RaviRaj_Mulasa_Final_project

Load data

```
credit_card = read.csv("/Users/ravirajmulasa/Downloads/creditcard.csv", header = TRUE)
head(credit_card, n=10L)
```

```
##    Time         V1          V2          V3         V4          V5
## 1     0 -1.3598071 -0.07278117  2.53634674  1.3781552 -0.33832077
## 2     0  1.1918571  0.26615071  0.16648011  0.4481541  0.06001765
## 3     1 -1.3583541 -1.34016307  1.77320934  0.3797796 -0.50319813
## 4     1 -0.9662717 -0.18522601  1.79299334 -0.8632913 -0.01030888
## 5     2 -1.1582331  0.87773675  1.54871785  0.4030339 -0.40719338
## 6     2 -0.4259659  0.96052304  1.14110934 -0.1682521  0.42098688
## 7     4  1.2296576  0.14100351  0.04537077  1.2026127  0.19188099
## 8     7 -0.6442694  1.41796355  1.07438038 -0.4921990  0.94893409
## 9     7 -0.8942861  0.28615720 -0.11319221 -0.2715261  2.66959866
## 10    9 -0.3382618  1.11959338  1.04436655 -0.2221873  0.49936081
##             V6           V7          V8         V9         V10        V11
## 1   0.46238778  0.239598554  0.09869790  0.3637870  0.09079417 -0.5515995
## 2  -0.08236081 -0.078802983  0.08510165 -0.2554251 -0.16697441  1.6127267
## 3   1.80049938  0.791460956  0.24767579 -1.5146543  0.20764287  0.6245015
## 4   1.24720317  0.237608940  0.37743587 -1.3870241 -0.05495192 -0.2264873
## 5   0.09592146  0.592940745 -0.27053268  0.8177393  0.75307443 -0.8228429
## 6  -0.02972755  0.476200949  0.26031433 -0.5686714 -0.37140720  1.3412620
## 7   0.27270812 -0.005159003  0.08121294  0.4649600 -0.09925432 -1.4169072
## 8   0.42811846  1.120631358 -3.80786424  0.6153747  1.24937618 -0.6194678
## 9   3.72181806  0.370145128  0.85108444 -0.3920476 -0.41043043 -0.7051166
## 10 -0.24676110  0.651583206  0.06953859 -0.7367273 -0.36684564  1.0176145
##            V12        V13         V14         V15        V16          V17
## 1  -0.61780086 -0.9913898 -0.31116935  1.46817697 -0.4704005  0.207971242
## 2   1.06523531  0.4890950 -0.14377230  0.63555809  0.4639170 -0.114804663
## 3   0.06608369  0.7172927 -0.16594592  2.34586495 -2.8900832  1.109969379
## 4   0.17822823  0.5077569 -0.28792375 -0.63141812 -1.0596472 -0.684092786
## 5   0.53819555  1.3458516 -1.11966983  0.17512113 -0.4514492 -0.237033239
## 6   0.35989384 -0.3580907 -0.13713370  0.51761681  0.4017259 -0.058132823
## 7  -0.15382583 -0.7510627  0.16737196  0.05014359 -0.4435868  0.002820512
## 8   0.29147435  1.7579642 -1.32386522  0.68613250 -0.0761270 -1.222127345
## 9  -0.11045226 -0.2862536  0.07435536 -0.32878305 -0.2100773 -0.499767969
## 10  0.83638957  1.0068435 -0.44352282  0.15021910  0.7394528 -0.540979922
##            V18         V19         V20          V21          V22
## 1   0.02579058  0.40399296  0.25141210 -0.018306778  0.277837576
## 2  -0.18336127 -0.14578304 -0.06908314 -0.225775248 -0.638671953
## 3  -0.12135931 -2.26185710  0.52497973  0.247998153  0.771679402
## 4   1.96577500 -1.23262197 -0.20803778 -0.108300452  0.005273597
## 5  -0.03819479  0.80348692  0.40854236 -0.009430697  0.798278495
## 6   0.06865315 -0.03319379  0.08496767 -0.208253515 -0.559824796
## 7  -0.61198734 -0.04557504 -0.21963255 -0.167716266 -0.270709726
## 8  -0.35822157  0.32450473 -0.15674185  1.943465340 -1.015454710
## 9   0.11876486  0.57032817  0.05273567 -0.073425100 -0.268091632
## 10  0.47667726  0.45177296  0.20371145 -0.246913937 -0.633752642
##            V23         V24         V25         V26          V27
```

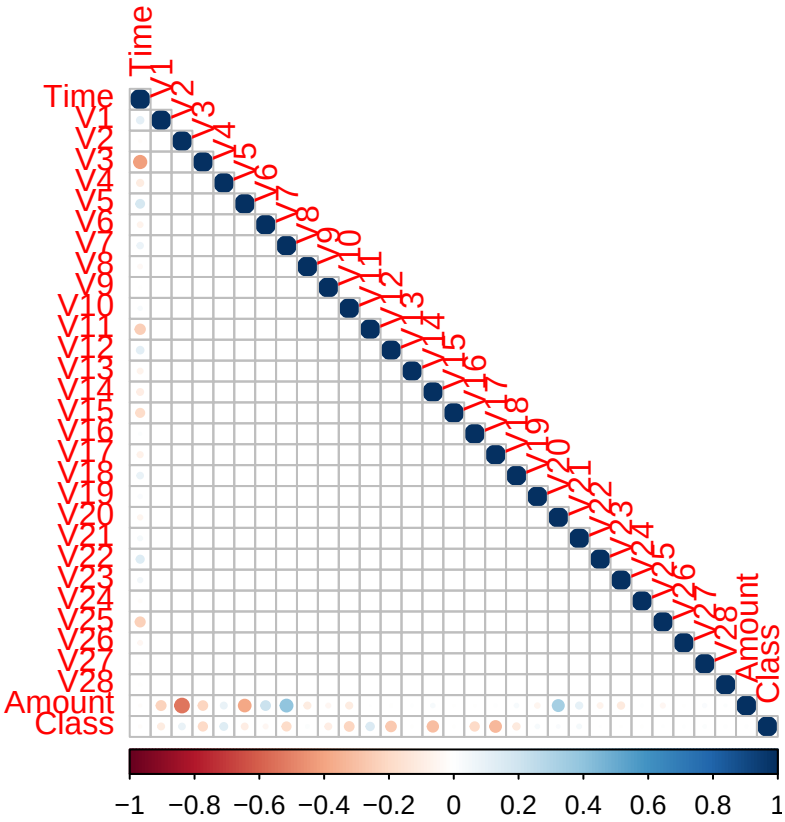
```
## 1  -0.11047391  0.06692807  0.12853936 -0.18911484  0.133558377
## 2   0.10128802 -0.33984648  0.16717040  0.12589453 -0.008983099
## 3   0.90941226 -0.68928096 -0.32764183 -0.13909657 -0.055352794
## 4  -0.19032052 -1.17557533  0.64737603 -0.22192884  0.062722849
## 5  -0.13745808  0.14126698 -0.20600959  0.50229222  0.219422230
## 6  -0.02639767 -0.37142658 -0.23279382  0.10591478  0.253844225
## 7  -0.15410379 -0.78005542  0.75013694 -0.25723685  0.034507430
## 8   0.05750353 -0.64970901 -0.41526657 -0.05163430 -1.206921081
## 9  -0.20423267  1.01159180  0.37320468 -0.38415731  0.011747356
## 10 -0.12079408 -0.38504993 -0.06973305  0.09419883  0.246219305
##             V28 Amount Class
## 1  -0.021053053 149.62     0
## 2   0.014724169   2.69     0
## 3  -0.059751841 378.66     0
## 4   0.061457629 123.50     0
## 5   0.215153147  69.99     0
## 6   0.081080257   3.67     0
## 7   0.005167769   4.99     0
## 8  -1.085339188  40.80     0
## 9   0.142404330  93.20     0
## 10  0.083075649   3.68     0
```

```
colSums(is.na(credit_card))
```

