Mobile Money Transaction Fraud Detection

Rashmy Patwari

3/4/2021

Abstract

This report is part of the final assignment for the Harvard Data Science Professional Program. Special thanks to Prof. of Biostatistics Rafael Irizarry from Harvard University. For this assignment I choose to use my Machine Learning understanding to classify Mobile Transaction Frauds into Legal / Fraudulant.

Contents

1 Executive Summary **2**

2 Exploratory Data Analysis **2**

2.1 The Dataset . . . 2

2.2 Source . . . 2

2.3 Dimensions . . . 2

2.4 Imbalanced Dataset . . . 2

2.5 Missing Values . . . 3

2.6 Top 10 Rows of `PaySim` dataset . . . 4

2.7 Frauds Amount Distributions . . . 4

2.8 Frauds over Time Distribution . . . 5

2.9 Correlations between each variables . . . 7

3 Data Pre-Processing **8**

4 Analysis - Models Building and Comparison **8**

4.1 Naive Baseline Algorithm - Predict Always "Legal" Transaction . . . 8

4.2 Naive Bayes . . . 10

4.3 KNN - K-Nearest Neighbors . . . 12

4.4 Random Forest . . . 16

4.5 GBM - Generalized Boosted Regression . . . 19

4.6 XGBoost . . . 24

5 Results **29**

6 Final Analysis **30**

1 Executive Summary

The Goal of this project is to evaluate Machine Learning Classification models and choose an optimal one. This model can predict if the mobile transaction is legal or fraudulent.

Due to imbalanced nature of the data, many observations could be predicted as False Negative, in this case Legal Transactions instead of Fraudulent Transaction. For example, a model that predict always **0** (Legal) can archive an Accuracy of **99.8**. For that reason, the metric used for measuring the score is the **Area Under The Precision-Recall Curve (AUCPR)** instead of the traditional AUC curve. Aiming to reach an AUCPR greater than **0.80**.

For achieving the task of classifying mobile transaction fraud detection, I trained several classification algorithms such as Naive Bayes Classifier, KNN, Random Forest, GBM and XGBoost.

2 Exploratory Data Analysis

2.1 The Dataset

PaySim simulates mobile money transactions based on a sample of real transactions extracted from one month of financial logs from a mobile money service implemented in an African country. The original logs were provided by a multinational company, who is the provider of the mobile financial service which is currently running in more than 14 countries all around the world.

This synthetic dataset is scaled down 1/4 of the original dataset and it is created just for Kaggle.

2.2 Source

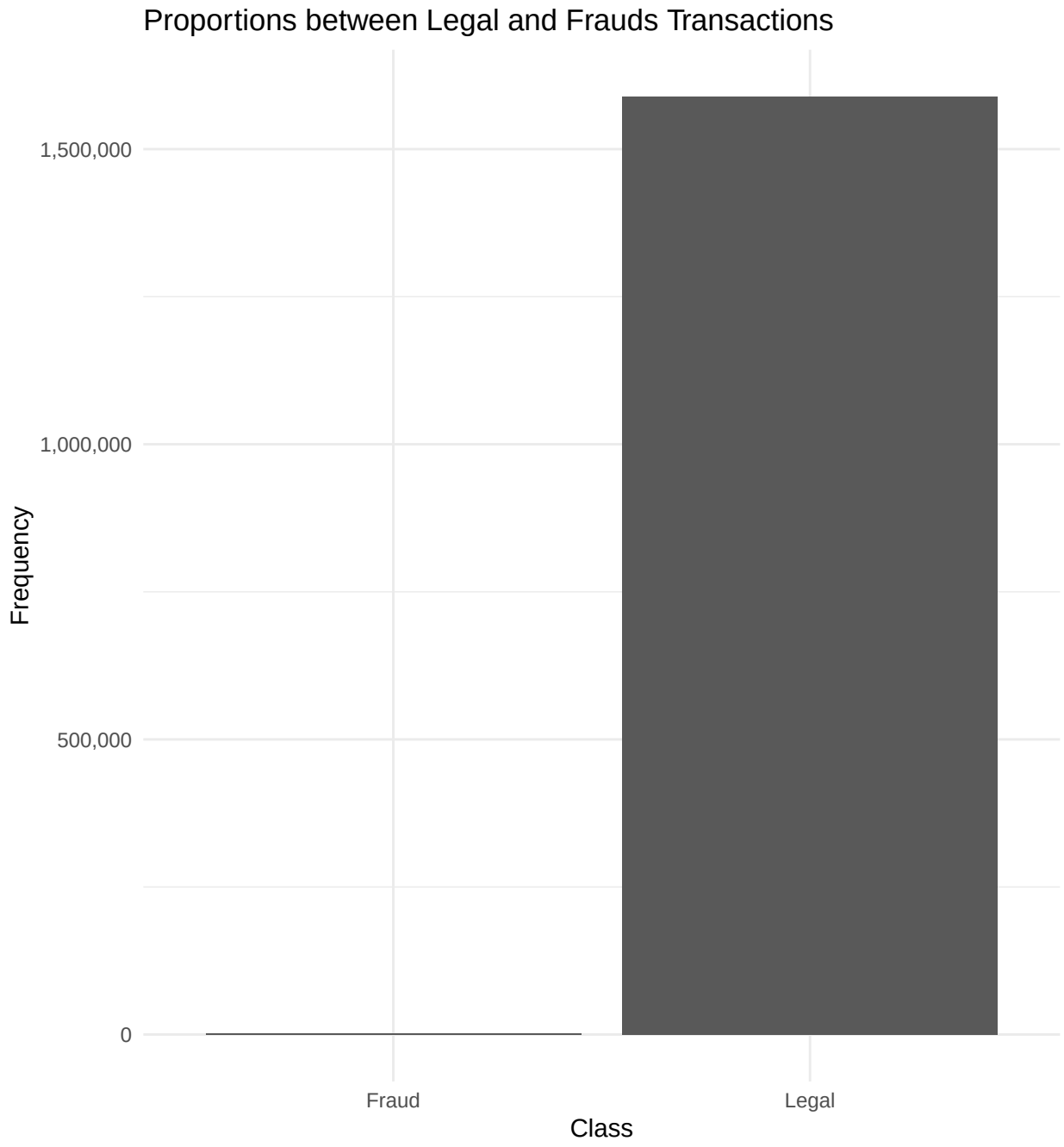
https://www.kaggle.com/ntnu-testimon/paysim1

2.3 Dimensions

Length	Columns
1590655	11

2.4 Imbalanced Dataset

This is a very imbalanced dataset. It means that there are few rows that represent a Fraud. In this case, only **133** transactions are frauds, represented by **1** and **158932** are not, represented by **0**.



isFraud	Count
0	1588893
1	1762

2.5 Missing Values

As the table below suggests, there aren't missing values in this dataframe.

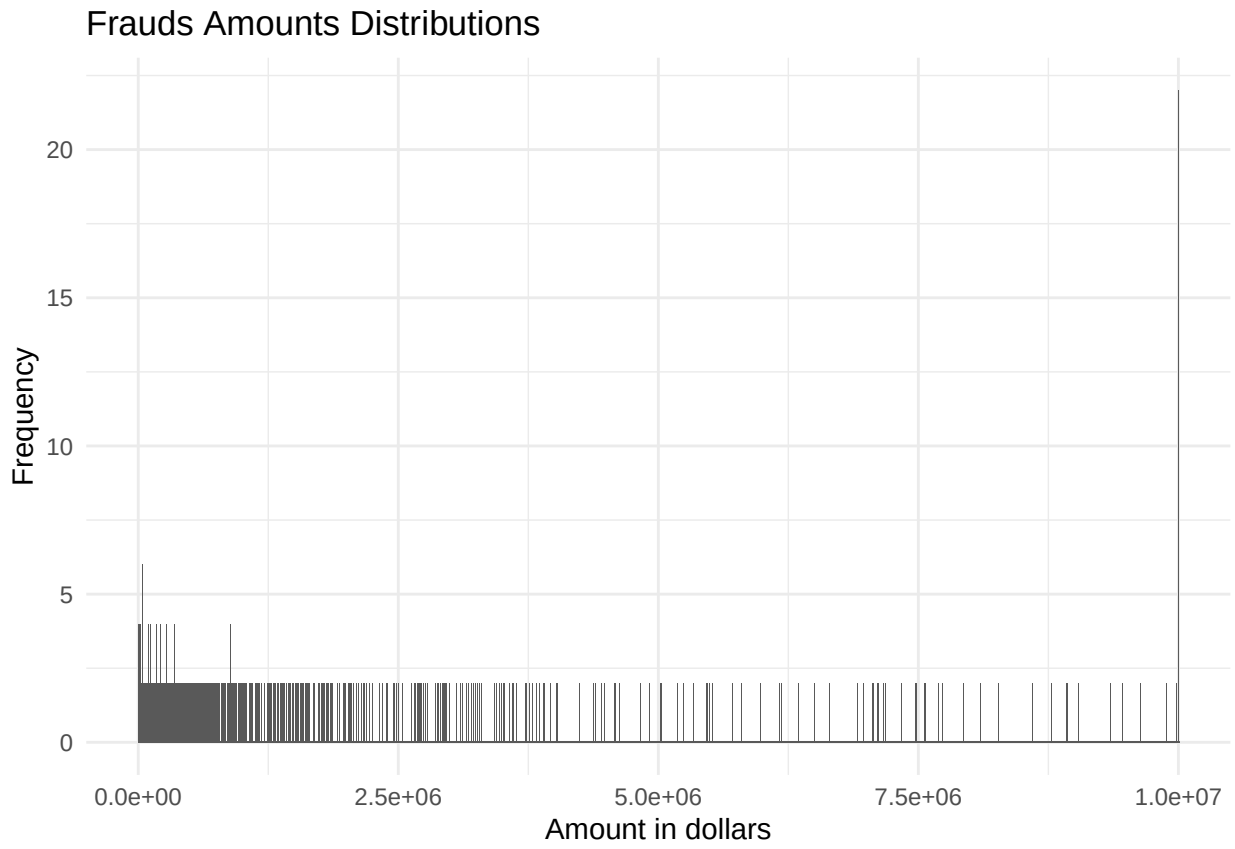
	Missing Values
step	0
type	0
amount	0
nameOrig	0
oldbalanceOrg	0
newbalanceOrig	0
nameDest	0
oldbalanceDest	0
newbalanceDest	0
isFraud	0
isFlaggedFraud	0

