kaggle credit card fraud prediction

```
library(caret)
```

```
## Loading required package: ggplot2

## Warning in register(): Can't find generic `scale_type` in package ggplot2 to
## register S3 method.

## Loading required package: lattice
```

```
library(ggplot2)
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.1.3

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v tibble  3.1.6     v purrr   0.3.4
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1

## Warning: package 'tidyr' was built under R version 4.1.3

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x purrr::lift()   masks caret::lift()
```

```
library(LiblineaR)
```

```
## Warning: package 'LiblineaR' was built under R version 4.1.3
```

```
library(recipes)
```

```
## Warning: package 'recipes' was built under R version 4.1.3

##
## Attaching package: 'recipes'
## The following object is masked from 'package:stringr':
##
##     fixed

## The following object is masked from 'package:stats':
##
##     step
```

```
library(themis)
```

```
## Warning: package 'themis' was built under R version 4.1.3

##
## Attaching package: 'themis'

## The following objects are masked from 'package:recipes':
##
##     step_downsample, step_upsample
```

```
library(kernlab)
```

```
##
## Attaching package: 'kernlab'

## The following object is masked from 'package:purrr':
##
##     cross

## The following object is masked from 'package:ggplot2':
##
##     alpha
```

```
library(pROC)
```

```
## Warning: package 'pROC' was built under R version 4.1.3

## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
library(ROSE)
```

```
## Warning: package 'ROSE' was built under R version 4.1.3

## Loaded ROSE 0.0-4
```

```
library(DMwR2)
```

```
## Warning: package 'DMwR2' was built under R version 4.1.3

## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
library(h2o)
```

```
## Warning: package 'h2o' was built under R version 4.1.3
```

```
##
## ----------------------------------------------------------------------
##
## Your next step is to start H2O:
##     > h2o.init()
##
## For H2O package documentation, ask for help:
##     > ??h2o
##
## After starting H2O, you can use the Web UI at http://localhost:54321
## For more information visit https://docs.h2o.ai
##
## ----------------------------------------------------------------------
```

```
##
## Attaching package: 'h2o'
```

```
## The following object is masked from 'package:pROC':
##
##     var
```

```
## The following objects are masked from 'package:stats':
##
##     cor, sd, var
```

```
## The following objects are masked from 'package:base':
##
##     %*%, %in%, &&, ||, apply, as.factor, as.numeric, colnames,
##     colnames<-, ifelse, is.character, is.factor, is.numeric, log,
##     log10, log1p, log2, round, signif, trunc
```

```
library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 4.1.3
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
## Loaded glmnet 4.1-3
```

```
library(xgboost)
```

```
## Warning: package 'xgboost' was built under R version 4.1.3
```

```
##
## Attaching package: 'xgboost'
```

```
## The following object is masked from 'package:dplyr':
##
##     slice
```

```
library(PRROC)
```

```
## Warning: package 'PRROC' was built under R version 4.1.3
```

```
##
## Attaching package: 'PRROC'

## The following object is masked from 'package:ROSE':
##
##     roc.curve
```

Load the data

```
setwd("C:/Users/ozge/Desktop/credit_card_deneme")
creditcard <- read.csv("creditcard.csv")
```

Summary of the credit card data

```
head(creditcard)
```

```
