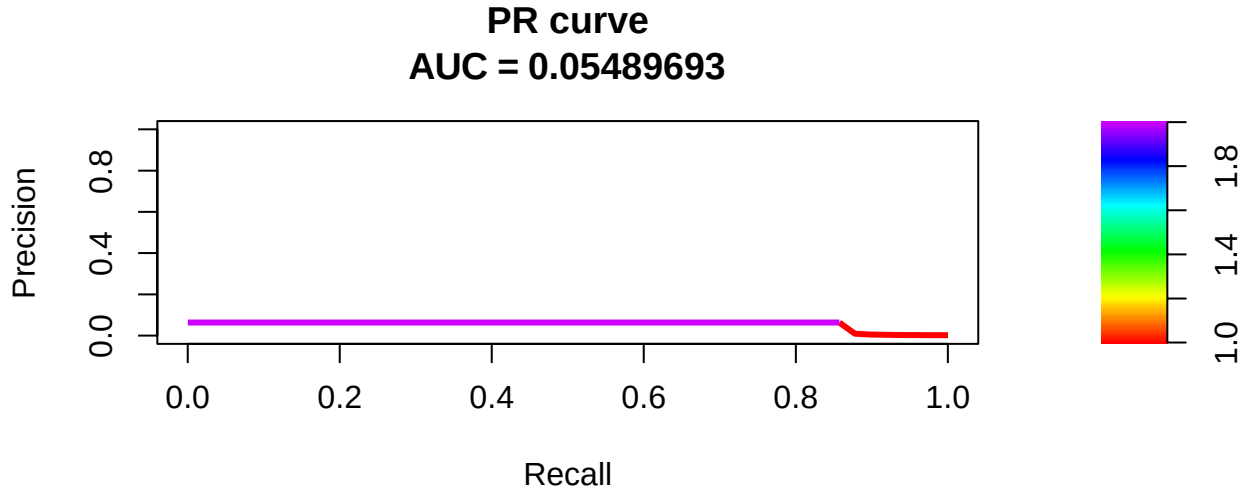
2. Split the dataset into train, test, cv dataset

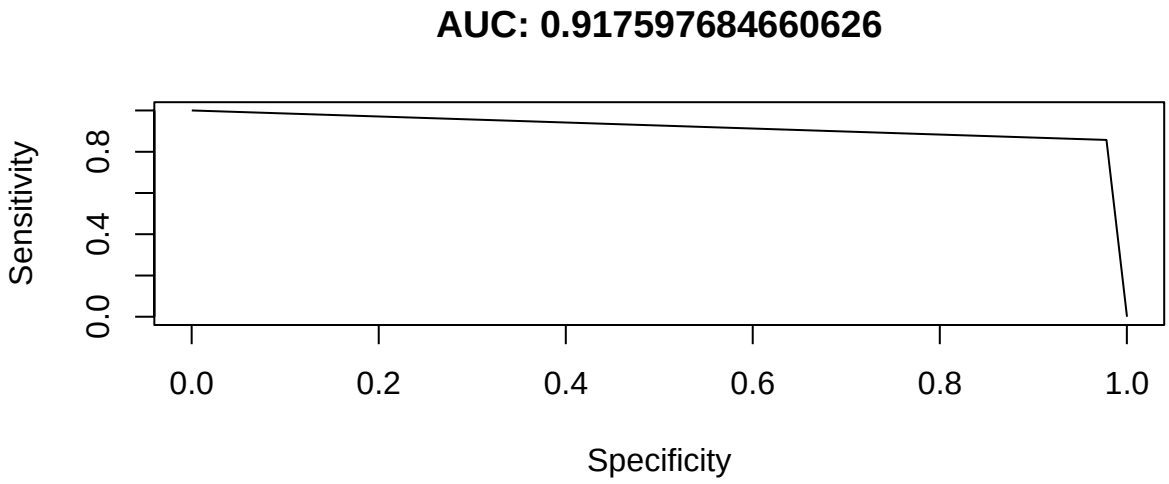
```
# Split the dataset into train, test and cross validation dataset train_index = createDataPartition(y =
df$Class,
                                                    p = .6, list = F)
train <- df[train_index,] test_cv <-df[-
train_index,]

test_index = createDataPartition(y = test_cv$Class,
                                                    p = .5, list = F)

 test <- test_cv[test_index,] cv <- test_cv[-test_index,] rm(train_index, test_index, test_cv)
```

