In [1]:

```
--------------Loading of the Libraries-----------------------
library(pROC)
library(gbm)
library(randomForest)
library(caret)
library(readr)
library(rpart.plot)
library(caTools)
library(rpart)
library(plyr)
library(Hmisc)
```

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


Attaching package: 'pROC'


The following objects are masked from 'package:stats':

    cov, smooth, var


Loaded gbm 2.1.8

randomForest 4.6-14

Type rfNews() to see new features/changes/bug fixes.

Loading required package: lattice

Loading required package: ggplot2


Attaching package: 'ggplot2'


The following object is masked from 'package:randomForest':

    margin


Loading required package: rpart

Loading required package: survival


Attaching package: 'survival'


The following object is masked from 'package:caret':

    cluster
```

```
Loading required package: Formula


Attaching package: 'Hmisc'


The following objects are masked from 'package:plyr':

    is.discrete, summarize


The following objects are masked from 'package:base':

    format.pval, units
```

In [2]:

```
---------------Loading of Dataset-----------------------
credit_card <- read.csv("C:\\Users\\13128\\Desktop\\creditcard.csv")
```

In [3]:

```
creditcard <- credit_card
```

In [4]:

```
--------------Checking any Missing values--------------------
apply(creditcard, 2, anyNA)
table(creditcard$Class)
```

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

```
     0      1 
284315    492 
```

In [5]:

```
-----------------Setting the Seed Value----------------
set.seed(4495)
creditcard$Time <- NULL
creditcard[is.na(creditcard)] = -9999
```

In [6]:

```
-----------------Replacing the NA values-----------------
replaceNAWithMean <- function(data) {
    for(i in 1:ncol(data)){
data[is.na(data[,i]), i] <- mean(data[,i], na.rm = TRUE)
}
}
replaceNAWithMean(creditcard)
```

In [7]:

```
----------------Splitting of the Dataset--------------
set.seed(4495)
t<-createDataPartition(p=0.5,y=creditcard$Class,list = F)
training<-creditcard[t,]
testing<-creditcard[-t,]
```

In [8]:

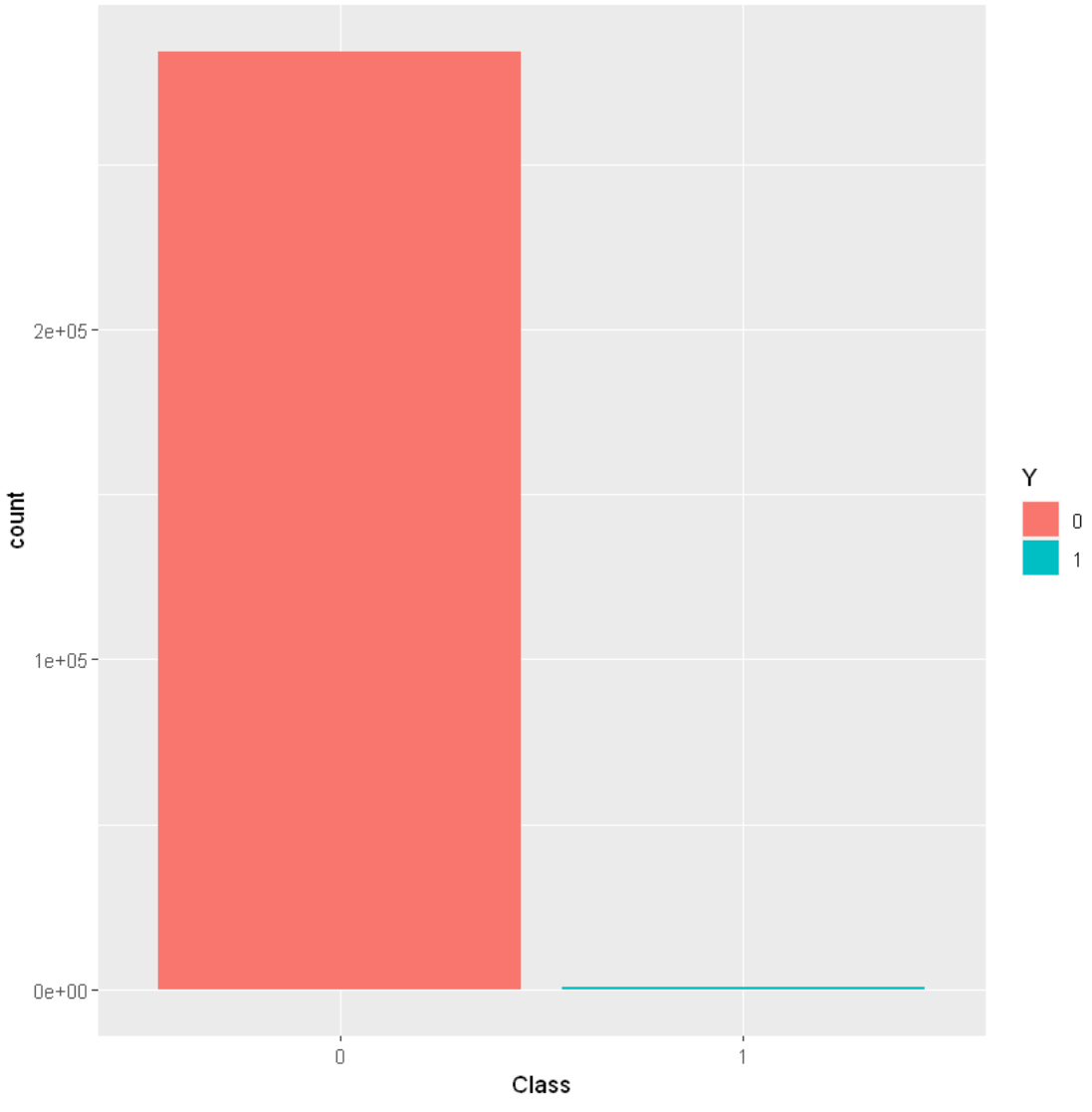
```
table(training$Class)
table(testing$Class)
```

