

Credit Card Fraud Detection

Shreeram Venkatesh

2023-10-01

PROJECT CODE :

```
library(ranger)

## Warning: package 'ranger' was built under R version 4.2.3

library(caret)

## Warning: package 'caret' was built under R version 4.2.3

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 4.2.3

## Loading required package: lattice

library(data.table)
library(pROC)

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

##
## Attaching package: 'pROC'

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

library(caTools)

## Warning: package 'caTools' was built under R version 4.2.3

creditcard_data <- read.csv("D:/SourceCodeforMSProjects/R-Project/Credit Card Fraud Detection/creditcar

dim(creditcard_data)

## [1] 284807      31
```

```
head(creditcard_data, 6)
```

```
##   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
```

```
tail(creditcard_data, 6)
```

```
##   Time      V1      V2      V3      V4      V5
## 284802 172785  0.1203164  0.93100513 -0.5460121 -0.7450968  1.13031398
## 284803 172786 -11.8811179 10.07178497 -9.8347835 -2.0666557 -5.36447278
## 284804 172787 -0.7327887 -0.05508049  2.0350297 -0.7385886  0.86822940
## 284805 172788  1.9195650 -0.30125385 -3.2496398 -0.5578281  2.63051512
## 284806 172788 -0.2404400  0.53048251  0.7025102  0.6897992 -0.37796113
## 284807 172792 -0.5334125 -0.18973334  0.7033374 -0.5062712 -0.01254568
##          V6      V7      V8      V9      V10     V11
## 284802 -0.2359732  0.8127221  0.1150929 -0.2040635 -0.6574221  0.6448373
## 284803 -2.6068373 -4.9182154  7.3053340  1.9144283  4.3561704 -1.5931053
## 284804  1.0584153  0.0243297  0.2948687  0.5848000 -0.9759261 -0.1501888
## 284805  3.0312601 -0.2968265  0.7084172  0.4324540 -0.4847818  0.4116137
## 284806  0.6237077 -0.6861800  0.6791455  0.3920867 -0.3991257 -1.9338488
```

```

## 284807 -0.6496167 1.5770063 -0.4146504 0.4861795 -0.9154266 -1.0404583
##          V12      V13      V14      V15      V16      V17
## 284802  0.19091623 -0.5463289 -0.73170658 -0.80803553  0.5996281  0.07044075
## 284803  2.71194079 -0.6892556  4.62694203 -0.92445871  1.1076406  1.99169111
## 284804  0.91580191  1.2147558 -0.67514296  1.16493091 -0.7117573 -0.02569286
## 284805  0.06311886 -0.1836987 -0.51060184  1.32928351  0.1407160  0.31350179
## 284806 -0.96288614 -1.0420817  0.44962444  1.96256312 -0.6085771  0.50992846
## 284807 -0.03151305 -0.1880929 -0.08431647  0.04133346 -0.3026201 -0.66037665
##          V18      V19      V20      V21      V22      V23
## 284802  0.3731103  0.1289038  0.0006758329 -0.3142046 -0.8085204  0.05034266
## 284803  0.5106323 -0.6829197  1.4758291347  0.2134541  0.1118637  1.01447990
## 284804 -1.2211789 -1.5455561  0.0596158999  0.2142053  0.9243836  0.01246304
## 284805  0.3956525 -0.5772518  0.0013959703  0.2320450  0.5782290 -0.03750086
## 284806  1.1139806  2.8978488  0.1274335158  0.2652449  0.8000487 -0.16329794
## 284807  0.1674299 -0.2561169  0.3829481049  0.2610573  0.6430784  0.37677701
##          V24      V25      V26      V27      V28 Amount Class
## 284802  0.102799590 -0.4358701  0.1240789  0.217939865  0.06880333  2.69  0
## 284803 -0.509348453  1.4368069  0.2500343  0.943651172  0.82373096  0.77  0
## 284804 -1.016225669 -0.6066240 -0.3952551  0.068472470 -0.05352739 24.79  0
## 284805  0.640133881  0.2657455 -0.0873706  0.004454772 -0.02656083 67.88  0
## 284806  0.123205244 -0.5691589  0.5466685  0.108820735  0.10453282 10.00  0
## 284807  0.008797379 -0.4736487 -0.8182671 -0.002415309  0.01364891 217.00  0