##   Time         V1          V2        V3         V4          V5          V6
## 1    0 -1.3598071 -0.07278117 2.5363467  1.3781552 -0.33832077  0.46238778
## 2    0  1.1918571  0.26615071 0.1664801  0.4481541  0.06001765 -0.08236081
## 3    1 -1.3583541 -1.34016307 1.7732093  0.3797796 -0.50319813  1.80049938
## 4    1 -0.9662717 -0.18522601 1.7929933 -0.8632913 -0.01030888  1.24720317
## 5    2 -1.1582331  0.87773675 1.5487178  0.4030339 -0.40719338  0.09592146
## 6    2 -0.4259659  0.96052304 1.1411093 -0.1682521  0.42098688 -0.02972755
##            V7          V8         V9         V10        V11         V12
## 1  0.23959855  0.09869790  0.3637870  0.09079417 -0.5515995 -0.61780086
## 2 -0.07880298  0.08510165 -0.2554251 -0.16697441  1.6127267  1.06523531
## 3  0.79146096  0.24767579 -1.5146543  0.20764287  0.6245015  0.06608369
## 4  0.23760894  0.37743587 -1.3870241 -0.05495192 -0.2264873  0.17822823
## 5  0.59294075 -0.27053268  0.8177393  0.75307443 -0.8228429  0.53819555
## 6  0.47620095  0.26031433 -0.5686714 -0.37140720  1.3412620  0.35989384
##           V13        V14        V15        V16         V17         V18
## 1 -0.9913898 -0.3111694  1.4681770 -0.4704005  0.20797124  0.02579058
## 2  0.4890950 -0.1437723  0.6355581  0.4639170 -0.11480466 -0.18336127
## 3  0.7172927 -0.1659459  2.3458649 -2.8900832  1.10996938 -0.12135931
## 4  0.5077569 -0.2879237 -0.6314181 -1.0596472 -0.68409279  1.96577500
## 5  1.3458516 -1.1196698  0.1751211 -0.4514492 -0.23703324 -0.03819479
## 6 -0.3580907 -0.1371337  0.5176168  0.4017259 -0.05813282  0.06865315
##            V19         V20          V21          V22         V23         V24
## 1  0.40399296  0.25141210 -0.018306778  0.277837576 -0.11047391  0.06692807
## 2 -0.14578304 -0.06908314 -0.225775248 -0.638671953  0.10128802 -0.33984648
## 3 -2.26185710  0.52497973  0.247998153  0.771679402  0.90941226 -0.68928096
## 4 -1.23262197 -0.20803778 -0.108300452  0.005273597 -0.19032052 -1.17557533
## 5  0.80348692  0.40854236 -0.009430697  0.798278495 -0.13745808  0.14126698
## 6 -0.03319379  0.08496767 -0.208253515 -0.559824796 -0.02639767 -0.37142658
##           V25        V26          V27         V28 Amount Class
## 1  0.1285394 -0.1891148  0.133558377 -0.02105305 149.62     0
## 2  0.1671704  0.1258945 -0.008983099  0.01472417   2.69     0
## 3 -0.3276418 -0.1390966 -0.055352794 -0.05975184 378.66     0
## 4  0.6473760 -0.2219288  0.062722849  0.06145763 123.50     0
## 5 -0.2060096  0.5022922  0.219422230  0.21515315  69.99     0
## 6 -0.2327938  0.1059148  0.253844225  0.08108026   3.67     0
```

```
summary(creditcard)
```

```
##       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
##  Min.   :-14.12985   Min.   :-25.16280   Min.   :-9.498746
##  1st Qu.: -0.46804   1st Qu.: -0.48375   1st Qu.:-0.498850
##  Median :  0.06641   Median : -0.06568   Median :-0.003636
##  Mean   :  0.00000   Mean   :  0.00000   Mean   : 0.000000
##  3rd Qu.:  0.52330   3rd Qu.:  0.39968   3rd Qu.: 0.500807
##  Max.   : 17.31511   Max.   :  9.25353   Max.   : 5.041069
##       V19                 V20                 V21
##  Min.   :-7.213527   Min.   :-54.49772   Min.   :-34.83038
##  1st Qu.:-0.456299   1st Qu.: -0.21172   1st Qu.: -0.22839
##  Median : 0.003735   Median : -0.06248   Median : -0.02945
##  Mean   : 0.000000   Mean   :  0.00000   Mean   :  0.00000
##  3rd Qu.: 0.458949   3rd Qu.:  0.13304   3rd Qu.:  0.18638
##  Max.   : 5.591971   Max.   : 39.42090   Max.   : 27.20284
##       V22                  V23                 V24
##  Min.   :-10.933144   Min.   :-44.80774   Min.   :-2.83663
##  1st Qu.: -0.542350   1st Qu.: -0.16185   1st Qu.:-0.35459
##  Median :  0.006782   Median : -0.01119   Median : 0.04098
##  Mean   :  0.000000   Mean   :  0.00000   Mean   : 0.00000
##  3rd Qu.:  0.528554   3rd Qu.:  0.14764   3rd Qu.: 0.43953
##  Max.   : 10.503090   Max.   : 22.52841   Max.   : 4.58455
##       V25                 V26                V27
##  Min.   :-10.29540   Min.   :-2.60455   Min.   :-22.565679
##  1st Qu.: -0.31715   1st Qu.:-0.32698   1st Qu.: -0.070840
##  Median :  0.01659   Median :-0.05214   Median :  0.001342
##  Mean   :  0.00000   Mean   : 0.00000   Mean   :  0.000000
##  3rd Qu.:  0.35072   3rd Qu.: 0.24095   3rd Qu.:  0.091045
##  Max.   :  7.51959   Max.   : 3.51735   Max.   : 31.612198
##       V28                 Amount              Class
##  Min.   :-15.43008   Min.   :    0.00   Min.   :0.000000
##  1st Qu.: -0.05296   1st Qu.:    5.60   1st Qu.:0.000000
##  Median :  0.01124   Median :   22.00   Median :0.000000
##  Mean   :  0.00000   Mean   :   88.35   Mean   :0.001728
##  3rd Qu.:  0.07828   3rd Qu.:   77.17   3rd Qu.:0.000000
##  Max.   : 33.84781   Max.   :25691.16   Max.   :1.000000
```

```
slice_sample(creditcard, n=10)
```

```