```
     0      1 
142149    255 

     0      1 
142166    237 
```

In [9]:

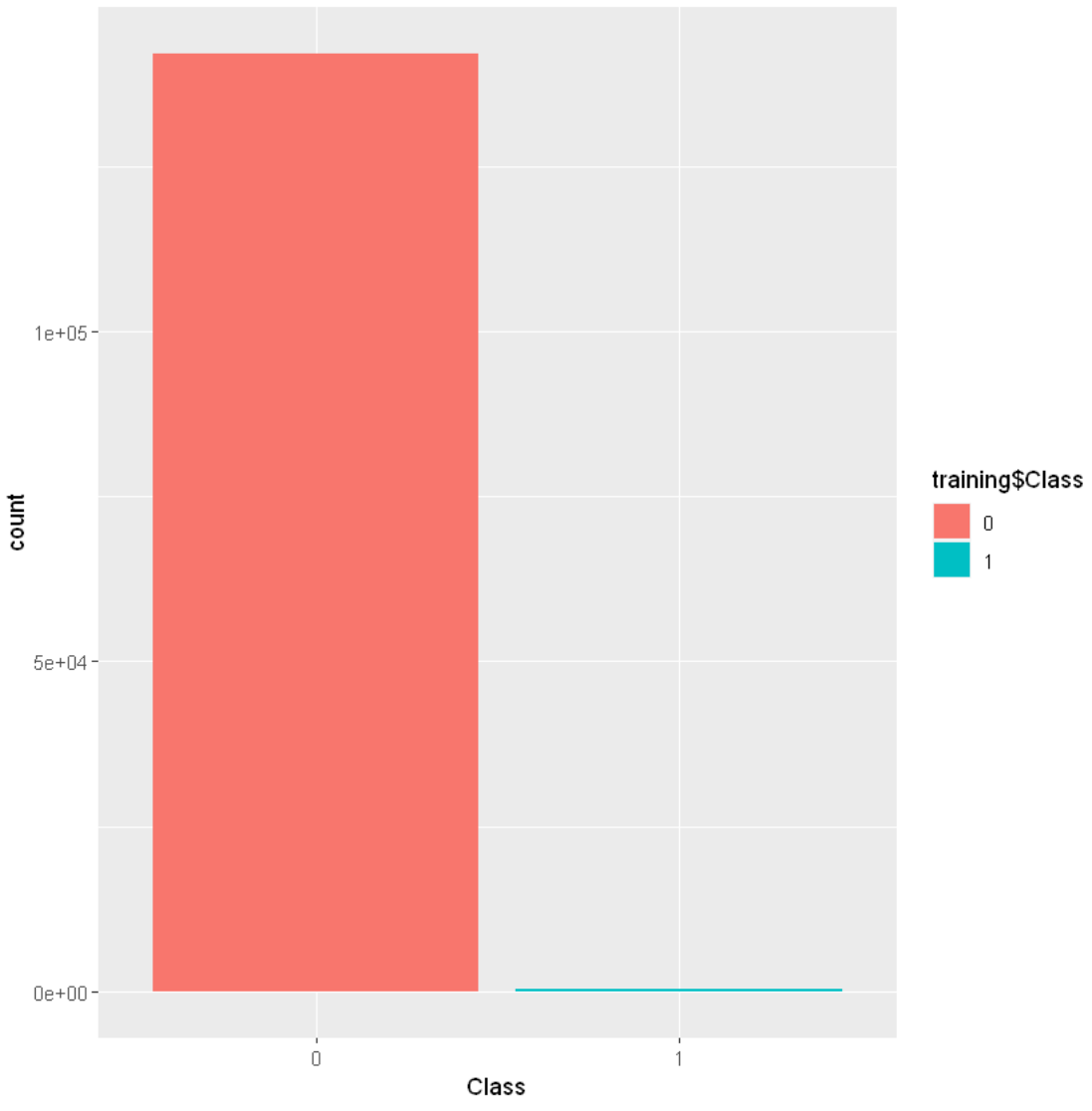
```
------------------------Visualizations-----------------------
library(ggplot2)
Y <- creditcard$Class
Y <- as.factor(Y)
ggplot(creditcard,aes(x = Y)) + geom_bar(aes(fill = Y)) + xlab('Class')
training$Class <- as.factor(training$Class)
ggplot(training,aes(x = training$Class)) + geom_bar(aes(fill = training$Class)
) + xlab('Class')
(table(Y)[1]/length(Y))*100
(table(Y)[2]/length(Y))*100
```

```
Warning message:
"Use of `training$Class` is discouraged. Use `Class` instead."
Warning message:
"Use of `training$Class` is discouraged. Use `Class` instead."
```



0: 99.827251436938

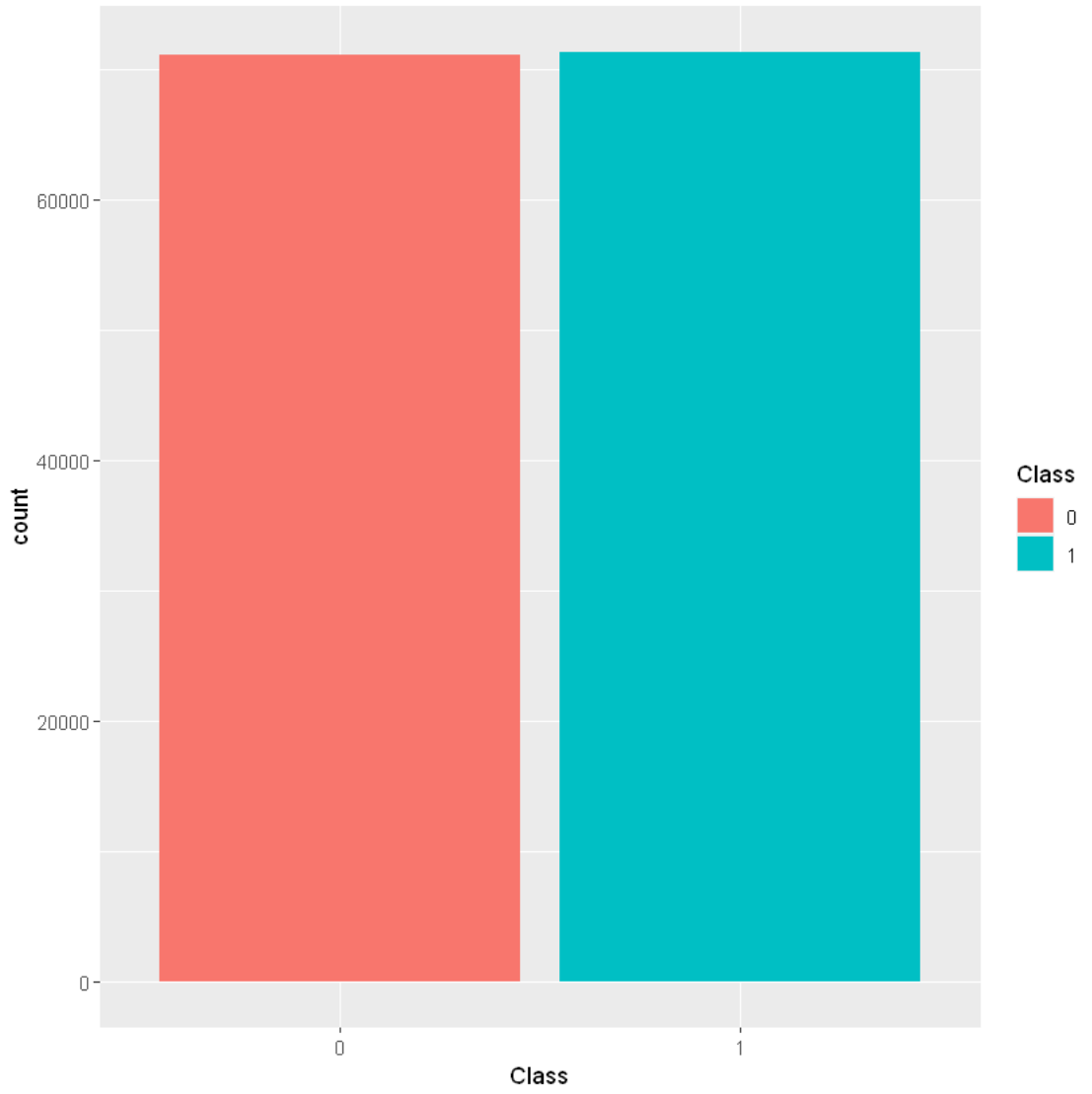
1: 0.172748563062003



In [10]:

```
----------------Synthetic Data Creation-----------------
library(ROSE)
attach(training)
set.seed(4495)
training_Rose <- ROSE(Class~.,data=training,seed = 4495)$data
training_Rose$Class <- as.factor(training_Rose$Class)
ggplot(training_Rose,aes(x = Class)) + geom_bar(aes(fill = Class))
```

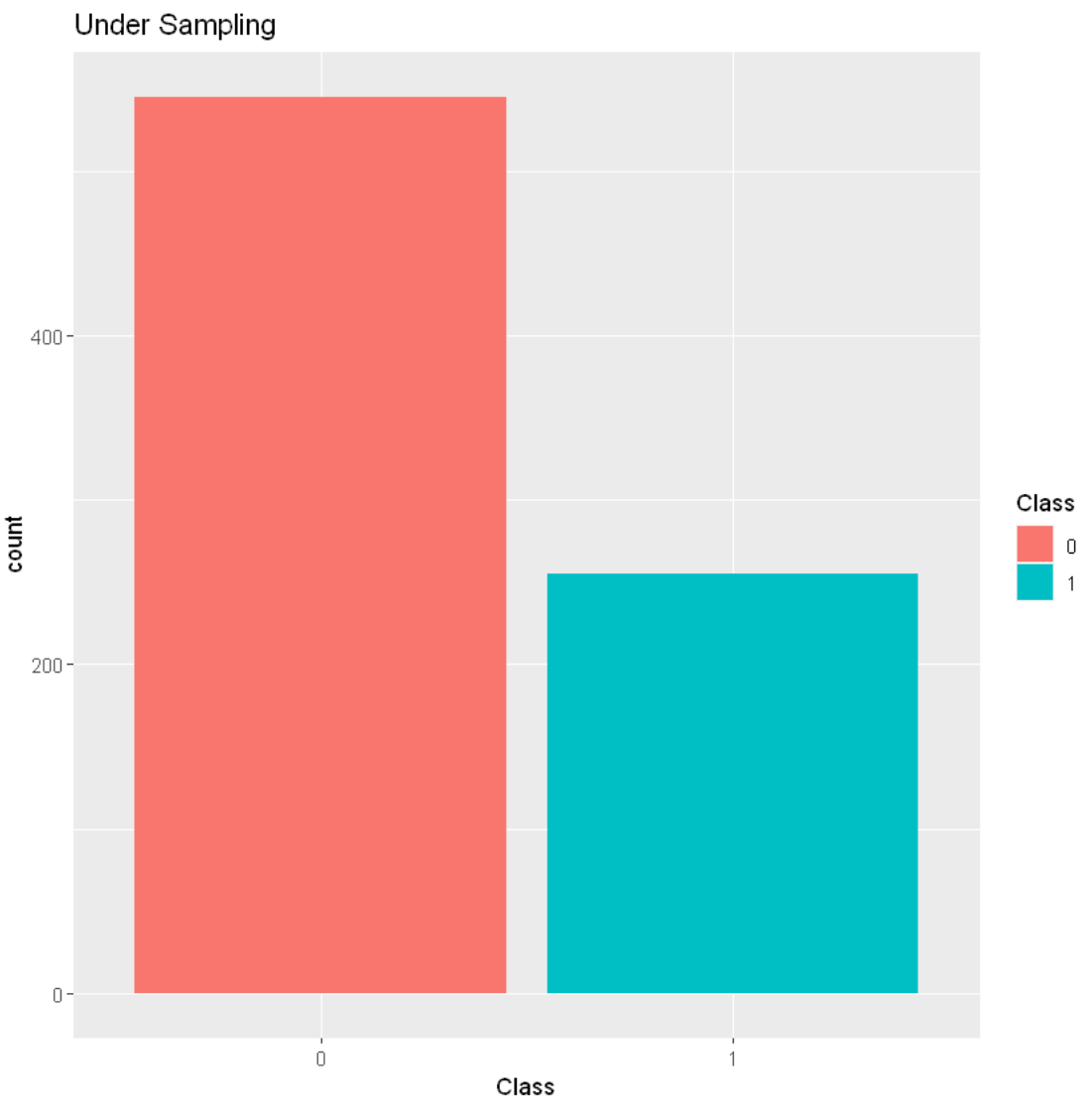
```
Loaded ROSE 0.0-3
```



In [11]:

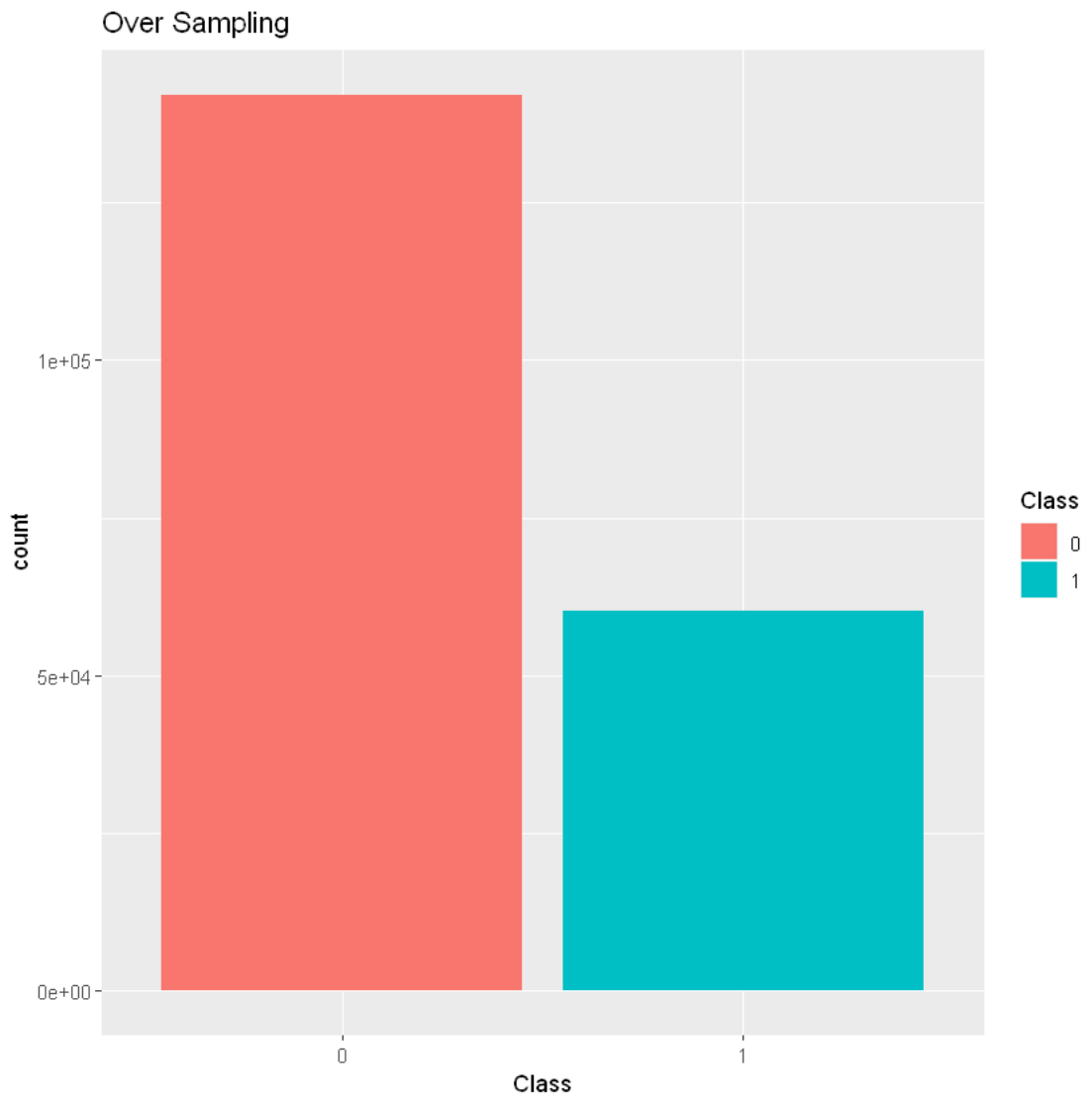
```
----------------------------Undersampling--------------------------
training <- na.omit(training)
attach(training)
training$Class <- as.factor(training$Class)
training_under <- ovun.sample(Class~.,data = training,method = "under",
N=800,seed=4495)$data
ggplot(training_under,aes(x = Class)) + geom_bar(aes(fill = Class))+ggtitle("U
nder Sampling")
```