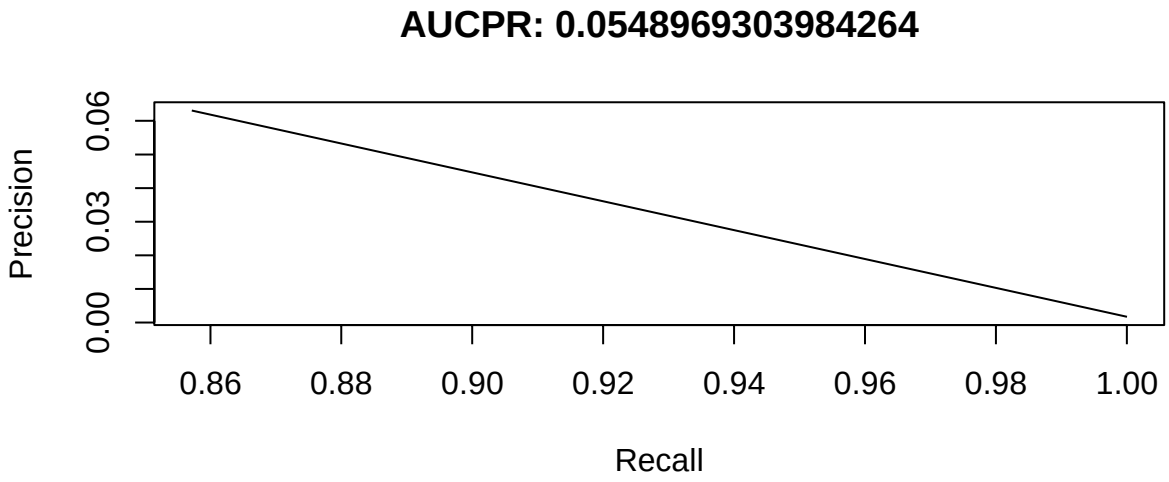
3.2 Classification Models

Classification is the process of predicting discrete variables (1/0, Yes/no, etc.). Given the case with our dataset, it will be more optimistic to deploy a classification model rather than any others. To better understand which algorithm would perform best on the given dataset, the following algorithms are used:Naive Bayes, KNN, Random Forest,XGBoost