##      Time         V1         V2         V3         V4          V5          V6
## 1  119092  2.0484749 -1.2735381 -0.8573912 -1.1010791 -0.76252637  0.15059396
## 2  131743 -0.6998622  0.1879334  1.5219314 -0.5037343  0.06530977 -0.55133748
## 3   68953  1.1098673 -0.7443700  0.8337334  0.6765046 -1.36572927 -0.37675780
## 4  137488  2.0208359 -1.1664213 -0.8455742 -0.7076253 -0.63909879  0.50557536
## 5   67674 -0.9896947 -0.1931960  3.1102324 -1.6027082 -1.11543190  0.96099322
## 6   47465  1.3550709  0.1671035 -0.3232397  0.3843779  0.12327349 -0.32244350
## 7   78866  0.4651004  1.9141040 -2.1683722  1.7275899  0.60544826 -1.93305530
## 8  141563  2.0737048  0.2612337 -1.6601118  0.4116396  0.52502658 -0.86639779
## 9   27145  1.4598037 -0.9878156  0.5493420 -1.3906454 -1.58195899 -0.81290873
## 10 123490  1.5574852 -0.7651202 -0.6395721  1.4171744 -0.44868645  0.01112118
##             V7           V8          V9        V10         V11         V12
## 1  -1.05055498  0.128780675  0.01272306  0.8657575 -0.07589796 -0.45875944
## 2   0.13021412  0.115707075  0.49672603 -0.5986593 -1.15154184  0.03078568
## 3  -0.62884131 -0.046157213 -0.41184687  0.6601026 -0.97207852 -0.12637919
## 4  -1.04488422  0.238684275 -0.08267961  0.9707196 -0.16435797 -0.07803315
## 5  -0.75723040  0.486876954  0.34966887  0.3433411  0.52499940 -0.16934129
## 6  -0.08741081  0.005894238  0.44790546 -0.2182571 -1.67800906 -1.34881923
## 7   0.59031674  0.159660277 -0.47029884 -1.4064328  0.51994900 -0.65131605
## 8   0.28179082 -0.318630619  0.27867297 -0.4195650 -0.29868299  0.90584094
## 9  -0.99936265 -0.122198822 -2.13598930  1.6055693  1.42181410 -0.06484767
## 10 -0.10688625  0.065051407  0.73863465  0.1925188  0.29369184  0.89274195
##            V13         V14         V15        V16         V17        V18
## 1  -0.47352743 -0.13291731 -0.26170732  1.7573987 -0.41752155 -0.6436270
## 2   0.11417404 -0.37804091 -0.01579561  0.1851264 -0.50256614  0.1309810
## 3   0.01496151 -0.18709893  0.35314256 -1.4249514 -0.04357242  1.4745777
## 4  -0.63316292  0.16745584 -0.05982872 -0.6388669 -0.68086698  1.6865396
## 5  -1.08812460 -1.46689305 -2.25928122  0.5385883  0.63645220 -1.1753483
## 6  -1.63817132  0.04805132  1.42761761  0.9856473 -0.39373682  0.4161649
## 7  -0.43622642 -3.14835290  1.07789484  0.9742979  3.01139635  1.5906487
## 8   1.46593894 -1.12252102  0.04842900  0.2409048  0.40490574 -0.4909754
## 9   0.48669324 -0.12003320 -0.15968571 -0.2691640  0.32579982  0.3742251
## 10 -0.52351079  0.26694055 -0.87109596  0.3335819 -0.72238516  0.1315779
##            V19         V20         V21         V22         V23         V24
## 1   1.23139892  0.09345787 -0.02538946 -0.27419771  0.19575145 -1.20363560
## 2  -0.48034032  0.16899859  0.30553613  1.02804378 -0.11277222  0.05151634
## 3  -1.14319534 -0.36537781 -0.18773202 -0.12919696 -0.13342785  0.37487723
## 4  -0.27962327 -0.49447058 -0.60407139 -1.41378438  0.42102228 -0.19901798
## 5   0.37640445  0.29877367  0.12654349  0.94759893 -0.41446493  0.06144162
## 6   0.34379038 -0.17234601 -0.40444859 -1.26000973 -0.01760917 -1.11480074
## 7  -0.41608651 -0.02801138 -0.10711610 -0.20226529  0.14771871  0.02916924
## 8   0.03109442 -0.06678915 -0.34628409 -0.82341669  0.32691633  0.60661400
## 9  -0.03290792 -0.31412453 -0.14726762 -0.06561503  0.04538533  0.52824398
## 10  0.22383710  0.10834328 -0.24174601 -1.00822955  0.21103680 -0.42719405
##           V25        V26          V27          V28 Amount Class
## 1  -0.4513554 -0.4365260 -0.007685852 -0.051446223  79.95     0
## 2  -0.3276416  0.6024382  0.362455837  0.234252410  29.99     0
## 3   0.4804928 -0.2320978  0.058022087  0.050445921 105.00     0
## 4  -0.7372957  0.2324978 -0.037909743 -0.043095646  64.50     0
## 5   0.4447344 -0.1214246  0.297566000 -0.048099213   2.00     0
## 6   0.3319059  0.1883976 -0.038526518  0.011939038   0.89     0
## 7  -0.4458099 -0.3977156  0.129018394 -0.047630319   4.99     0
## 8  -0.2148484  0.1649159 -0.053459557 -0.027094267   1.79     0
## 9   0.3335129 -0.2373641  0.026810360  0.013446093  15.00     0
## 10 -0.4406816 -1.0871935  0.006024886 -0.008783866 199.50     0
```

```
str(creditcard)
```