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

Imbalanced Data

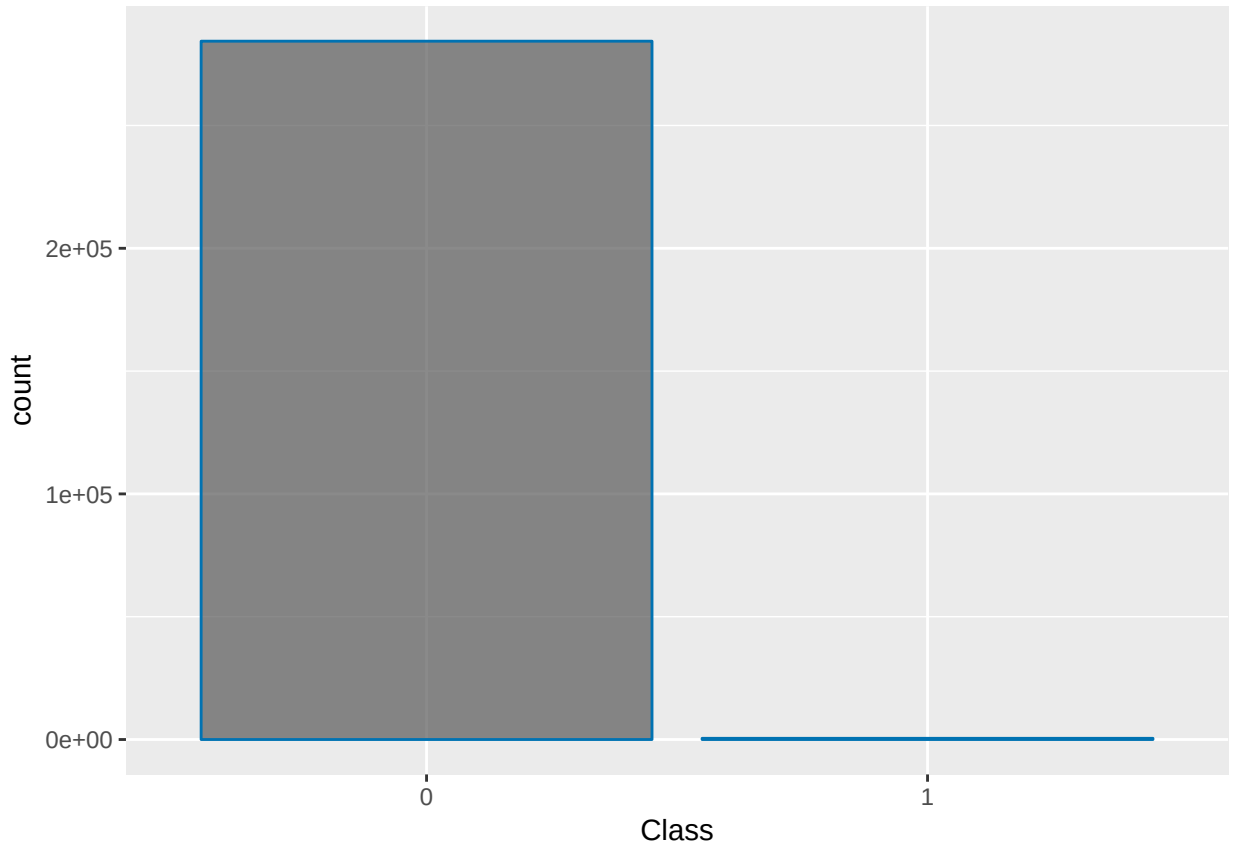
```
credit_card_corr <- cor(credit_card)
corrplot(credit_card_corr, type="lower")
```




```
credit_card$Class = as.factor(credit_card$Class)
prop.table(table(credit_card$Class))
```

```
##
##           0           1
## 0.998272514 0.001727486
```

```
ggplot(credit_card, aes(x=Class)) +
  geom_bar(alpha=0.7, colour="#0072B2")
```