2.6 Top 10 Rows of `PaySim` dataset

step	type	amount	nameOrig	nameDest	isFraud
1	PAYMENT	9839.64	C1231006815	M1979787155	0
1	PAYMENT	1864.28	C1666544295	M2044282225	0
1	TRANSFER	181.00	C1305486145	C553264065	1
1	CASH_OUT	181.00	C840083671	C38997010	1
1	PAYMENT	11668.14	C2048537720	M1230701703	0
1	PAYMENT	7817.71	C90045638	M573487274	0
1	PAYMENT	7107.77	C154988899	M408069119	0
1	PAYMENT	7861.64	C1912850431	M633326333	0
1	PAYMENT	4024.36	C1265012928	M1176932104	0
1	DEBIT	5337.77	C712410124	C195600860	0

2.7 Frauds Amount Distributions

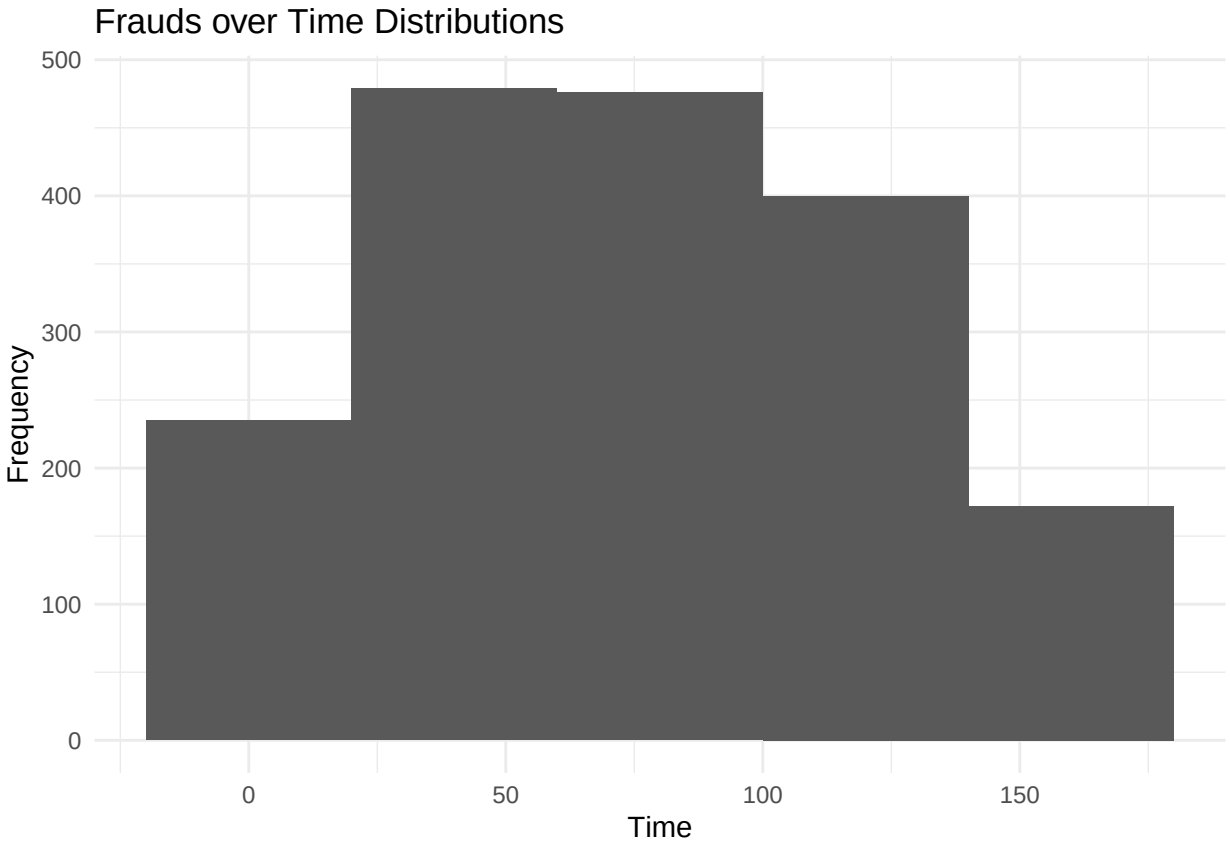
Large sums of money **10000000.00** are scammed most often.



amount	count
1.0000e+07	22
1.1900e+02	2
1.6400e+02	2
1.7000e+02	2
1.8100e+02	2
2.1583e+02	2
2.2200e+02	2
4.0800e+02	2
6.3600e+02	2
1.0550e+03	2

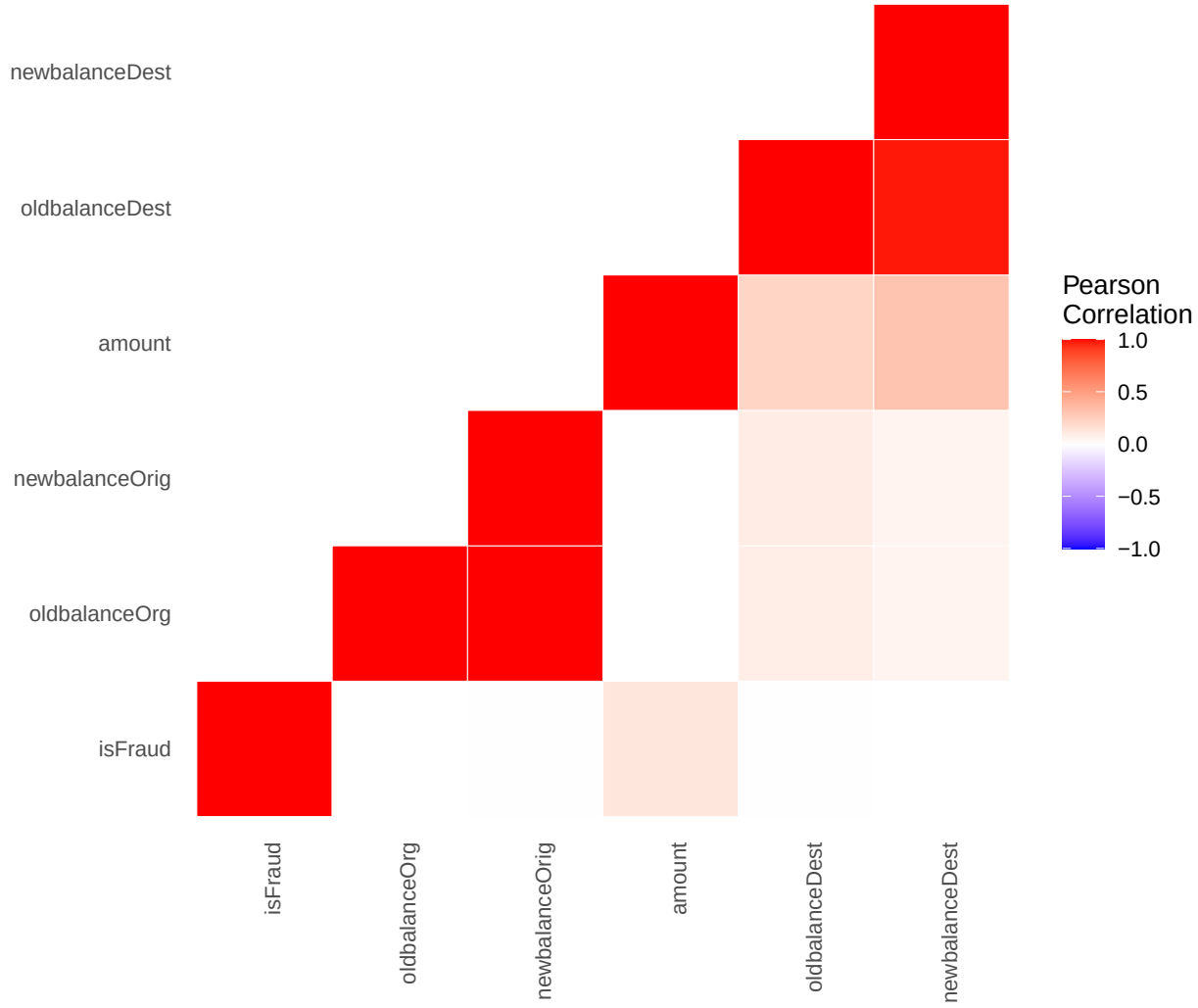
2.8 Frauds over Time Distribution

There aren't correlation between `time` and frauds. A fraud can happen anytime. It seems not particularly useful for the modelling phase. The correlation matrix below, confirms this assumpion.



step	count
66	24
22	23
6	22
34	22
74	22
149	22
47	21
15	20
58	20
59	20

2.9 Correlations between each variables



3 Data Pre-Processing

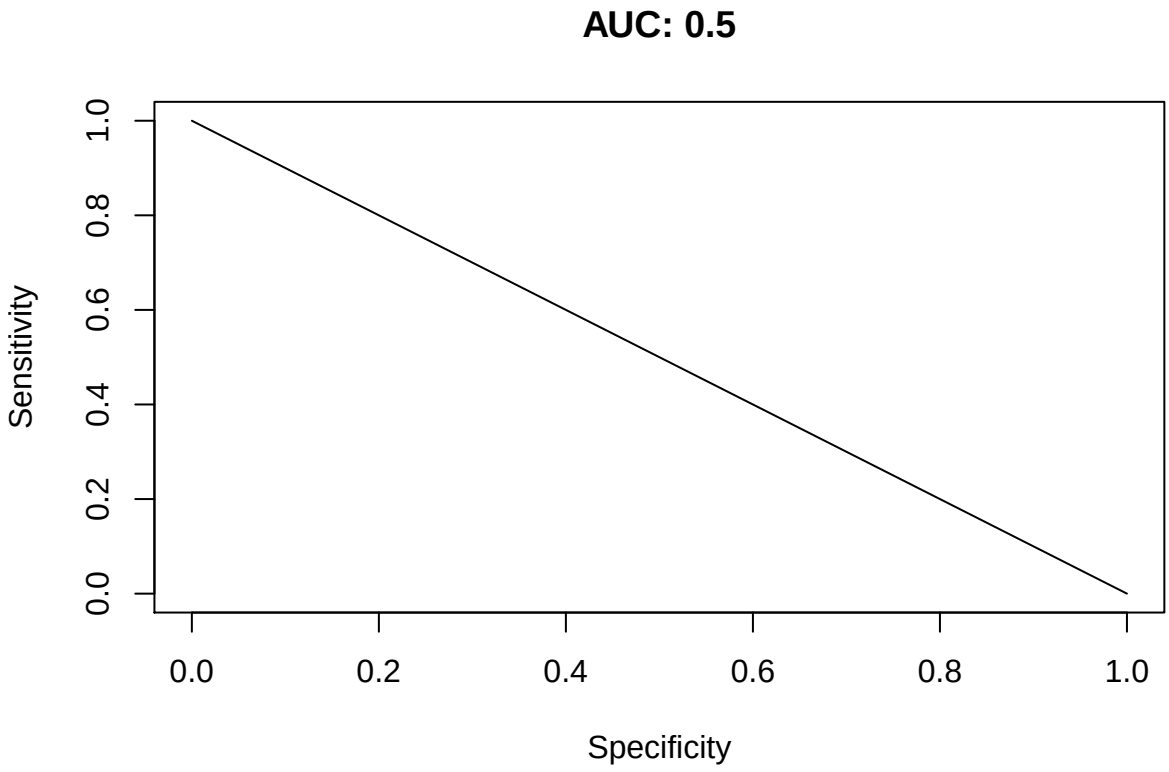
Before continuing to build models, lets do some data pre-processing:

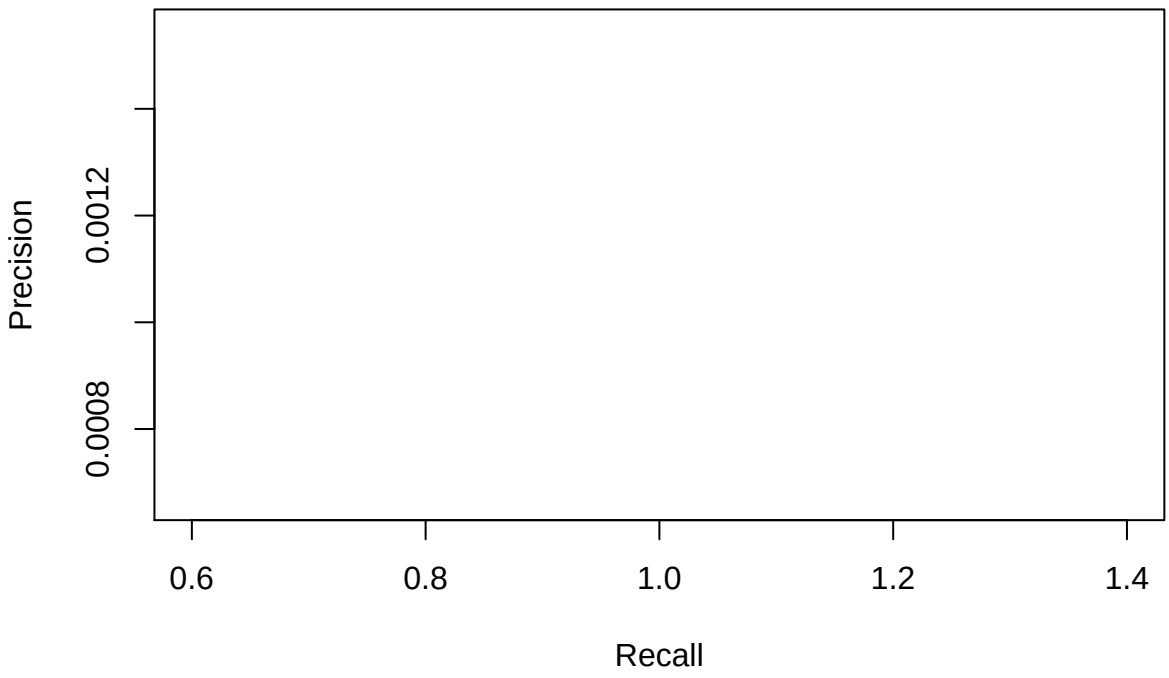
1. Remove the unwanted Columns. Step,type nameOrig,nameDest and isFlaggedFraud.
2. Split the dataset into train, test, cv dataset.

4 Analysis - Models Building and Comparison

4.1 Naive Baseline Algorithm - Predict Always "Legal" Transaction

Predicting always "Legal" transaction can achieve an impressive accuracy of **99.8** and an AUC of **0.5**. Because the recall and precision are **0**, it is impossible to compute the AUCPR, so that is **0**.