```

```
table(creditcard_data$Class)
```

```

##
##      0      1
## 284315    492

```

```
summary(creditcard_data$Amount)
```

```

##      Min.   1st Qu.   Median   Mean   3rd Qu.   Max.
##      0.00    5.60   22.00  88.35   77.17 25691.16

```

```
names(creditcard_data)
```

```

## [1] "Time"     "V1"       "V2"       "V3"       "V4"       "V5"       "V6"       "V7"
## [9] "V8"       "V9"       "V10"      "V11"      "V12"      "V13"      "V14"      "V15"
## [17] "V16"      "V17"      "V18"      "V19"      "V20"      "V21"      "V22"      "V23"
## [25] "V24"      "V25"      "V26"      "V27"      "V28"      "Amount"   "Class"

```

```
var(creditcard_data$Amount)
```

```
## [1] 62560.07
```

```
sd(creditcard_data$Amount)
```

```
## [1] 250.1201
```

```

head(creditcard_data)

##      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

```

```

# Assuming 'creditcard_data' is your dataset

creditcard_data$Amount <- scale(creditcard_data$Amount)
NewData=creditcard_data[,-c(1)]
head(NewData)

```

```

##      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 0.24496383 0
## 2 0.1671704 0.1258945 -0.008983099 0.01472417 -0.34247394 0
## 3 -0.3276418 -0.1390966 -0.055352794 -0.05975184 1.16068389 0
## 4 0.6473760 -0.2219288 0.062722849 0.06145763 0.14053401 0
## 5 -0.2060096 0.5022922 0.219422230 0.21515315 -0.07340321 0
## 6 -0.2327938 0.1059148 0.253844225 0.08108026 -0.33855582 0

```

```

set.seed(123)
data_sample = sample.split(NewData$Class, SplitRatio=0.80)
train_data = subset(NewData, data_sample==TRUE)
test_data = subset(NewData, data_sample==FALSE)
dim(train_data)