DATA PREPARATION: ### Separate fraudulent and non-fraudulent data,

```
dropped_cols <- c("Time")
credit_card[ , dropped_cols] <- list(NULL)
credit_card_frauds  = filter(credit_card,credit_card$Class == 1)
head(credit_card_frauds, n=10L)
```

```
##              V1        V2         V3       V4         V5          V6
## 1  -2.312226542  1.951992 -1.6098507 3.997906 -0.5221879 -1.42654532
## 2  -3.043540624 -3.157307  1.0884628 2.288644  1.3598051 -1.06482252
## 3  -2.303349568  1.759247 -0.3597447 2.330243 -0.8216283 -0.07578757
## 4  -4.397974442  1.358367 -2.5928442 2.679787 -1.1281309 -1.70653639
## 5   1.234235046  3.019740 -4.3045969 4.732795  3.6242008 -1.35774566
## 6   0.008430365  4.137837 -6.2406966 6.675732  0.7683070 -3.35305955
## 7   0.026779226  4.132464 -6.5606000 6.348557  1.3296657 -2.51347885
## 8   0.329594333  3.712889 -5.7759351 6.078266  1.6673590 -2.42016841
## 9   0.316459000  3.809076 -5.6151590 6.047445  1.5540260 -2.65135311
## 10  0.725645740  2.300894 -5.3299762 4.007683 -1.7304106 -1.73219257
##             V7          V8         V9         V10        V11         V12
## 1  -2.5373873  1.39165725 -2.7700893 -2.7722721  3.2020332  -2.8999074
## 2   0.3255743 -0.06779365 -0.2709528 -0.8385866 -0.4145754  -0.5031409
## 3   0.5623198 -0.39914658 -0.2382534 -1.5254116  2.0329122  -6.5601243
## 4  -3.4961973 -0.24877774 -0.2477679 -4.8016374  4.8958442 -10.9128193
## 5   1.7134450 -0.49635849 -1.2828578 -2.4474693  2.1013439  -4.6096284
## 6  -1.6317347  0.15461245 -2.7958925 -6.1878906  5.6643947  -9.8544848
## 7  -1.6891022  0.30325280 -3.1394091 -6.0454678  6.7546254  -8.9481786
## 8  -0.8128912  0.13308012 -2.2143113 -5.1344545  4.5607201  -8.8737484
## 9  -0.7465793  0.05558631 -2.6786785 -4.9594929  6.4390534  -7.5201174
```

```
## 10 -3.9685926  1.06372815 -0.4860966 -4.6249850  5.5887239  -7.1482426
##            V13        V14          V15        V16         V17         V18
## 1  -0.59522188  -4.289254  0.389724120 -1.1407472  -2.8300557 -0.01682247
## 2   0.67650154  -1.692029  2.000634839  0.6667797   0.5997174  1.72532101
## 3   0.02293732  -1.470102 -0.698826069 -2.2821938  -4.7818309 -2.61566494
## 4   0.18437169  -6.771097 -0.007326183 -7.3580832 -12.5984185 -5.13154863
## 5   1.46437762  -6.079337 -0.339237373  2.5818510   6.7393844  3.04249318
## 6  -0.30616666 -10.691196 -0.638498193 -2.0419738  -1.1290559  0.11645252
## 7   0.70272500 -10.733854 -1.379519857 -1.6389601  -1.7463501  0.77674410
## 8  -0.79748360  -9.177166 -0.257024775 -0.8716885   1.3130136  0.77391387
## 9   0.38635167  -9.252307 -1.365188415 -0.5023622   0.7844266  1.49430461
## 10  1.68045074  -6.210258  0.495282118 -3.5995402  -4.8303242 -0.64909012
##           V19          V20        V21         V22        V23         V24
## 1   0.4169557  0.126910559  0.5172324 -0.03504937 -0.4652111  0.32019820
## 2   0.2833448  2.102338793  0.6616959  0.43547721  1.3759657 -0.29380315
## 3  -1.3344411 -0.430021867 -0.2941663 -0.93239106  0.1727263 -0.08732954
## 4   0.3083339 -0.171607879  0.5735741  0.17696772 -0.4362069 -0.05350186
## 5  -2.7218531  0.009060836 -0.3790683 -0.70418103 -0.6568048 -1.63265296
## 6  -1.9346657  0.488378221  0.3645142 -0.60805713 -0.5395279  0.12893998
## 7  -1.3273566  0.587743219  0.3705087 -0.57675247 -0.6696054 -0.75990753
## 8  -2.3705995  0.269772776  0.1566172 -0.65245044 -0.5515722 -0.71652164
## 9  -1.8080122  0.388307428  0.2088284 -0.51174662 -0.5838132 -0.21984503
## 10  2.2501232  0.504646226  0.5896691  0.10954132  0.6010453 -0.36470028
##            V25        V26         V27         V28 Amount Class
## 1   0.04451917  0.1778398  0.26114500 -0.14327587   0.00     1
## 2   0.27979803 -0.1453617 -0.25277312  0.03576423 529.00     1
## 3  -0.15611426 -0.5426279  0.03956599 -0.15302880 239.93     1
## 4   0.25240526 -0.6574878 -0.82713571  0.84957338  59.00     1
## 5   1.48890145  0.5667973 -0.01001622  0.14679273   1.00     1
## 6   1.48848121  0.5079627  0.73582164  0.51357374   1.00     1
## 7   1.60505555  0.5406754  0.73704038  0.49669911   1.00     1
## 8   1.41571662  0.5552647  0.53050739  0.40447405   1.00     1
## 9   1.47475258  0.4911919  0.51886828  0.40252807   1.00     1
## 10 -1.84307769  0.3519093  0.59454998  0.09937224   1.00     1
```

```
credit_card_non_frauds  = filter(credit_card,credit_card$Class == 0)
head(credit_card_non_frauds, n=10L)
```

```
##            V1          V2          V3         V4          V5          V6
## 1  -1.3598071 -0.07278117  2.53634674  1.3781552 -0.33832077  0.46238778
## 2   1.1918571  0.26615071  0.16648011  0.4481541  0.06001765 -0.08236081
## 3  -1.3583541 -1.34016307  1.77320934  0.3797796 -0.50319813  1.80049938
## 4  -0.9662717 -0.18522601  1.79299334 -0.8632913 -0.01030888  1.24720317
## 5  -1.1582331  0.87773675  1.54871785  0.4030339 -0.40719338  0.09592146
## 6  -0.4259659  0.96052304  1.14110934 -0.1682521  0.42098688 -0.02972755
## 7   1.2296576  0.14100351  0.04537077  1.2026127  0.19188099  0.27270812
## 8  -0.6442694  1.41796355  1.07438038 -0.4921990  0.94893409  0.42811846
## 9  -0.8942861  0.28615720 -0.11319221 -0.2715261  2.66959866  3.72181806
## 10 -0.3382618  1.11959338  1.04436655 -0.2221873  0.49936081 -0.24676110
##             V7          V8         V9          V10        V11         V12
## 1   0.239598554  0.09869790  0.3637870  0.09079417 -0.5515995 -0.61780086
## 2  -0.078802983  0.08510165 -0.2554251 -0.16697441  1.6127267  1.06523531
## 3   0.791460956  0.24767579 -1.5146543  0.20764287  0.6245015  0.06608369
## 4   0.237608940  0.37743587 -1.3870241 -0.05495192 -0.2264873  0.17822823
## 5   0.592940745 -0.27053268  0.8177393  0.75307443 -0.8228429  0.53819555
```

```
## 6   0.476200949  0.26031433 -0.5686714 -0.37140720  1.3412620  0.35989384
## 7  -0.005159003  0.08121294  0.4649600 -0.09925432 -1.4169072 -0.15382583
## 8   1.120631358 -3.80786424  0.6153747  1.24937618 -0.6194678  0.29147435
## 9   0.370145128  0.85108444 -0.3920476 -0.41043043 -0.7051166 -0.11045226
## 10  0.651583206  0.06953859 -0.7367273 -0.36684564  1.0176145  0.83638957
##           V13         V14         V15         V16          V17         V18
## 1  -0.9913898 -0.31116935  1.46817697 -0.4704005  0.207971242  0.02579058
## 2   0.4890950 -0.14377230  0.63555809  0.4639170 -0.114804663 -0.18336127
## 3   0.7172927 -0.16594592  2.34586495 -2.8900832  1.109969379 -0.12135931
## 4   0.5077569 -0.28792375 -0.63141812 -1.0596472 -0.684092786  1.96577500
## 5   1.3458516 -1.11966983  0.17512113 -0.4514492 -0.237033239 -0.03819479
## 6  -0.3580907 -0.13713370  0.51761681  0.4017259 -0.058132823  0.06865315
## 7  -0.7510627  0.16737196  0.05014359 -0.4435868  0.002820512 -0.61198734
## 8   1.7579642 -1.32386522  0.68613250 -0.0761270 -1.222127345 -0.35822157
## 9  -0.2862536  0.07435536 -0.32878305 -0.2100773 -0.499767969  0.11876486
## 10  1.0068435 -0.44352282  0.15021910  0.7394528 -0.540979922  0.47667726
##            V19         V20          V21          V22         V23
## 1   0.40399296  0.25141210 -0.018306778  0.277837576 -0.11047391
## 2  -0.14578304 -0.06908314 -0.225775248 -0.638671953  0.10128802
## 3  -2.26185710  0.52497973  0.247998153  0.771679402  0.90941226
## 4  -1.23262197 -0.20803778 -0.108300452  0.005273597 -0.19032052
## 5   0.80348692  0.40854236 -0.009430697  0.798278495 -0.13745808
## 6  -0.03319379  0.08496767 -0.208253515 -0.559824796 -0.02639767
## 7  -0.04557504 -0.21963255 -0.167716266 -0.270709726 -0.15410379
## 8   0.32450473 -0.15674185  1.943465340 -1.015454710  0.05750353
## 9   0.57032817  0.05273567 -0.073425100 -0.268091632 -0.20423267
## 10  0.45177296  0.20371145 -0.246913937 -0.633752642 -0.12079408
##            V24         V25         V26          V27          V28 Amount
## 1   0.06692807  0.12853936 -0.18911484  0.133558377 -0.021053053 149.62
## 2  -0.33984648  0.16717040  0.12589453 -0.008983099  0.014724169   2.69
## 3  -0.68928096 -0.32764183 -0.13909657 -0.055352794 -0.059751841 378.66
## 4  -1.17557533  0.64737603 -0.22192884  0.062722849  0.061457629 123.50
## 5   0.14126698 -0.20600959  0.50229222  0.219422230  0.215153147  69.99
## 6  -0.37142658 -0.23279382  0.10591478  0.253844225  0.081080257   3.67
## 7  -0.78005542  0.75013694 -0.25723685  0.034507430  0.005167769   4.99
## 8  -0.64970901 -0.41526657 -0.05163430 -1.206921081 -1.085339188  40.80
## 9   1.01159180  0.37320468 -0.38415731  0.011747356  0.142404330  93.20
## 10 -0.38504993 -0.06973305  0.09419883  0.246219305  0.083075649   3.68
##    Class
## 1      0
## 2      0
## 3      0
## 4      0
## 5      0
## 6      0
## 7      0
## 8      0
## 9      0
## 10     0
```