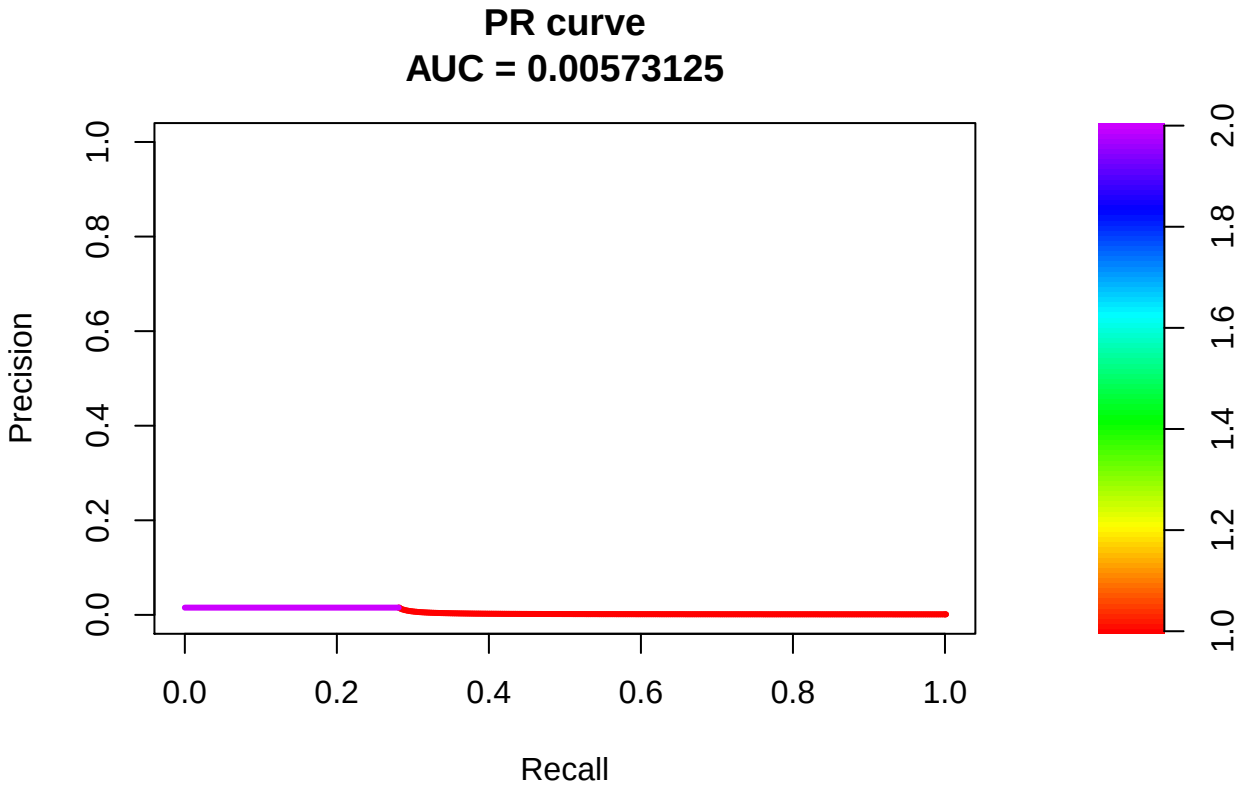


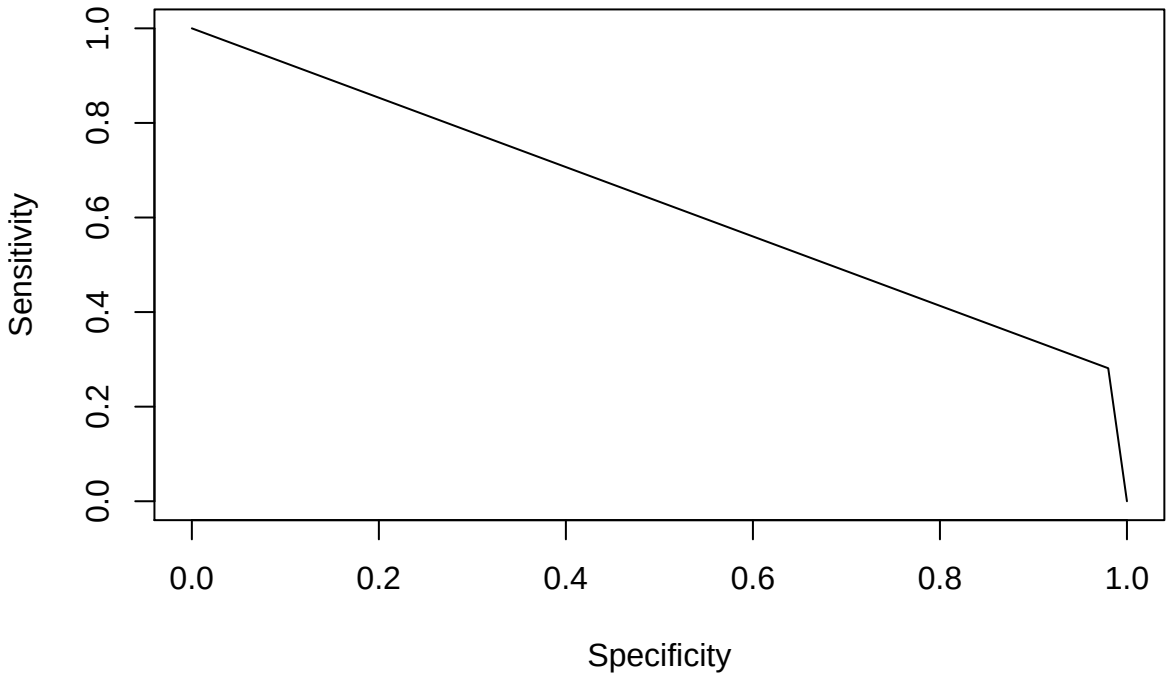


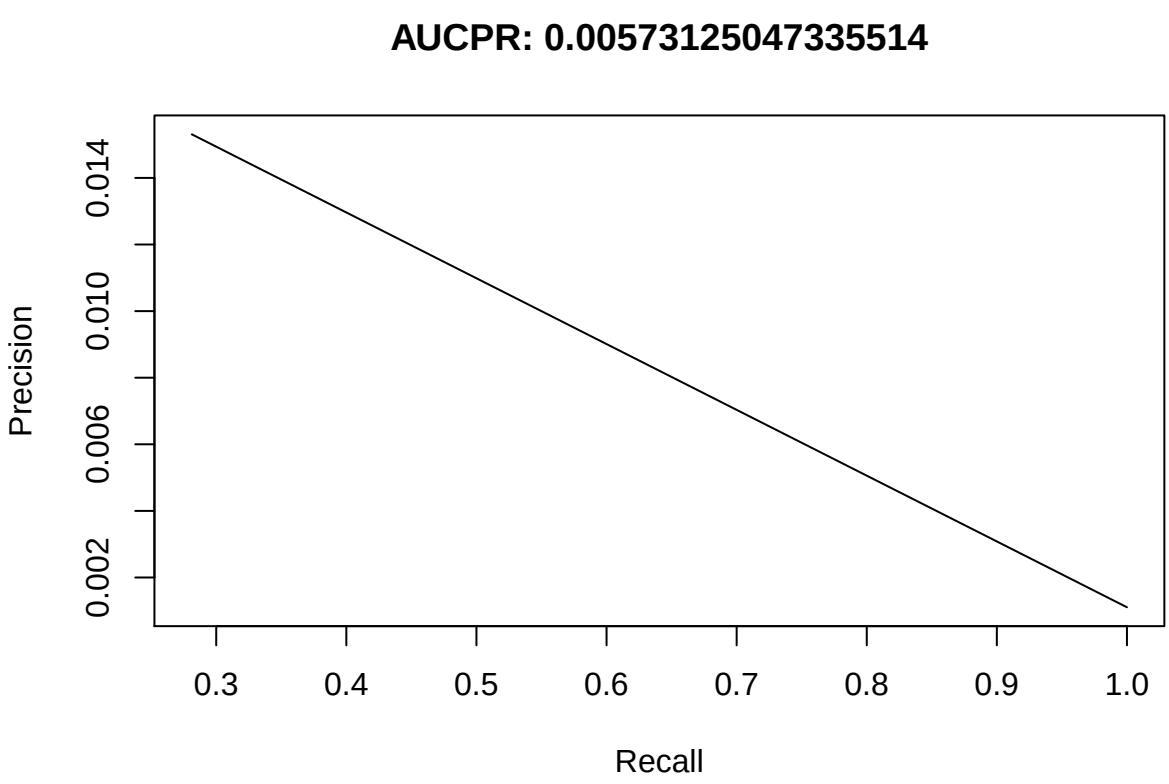
Model	AUC	AUCPR
Naive Baseline - Predict Always Legal	0.5	0

4.2 Naive Bayes

A step forward is building a Naive Bayes Classifier. The performance improve a little bit: AUC is **0.630** and now we have an AUCPR of **0.0057**. It is a poor result according to the metric of interest and it is easy to improve.



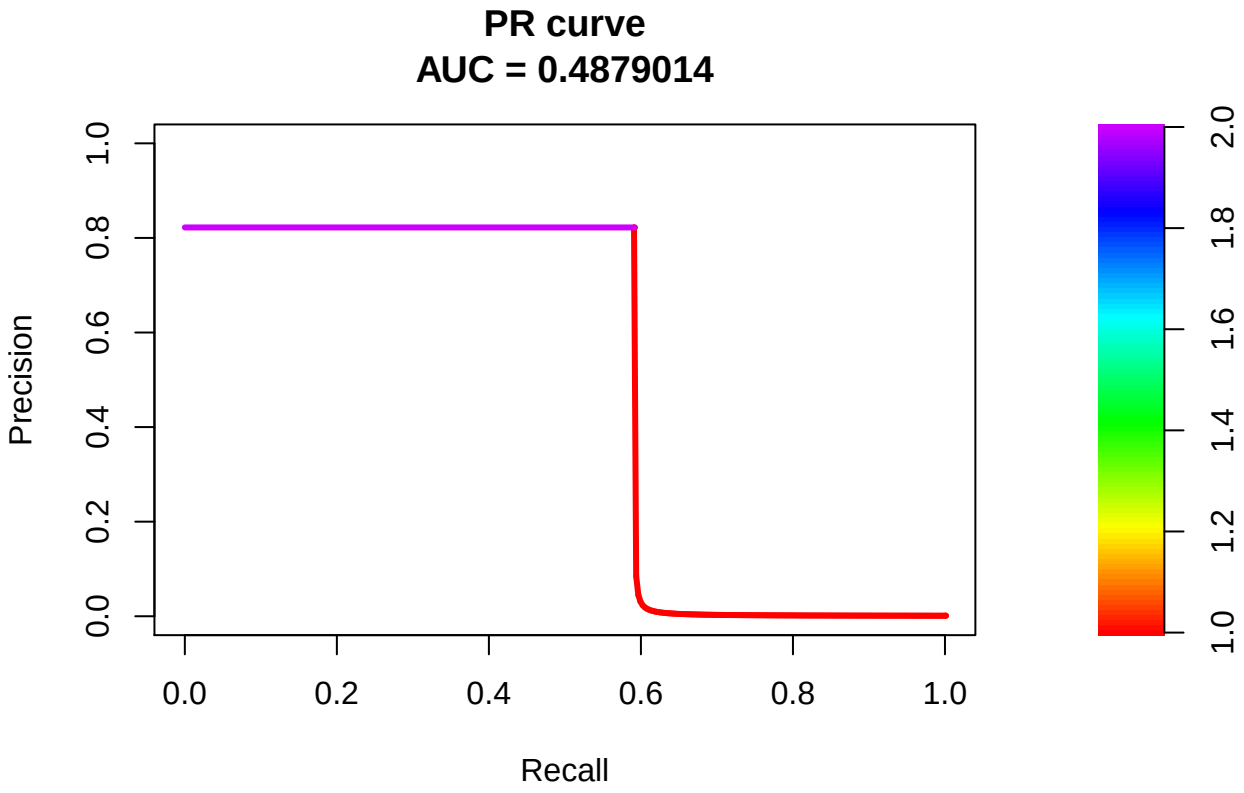

AUC: 0.630605458117119
Sensitivity
0.0
0.2
0.4
0.6
0.8
1.0
Specificity
0.0
0.2
0.4
0.6
0.8
1.0

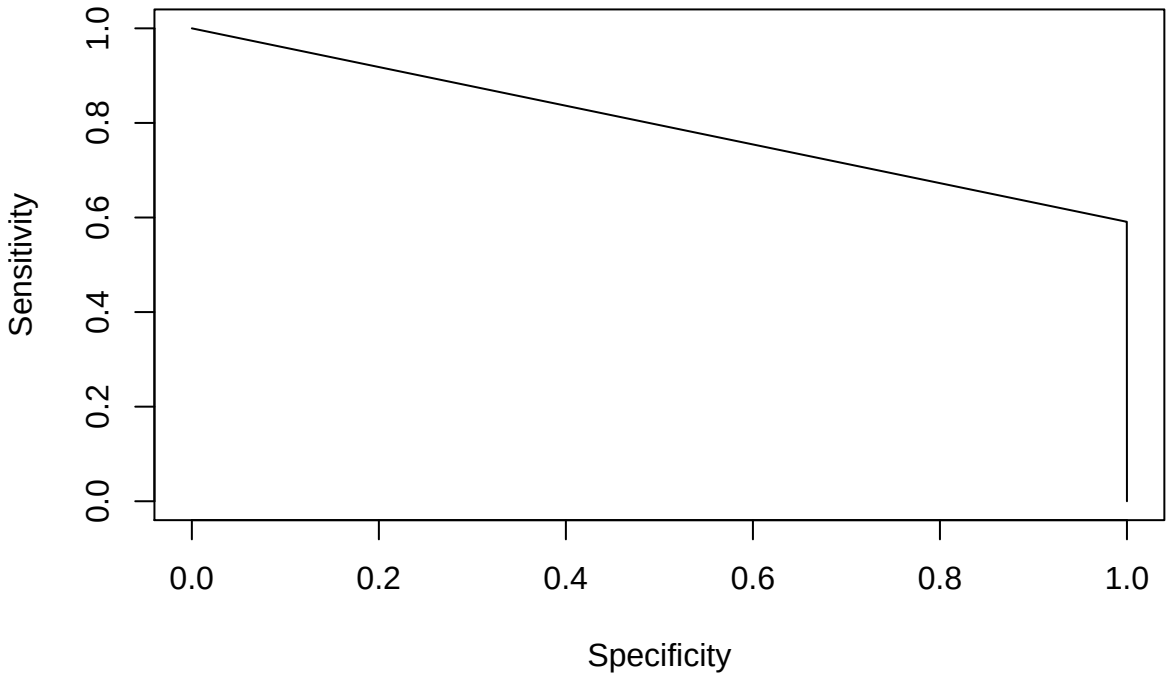


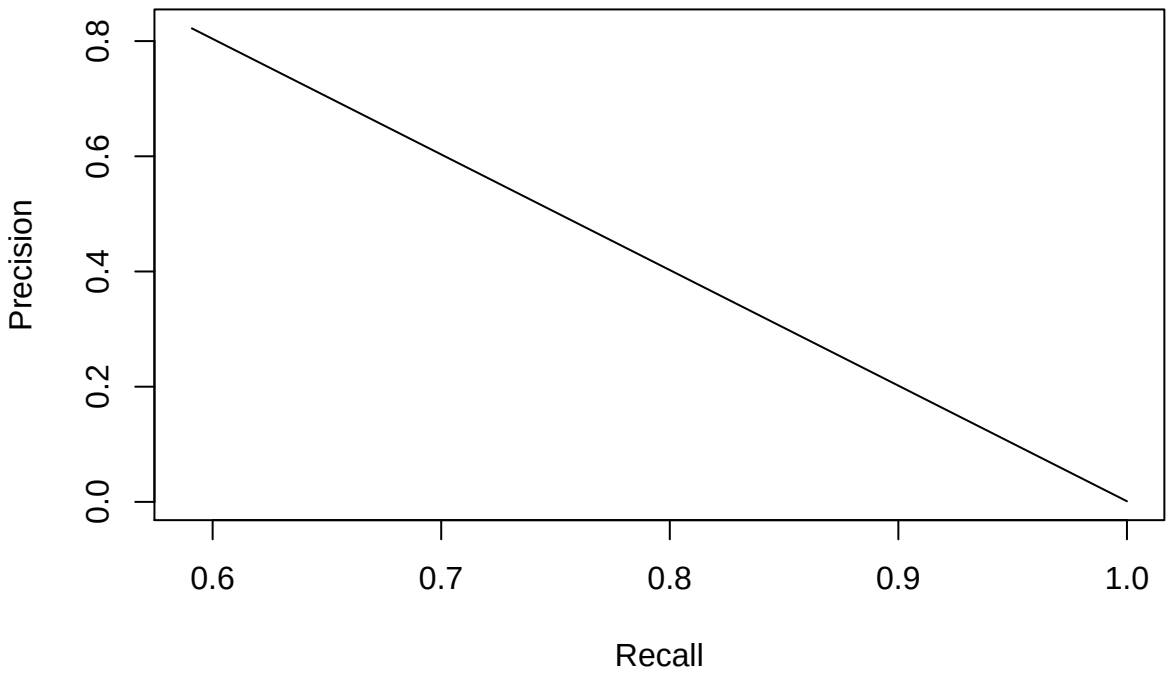
Model	AUC	AUCPR
Naive Baseline - Predict Always Legal	0.5000000	0.0000000
Naive Bayes	0.6306055	0.0057313

4.3 KNN - K-Nearest Neighbors

A KNN Model with k=5 can achieve a significant improvement in respect to the previous models, as regard AUCPR of **0.487** and AUC **0.795**.