```
The following objects are masked from training (pos = 3):

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



In [12]:

```
----------------------------Oversampling----------------------
training_over <- ovun.sample(Class~.,data = training,method = "over",
N=202404,seed=4495)$data
ggplot(training_over,aes(x = Class)) + geom_bar(aes(fill = Class))+ggtitle("Ov
er Sampling")
```



In [13]:

```
----------------------Logistic Regression Model-----------------
attach(training)
log <- glm(Class~., data = training,family=binomial)
```

```
The following objects are masked from training (pos = 3):

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


The following objects are masked from training (pos = 4):

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


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

In [14]:

```
log2 <- glm(training_Rose$Class~., data = training_Rose,family=binomial(logit)
)
```

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

In [15]:

```
log3 <- glm(training_under$Class~.,data = training_under,family=binomial(logit
))
```

```
Warning message:
"glm.fit: algorithm did not converge"
Warning message:
"glm.fit: fitted probabilities numerically 0 or 1 occurred"
```

In [16]:

```
log4 <- glm(training_over$Class~.,data = training_over,family=binomial(logit))
```

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

In [17]:

```
----------------------Prediction for Logistc Regression------------------
pred <- predict(log4, testing,type="response")
pred <- round(pred)
```

In [18]:

```
----------------------Finding the Accuracy---------------------------
accuracy <- (1-mean(pred != testing$Class))*100
accuracy
```

99.073053236238

In [19]:

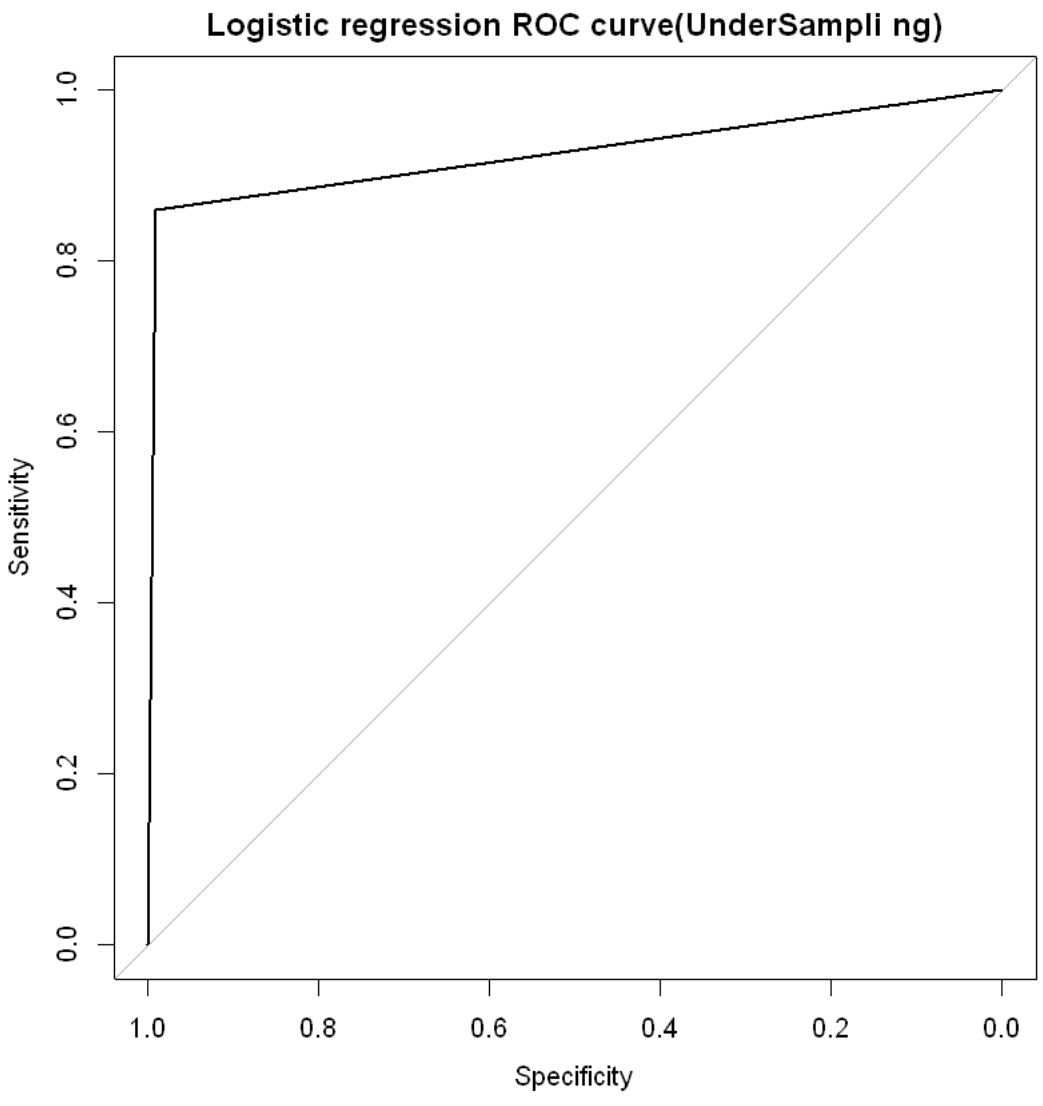
```
------------------------------Confusion Matrix of Logistic Regression--------
----------
confusionMatrix(table(pred,testing$Class))
mat <- as.matrix(confusionMatrix(table(pred,testing$Class)))
```