```

```
## [1] 227846     30
```

```
dim(test_data)
```

```
## [1] 56961     30
```

```
Logistic_Model = glm(Class~., test_data, family=binomial())
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(Logistic_Model)
```

```

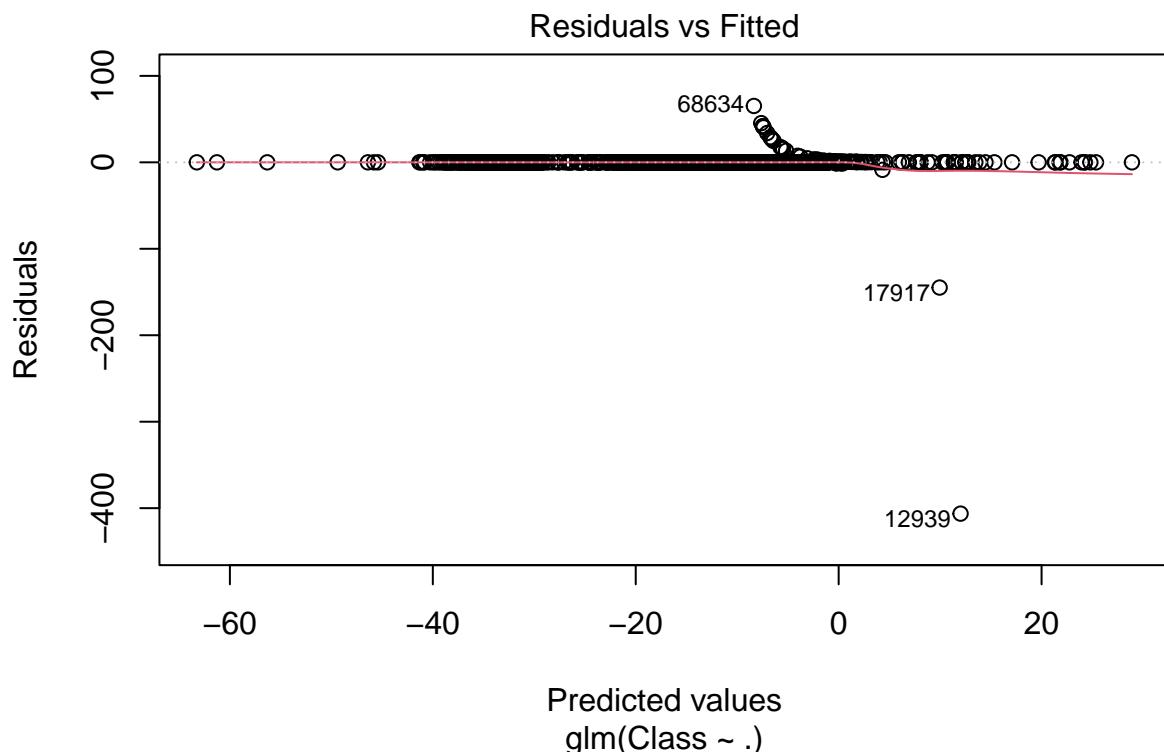
##
## Call:
## glm(formula = Class ~ ., family = binomial(), data = test_data)
##
## Deviance Residuals:
##      Min      1Q  Median      3Q      Max
##
```