PR curve
AUC = 0.4879014
Precision
0.0
0.2
0.4
0.6
0.8
1.0
Recall
0.0
0.2
0.4
0.6
0.8
1.0
1.0
1.2
1.4
1.6
1.8
2.0

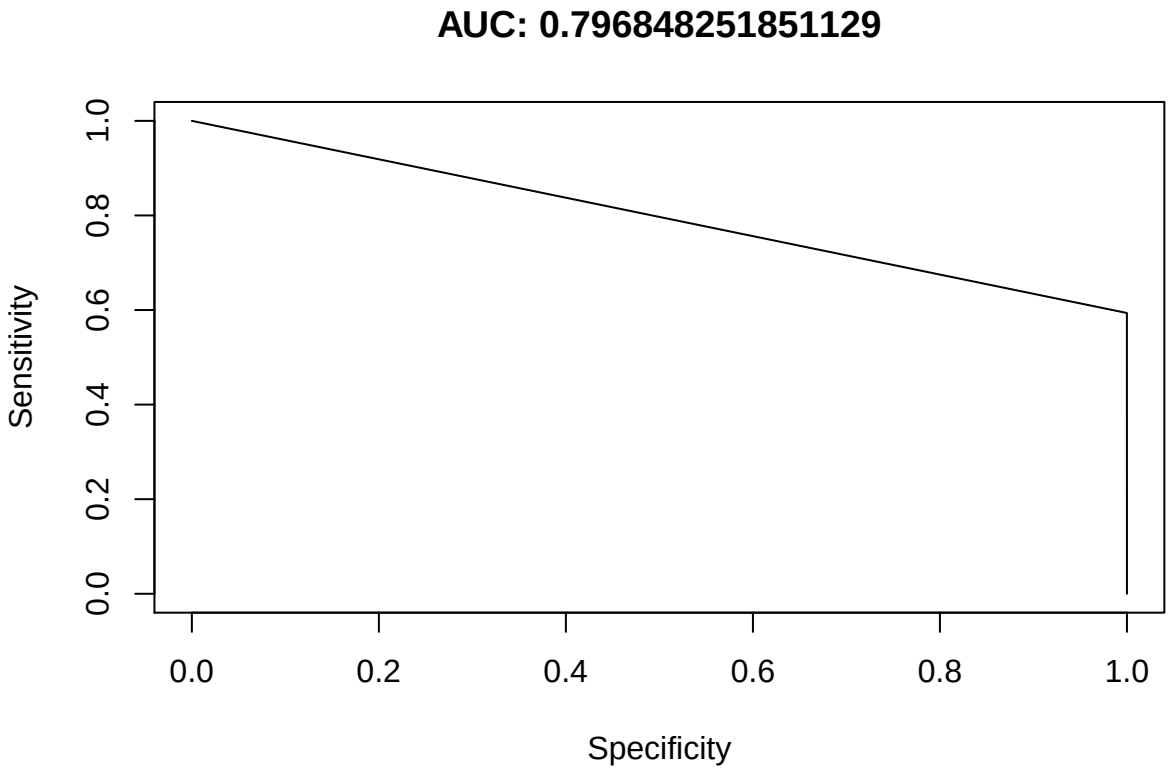

AUC: 0.795383741531064
Sensitivity
0.0
0.2
0.4
0.6
0.8
1.0
Specificity
0.0
0.2
0.4
0.6
0.8
1.0

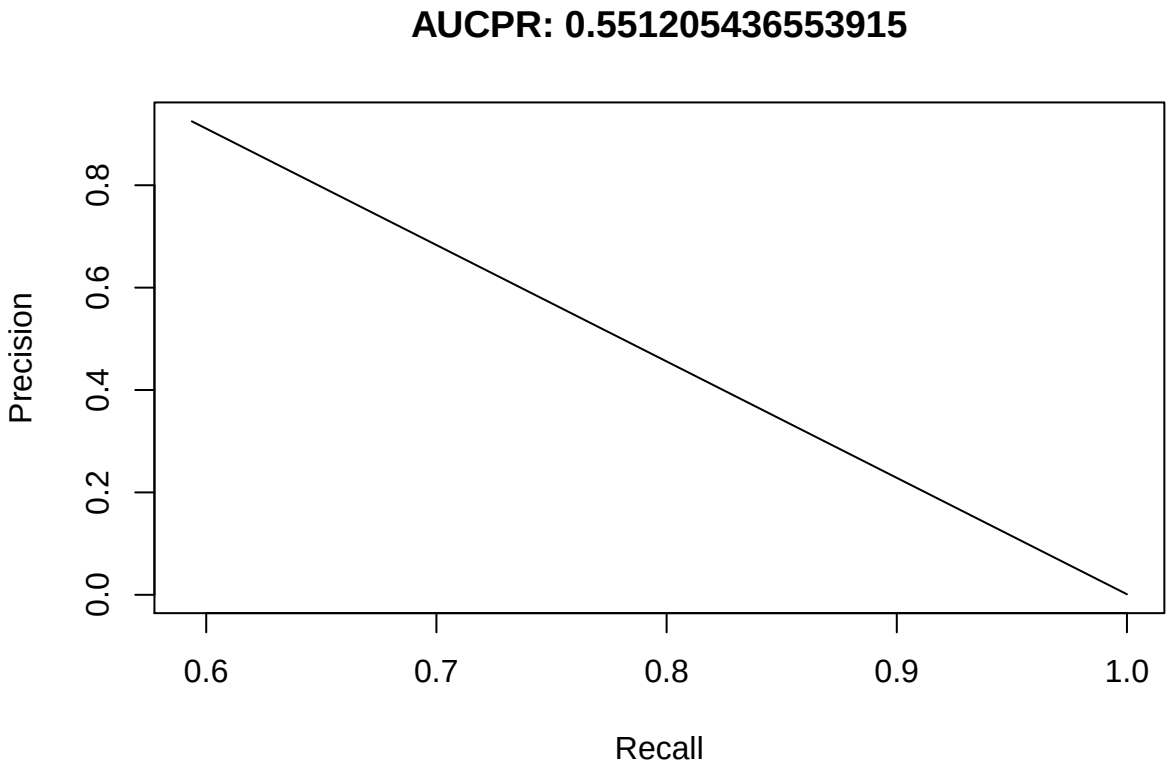


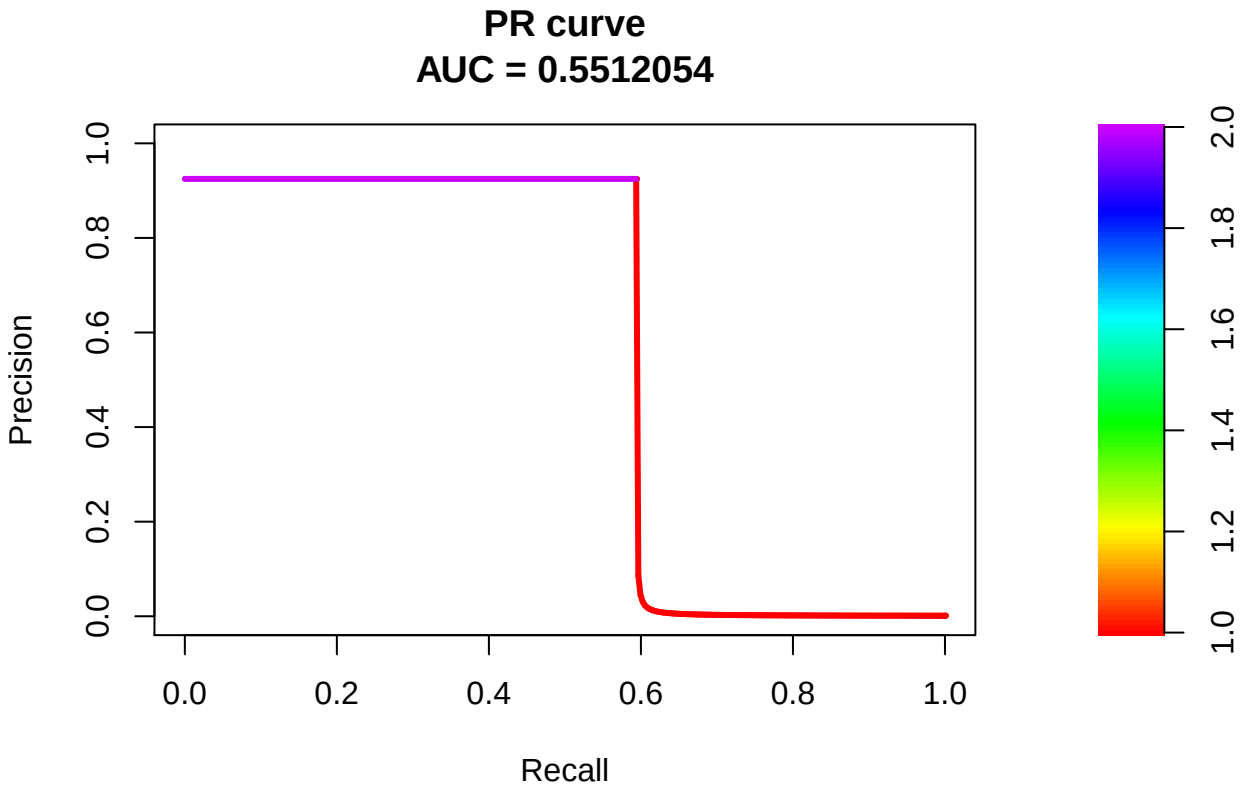
Model	AUC	AUCPR
Naive Baseline - Predict Always Legal	0.5000000	0.0000000
Naive Bayes	0.6306055	0.0057313
K-Nearest Neighbors k=5	0.7953837	0.4879014

4.4 Random Forest

The ensemble methods are capable of a significant increase in performance. Although there is a huge step forward in terms of AUCPR, that is **0.77** the model ha not reached the performance goal (AUCPR > 0.80). Another interesting discovery is the influence of predictors useful for classifying a fraud.In this case **oldbalanceOrg** and **amount** top the list.