```
Confusion Matrix and Statistics


pred      0      1
   0 140879     33
   1   1287    204

               Accuracy : 0.9907
                 95% CI : (0.9902, 0.9912)
    No Information Rate : 0.9983
    P-Value [Acc > NIR] : 1

                  Kappa : 0.2339

 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.9909
            Specificity : 0.8608
         Pos Pred Value : 0.9998
         Neg Pred Value : 0.1368
             Prevalence : 0.9983
         Detection Rate : 0.9893
   Detection Prevalence : 0.9895
      Balanced Accuracy : 0.9259

       'Positive' Class : 0
```

In [20]:

```
-----------------Getting AUROC values-----------------
print(roc(testing$Class,pred))
plot(roc(testing$Class,pred),main = "Logistic regression ROC curve(UnderSampli
ng)")
```

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

Setting direction: controls < cases


Call:
roc.default(response = testing$Class, predictor = pred)

Data: pred in 142166 controls (testing$Class 0) < 237 cases (testi
ng$Class 1).
Area under the curve: 0.9259

Setting levels: control = 0, case = 1

Setting direction: controls < cases
```



In [21]:

```
---------------------------Variable Importance--------------
logrgImp <- varImp(log4,scale = FALSE)
```

In [22]:

```
logrgImp
```

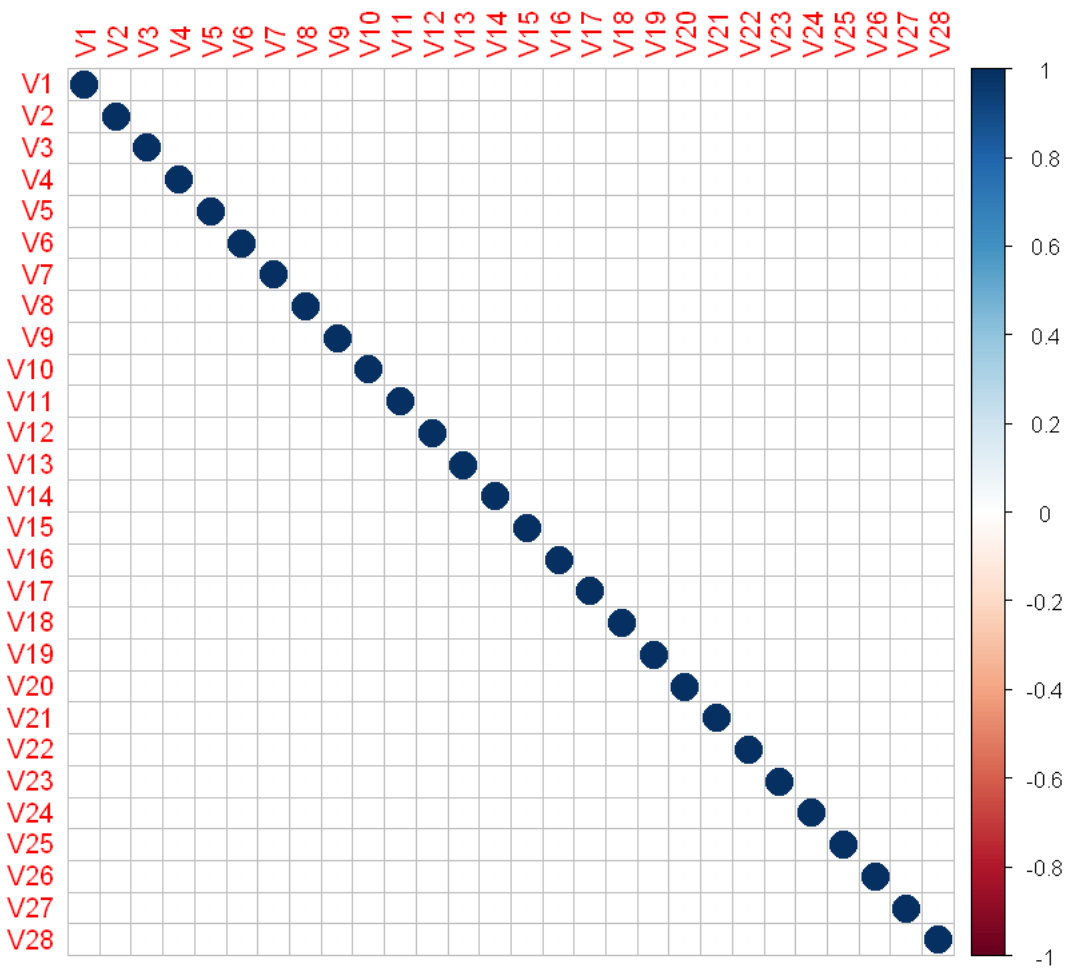
A data.frame: 29 × 1

	Overall
	<dbl>
V1	20.7734055
V2	9.1857692
V3	16.7970361
V4	55.7589186
V5	20.1315767
V6	12.3605734
V7	12.1023691
V8	18.5051715
V9	15.2503371
V10	30.2930195
V11	24.5119709
V12	34.2939813
V13	30.5482485
V14	41.9922865
V15	3.3906752
V16	24.8075535
V17	17.7684252
V18	3.8484204
V19	4.5958506
V20	21.6016873
V21	10.1083491
V22	35.3937924
V23	2.0585542
V24	0.5703025
V25	3.9919142
V26	29.9084167
V27	9.4307256
V28	6.1087944
Amount	17.7063166

In [23]:

```
----------------------------Correlation Matrix-----------------------
library(corrplot)
correlations <- cor(creditcard[,1:28])
corrplot(correlations, method="circle")
```

```
corrplot 0.84 loaded
```



In [24]:

```
------------------------Random Forest Regression-----------------
set.seed(4495)
library(e1071)
```

```
Attaching package: 'e1071'


The following object is masked from 'package:Hmisc':

    impute
```

In [25]:

```
library(randomForest)
```

In [26]:

```
set.seed(4495)
system.time(rand_model <- randomForest(Class~., data = training, ntree = 200))
```

```
   user  system elapsed 
 452.76   14.23  471.91 
```