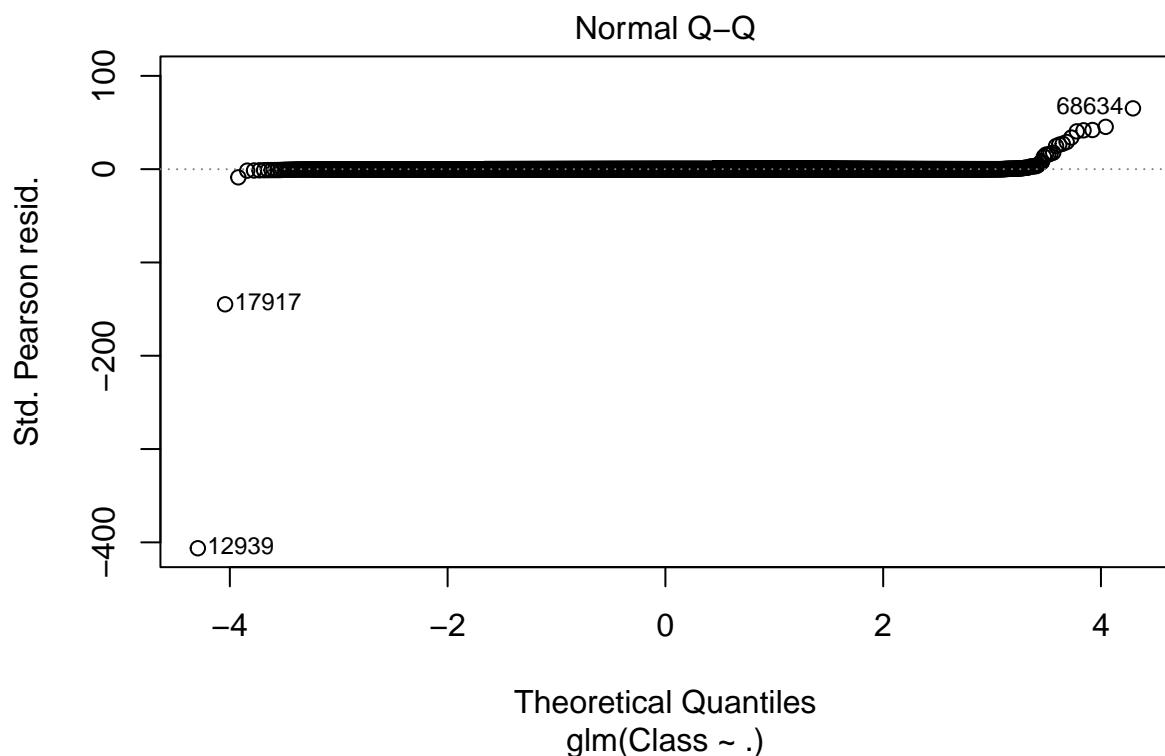
```

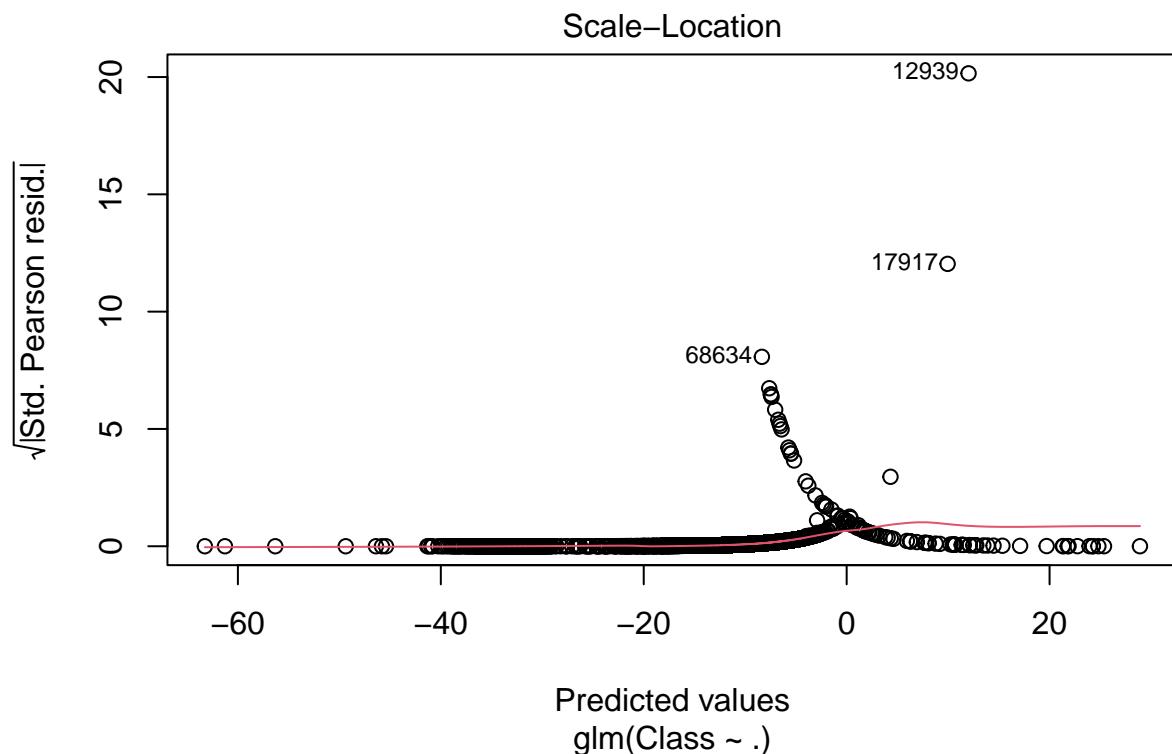
## -4.9019 -0.0254 -0.0156 -0.0078 4.0877
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -12.52800 10.30537 -1.216 0.2241
## V1          -0.17299 1.27381 -0.136 0.8920
## V2           1.44512 4.23062  0.342 0.7327
## V3           0.17897 0.24058  0.744 0.4569
## V4           3.13593 7.17768  0.437 0.6622
## V5           1.49014 3.80369  0.392 0.6952
## V6          -0.12428 0.22202 -0.560 0.5756
## V7           1.40903 4.22644  0.333 0.7388
## V8          -0.35254 0.17462 -2.019 0.0435 *
## V9           3.02176 8.67262  0.348 0.7275
## V10          -2.89571 6.62383 -0.437 0.6620
## V11          -0.09769 0.28270 -0.346 0.7297
## V12          1.97992 6.56699  0.301 0.7630
## V13          -0.71674 1.25649 -0.570 0.5684
## V14          0.19316 3.28868  0.059 0.9532
## V15          1.03868 2.89256  0.359 0.7195
## V16          -2.98194 7.11391 -0.419 0.6751
## V17          -1.81809 4.99764 -0.364 0.7160
## V18          2.74772 8.13188  0.338 0.7354
## V19          -1.63246 4.77228 -0.342 0.7323
## V20          -0.69925 1.15114 -0.607 0.5436
## V21          -0.45082 1.99182 -0.226 0.8209
## V22          -1.40395 5.18980 -0.271 0.7868
## V23          0.19026 0.61195  0.311 0.7559
## V24          -0.12889 0.44701 -0.288 0.7731
## V25          -0.57835 1.94988 -0.297 0.7668
## V26          2.65938 9.34957  0.284 0.7761
## V27          -0.45396 0.81502 -0.557 0.5775
## V28          -0.06639 0.35730 -0.186 0.8526
## Amount       0.22576 0.71892  0.314 0.7535
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 1443.40 on 56960 degrees of freedom
## Residual deviance: 378.59 on 56931 degrees of freedom
## AIC: 438.59
##
## Number of Fisher Scoring iterations: 17