```
## 'data.frame':    284807 obs. of  31 variables:
##  $ Time  : num  0 0 1 1 2 2 4 7 7 9 ...
##  $ V1    : num  -1.36 1.192 -1.358 -0.966 -1.158 ...
##  $ V2    : num  -0.0728 0.2662 -1.3402 -0.1852 0.8777 ...
##  $ V3    : num  2.536 0.166 1.773 1.793 1.549 ...
##  $ V4    : num  1.378 0.448 0.38 -0.863 0.403 ...
##  $ V5    : num  -0.3383 0.06 -0.5032 -0.0103 -0.4072 ...
##  $ V6    : num  0.4624 -0.0824 1.8005 1.2472 0.0959 ...
##  $ V7    : num  0.2396 -0.0788 0.7915 0.2376 0.5929 ...
##  $ V8    : num  0.0987 0.0851 0.2477 0.3774 -0.2705 ...
##  $ V9    : num  0.364 -0.255 -1.515 -1.387 0.818 ...
##  $ V10   : num  0.0908 -0.167 0.2076 -0.055 0.7531 ...
##  $ V11   : num  -0.552 1.613 0.625 -0.226 -0.823 ...
##  $ V12   : num  -0.6178 1.0652 0.0661 0.1782 0.5382 ...
##  $ V13   : num  -0.991 0.489 0.717 0.508 1.346 ...
##  $ V14   : num  -0.311 -0.144 -0.166 -0.288 -1.12 ...
##  $ V15   : num  1.468 0.636 2.346 -0.631 0.175 ...
##  $ V16   : num  -0.47 0.464 -2.89 -1.06 -0.451 ...
##  $ V17   : num  0.208 -0.115 1.11 -0.684 -0.237 ...
##  $ V18   : num  0.0258 -0.1834 -0.1214 1.9658 -0.0382 ...
##  $ V19   : num  0.404 -0.146 -2.262 -1.233 0.803 ...
##  $ V20   : num  0.2514 -0.0691 0.525 -0.208 0.4085 ...
##  $ V21   : num  -0.01831 -0.22578 0.248 -0.1083 -0.00943 ...
##  $ V22   : num  0.27784 -0.63867 0.77168 0.00527 0.79828 ...
##  $ V23   : num  -0.11 0.101 0.909 -0.19 -0.137 ...
##  $ V24   : num  0.0669 -0.3398 -0.6893 -1.1756 0.1413 ...
##  $ V25   : num  0.129 0.167 -0.328 0.647 -0.206 ...
##  $ V26   : num  -0.189 0.126 -0.139 -0.222 0.502 ...
##  $ V27   : num  0.13356 -0.00898 -0.05535 0.06272 0.21942 ...
##  $ V28   : num  -0.0211 0.0147 -0.0598 0.0615 0.2152 ...
##  $ Amount: num  149.62 2.69 378.66 123.5 69.99 ...
##  $ Class : int  0 0 0 0 0 0 0 0 0 0 ...
```

```