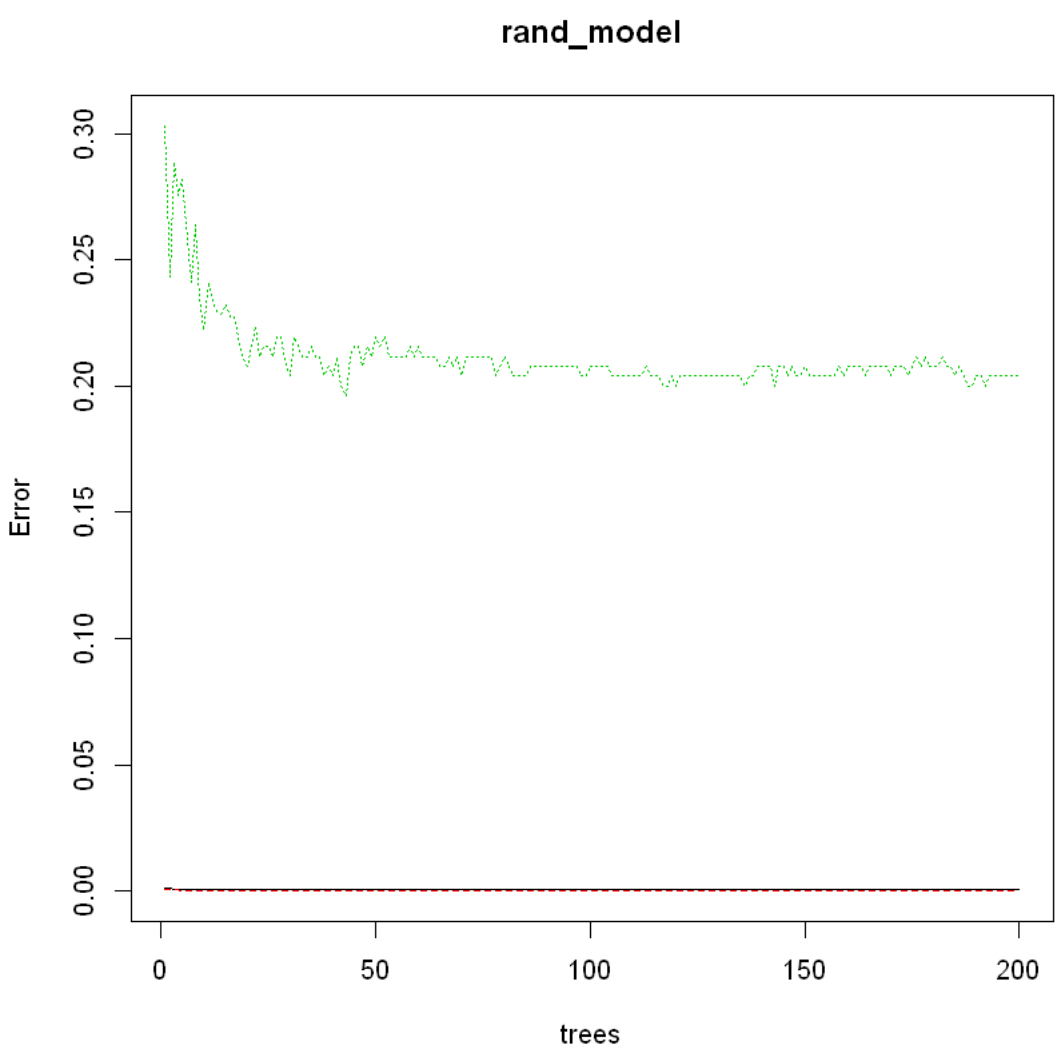
In [27]:

```
------------------Confusion Matrix-------------------
rand_model
```

```
Call:
 randomForest(formula = Class ~ ., data = training, ntree = 200) 
               Type of random forest: classification
                     Number of trees: 200
No. of variables tried at each split: 5

        OOB estimate of  error rate: 0.05%
Confusion matrix:
       0   1  class.error
0 142133  16 0.0001125579
1     52 203 0.2039215686
```

In [28]:

```
plot(rand_model)
```



In [29]:

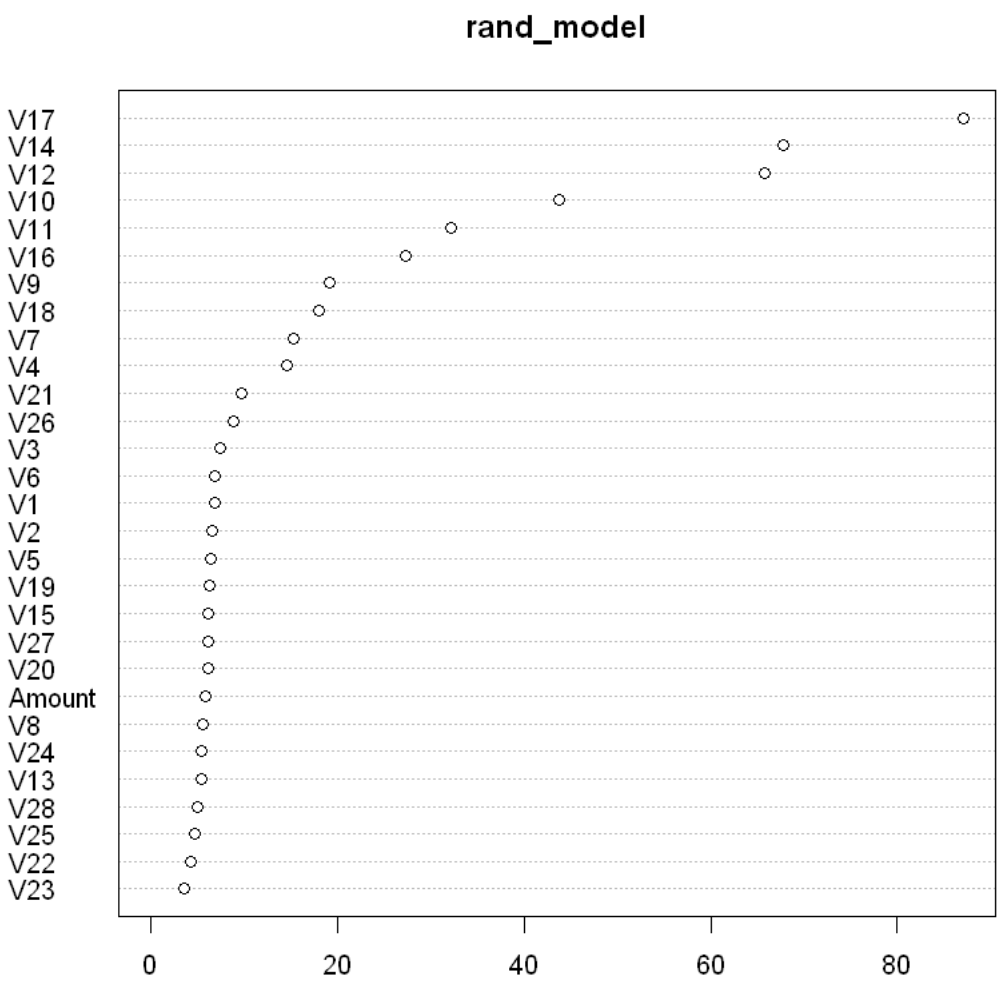
```
------------------------Variable Importance----------------------
importance(rand_model)
```

A matrix: 29 × 1 of type dbl

	MeanDecreaseGini
V1	6.806659
V2	6.614172
V3	7.468481
V4	14.529754
V5	6.334535
V6	6.859376
V7	15.204911
V8	5.526452
V9	19.165079
V10	43.776896
V11	32.120139
V12	65.764325
V13	5.374850
V14	67.786611
V15	6.183966
V16	27.281721
V17	87.046434
V18	17.968638
V19	6.311618
V20	6.123274
V21	9.680153
V22	4.204290
V23	3.475204
V24	5.456588
V25	4.655896
V26	8.804864
V27	6.132533
V28	4.923242
Amount	5.818262