Create train and test data - 75% AND 25%

```
credit_card$Amount <- scale(as.vector(credit_card$Amount))
credit_card$Amount = as.numeric(credit_card$Amount)
set.seed(3456)
train_index <- createDataPartition(credit_card$Class, p = .75,
                                   list = FALSE,
                                   times = 1)
class(train_index)
```

```
## [1] "matrix"
```

```
length(train_index)
```

```
## [1] 213606
```

```
nrow(credit_card)
```

```
## [1] 284807
```

```
credit_card_split <- list()
credit_card_split$train <- credit_card[train_index, ]
credit_card_split$test  <- credit_card[-train_index, ]
head(credit_card_split$train)
```

```
##            V1          V2         V3         V4          V5          V6
## 1  -1.3598071 -0.07278117  2.5363467  1.3781552 -0.33832077  0.46238778
## 2   1.1918571  0.26615071  0.1664801  0.4481541  0.06001765 -0.08236081
## 6  -0.4259659  0.96052304  1.1411093 -0.1682521  0.42098688 -0.02972755
## 8  -0.6442694  1.41796355  1.0743804 -0.4921990  0.94893409  0.42811846
## 9  -0.8942861  0.28615720 -0.1131922 -0.2715261  2.66959866  3.72181806
## 13  1.2499987 -1.22163681  0.3839302 -1.2348987 -1.48541947 -0.75323016
##             V7          V8         V9         V10        V11        V12
## 1   0.23959855  0.09869790  0.3637870  0.09079417 -0.5515995 -0.6178009
## 2  -0.07880298  0.08510165 -0.2554251 -0.16697441  1.6127267  1.0652353
## 6   0.47620095  0.26031433 -0.5686714 -0.37140720  1.3412620  0.3598938
## 8   1.12063136 -3.80786424  0.6153747  1.24937618 -0.6194678  0.2914744
## 9   0.37014513  0.85108444 -0.3920476 -0.41043043 -0.7051166 -0.1104523
## 13 -0.68940498 -0.22748723 -2.0940106  1.32372927  0.2276662 -0.2426820
##           V13         V14        V15        V16         V17         V18
## 1  -0.9913898 -0.31116935  1.4681770 -0.4704005  0.20797124  0.02579058
## 2   0.4890950 -0.14377230  0.6355581  0.4639170 -0.11480466 -0.18336127
## 6  -0.3580907 -0.13713370  0.5176168  0.4017259 -0.05813282  0.06865315
## 8   1.7579642 -1.32386522  0.6861325 -0.0761270 -1.22212735 -0.35822157
## 9  -0.2862536  0.07435536 -0.3287831 -0.2100773 -0.49976797  0.11876486
## 13  1.2054168 -0.31763053  0.7256750 -0.8156122  0.87393645 -0.84778860
##            V19         V20         V21        V22         V23         V24
## 1   0.40399296  0.25141210 -0.01830678  0.2778376 -0.11047391  0.06692807
## 2  -0.14578304 -0.06908314 -0.22577525 -0.6386720  0.10128802 -0.33984648
## 6  -0.03319379  0.08496767 -0.20825351 -0.5598248 -0.02639767 -0.37142658
## 8   0.32450473 -0.15674185  1.94346534 -1.0154547  0.05750353 -0.64970901
## 9   0.57032817  0.05273567 -0.07342510 -0.2680916 -0.20423267  1.01159180
## 13 -0.68319263 -0.10275594 -0.23180924 -0.4832853  0.08466769  0.39283089
##           V25        V26          V27         V28      Amount Class
## 1   0.1285394 -0.1891148  0.133558377 -0.02105305  0.24496383     0
## 2   0.1671704  0.1258945 -0.008983099  0.01472417 -0.34247394     0
```

```
## 6  -0.2327938  0.1059148  0.253844225  0.08108026 -0.33855582     0
## 8  -0.4152666 -0.0516343 -1.206921081 -1.08533919 -0.19010714     0
## 9   0.3732047 -0.3841573  0.011747356  0.14240433  0.01939221     0
## 13  0.1611346 -0.3549900  0.026415549  0.04242209  0.13253785     0
```

```
head(credit_card_split$test)
```

```
##            V1         V2         V3         V4          V5          V6
## 3  -1.3583541 -1.3401631 1.77320934  0.3797796 -0.50319813  1.80049938
## 4  -0.9662717 -0.1852260 1.79299334 -0.8632913 -0.01030888  1.24720317
## 5  -1.1582331  0.8777368 1.54871785  0.4030339 -0.40719338  0.09592146
## 7   1.2296576  0.1410035 0.04537077  1.2026127  0.19188099  0.27270812
## 10 -0.3382618  1.1195934 1.04436655 -0.2221873  0.49936081 -0.24676110
## 11  1.4490438 -1.1763388 0.91385983 -1.3756667 -1.97138317 -0.62915214
##              V7          V8         V9          V10        V11         V12
## 3   0.791460956  0.24767579 -1.5146543  0.20764287  0.6245015  0.06608369
## 4   0.237608940  0.37743587 -1.3870241 -0.05495192 -0.2264873  0.17822823
## 5   0.592940745 -0.27053268  0.8177393  0.75307443 -0.8228429  0.53819555
## 7  -0.005159003  0.08121294  0.4649600 -0.09925432 -1.4169072 -0.15382583
## 10  0.651583206  0.06953859 -0.7367273 -0.36684564  1.0176145  0.83638957
## 11 -1.423235601  0.04845589 -1.7204084  1.62665906  1.1996439 -0.67143978
##           V13         V14         V15         V16          V17         V18
## 3   0.7172927 -0.16594592  2.34586495 -2.89008319  1.109969379 -0.12135931
## 4   0.5077569 -0.28792375 -0.63141812 -1.05964725 -0.684092786  1.96577500
## 5   1.3458516 -1.11966983  0.17512113 -0.45144918 -0.237033239 -0.03819479
## 7  -0.7510627  0.16737196  0.05014359 -0.44358680  0.002820512 -0.61198734
## 10  1.0068435 -0.44352282  0.15021910  0.73945278 -0.540979922  0.47667726
## 11 -0.5139472 -0.09504505  0.23093041  0.03196747  0.253414716  0.85434381
##            V19        V20          V21          V22         V23         V24
## 3  -2.26185710  0.5249797  0.247998153  0.771679402  0.90941226 -0.6892810
## 4  -1.23262197 -0.2080378 -0.108300452  0.005273597 -0.19032052 -1.1755753
## 5   0.80348692  0.4085424 -0.009430697  0.798278495 -0.13745808  0.1412670
## 7  -0.04557504 -0.2196326 -0.167716266 -0.270709726 -0.15410379 -0.7800554
## 10  0.45177296  0.2037115 -0.246913937 -0.633752642 -0.12079408 -0.3850499
## 11 -0.22136541 -0.3872265 -0.009301897  0.313894411  0.02774016  0.5005123
##            V25         V26         V27          V28      Amount Class
## 3  -0.32764183 -0.13909657 -0.05535279 -0.059751841  1.16068389     0
## 4   0.64737603 -0.22192884  0.06272285  0.061457629  0.14053401     0
## 5  -0.20600959  0.50229222  0.21942223  0.215153147 -0.07340321     0
## 7   0.75013694 -0.25723685  0.03450743  0.005167769 -0.33327836     0
## 10 -0.06973305  0.09419883  0.24621930  0.083075649 -0.33851584     0
## 11  0.25136736 -0.12947795  0.04284987  0.016253262 -0.32204376     0
```

```
prop.table(table(credit_card_split$train$Class))
```

```
##
##          0          1
## 0.99827252 0.00172748
```

```
prop.table(table(credit_card_split$test$Class))
```

```
##
##           0           1
## 0.998272496 0.001727504
```