3.2.1 Naive Algorithm



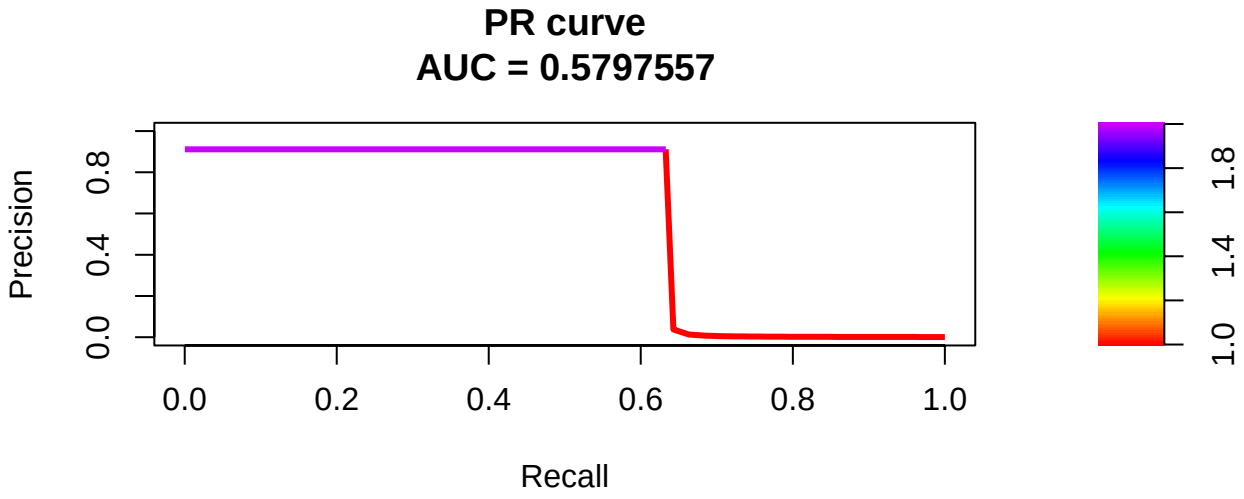


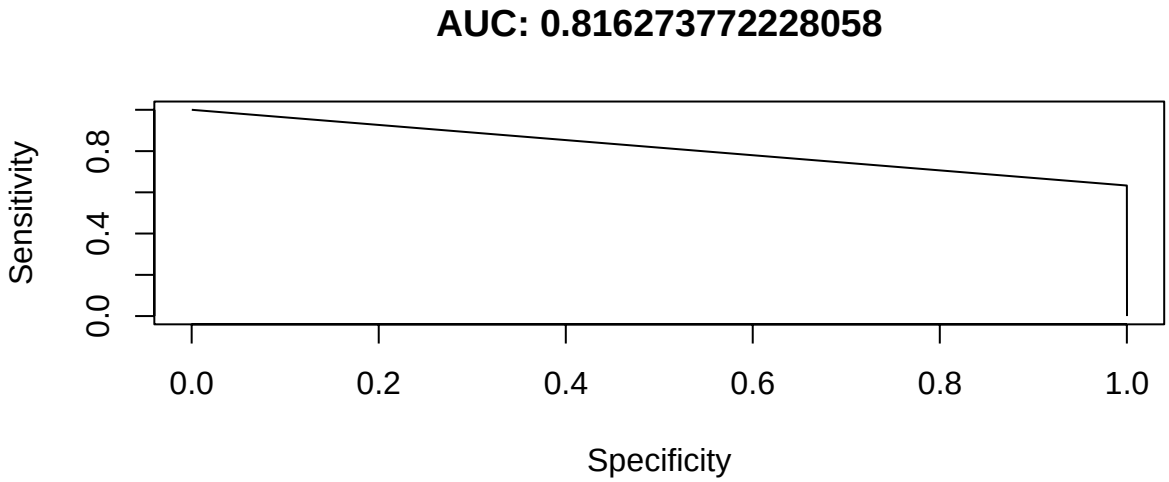


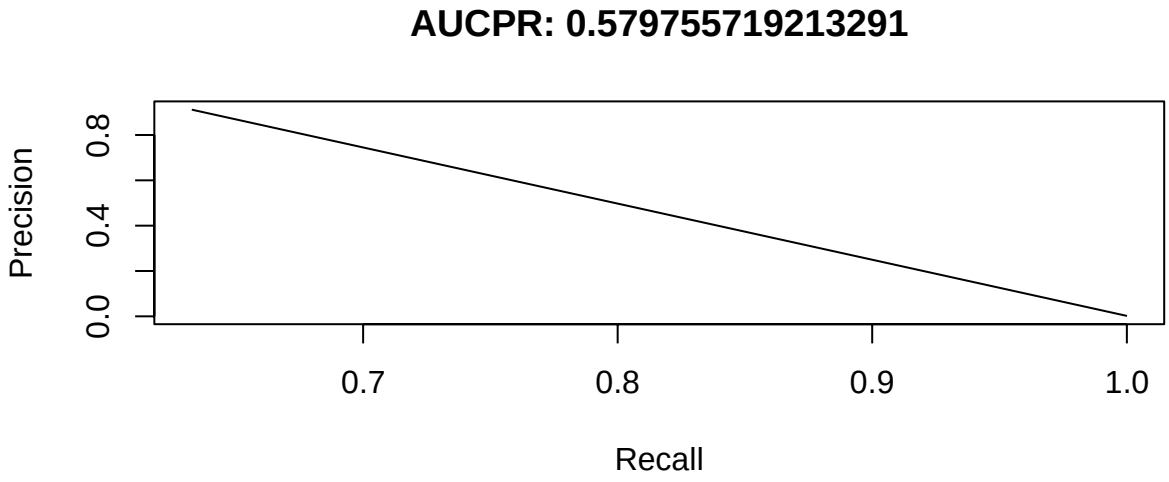
Model	AUC	AUCPR
Naive Bayes	0.9175977	0.0548969