In [30]:

```
------------------------Variable Importance Plot------------------
varImpPlot(rand_model)
```



In [31]:

```
-------------------Predictions---------------
pred1 <- predict(rand_model,training,type = "class")
```

In [32]:

```
table(pred1,training$Class )
```

```
pred1       0       1
    0  142149       0
    1       0     255
```

In [33]:

```
pred2 <- predict(rand_model,testing,type = "class")
```

In [34]:

```
-------------------Getting Accuracy---------------------
accuracy_rf <- mean(pred2 == testing$Class)
```

In [35]:

```
accuracy_rf
```

0.999522481970183

In [36]:

```
training$Class <- as.numeric(training$Class)
```

In [37]:

```
pred_rf <- predict(rand_model,type = "prob")
```

In [38]:

```
-----------------AUROC CURVE genration-------------------------
library(ROCR)
perf <- prediction(pred_rf[,2], training$Class)
```

In [39]:

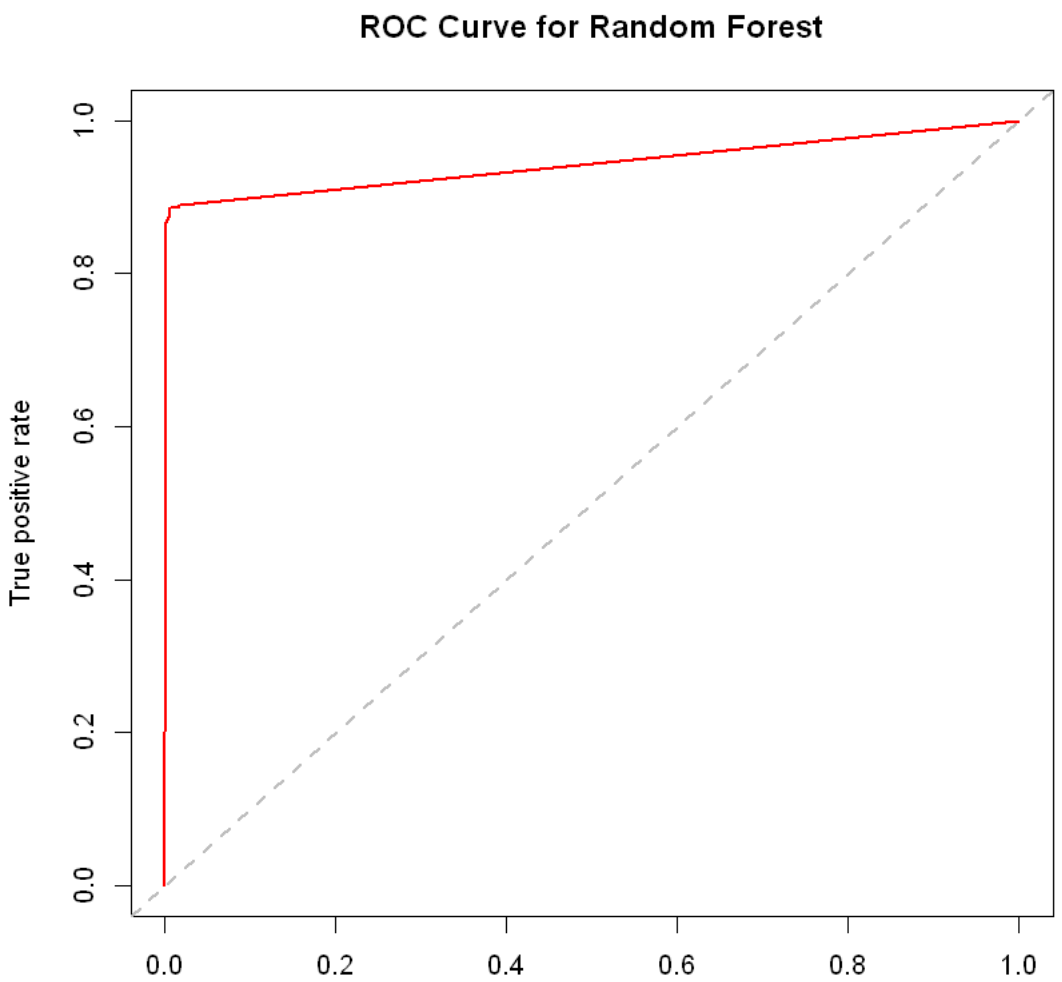
```
auc <- performance(perf, "auc")
```

In [40]:

```
pred3 <- performance(perf, "tpr","fpr")
```

In [41]:

```
----------------------Plotting of AUROC Curve------------------
plot(pred3,main="ROC Curve for Random Forest",col=2,lwd=2)
abline(a=0,b=1,lwd=2,lty=2,col="gray")
```



In [42]:

```
-------------------------Decision Tree Classification-----------------------
library(rpart)
set.seed(4495)
```

In [43]:

```
---------------------------Creating trainning model------------
tree.model <- rpart(Class ~ ., data = training, method = "class", minbucket =
20)
```

In [45]:

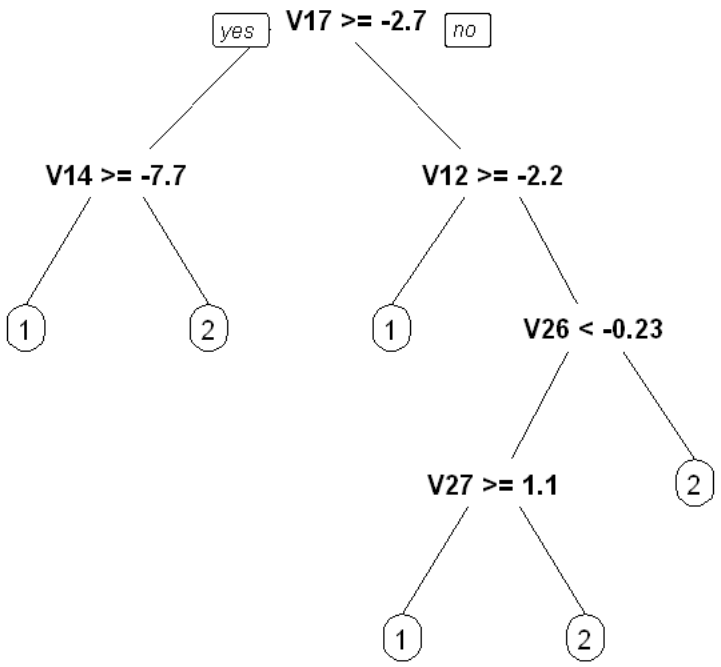
```
tree.model3 <- rpart(training_under$Class ~ ., data = training_under, method =
"class", minbucket = 20)
```