SMOTE

```
library(DMwR)
```

```
## Loading required package: grid
```

```
##
## Attaching package: 'DMwR'
```

```
## The following object is masked from 'package:graph':
##
##     join
```

```
credit_card_somte_train <- cbind(credit_card_split$train)
credit_card_somte_train <- DMwR::SMOTE(Class ~ ., credit_card_somte_train, perc.over = 100, perc.under=
prop.table(table(credit_card_split$train$Class))
```

```
##
##          0          1
## 0.99827252 0.00172748
```

```
prop.table(table(credit_card_somte_train$Class))
```

```
##
##   0   1
## 0.5 0.5
```

ROSE

```
library(ROSE)
```

```
## Loaded ROSE 0.0-3
```

```
##
## Attaching package: 'ROSE'
```

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

```
credit_card_rose_train <- cbind(credit_card_split$train)
credit_card_rose_train <- ROSE::ROSE(Class ~ ., data=credit_card_rose_train,  seed = 1)$data
prop.table(table(credit_card_split$train$Class))
```

```
##
##          0          1
## 0.99827252 0.00172748
```

```
prop.table(table(credit_card_rose_train$Class))
```

```
##
##         0         1
## 0.4995038 0.5004962
```

Run RandomForest with SMOTE

```
predictors = paste(names(credit_card)[names(credit_card) != 'Class'], collapse = "+")
model_formula = paste("Class", predictors, sep="~")
rf_titanic_train <- randomForest(formula(model_formula),
                      data=credit_card_somte_train,
                      importance=TRUE, type="classification",
                      ntree=25)
summary(rf_titanic_train)
```

```
##                 Length Class  Mode
## call               6   -none- call
## type               1   -none- character
## predicted       1476   factor numeric
## err.rate          75   -none- numeric
## confusion          6   -none- numeric
## votes           2952   matrix numeric
## oob.times       1476   -none- numeric
## classes            2   -none- character
## importance       116   -none- numeric
## importanceSD      87   -none- numeric
## localImportance    0   -none- NULL
## proximity          0   -none- NULL
## ntree              1   -none- numeric
## mtry               1   -none- numeric
## forest            14   -none- list
## y               1476   factor numeric
## test               0   -none- NULL
## inbag              0   -none- NULL
## terms              3   terms  call
```

```
rf_credit_card_fraud_preds = predict(rf_titanic_train, newdata = credit_card_split$test)
head(rf_credit_card_fraud_preds)
```