3.2.2 KNN

A KNN Model with k=5 can achieve a significant improvement in respect to the previous models, as regard AUCPR of **0.58** at the expense of a little drop off AUC, that is **0.81**.



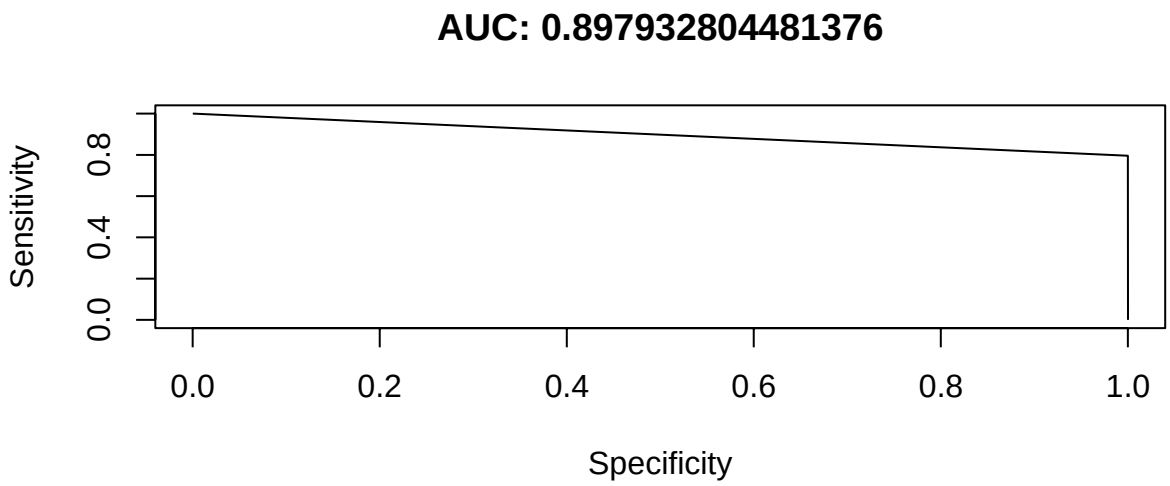


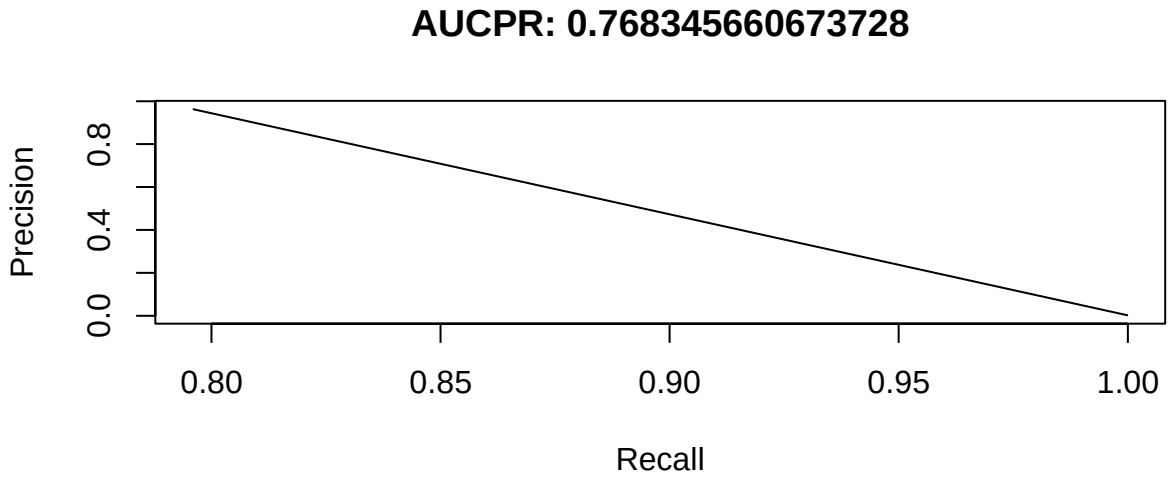


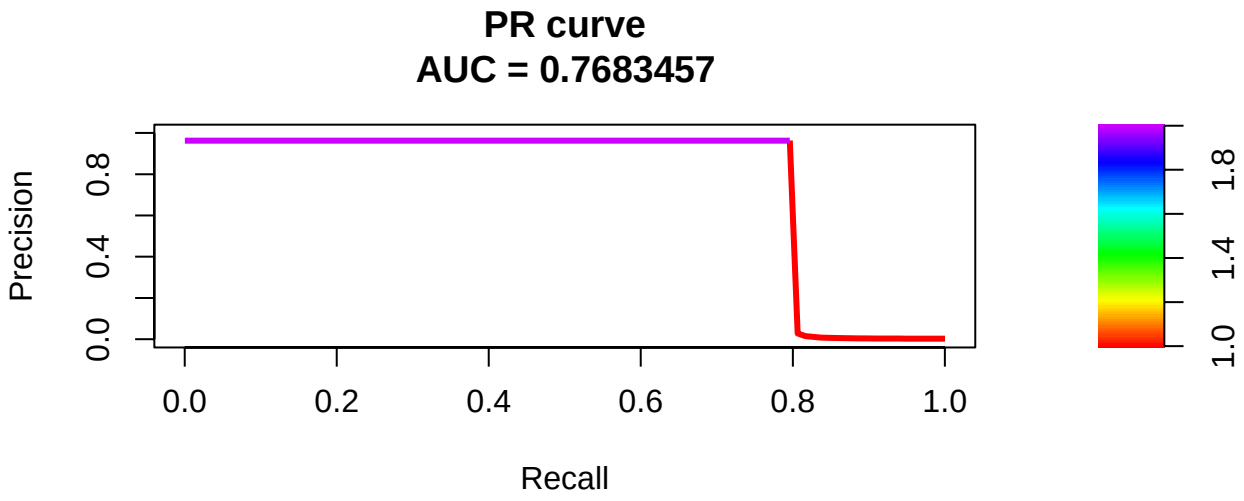
Model	AUC	AUCPR
Naive Bayes	0.9175977	0.0548969
K-Nearest Neighbors k=5	0.8162738	0.5797557

3.2.3 Random Forest

The ensemble methods are capable of a significant increase in performance. At the expense of another little drop off in terms of AUC (**0.9**) respect to the Naive Bayes model, there is a huge step forward in terms of AUCPR, that is **0.77**. This model doesn't reach the desidered performance (AUCPR > 0.85), but it's close to it. As the plot and the table below suggest, there are few predictors like **V17**, **V12** and **V14** that are particularly useful for classifying a fraud.