creditcard$Amount <- scale(creditcard$Amount, center = TRUE, scale = TRUE)
summary(creditcard)
```

```
##       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
##  Min.   :-14.12985   Min.   :-25.16280   Min.   :-9.498746
##  1st Qu.: -0.46804   1st Qu.: -0.48375   1st Qu.:-0.498850
##  Median :  0.06641   Median : -0.06568   Median :-0.003636
##  Mean   :  0.00000   Mean   :  0.00000   Mean   : 0.000000
##  3rd Qu.:  0.52330   3rd Qu.:  0.39968   3rd Qu.: 0.500807
##  Max.   : 17.31511   Max.   :  9.25353   Max.   : 5.041069
##       V19                 V20                 V21
##  Min.   :-7.213527   Min.   :-54.49772   Min.   :-34.83038
##  1st Qu.:-0.456299   1st Qu.: -0.21172   1st Qu.: -0.22839
##  Median : 0.003735   Median : -0.06248   Median : -0.02945
##  Mean   : 0.000000   Mean   :  0.00000   Mean   :  0.00000
##  3rd Qu.: 0.458949   3rd Qu.:  0.13304   3rd Qu.:  0.18638
##  Max.   : 5.591971   Max.   : 39.42090   Max.   : 27.20284
##       V22                  V23                 V24
##  Min.   :-10.933144   Min.   :-44.80774   Min.   :-2.83663
##  1st Qu.: -0.542350   1st Qu.: -0.16185   1st Qu.:-0.35459
##  Median :  0.006782   Median : -0.01119   Median : 0.04098
##  Mean   :  0.000000   Mean   :  0.00000   Mean   : 0.00000
##  3rd Qu.:  0.528554   3rd Qu.:  0.14764   3rd Qu.: 0.43953
##  Max.   : 10.503090   Max.   : 22.52841   Max.   : 4.58455
##       V25                 V26                V27
##  Min.   :-10.29540   Min.   :-2.60455   Min.   :-22.565679
##  1st Qu.: -0.31715   1st Qu.:-0.32698   1st Qu.: -0.070840
##  Median :  0.01659   Median :-0.05214   Median :  0.001342
##  Mean   :  0.00000   Mean   : 0.00000   Mean   :  0.000000
##  3rd Qu.:  0.35072   3rd Qu.: 0.24095   3rd Qu.:  0.091045
##  Max.   :  7.51959   Max.   : 3.51735   Max.   : 31.612198
##       V28                 Amount.V1          Class
##  Min.   :-15.43008   Min.   : -0.35323   Min.   :0.000000
##  1st Qu.: -0.05296   1st Qu.: -0.33084   1st Qu.:0.000000
##  Median :  0.01124   Median : -0.26527   Median :0.000000
##  Mean   :  0.00000   Mean   :  0.00000   Mean   :0.001728
##  3rd Qu.:  0.07828   3rd Qu.: -0.04472   3rd Qu.:0.000000
##  Max.   : 33.84781   Max.   :102.36206   Max.   :1.000000
```

```
#Baseline occurrence of fraud

credit_table <- table(creditcard$Class)
print(credit_table)
```

```
##
##      0      1
## 284315    492
```

```
print(credit_table[2]/(credit_table[1]+credit_table[2]))
```

```
##           1
## 0.001727486
```

```
creditcard$Class<- factor(make.names(creditcard$Class), labels = c("non_fraud", "fraud"))
creditcard<-subset(creditcard, select = -c(Time))
```

```
# Split data

set.seed(77)
partition <- caret::createDataPartition(y=creditcard$Class, p=.75, list=FALSE)
imbal_train <- creditcard[partition,]
imbal_test <- creditcard[-partition,]
print(nrow(imbal_train)/(nrow(imbal_test)+nrow(imbal_train)))
```

```
## [1] 0.7500026
```

```
#Different versions of training set
set.seed(9560)
down_train <- downSample(x = imbal_train[, -ncol(imbal_train)],
                         y = imbal_train$Class)
table(down_train$Class)
```

```
##
## non_fraud     fraud
##       369       369
```

```
set.seed(9560)
up_train <- upSample(x = imbal_train[, -ncol(imbal_train)],
                     y = imbal_train$Class)
table(up_train$Class)
```

```
##
## non_fraud     fraud
##    213237    213237
```

```
set.seed(9560)
smote_train <- smote(imbal_train,var="Class",over_ratio = 0.5)
table(smote_train$Class)
```

```
##
```

```
## non_fraud     fraud
##    213237    106618
```

```
set.seed(9560)
rose_train <- ovun.sample(Class ~ ., data  = imbal_train,method="both",p=0.5)$data