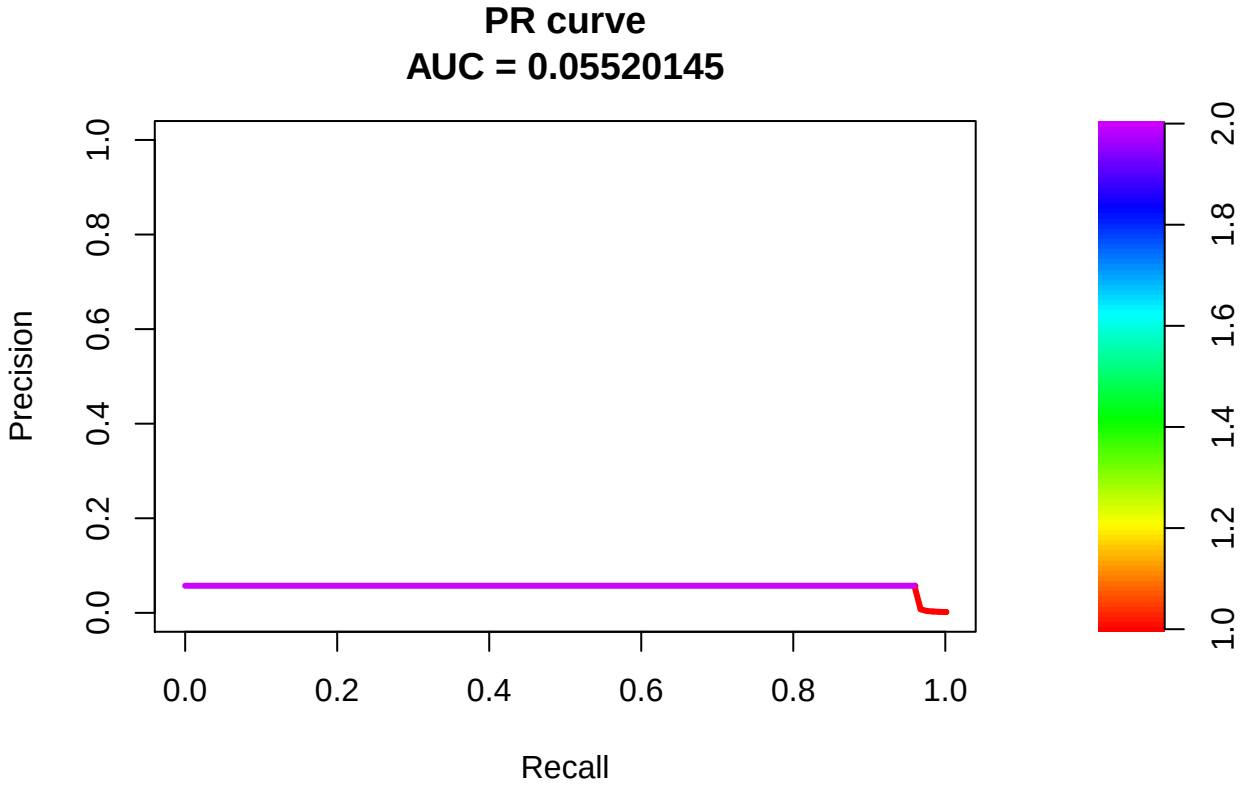
```
##  3  4  5  7 10 11
##  0  0  0  0  0  0
## Levels: 0 1
```

```
#Confusion Matrix
confusionMatrix(table(rf_credit_card_fraud_preds, credit_card_split$test$Class))
```

```
## Confusion Matrix and Statistics
##
##
## rf_credit_card_fraud_preds     0     1
##                          0 69136     5
##                          1  1942   118
##
##                Accuracy : 0.9727
##                  95% CI : (0.9714, 0.9738)
##     No Information Rate : 0.9983
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1052
##  Mcnemar's Test P-Value : <2e-16
```

```
##
##             Sensitivity : 0.97268
##             Specificity : 0.95935
##          Pos Pred Value : 0.99993
##          Neg Pred Value : 0.05728
##              Prevalence : 0.99827
##          Detection Rate : 0.97100
##    Detection Prevalence : 0.97107
##       Balanced Accuracy : 0.96601
##
##        'Positive' Class : 0
##
```

```
fg <- rf_credit_card_fraud_preds[credit_card_split$test$Class == 1]
bg <- rf_credit_card_fraud_preds[credit_card_split$test$Class == 0]
# PR Curve
pr <- pr.curve(scores.class0 = fg, scores.class1 = bg, curve = T)
plot(pr)
```



Run RandomForest with ROSE

```
rf_titanic_train_1 <- randomForest(formula(model_formula),
                      data=credit_card_rose_train,
                      importance=TRUE, type="classification",
                      ntree=25)
summary(rf_titanic_train_1)
```

```
##                 Length Class  Mode
## call                 6 -none- call
## type                 1 -none- character
## predicted       213606 factor numeric
```

```
## err.rate            75 -none- numeric
## confusion            6 -none- numeric
## votes           427212 matrix numeric
## oob.times       213606 -none- numeric
## classes              2 -none- character
## importance         116 -none- numeric
## importanceSD        87 -none- numeric
## localImportance      0 -none- NULL
## proximity            0 -none- NULL
## ntree                1 -none- numeric
## mtry                 1 -none- numeric
## forest              14 -none- list
## y               213606 factor numeric
## test                 0 -none- NULL
## inbag                0 -none- NULL
## terms                3 terms  call
```

```
rf_credit_card_fraud_preds = predict(rf_titanic_train_1, newdata = credit_card_split$test)
head(rf_credit_card_fraud_preds)
```

