# IDS 572 – BUSINESS DATA MINING

## ASSIGNMENT -4

Group Members :

**Deepika Kolli** (UIN - 668938591)

**Rini Rose Thomas** (UIN - 653946671)

**Sruthi Reddy Narapareddy** (UIN – 660028172)

**Q1**

Gauging customer satisfaction in order to improve processes that ultimately lead to improved patient experience. Customer service was being billed as the key to sustaining sales, client loyalty and profits. The NPS score is used as an indicator to stratify and target customers accordingly. Detractors are likely to spread negative sentiments. This converts into a multiclassification analytics problem. To identify potential Detractors, passive customers and Promoters. Using analytics and machine learning on the data collected where NPS being the effective tool will help to understand the significant factors concerning detractors and to analyze the improvement opportunities within the departments.

**Q2**

There are no Missing values in the data.

**Q3**

Sensitivity of each class can be calculated from its TP/(TP+FN) and
Specificity of each class can be calculated from its TN/(TN+FP)

For multi-class where there are three classes: C1, C2, and C3
"TP of C1" is all C1 instances that are classified as C1.
"TN of C1" is all non-C1 instances that are not classified as C1.
"FP of C1" is all non-C1 instances that are classified as C1.
"FN of C1" is all C1 instances that are not classified as C1.
Similarly, for other classes.

**Confusion Matrix**

| | | **Actual** | | |
|---|---|---|---|---|
| | | C1 | C2 | C3 |
| | C1 | a | B | c |
| | C2 | d | E | f |
| **Predicted** | C3 | g | H | i |

**Sensitivity**

| | | |
|---|---|---|
| For Class 1 Sensitivity | = | a/a+d+g |
| For Class 2 Sensitivity | = | e/e+b+h |
| For Class 3 Sensitivity | = | i/c+f+i |

**Specificity**

| | | |
|---|---|---|
| For Class 1 Specificity | = | e+i/e+i+b+c |
| For Class 2 Specificity | = | a+i/a+i+d+f |
| For Class 3 Specificity | = | a+e/a+e+g+h |

**Q4. what is quasi-complete separation?**

Quasi-complete separation occurs when the data are not completely separated and a vector of pseudo estimates correctly allocates all but a nonempty set of observations to their response groups. If a quasi-complete separation exists in the sample points, then the maximum likelihood estimate does not exist. Some remedies can be attempted to relieve the separation of the sample data, including increasing the sample size, categorizing quantitative variables and reducing the number of explanatory variables.

Separation occurs because:

1. The independent variable is a variant of the Target variable and there is high correlation between the Target variable and the independent variable.

2. If the data size is small and the distribution is extreme. In this case, obtaining more data might resolve the problem.

In the presence of variables leading to quasi-complete separation, the maximum likelihood estimate for that variable does not exist in a Logistic Regression.

Variables that are leading to Quasi- complete separation in the dataset are "EM_DOCTOR", "STATEZONE", "Country", "State", "BedCategory" , "MaritalStatus", "OVS_OVERALLSTAFFATTITUDE", "EM_DOCTOR", "NS_NURSESATTITUDE","DOC_TREATMENTEFFECTIVENESS","AE_PATIENTSTATUSINFO","NS_NURSEPATIENCE","DOC_TREATMENTEXPLAINATION"

**Q5. Orthogonal polynomial coding**

Orthogonal polynomial coding is a form of trend analysis in that it is looking for the linear, quadratic and cubic trends in the categorical variable. This type of coding system should be used only with an ordinal variable in which the levels are equally spaced.

If we have 2 variables, that are highly correlated, regressing them will affect our estimates. Correlation from higher order polynomials leads to higher standard errors, thus affecting our t-stats. So, we choose to orthogonalize our polynomials before regressing them.

**Q 6**.

We can convert it to binary by considering only two classes say detractors and promoters. The promoters class includes instances of both promoters and passive. Thereby understanding how independent variables effect or result in the detractors.