```

```
plot(Logistic_Model)
```

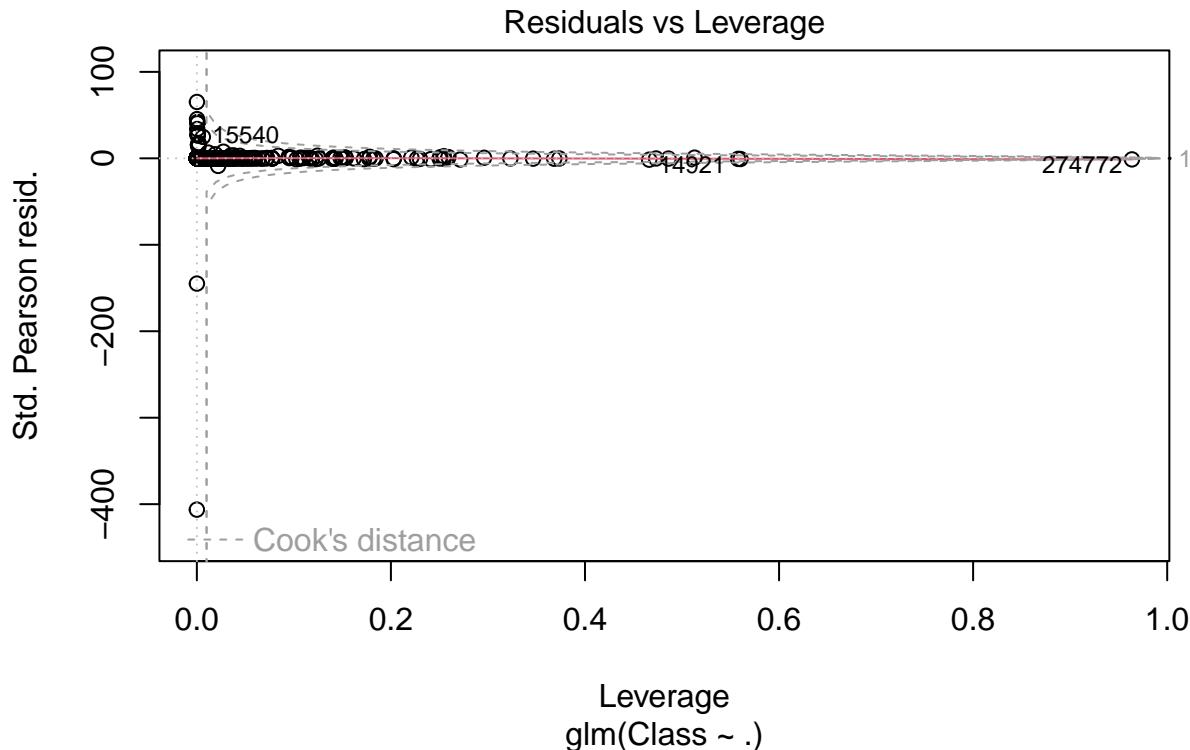






```
## Warning in sqrt(crit * p * (1 - hh)/hh): NaNs produced
```

```
## Warning in sqrt(crit * p * (1 - hh)/hh): NaNs produced
```



```
Logistic_Model = glm(Class~.,train_data,family=binomial())
summary(Logistic_Model)
```

```
##
## Call:
## glm(formula = Class ~ ., family = binomial(), data = train_data)
##
## Deviance Residuals:
##      Min        1Q     Median        3Q       Max
## -4.6108   -0.0292   -0.0194   -0.0125    4.6021
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -8.651305  0.160212 -53.999 < 2e-16 ***
## V1          0.072540  0.044144   1.643 0.100332
## V2          0.014818  0.059777   0.248 0.804220
## V3          0.026109  0.049776   0.525 0.599906
## V4          0.681286  0.078071   8.726 < 2e-16 ***
## V5          0.087938  0.071553   1.229 0.219079
## V6         -0.148083  0.085192  -1.738 0.082170 .
## V7         -0.117344  0.068940  -1.702 0.088731 .
## V8         -0.146045  0.035667  -4.095 4.23e-05 ***
## V9         -0.339828  0.117595  -2.890 0.003855 **
## V10        -0.785462  0.098486  -7.975 1.52e-15 ***
## V11         0.001492  0.085147   0.018 0.986018
```

```