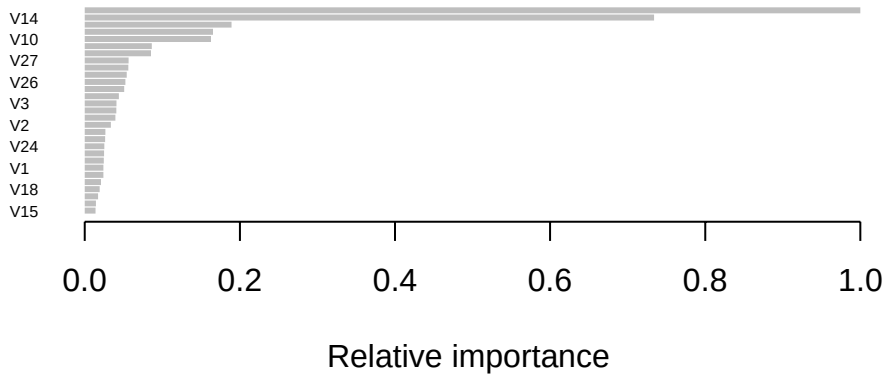
Model	AUC	AUCPR
Naive Bayes	0.9175977	0.0548969
K-Nearest Neighbors k=5	0.8162738	0.5797557
Random Forest	0.8979328	0.7683457

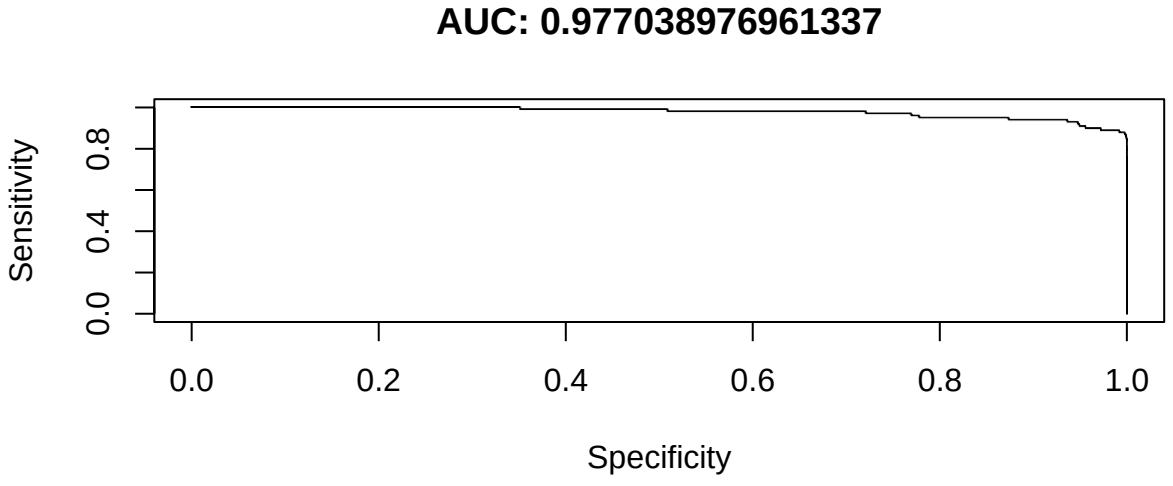
	MeanDecreaseGini
V1	8.708982
V2	7.784292
V3	8.985490
V4	17.257080
V5	7.772203
V6	8.821890
V7	19.072039
V8	7.013489
V9	23.520504
V10	43.772484
V11	44.997607
V12	73.056009
V13	6.829304
V14	63.479173
V15	6.388524
V16	40.124086
V17	105.084852
V18	16.236771
V19	8.041600
V20	8.359602
V21	10.723973
V22	5.886333
V23	4.705090
V24	6.127916
V25	5.290926
V26	10.888757
V27	9.216603
V28	6.266699
Amount	7.974071