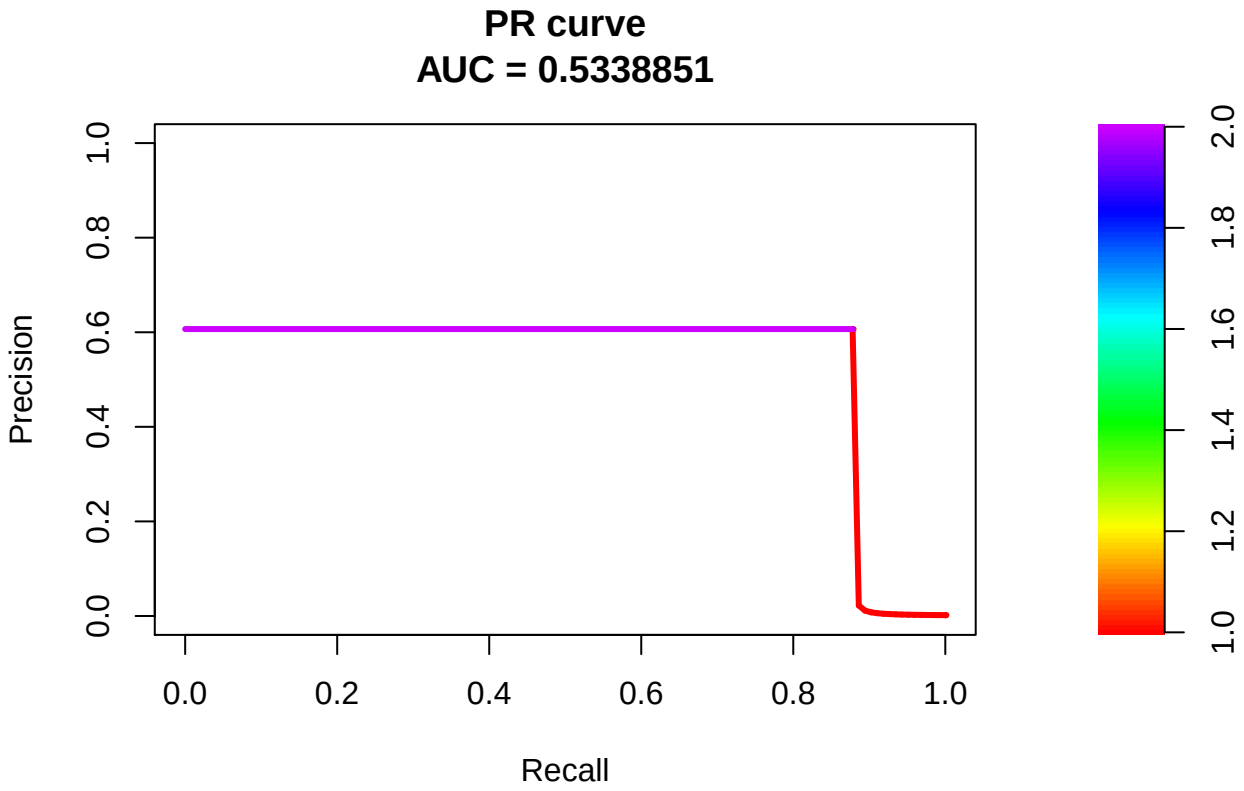
```
##  3  4  5  7 10 11
##  0  0  0  0  0  0
## Levels: 0 1
```

```
#Confusion Matrix
confusionMatrix(table(rf_credit_card_fraud_preds, credit_card_split$test$Class))
```

```
## Confusion Matrix and Statistics
##
##
## rf_credit_card_fraud_preds     0     1
##                          0 71008    15
##                          1    70   108
##
##                Accuracy : 0.9988
##                  95% CI : (0.9985, 0.999)
##     No Information Rate : 0.9983
##     P-Value [Acc > NIR] : 0.000181
##
##                   Kappa : 0.717
##  Mcnemar's Test P-Value : 4.71e-09
##
##             Sensitivity : 0.9990
##             Specificity : 0.8780
##          Pos Pred Value : 0.9998
##          Neg Pred Value : 0.6067
##              Prevalence : 0.9983
##          Detection Rate : 0.9973
##    Detection Prevalence : 0.9975
##       Balanced Accuracy : 0.9385
##
##        'Positive' Class : 0
##
```

```
fg <- rf_credit_card_fraud_preds[credit_card_split$test$Class == 1]
bg <- rf_credit_card_fraud_preds[credit_card_split$test$Class == 0]
```

```
# PR Curve
pr <- pr.curve(scores.class0 = fg, scores.class1 = bg, curve = T)
plot(pr)
```



Run Gradeint Boosting with SMOTE

```
library(xgboost)
```

```
## Warning: package 'xgboost' was built under R version 3.3.2
```

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

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

```
bst<- xgboost(data = as.matrix(credit_card_somte_train[, -ncol(credit_card_somte_train)])
                      , label = as.numeric(as.character(credit_card_somte_train$Class))
                      , max_depth = 2
                      , eta = 1
                      , nthread = 2
                      , nrounds = 2
                      , objective = "binary:logistic"
)
```

```
## [1]  train-error:0.081301
## [2]  train-error:0.065718
```

```
summary(bst)
```

```
##                Length Class              Mode
## handle            1   xgb.Booster.handle externalptr
```

```
## raw            1171   -none-            raw
## niter             1   -none-            numeric
## evaluation_log    2   data.table        list
## call             17   -none-            call
## params            5   -none-            list
## callbacks         3   -none-            list
```

```
credit_card_test_smote = cbind(credit_card_split$test)
credit_card_test_smote$Class = as.numeric(as.character(credit_card_test_smote$Class))
rf_credit_card_fraud_preds = predict(bst, newdata = as.matrix(credit_card_test_smote))
head(rf_credit_card_fraud_preds)
```

```
## [1] 0.07095362 0.07095362 0.20324460 0.07095362 0.07095362 0.07095362
```

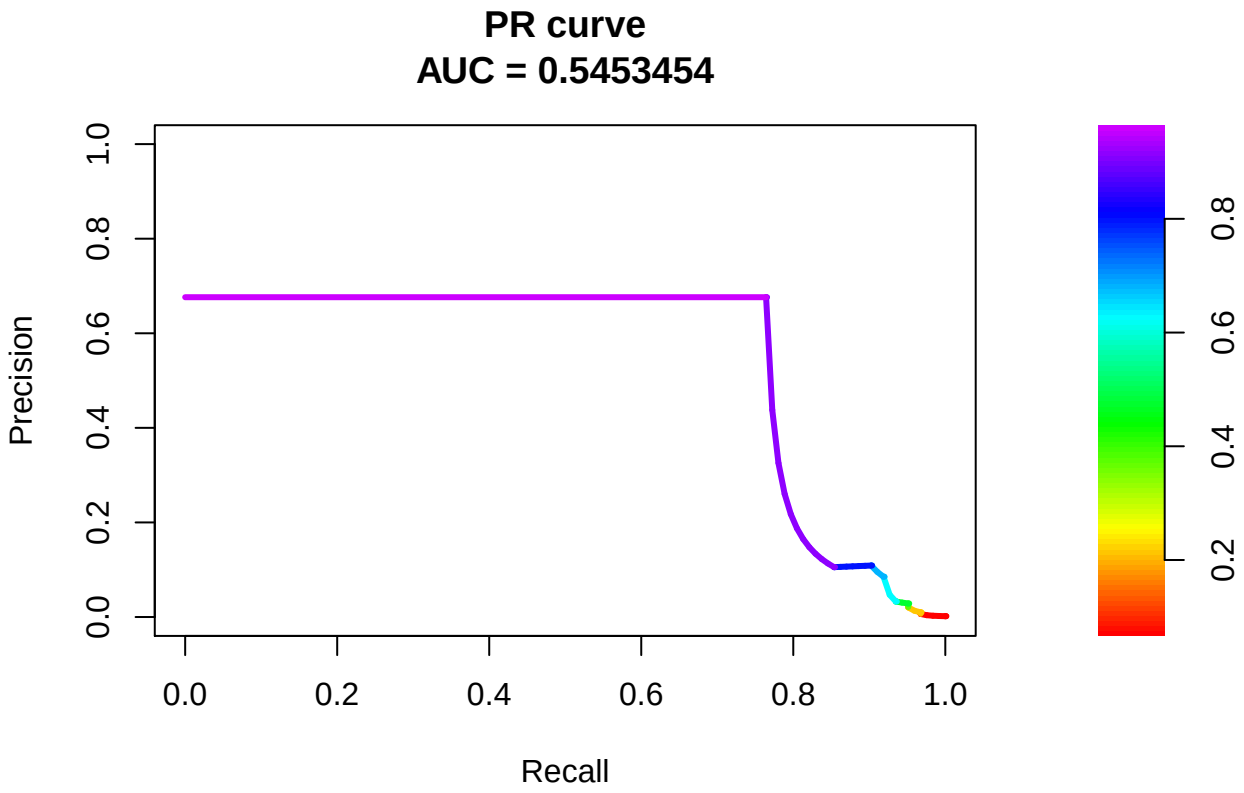
```
length(rf_credit_card_fraud_preds)
```

```
## [1] 71201
```

```
nrow(credit_card_test_smote)
```

```
## [1] 71201
```

```
#Confusion Matrix
credit_card_test_smote$Class = as.factor(credit_card_test_smote$Class)
#confusionMatrix(rf_credit_card_fraud_preds, credit_card_test_smote$Class)
fg <- rf_credit_card_fraud_preds[credit_card_test_smote$Class == 1]
bg <- rf_credit_card_fraud_preds[credit_card_test_smote$Class == 0]
# PR Curve
pr <- pr.curve(scores.class0 = fg, scores.class1 = bg, curve = T)
plot(pr)
```