## V12      0.087106  0.094869  0.918 0.358532
## V13     -0.343792  0.092381 -3.721 0.000198 ***
## V14     -0.526828  0.067084 -7.853 4.05e-15 ***
## V15     -0.095471  0.094037 -1.015 0.309991
## V16     -0.130225  0.138629 -0.939 0.347537
## V17      0.032463  0.074471  0.436 0.662900
## V18     -0.100964  0.140985 -0.716 0.473909
## V19      0.083711  0.105134  0.796 0.425897
## V20     -0.463946  0.081871 -5.667 1.46e-08 ***
## V21      0.381206  0.065880  5.786 7.19e-09 ***
## V22      0.610874  0.142086  4.299 1.71e-05 ***
## V23     -0.071406  0.058799 -1.214 0.224589
## V24      0.255791  0.170568  1.500 0.133706
## V25     -0.073955  0.142634 -0.519 0.604109
## V26      0.120841  0.202553  0.597 0.550783
## V27     -0.852018  0.118391 -7.197 6.17e-13 ***
## V28     -0.323854  0.090075 -3.595 0.000324 ***
## Amount    0.292477  0.092075  3.177 0.001491 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 5799.1  on 227845  degrees of freedom
## Residual deviance: 1790.9  on 227816  degrees of freedom
## AIC: 1850.9
##
## Number of Fisher Scoring iterations: 12

library(pROC)  # display and analyze ROC curves
lr.predict <- predict(Logistic_Model,test_data, probability = TRUE)
auc.gbm = roc(test_data$Class, lr.predict, plot = TRUE, col = "blue")

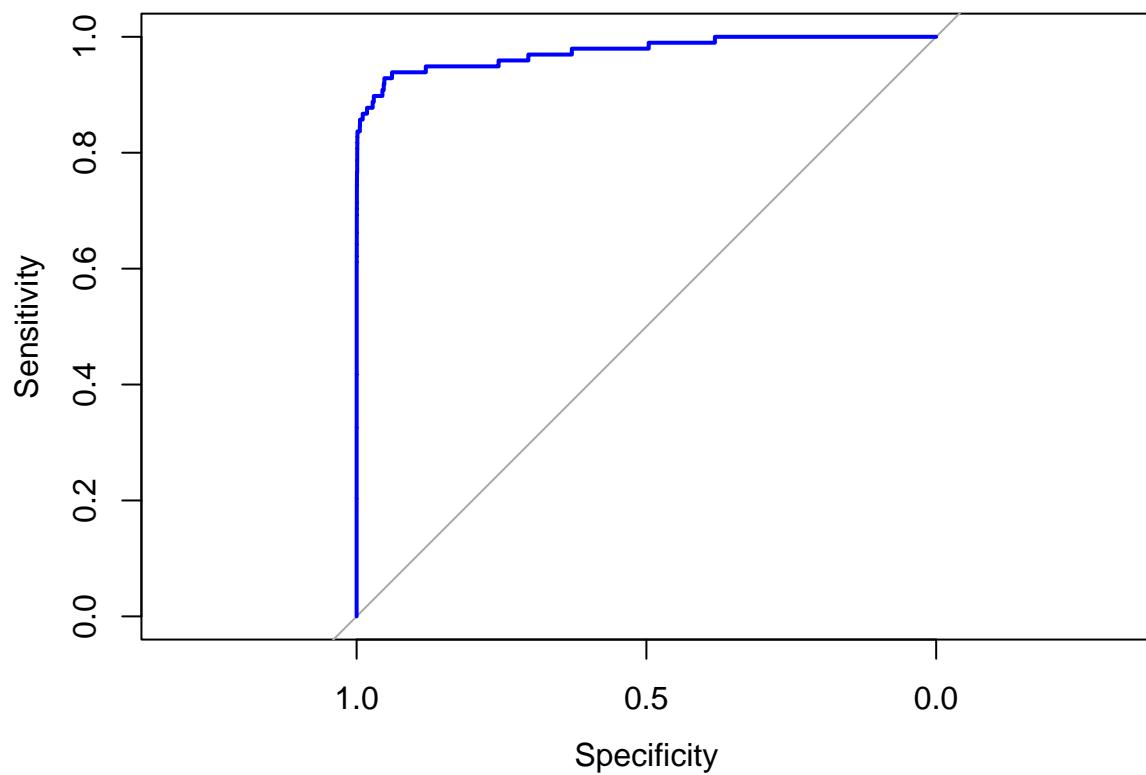
```

```

## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