AUCPR: 0.551205436553915
Precision
0.0
0.2
0.4
0.6
0.8
0.6
0.7
0.8
0.9
1.0
Recall

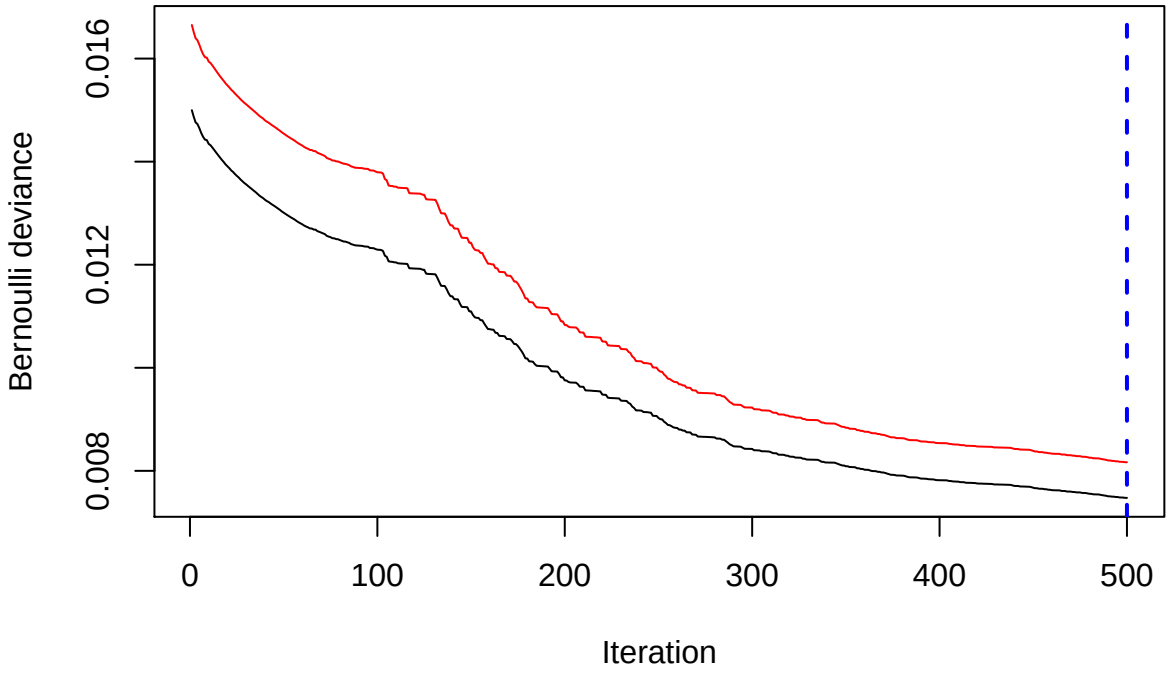


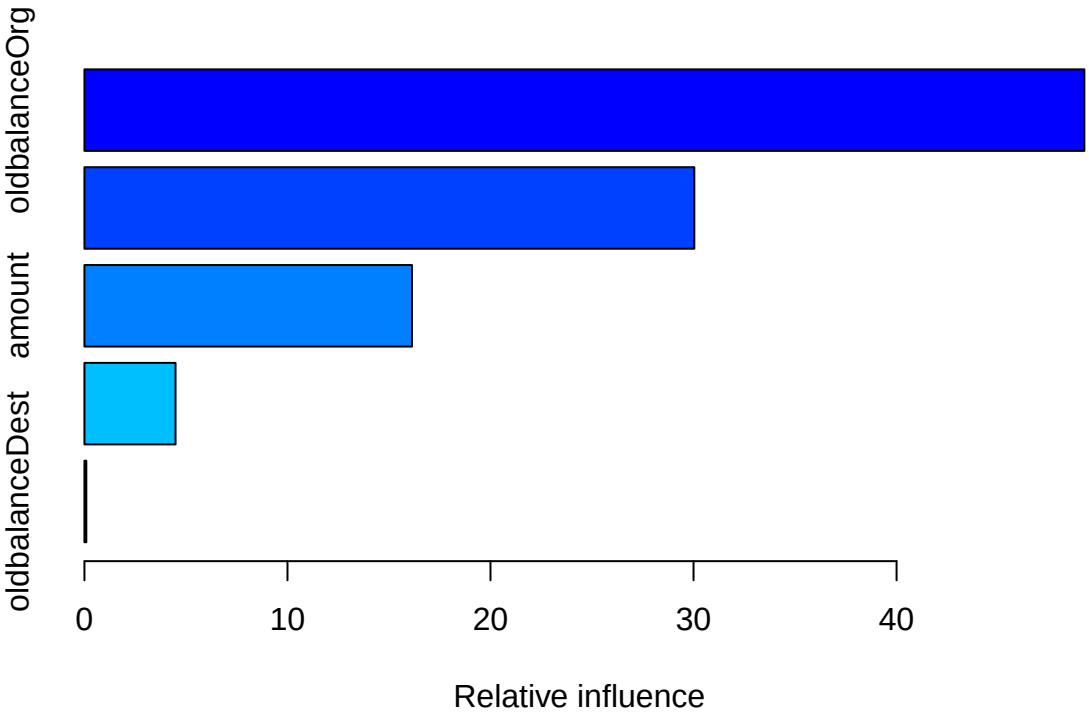
Model	AUC	AUCPR
Naive Baseline - Predict Always Legal	0.5000000	0.0000000
Naive Bayes	0.6306055	0.0057313
K-Nearest Neighbors k=5	0.7953837	0.4879014
Random Forest	0.7968483	0.5512054

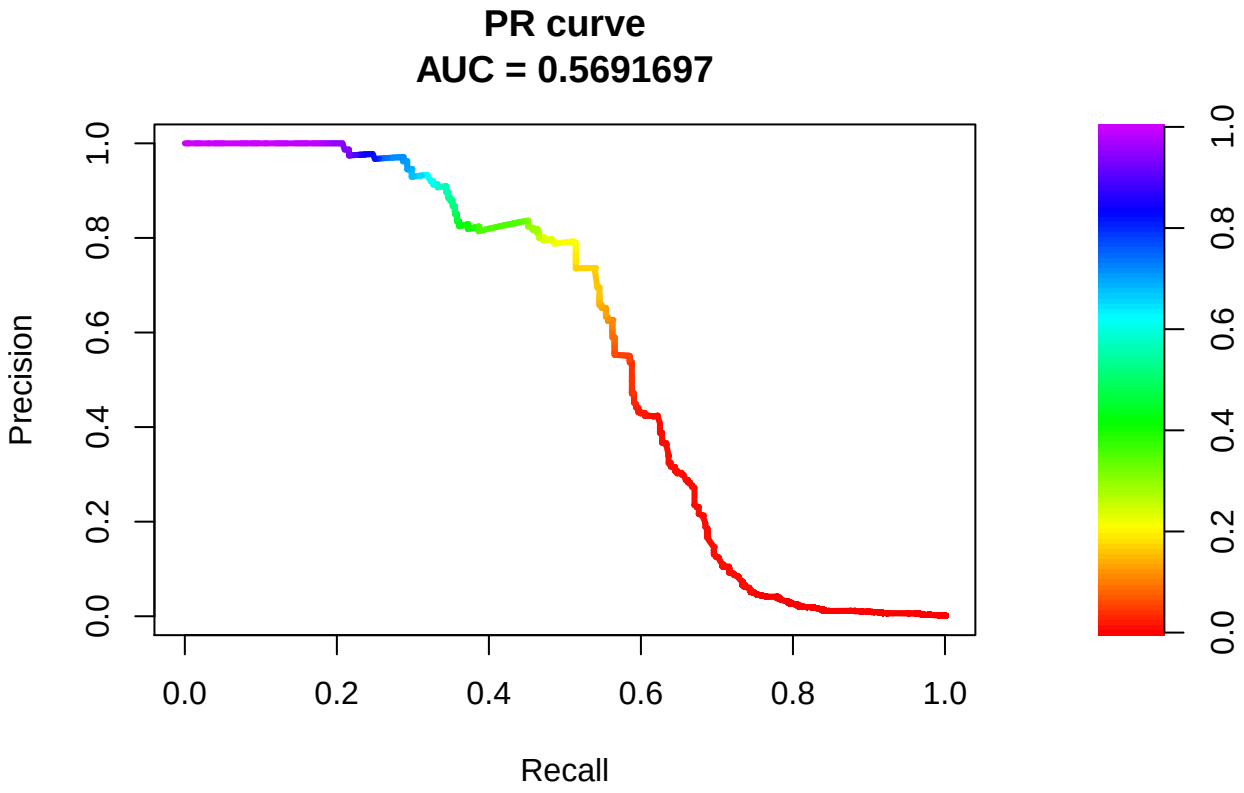
	MeanDecreaseGini
amount	517.28317
oldbalanceOrg	768.65320
newbalanceOrig	75.52712
oldbalanceDest	166.97924
newbalanceDest	371.51761