**R Code -**

```
#install.packages("xlsx")
library(xlsx)
library("openxlsx")

##
## Attaching package: 'openxlsx'

## The following objects are masked from 'package:xlsx':
##
##     createWorkbook, loadWorkbook, read.xlsx, saveWorkbook,
##     write.xlsx

train_binarydata= read.xlsx("E:/MS_Studies/572/assignments/assignment4/IMB651
-XLS-ENG.xlsx",sheet = 2)
test_binarydata=read.xlsx("E:/MS_Studies/572/assignments/assignment4/IMB651-X
LS-ENG.xlsx",sheet = 3)

train_multidata=read.xlsx("E:/MS_Studies/572/assignments/assignment4/IMB651-X
LS-ENG.xlsx",sheet = 4)
test_multidata=read.xlsx("E:/MS_Studies/572/assignments/assignment4/IMB651-XL
S-ENG.xlsx",sheet = 5)
```

##1.2

```
i=0 x=c() for(i in
1:ncol(train_binarydata))
x[i]=sum(is.null(train_binarydata))

##no missing data in the provided training and test data set
```

##1.4 Quasi complete separation

```
#install.packages("brglm2")
library(brglm2)
unnecessary_var=which(names(train_binarydata) %in%
c("State","Country","Admis sionDate","DischargeDate"))
train_binary_final_Data=train_binarydata[,-unnecessary_var]
new_train_binary=train_binary_final_Data[,-47]
```

```
## DOC_TREATMENTEFFECTIVENESS                        0
## NS_CALLBELLRESPONSE                               0
## NS_NURSESATTITUDE                                 0
## NS_NURSEPROACTIVENESS                             0
## NS_NURSEPATIENCE                                  0
## OVS_OVERALLSTAFFATTITUDE                          0
## OVS_OVERALLSTAFFPROMPTNESS                        0
## OVS_SECURITYATTITUDE                              0
## DP_DISCHARGETIME                                  0
## DP_DISCHARGEQUERIES                               0
## DP_DISCHARGEPROCESS                               0
## LengthofStay                                      0
```

```
#x=match(final_var,names(new_train_binary))
final_var=c("HospitalNo2","AgeYrs","Sex","Department","Estimatedcost","InsPay
orcategory","CE_ACCESSIBILITY","CE_CSAT"
,"CE_VALUEFORMONEY","EM_IMMEDIATEATTENTION","EM_NURSING"
,"EM_DOCTOR"
,"EM_OVERALL"
,"AD_TIME","AD_TARRIFFPACKAGESEXPLAINATION"
,"AD_STAFFATTITUDE"
,"INR_ROOMCLEANLINESS"
,"INR_ROOMPEACE"
,"INR_ROOMEQUIPMENT"
,"INR_ROOMAMBIENCE"
,"FNB_FOODQUALITY"
,"FNB_FOODDELIVERYTIME"
,"FNB_DIETICIAN"
,"FNB_STAFFATTITUDE"
,"AE_ATTENDEECARE"
,"AE_PATIENTSTATUSINFO"
,"AE_ATTENDEEFOOD"
,"DOC_TREATMENTEXPLAINATION"
,"DOC_ATTITUDE"
,"DOC_VISITS"
,"DOC_TREATMENTEFFECTIVENESS"
,"NS_CALLBELLRESPONSE","NS_NURSESATTITUDE"  ,"NS_NURSEPROACTIVENESS",   "NS_N
URSEPATIENCE"
,"OVS_OVERALLSTAFFATTITUDE",    "OVS_OVERALLSTAFFPROMPTNESS"    ,"OVS_SECURIT
YATTITUDE",    "DP_DISCHARGETIME", "DP_DISCHARGEQUERIES"   ,"DP_DISCHARGEPROC
ESS","NPS_Status"
)
train_binary_final_Data= new_train_binary[,final_var]
test_binary_final_data=test_binarydata[,final_var]
train_multi_final_data=train_multidata[,final_var]
test_multi_final_data=test_multidata[,final_var]
```

```
x= which(names(train_binarydata) %in% final_var)
```

```
removed_variables=names(train_binarydata[,-x]) removed_variables

##  [1] "SN"            "MaritalStatus" "BedCategory"   "State"         ##
[5] "Country"       "STATEZONE"     "AdmissionDate" "DischargeDate" ##
[9] "LengthofStay"  "CE_NPS"
```