```



```

library(rpart) # Recursive partitioning for classification, regression and survival trees.

## Warning: package 'rpart' was built under R version 4.2.3

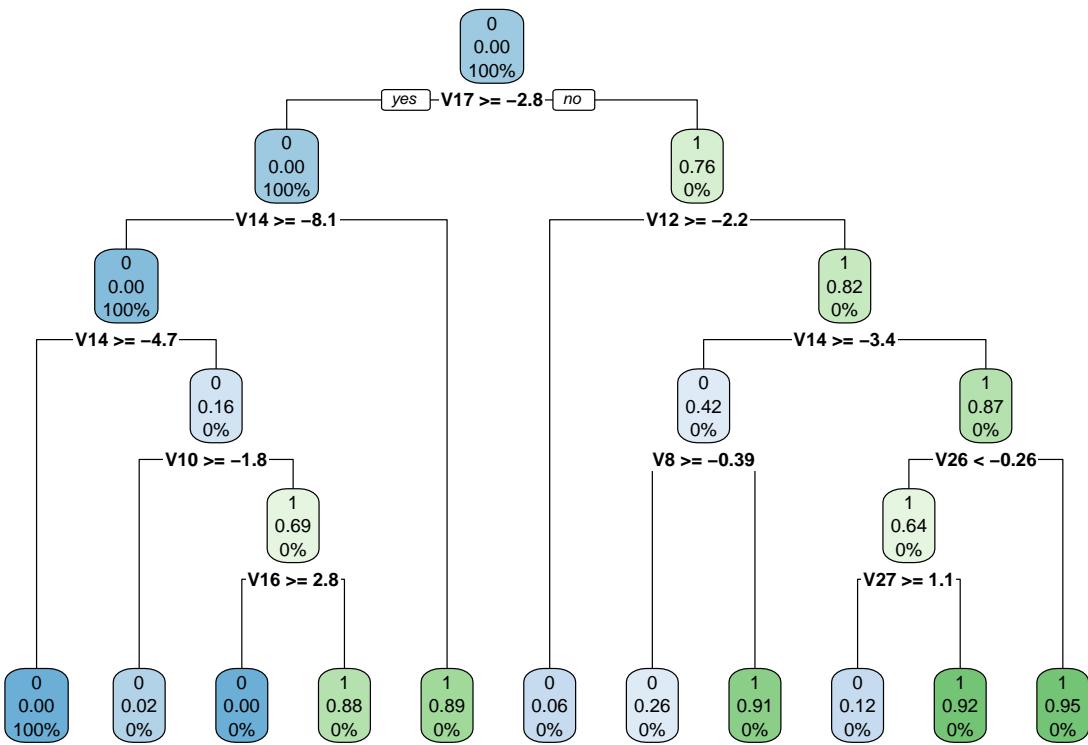
library(rpart.plot)

## Warning: package 'rpart.plot' was built under R version 4.2.3

decisionTree_model <- rpart(Class ~ . , creditcard_data, method = 'class')
predicted_val <- predict(decisionTree_model, creditcard_data, type = 'class')
probability <- predict(decisionTree_model, creditcard_data, type = 'prob')

rpart.plot(decisionTree_model)