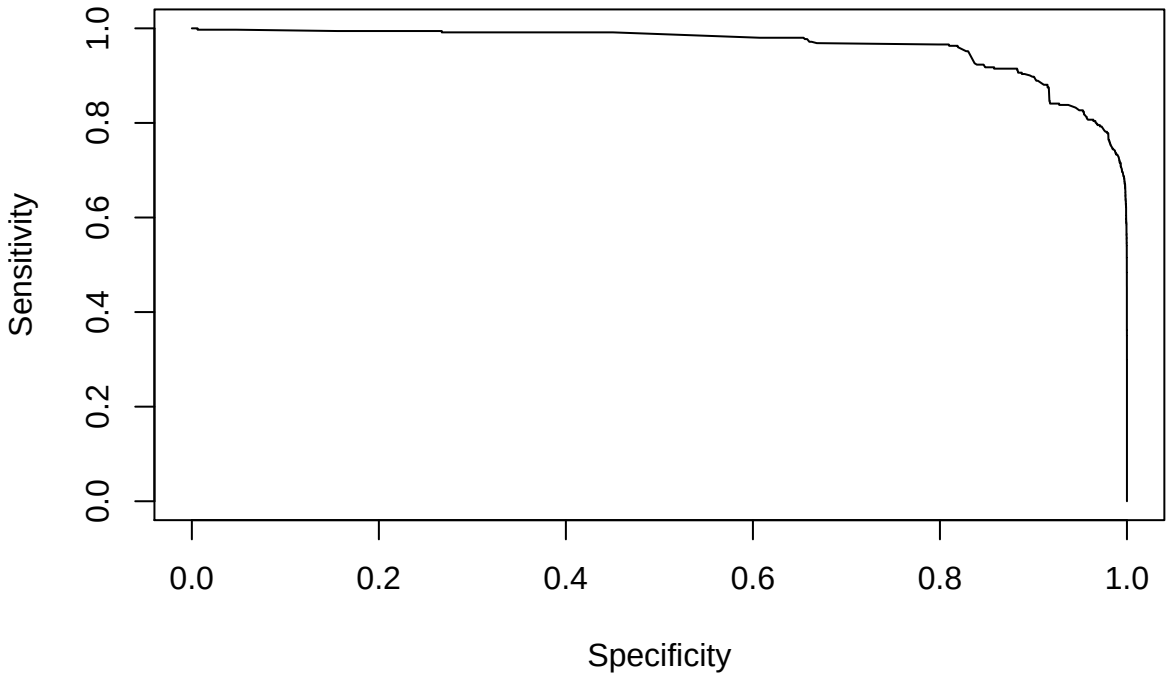
4.5 GBM - Generalized Boosted Regression

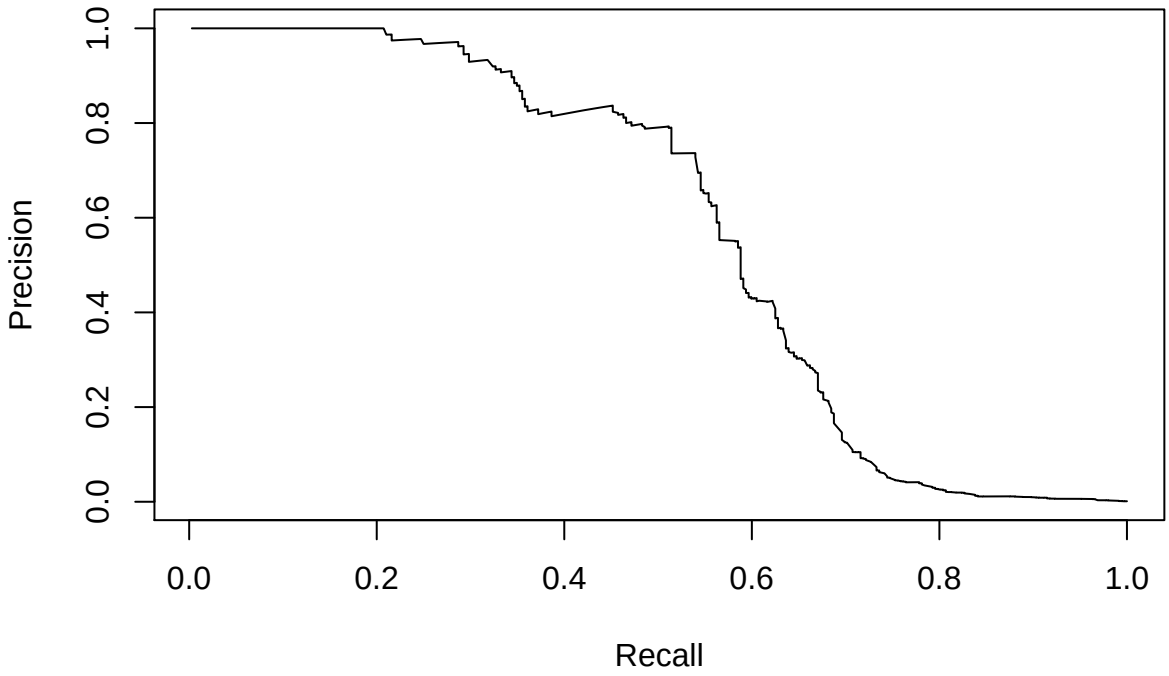
The GBM performance are really good: with an AUC of **0.96** and AUCPR of **0.56**, It still does not achieve the target. As the Random Forest model shows, the **amount** and **oldbalanceOrg** are still relevant to predict a fraud.




oldbalanceOrg
amount
oldbalanceDest
0
10
20
30
40
Relative influence


PR curve
AUC = 0.5691697
Precision
0.0
0.2
0.4
0.6
0.8
1.0
Recall
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0


AUC: 0.96335719950031
Sensitivity
0.0
0.2
0.4
0.6
0.8
1.0
Specificity



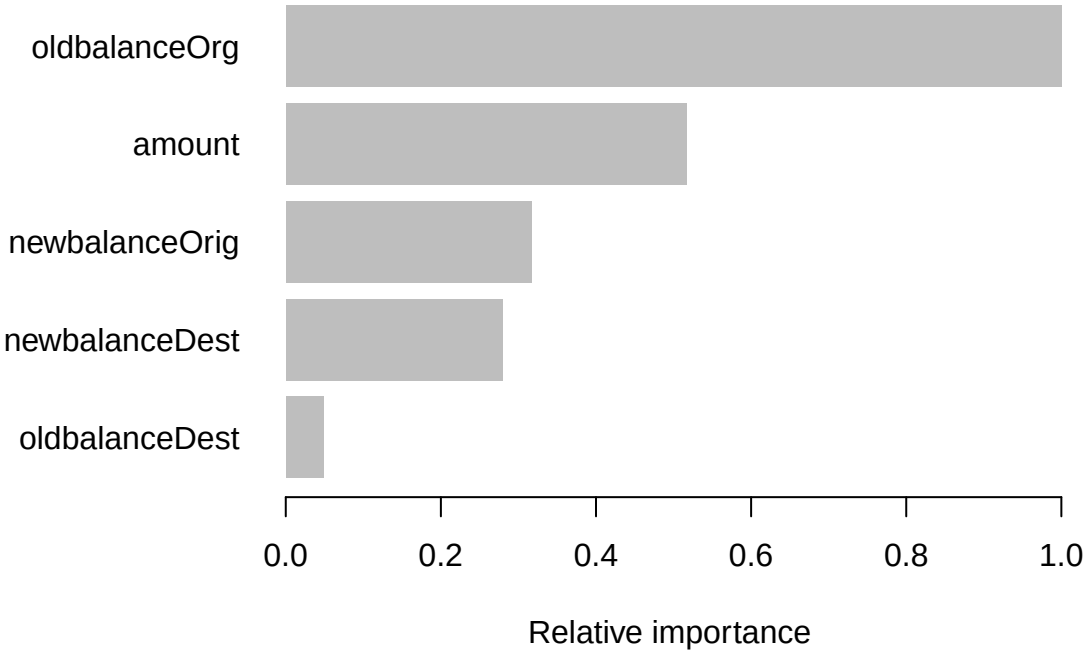
Model	AUC	AUCPR
Naive Baseline - Predict Always Legal	0.5000000	0.0000000
Naive Bayes	0.6306055	0.0057313
K-Nearest Neighbors k=5	0.7953837	0.4879014
Random Forest	0.7968483	0.5512054
GBM - Generalized Boosted Regression	0.9633572	0.5691697

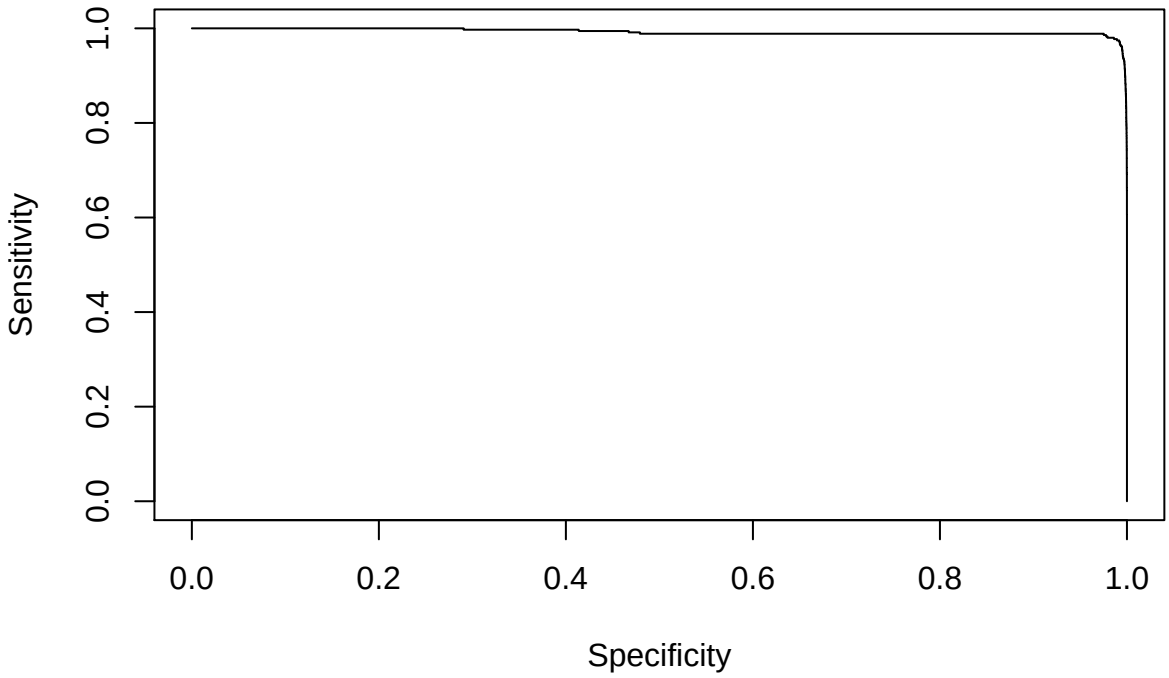
	var	rel.inf
oldbalanceOrg	oldbalanceOrg	49.2498267
newbalanceDest	newbalanceDest	30.0377364
amount	amount	16.1333701
newbalanceOrig	newbalanceOrig	4.4870431
oldbalanceDest	oldbalanceDest	0.0920237