Run Gradeint Boosting with ROSE

```
bst_rose<- xgboost(data = as.matrix(credit_card_rose_train[, -ncol(credit_card_rose_train)])
                   , label = as.numeric(as.character(credit_card_rose_train$Class))
                   , max_depth = 2
                   , eta = 1
                   , nthread = 2
                   , nrounds = 2
                   , objective = "binary:logistic"
)
```

```
## [1]  train-error:0.065986
## [2]  train-error:0.042728
```

```
summary(bst_rose)
```

```
##                Length Class              Mode
## handle            1   xgb.Booster.handle externalptr
## raw            1171   -none-             raw
## niter             1   -none-             numeric
## evaluation_log    2   data.table         list
## call             17   -none-             call
## params            5   -none-             list
## callbacks         3   -none-             list
```

```
credit_card_test_rose = cbind(credit_card_split$test)
credit_card_test_rose$Class = as.numeric(as.character(credit_card_test_rose$Class))
rf_credit_card_fraud_preds = predict(bst_rose, newdata = as.matrix(credit_card_test_rose))
head(rf_credit_card_fraud_preds)
```

```
## [1] 0.08125161 0.08125161 0.08125161 0.08125161 0.08125161 0.08125161
```

```
length(rf_credit_card_fraud_preds)
```

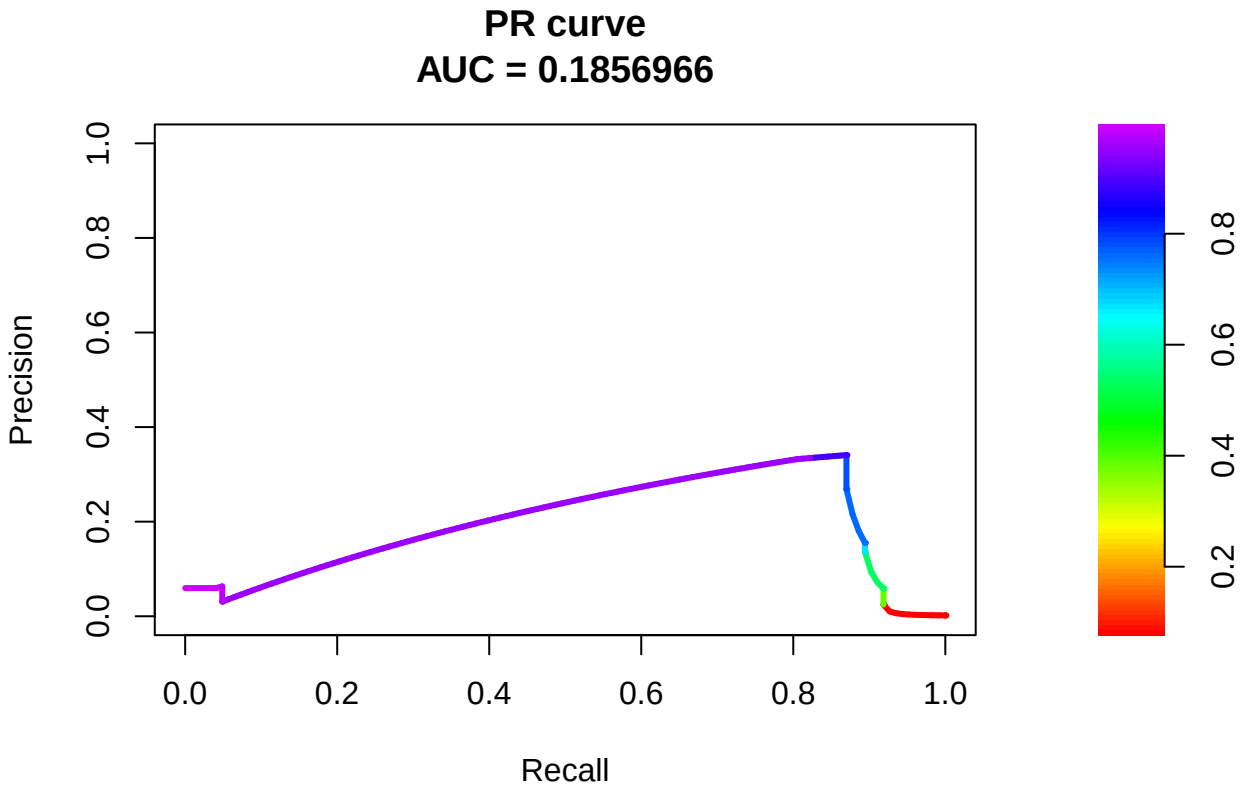
```
## [1] 71201
```

```
nrow(credit_card_test_rose)
```

```
## [1] 71201
```

```
#Confusion Matrix

credit_card_test_rose$Class = as.factor(credit_card_test_rose$Class)
#confusionMatrix(rf_credit_card_fraud_preds, credit_card_test_smote$Class)
fg <- rf_credit_card_fraud_preds[credit_card_test_rose$Class == 1]
bg <- rf_credit_card_fraud_preds[credit_card_test_rose$Class == 0]
# PR Curve
pr <- pr.curve(scores.class0 = fg, scores.class1 = bg, curve = T)
plot(pr)
```


PR curve
AUC = 0.1856966
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
0.2
0.4
0.6
0.8