```



```

library(neuralnet)

## Warning: package 'neuralnet' was built under R version 4.2.3

ANN_model <- neuralnet(Class~., data=train_data, linear.output=F)
plot(ANN_model)

predANN=compute(ANN_model,test_data)
resultANN=predANN$net.result
resultANN=ifelse(resultANN>0.5,1,0)
library(gbm, quietly=TRUE)

## Warning: package 'gbm' was built under R version 4.2.3

## Loaded gbm 2.1.8.1

# Get the time to train the GBM model
system.time(
  model_gbm <- gbm(Class ~ .
    , distribution = "bernoulli"
    , data = rbind(train_data, test_data)
    , n.trees = 500
    , interaction.depth = 3
    , n.minobsinnode = 100
)

```

```

        , shrinkage = 0.01
        , bag.fraction = 0.5
        , train.fraction = nrow(train_data) / (nrow(train_data) + nrow(test_data))
    )
)

##      user  system elapsed
##  308.95    0.36  311.11

# Determine best iteration based on test data
gbm.ITER = gbm.perf(model_gbm, method = "test")
model.influence = relative.influence(model_gbm, n.trees = gbm.ITER, sort. = TRUE)

plot(model_gbm)
gbm_test = predict(model_gbm, newdata = test_data, n.trees = gbm.ITER)
gbm_auc = roc(test_data$Class, gbm_test, plot = TRUE, col = "red")

## Setting levels: control = 0, case = 1
## Setting direction: controls < cases

print(gbm_auc)

##
## Call:
## roc.default(response = test_data$Class, predictor = gbm_test,      plot = TRUE, col = "red")
##
## Data: gbm_test in 56863 controls (test_data$Class 0) < 98 cases (test_data$Class 1).
## Area under the curve: 0.9555

```