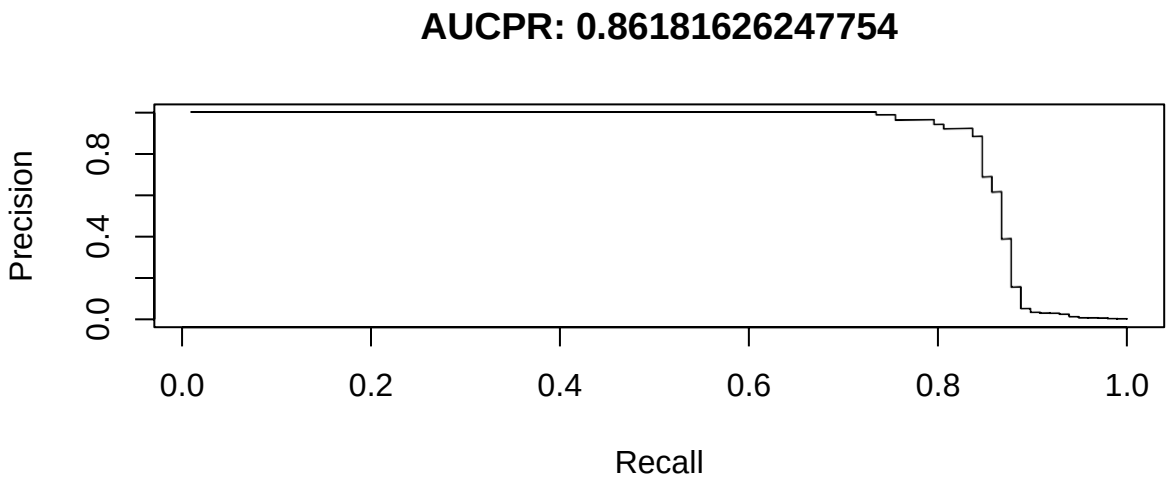
3.2.4 XGBoost

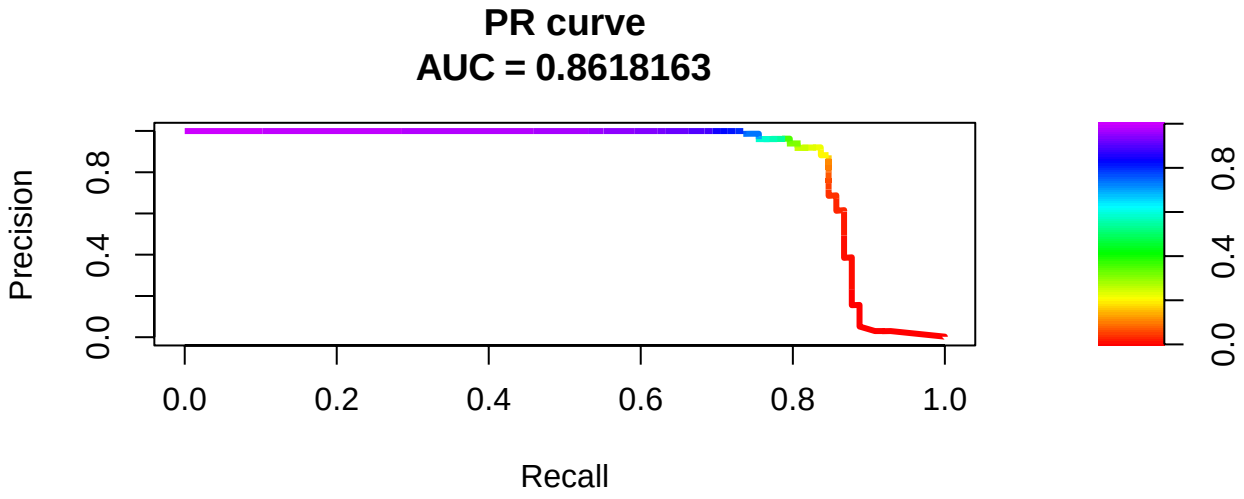
XGBoost are a top class model. It always stays on TOP5 (or wins them) in every competitions on Kaggle and in this case, its' very fast to train and its performance are awesome. With an AUC of **0.98** and an AUCPR of **0.86** it reach and overtake the desidered performance. As the previous model shown, **V17** and **V14** are still relevant to predict a fraud.

```
## [1]    test-aucpr:0.658215 cv-aucpr:0.651097
## Multiple eval metrics are present. Will use cv_aucpr for early stopping.
## Will train until cv_aucpr hasn't improved in 40 rounds.
##
## [101] test-aucpr:0.857385 cv-aucpr:0.877270
## [201] test-aucpr:0.862116 cv-aucpr:0.886406
## Stopping. Best iteration:
## [190] test-aucpr:0.861816 cv-aucpr:0.887686
```