##1.6 converting attributes to ordinal variables

```
train_binary_final_Data1=train_binary_final_Data i=0
for (i in 1:(ncol(train_binary_final_Data)-1))
{

if(class(train_binary_final_Data[[i]])=="factor" | (is.numeric(train_binary_f
inal_Data[[i]])))
{

   train_binary_final_Data1[[i]]=as.factor(train_binary_final_Data[[i]])
if (nlevels(train_binary_final_Data1[[i]])<5)
     train_binary_final_Data1[[i]]=as.ordered(train_binary_final_Data1[[i]])
else
      train_binary_final_Data1[[i]]=as.numeric(train_binary_final_Data[[i]])

}



}

##doing the same for test data
test_binary_final_data1=test_binary_final_data i=0
for (i in 1:(ncol(test_binary_final_data1)-1))
{

if(class(test_binary_final_data[[i]])=="factor" | (is.numeric(test_binary_fin
al_data[[i]])))
{

   test_binary_final_data1[[i]]=as.factor(test_binary_final_data[[i]])
if (nlevels(test_binary_final_data1[[i]])<5)
     test_binary_final_data1[[i]]=as.ordered(test_binary_final_data1[[i]])
else
      test_binary_final_data1[[i]]=as.numeric(test_binary_final_data[[i]])

}



}
```

```
##converting to ordinal for train and test of multi class
train_multi_final_data1=train_multi_final_data i=0
for (i in 1:(ncol(train_multi_final_data1)-1))
{

if(class(train_multi_final_data[[i]])=="factor" | (is.numeric(train_multi_fin
al_data[[i]])))
{

   train_multi_final_data1[[i]]=as.factor(train_multi_final_data[[i]])
if (nlevels(train_multi_final_data1[[i]])<5)
     train_multi_final_data1[[i]]=as.ordered(train_multi_final_data1[[i]])
else
      train_multi_final_data1[[i]]=as.numeric(train_multi_final_data[[i]])

}
}
train_multi_final_data1[[42]]=as.factor(train_multi_final_data[[42]])


test_multi_final_data1=test_multi_final_data i=0
for (i in 1:(ncol(test_multi_final_data1)-1))
{
  if(class(test_multi_final_data[[i]])=="factor" |
(is.numeric(test_multi_final _data[[i]])))
{

   test_multi_final_data1[[i]]=as.factor(test_multi_final_data[[i]])
if (nlevels(test_multi_final_data1[[i]])<5)
     test_multi_final_data1[[i]]=as.ordered(test_multi_final_data1[[i]])
else
      test_multi_final_data1[[i]]=as.numeric(test_multi_final_data[[i]])

}


}




test_multi_final_data1[[42]]=as.factor(test_multi_final_data[[42]])
```