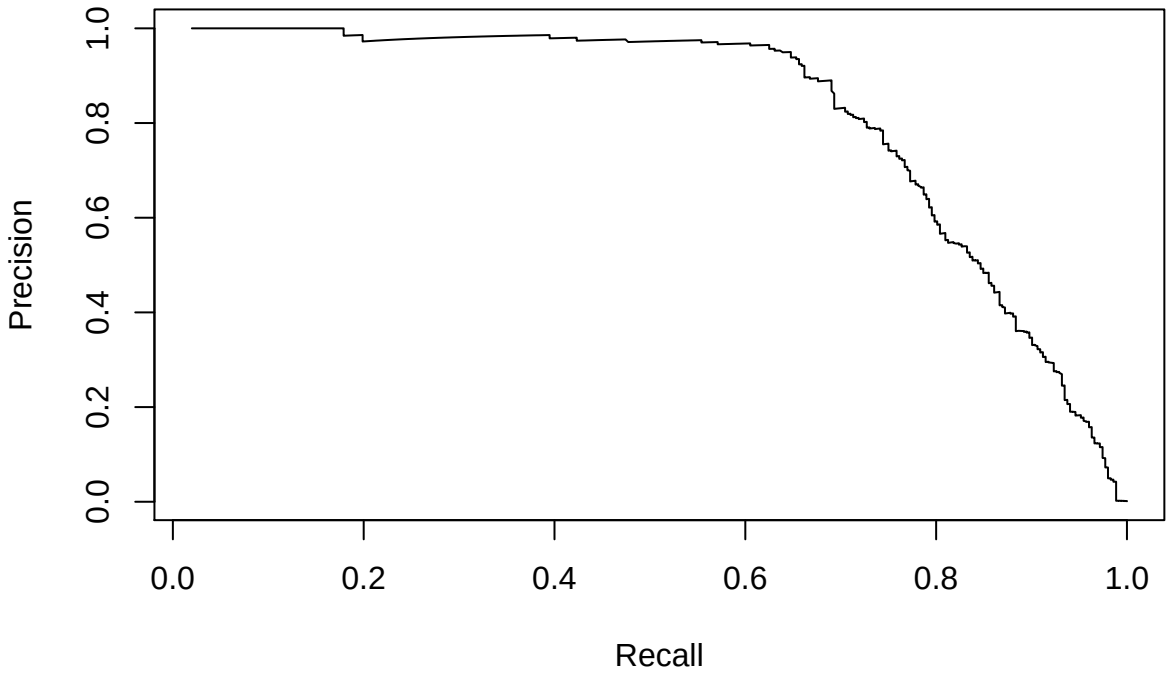
4.6 XGBoost

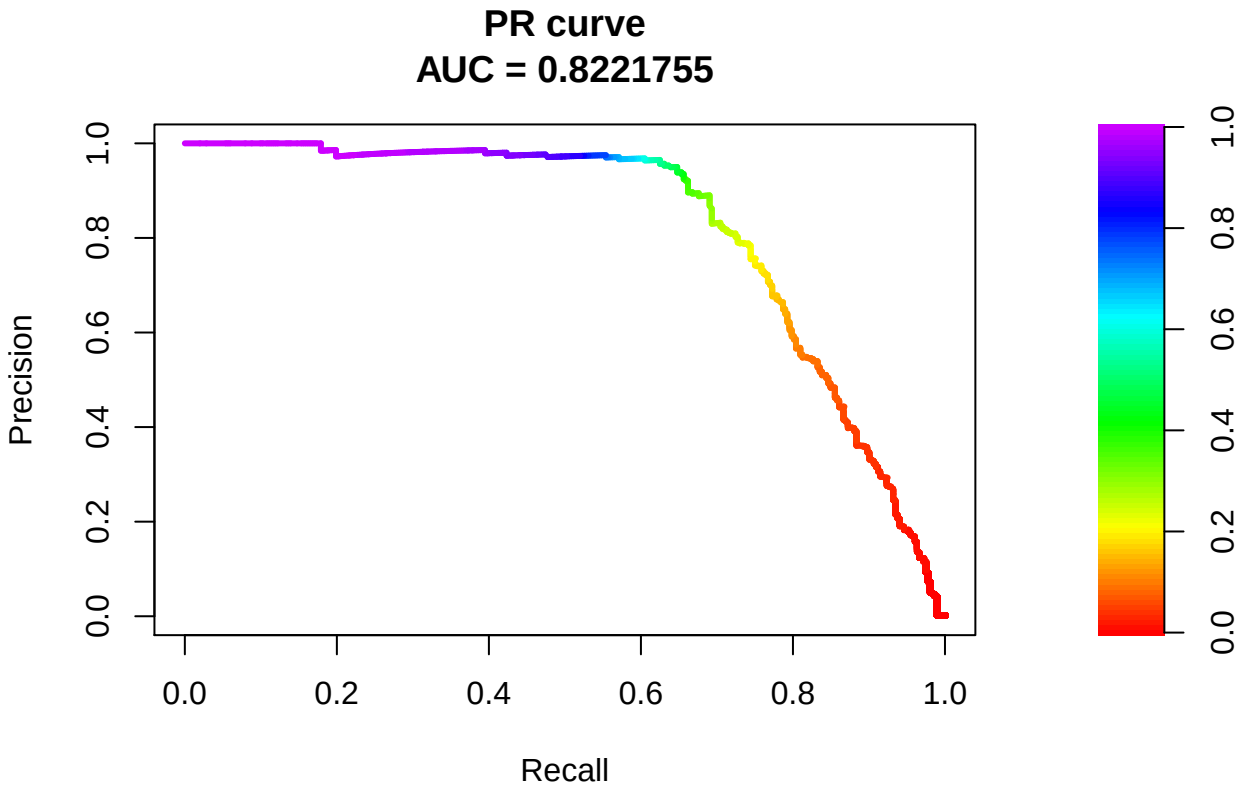
XGBoost is a very popular model widely used in many applications. Fast to train with awesome performance. With an AUC of **0.99** and an AUCPR of **0.82** it meets the performance target. **amount** and **oldbalanceOrg** are still relevant to predict a fraud.

```
## [1]  test-aucpr:0.091797 cv-aucpr:0.114824
## Multiple eval metrics are present. Will use cv_aucpr for early stopping.
## Will train until cv_aucpr hasn't improved in 40 rounds.
##
## [21] test-aucpr:0.321219 cv-aucpr:0.319132
## [41] test-aucpr:0.581121 cv-aucpr:0.566498
## [61] test-aucpr:0.637505 cv-aucpr:0.615435
## [81] test-aucpr:0.666710 cv-aucpr:0.643613
## [101]    test-aucpr:0.693708 cv-aucpr:0.678843
## [121]    test-aucpr:0.713601 cv-aucpr:0.707472
## [141]    test-aucpr:0.733071 cv-aucpr:0.728429
## [161]    test-aucpr:0.746217 cv-aucpr:0.746945
## [181]    test-aucpr:0.760542 cv-aucpr:0.760213
## [201]    test-aucpr:0.769786 cv-aucpr:0.773023
## [221]    test-aucpr:0.777651 cv-aucpr:0.783752
## [241]    test-aucpr:0.786091 cv-aucpr:0.792355
## [261]    test-aucpr:0.790901 cv-aucpr:0.799891
## [281]    test-aucpr:0.794572 cv-aucpr:0.806256
## [301]    test-aucpr:0.797797 cv-aucpr:0.809908
## [321]    test-aucpr:0.799836 cv-aucpr:0.812579
## [341]    test-aucpr:0.804892 cv-aucpr:0.816439
## [361]    test-aucpr:0.807272 cv-aucpr:0.820282
## [381]    test-aucpr:0.808830 cv-aucpr:0.822139
## [401]    test-aucpr:0.812535 cv-aucpr:0.825087
## [421]    test-aucpr:0.814194 cv-aucpr:0.826569
## [441]    test-aucpr:0.816195 cv-aucpr:0.828887
## [461]    test-aucpr:0.818060 cv-aucpr:0.829859
## [481]    test-aucpr:0.820152 cv-aucpr:0.832942
## [500]    test-aucpr:0.822176 cv-aucpr:0.835680
```


oldbalanceOrg
amount
newbalanceOrig
newbalanceDest
oldbalanceDest
0.0
0.2
0.4
0.6
0.8
1.0
Relative importance


AUC: 0.992606590837367
Sensitivity
0.0
0.2
0.4
0.6
0.8
1.0
Specificity


AUCPR: 0.822175541362254
Precision
0.0
0.2
0.4
0.6
0.8
1.0
Recall
0.0
0.2
0.4
0.6
0.8
1.0



Model	AUC	AUCPR
Naive Baseline - Predict Always Legal	0.5000000	0.0000000
Naive Bayes	0.6306055	0.0057313
K-Nearest Neighbors k=5	0.7953837	0.4879014
Random Forest	0.7968483	0.5512054
GBM - Generalized Boosted Regression	0.9633572	0.5691697
XGBoost	0.9926066	0.8221755

Feature	Gain	Cover	Frequency	Importance
oldbalanceOrg	0.4627007	0.0804418	0.3597792	0.4627007
amount	0.2389313	0.7785121	0.4122111	0.2389313
newbalanceOrig	0.1466333	0.1335095	0.0996895	0.1466333
newbalanceDest	0.1291897	0.0045532	0.0748534	0.1291897
oldbalanceDest	0.0225450	0.0029833	0.0534667	0.0225450

5 Results

This is the summary results for all the models.

Model	AUC	AUCPR
Naive Baseline - Predict Always Legal	0.5000000	0.0000000
Naive Bayes	0.6306055	0.0057313
K-Nearest Neighbors k=5	0.7953837	0.4879014
Random Forest	0.7968483	0.5512054
GBM - Generalized Boosted Regression	0.9633572	0.5691697
XGBoost	0.9926066	0.8221755

6 Final Analysis

The ensemble methods used prove themselves as the best models out there. In this task, a XGBoost model can achieve a very good AUCPR result of **0.82** and the others ensemble methods are very close to it.

As the features importance plots and table show, there are few predictors like **amount** and **oldbalanceOrg** that are particularly useful for classifying a fraud.