table(rose_train$Class)
```

```
##
## non_fraud     fraud
##    106996    106610
```

Train control parameters

```
ctrl <- trainControl(method = "cv",
                     number = 5,
                     classProbs = TRUE,
                     summaryFunction = twoClassSummary)
```

Train Model: Exteme Gradient Boosting with L1 and L2 Regularization

```
train <- train(Class ~., data = smote_train, method = 'xgbLinear',trControl = ctrl)
```

```
## Warning in train.default(x, y, weights = w, ...): The metric "Accuracy" was not
## in the result set. ROC will be used instead.
```

```
train_xgb<-train
```

Predictions and probabilities

```
prediction_probability_xgb <- predict(train_xgb, imbal_test, type="prob")


prediction_raw_xgb <- predict(train_xgb, imbal_test, type="raw")
```

```
fraud_probs_xgb <- predict(train_xgb, imbal_test, type="prob")[,2]
non_fraud_probs_xgb <- predict(train_xgb, imbal_test, type="prob")[,1]
```

Confusion Matrix

```
pred_xgb <- factor(ifelse(fraud_probs_xgb >= .5, "fraud", "non_fraud"))

prediction_raw_xgb<-relevel(prediction_raw_xgb,ref=c("fraud"))

imbal_test$Class<-relevel(imbal_test$Class,ref=c("fraud"))

confusionMatrix(data = pred_xgb, reference = factor(imbal_test$Class,levels=c("fraud","non_fraud")))
```

```
## Confusion Matrix and Statistics
##
```

```
##            Reference
## Prediction  fraud non_fraud
##   fraud       101        24
##   non_fraud    22     71054
##
##                Accuracy : 0.9994
##                  95% CI : (0.9991, 0.9995)
##     No Information Rate : 0.9983
##     P-Value [Acc > NIR] : 1.441e-15
##
##                   Kappa : 0.8142
##
##  Mcnemar's Test P-Value : 0.8828
##
##             Sensitivity : 0.821138
##             Specificity : 0.999662
##          Pos Pred Value : 0.808000
##          Neg Pred Value : 0.999690
##              Prevalence : 0.001728
##          Detection Rate : 0.001419
##    Detection Prevalence : 0.001756
##       Balanced Accuracy : 0.910400
##
##        'Positive' Class : fraud
##
```