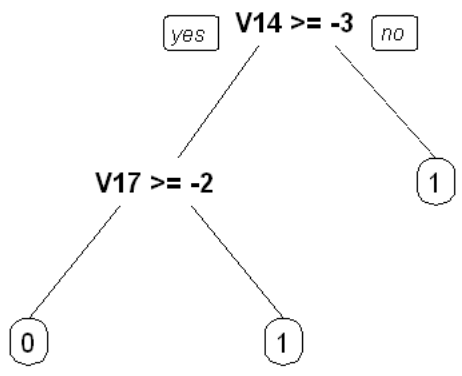
In [47]:

```
tree.model4 <- rpart(training_over$Class~.,data = training_over,method = "clas
s",minbucket = 20)
```

In [48]:

```
--------------------------------Generation of Decision Tree-------------------
-----
prp(tree.model)
prp(tree.model4)
```


yes
V17 >= -2.7
no
V14 >= -7.7
V12 >= -2.2
1
2
1
V26 < -0.23
V27 >= 1.1
2
1
2


yes
V14 >= -3
no
V17 >= -2
1
0
1

In [49]:

```
----------------------------Prediction and Accuracy----------------------
tree.predict <- predict(tree.model4,testing,type = "class")

accuracy_dt <-(1-mean(tree.predict != testing$Class))*100
```

In [50]:

```
accuracy_dt
```

99.2212242719606

In [51]:

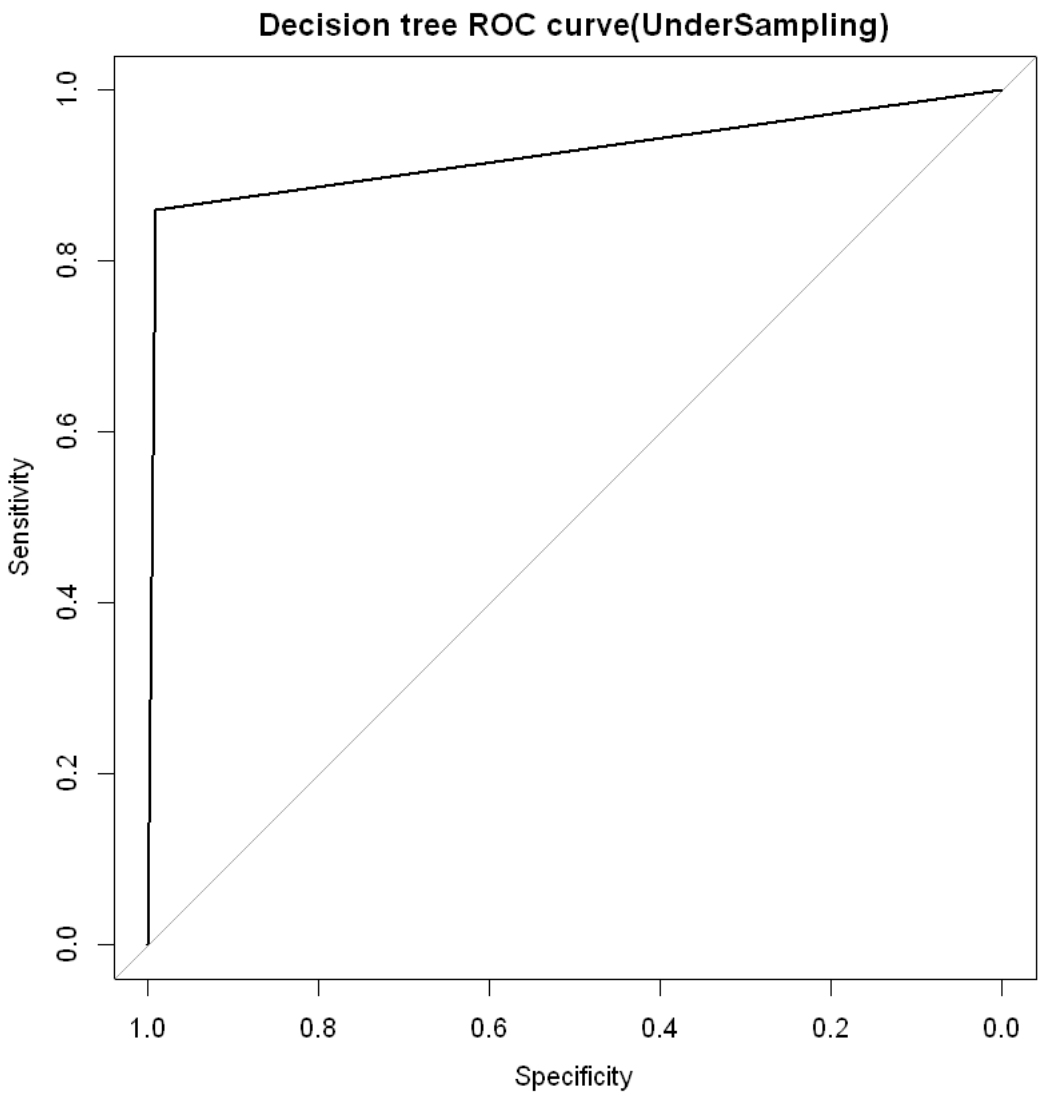
```
-------------------------Generating Confusion Matrix--------------------
confusionMatrix(table(tree.predict,testing$Class))
mat <- as.matrix(confusionMatrix(table(tree.predict,testing$Class)))
```

```
Confusion Matrix and Statistics


tree.predict      0      1
           0 141097     40
           1   1069    197

               Accuracy : 0.9922
                 95% CI : (0.9917, 0.9927)
    No Information Rate : 0.9983
    P-Value [Acc > NIR] : 1

                  Kappa : 0.2601

 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.9925
            Specificity : 0.8312
         Pos Pred Value : 0.9997
         Neg Pred Value : 0.1556
             Prevalence : 0.9983
         Detection Rate : 0.9908
   Detection Prevalence : 0.9911
      Balanced Accuracy : 0.9119

       'Positive' Class : 0
```

In [52]:

```
----------------------------Generating and plotting of AUROC curve-------------
----------
print(roc(testing$Class,pred))
plot(roc(testing$Class,pred),main = "Decision tree ROC curve(UnderSampling)")
```

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

Setting direction: controls < cases


Call:
roc.default(response = testing$Class, predictor = pred)

Data: pred in 142166 controls (testing$Class 0) < 237 cases (testi
ng$Class 1).
Area under the curve: 0.9259

Setting levels: control = 0, case = 1

Setting direction: controls < cases
```



In [53]:

```
----------------------------Variable Importance-----------------------
varImpPlot(tree.model4)
```

A data.frame: 29 × 1

	Overall
	<dbl>
V10	64393.638
V11	55568.386
V12	56527.482
V14	67253.315
V17	60349.214
V20	1689.982
V4	2176.235
V1	0.000
V2	0.000
V3	0.000
V5	0.000
V6	0.000
V7	0.000
V8	0.000
V9	0.000
V13	0.000
V15	0.000
V16	0.000
V18	0.000
V19	0.000
V21	0.000
V22	0.000
V23	0.000
V24	0.000
V25	0.000
V26	0.000
V27	0.000
V28	0.000
Amount	0.000