Model	AUC	AUCPR
Naive Bayes	0.9175977	0.0548969
K-Nearest Neighbors k=5	0.8162738	0.5797557
Random Forest	0.8979328	0.7683457
XGBoost	0.9770390	0.8618163

Feature	Gain	Cover	Frequency	Importance
V17	0.3171657	0.3376840	0.0590406	0.3171657
V14	0.2328285	0.4247761	0.0974170	0.2328285
V4	0.0600361	0.0149544	0.0900369	0.0600361
V7	0.0524206	0.0016778	0.0487085	0.0524206
V10	0.0515966	0.0024414	0.0442804	0.0515966
V12	0.0274032	0.1442810	0.0457565	0.0274032
Amount	0.0270669	0.0014754	0.0568266	0.0270669
V27	0.0179538	0.0006398	0.0265683	0.0179538
V28	0.0178111	0.0008319	0.0324723	0.0178111
V20	0.0171806	0.0008593	0.0250923	0.0171806
V26	0.0166046	0.0006860	0.0332103	0.0166046
V9	0.0161372	0.0059450	0.0265683	0.0161372
V19	0.0139521	0.0008483	0.0346863	0.0139521
V3	0.0129482	0.0014248	0.0391144	0.0129482
V8	0.0128923	0.0008873	0.0280443	0.0128923
V5	0.0125336	0.0188990	0.0324723	0.0125336
V2	0.0106854	0.0006103	0.0228782	0.0106854
V21	0.0084312	0.0007444	0.0191882	0.0084312
V23	0.0083561	0.0280382	0.0265683	0.0083561
V24	0.0079779	0.0005232	0.0250923	0.0079779
V22	0.0079069	0.0011115	0.0228782	0.0079069
V13	0.0077632	0.0008035	0.0243542	0.0077632
V1	0.0076040	0.0006159	0.0295203	0.0076040
V16	0.0076017	0.0069315	0.0258303	0.0076017
V11	0.0066428	0.0006218	0.0177122	0.0066428
V18	0.0060901	0.0004219	0.0199262	0.0060901
V6	0.0054157	0.0004609	0.0169742	0.0054157
V25	0.0045781	0.0004818	0.0169742	0.0045781
V15	0.0044156	0.0003236	0.0118081	0.0044156

4 Results

This is the summary results for all the models builted, trained and validated.

Model	AUC	AUCPR
Naive Bayes	0.9175977	0.0548969
K-Nearest Neighbors k=5	0.8162738	0.5797557
Random Forest	0.8979328	0.7683457
XGBoost	0.9770390	0.8618163

5 Conclusion

The ensemble methods once again confirm themselves as among the best models out there. It easy to find them as a winners of numerous Kaggle's competitions or on TOP5 of them. In this task, a XGBoost model can achieve a very good AUCPR result of **0.86** and the others ensembe methods are very close to it. As the features importance plots and table show, there are few predictors like **V17** and **V14** that are particularly useful for classifying a fraud. The SMOTE technique (a technique for dealing with imbalanced data) could improve the performance a little bit.

6 Literature

1.Rafael A. Irizarry, Introduction to Data Science

2.HarvardX Data Science Program