```
dat_xgb<-data.frame(obs=imbal_test$Class,pred=prediction_raw_xgb,prediction_probability_xgb)

twoClassSummary(dat_xgb,lev=levels(imbal_test$Class))
```

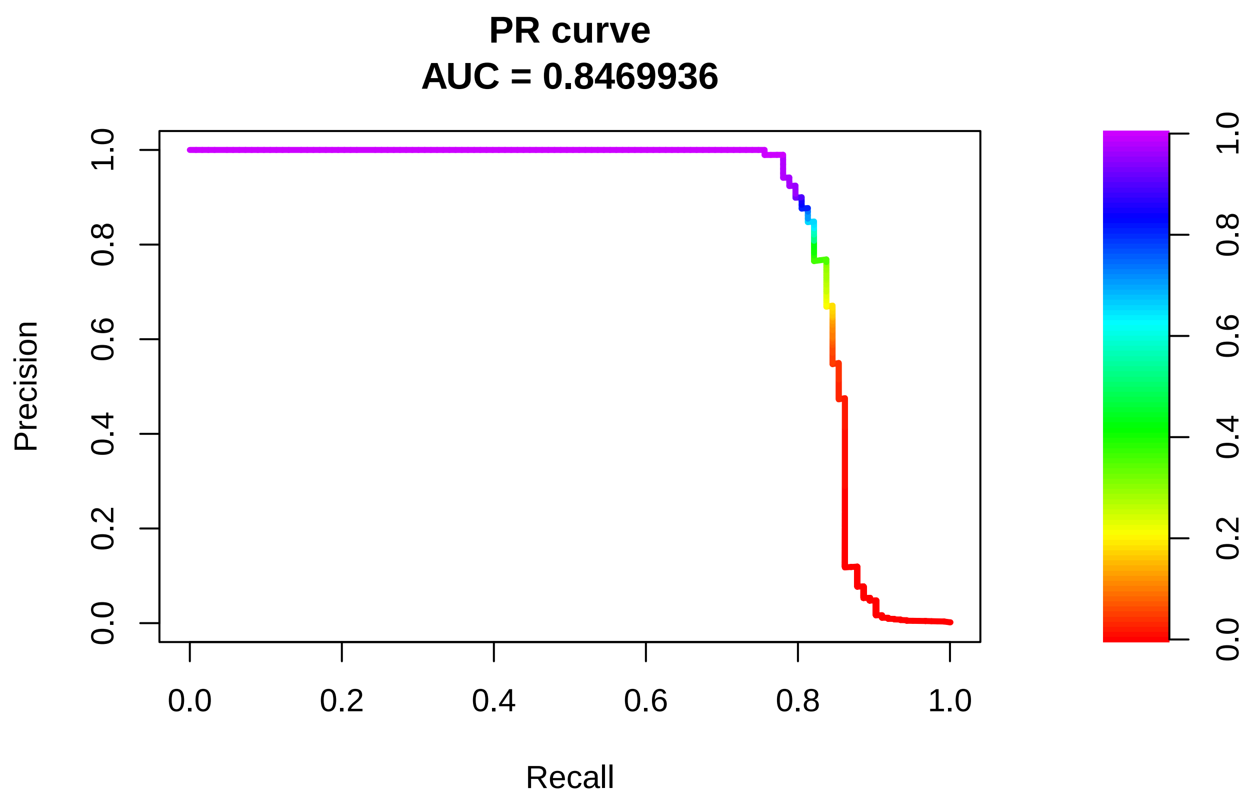
```
##       ROC      Sens      Spec
## 0.9670741 0.8211382 0.9996623
```

```
prSummary(dat_xgb, lev=levels(imbal_test$Class))
```

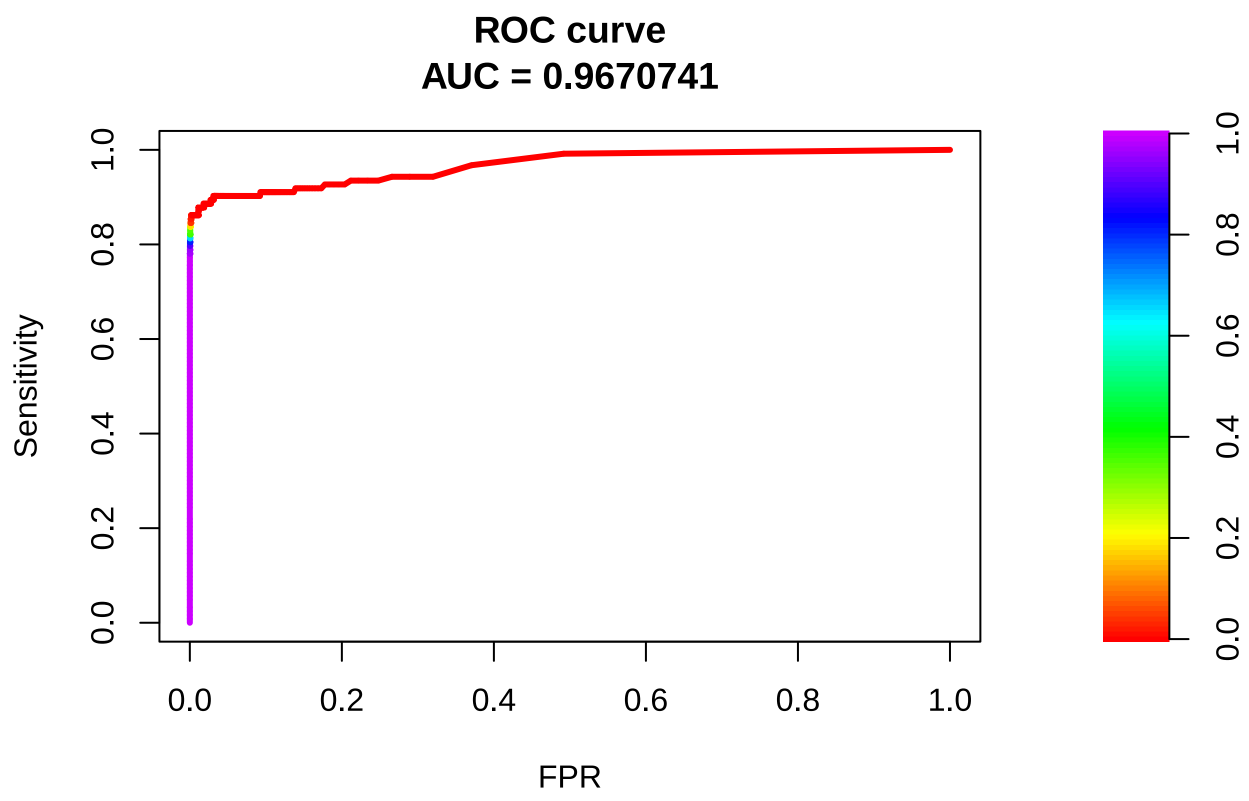
```
##       AUC Precision    Recall         F
## 0.8388666 0.8080000 0.8211382 0.8145161
```

```
positive_xgb<-fraud_probs_xgb[imbal_test[,30]==c("fraud")]
negative_xgb<-fraud_probs_xgb[imbal_test[,30]==c("non_fraud")]

PRC <- pr.curve(positive_xgb, negative_xgb, curve=TRUE)
plot(PRC)
```



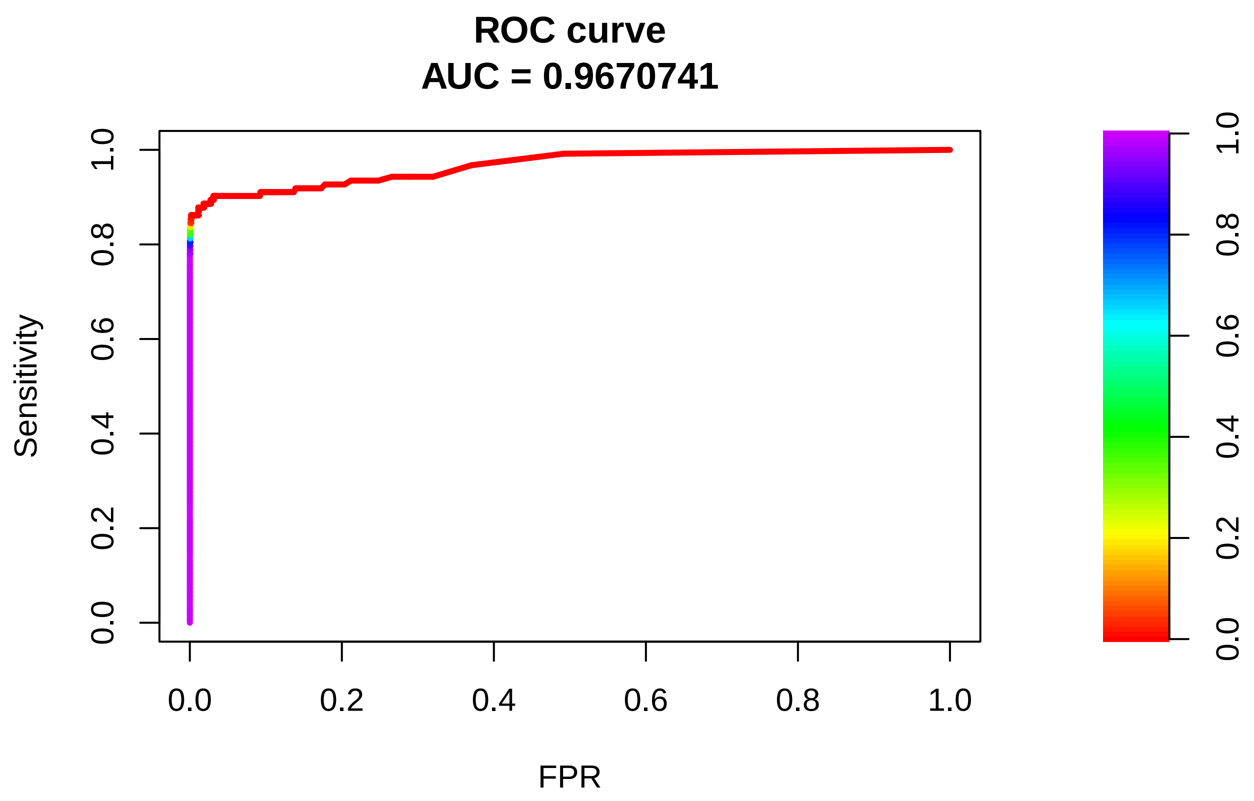
```
ROC<-roc.curve(positive_xgb, negative_xgb, curve=TRUE)
plot(ROC)
```



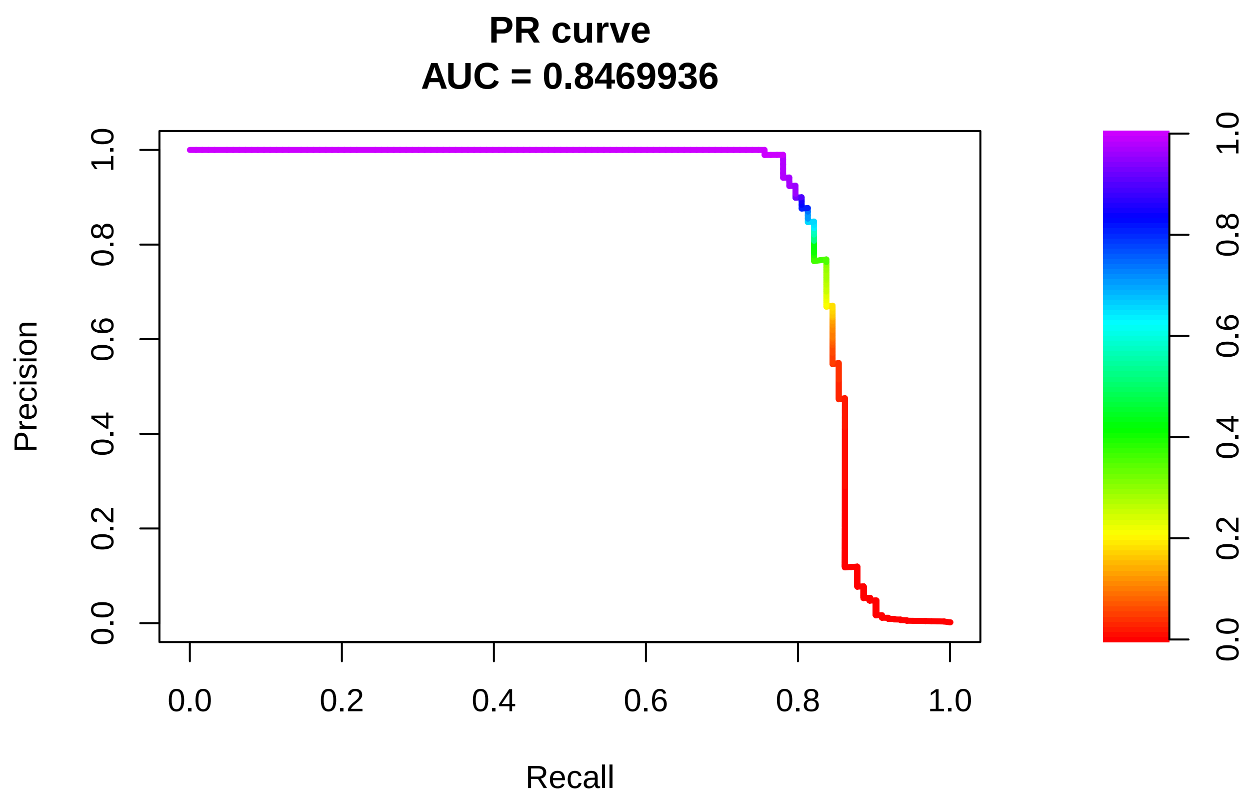
Second way of calculating ROC Curve and PR Curve

```
prediction_probability_xgb_scores<-data.frame(event_prob = prediction_probability_xgb$fraud, labels = in

roc <- PRROC::roc.curve(scores.class0 = prediction_probability_xgb_scores[prediction_probability_xgb_sco
plot(roc)
```



```
pr<-PRROC::pr.curve(scores.class0 = prediction_probability_xgb_scores[prediction_probability_xgb_scores$
                    scores.class1 = prediction_probability_xgb_scores[prediction_probability_xgb_scores$
                     curve=T)
plot(pr)
```



```
paste("Area under the Precision-Recall curve:", round(pr$auc.integral, 7))
```

```
## [1] "Area under the Precision-Recall curve: 0.8469936"
```

```
paste("Area under the ROC curve:", round(roc$auc, 7))
```

```
## [1] "Area under the ROC curve: 0.9670741"
```