##1.6 logistic regression on binary after conversion to ordinal variables

```r
library(MASS)
model <- glm(train_binary_final_Data1$NPS_Status~., data = train_binary_final
_Data1, family = binomial("logit"))

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

step_model=stepAIC(model,trace = FALSE,direction="both")

summary(step_model) step_model$anova



## Stepwise Model Path
## Analysis of Deviance Table
##
## Initial Model:
## train_binary_final_Data1$NPS_Status ~ HospitalNo2 + AgeYrs +
##     Sex + Department + Estimatedcost + InsPayorcategory + CE_ACCESSIBILITY
+
##     CE_CSAT + CE_VALUEFORMONEY + EM_IMMEDIATEATTENTION + EM_NURSING +
##     EM_DOCTOR + EM_OVERALL + AD_TIME + AD_TARRIFFPACKAGESEXPLAINATION +
##     AD_STAFFATTITUDE + INR_ROOMCLEANLINESS + INR_ROOMPEACE +
##     INR_ROOMEQUIPMENT + INR_ROOMAMBIENCE + FNB_FOODQUALITY +
##     FNB_FOODDELIVERYTIME + FNB_DIETICIAN + FNB_STAFFATTITUDE +
##     AE_ATTENDEECARE + AE_PATIENTSTATUSINFO + AE_ATTENDEEFOOD +
##     DOC_TREATMENTEXPLAINATION + DOC_ATTITUDE + DOC_VISITS + DOC_TREATMENTE
FFECTIVENESS +
##     NS_CALLBELLRESPONSE + NS_NURSESATTITUDE + NS_NURSEPROACTIVENESS +
##     NS_NURSEPATIENCE + OVS_OVERALLSTAFFATTITUDE + OVS_OVERALLSTAFFPROMPTNE
SS +
##     OVS_SECURITYATTITUDE + DP_DISCHARGETIME + DP_DISCHARGEQUERIES +  ##
DP_DISCHARGEPROCESS
##
## Final Model:
## train_binary_final_Data1$NPS_Status ~ HospitalNo2 + Department +
##     Estimatedcost + CE_ACCESSIBILITY + CE_CSAT + CE_VALUEFORMONEY +
##     EM_NURSING + EM_DOCTOR + AD_TARRIFFPACKAGESEXPLAINATION +
##     AD_STAFFATTITUDE + INR_ROOMCLEANLINESS + INR_ROOMAMBIENCE +
##     FNB_FOODDELIVERYTIME + AE_PATIENTSTATUSINFO + AE_ATTENDEEFOOD +
##     DOC_TREATMENTEXPLAINATION + DOC_VISITS + NS_CALLBELLRESPONSE +
##     NS_NURSEPROACTIVENESS + OVS_OVERALLSTAFFPROMPTNESS + DP_DISCHARGEQUERI
ES
##
##
##                             Step Df  Deviance Resid. Df Resid. Dev      AIC
## 1                                                 4870   5006.132 5244.132
## 2         - DP_DISCHARGEPROCESS  3 0.7761258      4873   5006.908 5238.908
```

```
## 3               - INR_ROOMPEACE  3 1.0079877      4876   5007.916 5233.916
## 4  - DOC_TREATMENTEFFECTIVENESS  3 1.1457654      4879   5009.062 5229.062
## 5                     - AD_TIME  3 1.4440053      4882   5010.506 5224.506
## 6             - AE_ATTENDEECARE  3 1.6052474      4885   5012.111 5220.111
## 7            - NS_NURSEPATIENCE  3 1.6509205      4888   5013.762 5215.762
## 8               - FNB_DIETICIAN  3 1.8085467      4891   5015.571 5211.571
## 9            - DP_DISCHARGETIME  3 2.0777223      4894   5017.649 5207.649
## 10                - EM_OVERALL  3 2.3096636      4897   5019.958 5203.958
## 11               - DOC_ATTITUDE  3 2.1817163      4900   5022.140 5200.140
## 12          - FNB_STAFFATTITUDE  3 2.6332058      4903   5024.773 5196.773
## 13          - INR_ROOMEQUIPMENT  3 2.8420964      4906   5027.615 5193.615
## 14   - OVS_OVERALLSTAFFATTITUDE  3 3.2401183      4909   5030.855 5190.855
## 15      - EM_IMMEDIATEATTENTION  3 3.4636049      4912   5034.319 5188.319
## 16                    - AgeYrs  1 0.3008170      4913   5034.620 5186.620
## 17          - NS_NURSESATTITUDE  3 4.5508643      4916   5039.171 5185.171
## 18                        - Sex  1 0.9052000      4917   5040.076 5184.076
## 19            - FNB_FOODQUALITY  3 5.0262672      4920   5045.102 5183.102
## 20       - OVS_SECURITYATTITUDE  3 4.9992976      4923   5050.102 5182.102
## 21           - InsPayorcategory  4 7.9691306      4927   5058.071 5182.071
```

```
test_binary_final_data1 <- subset(test_binary_final_data1,test_binary_final_d
ata1$NS_NURSEPROACTIVENESS!="1")

log_pred=predict(step_model,test_binary_final_data1,type="response") log_pred=
ifelse(log_pred>0.5,"Promotor","Detractor")
confusion_matrix=table(log_pred,test_binary_final_data1$NPS_Status)
confusion_matrix
```

```
##
## log_pred    Detractor Promotor
##   Detractor        80       32 ##
Promotor         78      170
```

```
accuracy=sum(diag(confusion_matrix))/sum(confusion_matrix)

accuracy
```

```
## [1] 0.6944444 cat("accuracy for step wise
model --",accuracy) ## accuracy for step wise
model -- 0.6944444
```

##NA check

```
x=c() i=0
for (i in 1: ncol(train_binary_final_Data1))
x[i]=sum(is.na(train_binary_final_Data1[[i]])) x
```

```
##  [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0## [36] 0 0 0 0 0 0 0
```

**Q 7.**

Random forest and Adaboost models both are giving better accuracy with binary classification than the multi class problem.

**Random Forest -**

Accuracy for binary model = 0.7433333

Accuracy for multiclass model=0.6805556

**Adaboost -**

Accuracy for binary model = 0.76

Accuracy for multiclass model = 0.67

**R Code :**

## Randomforest for Binary and Multiclass Problem R Markdown

##1.7 random forest for binary class vs multiclass ( conisdering the variables from step wise)

##formula = train_binary_final_Data1$NPS_Status ~ HospitalNo2 + ## Department + Estimatedcost + CE_ACCESSIBILITY + CE_CSAT + ## CE_VALUEFORMONEY + EM_NURSING
+ EM_DOCTOR + AD_TARRIFFPACKAGESEXPLAINATION + ## AD_STAFFATTITUDE + INR_ROOMCLEANLINESS + INR_ROOMAMBIENCE + ## FNB_FOODDELIVERYTIME + AE_PATIENTSTATUSINFO + AE_ATTENDEEFOOD + ## DOC_TREATMENTEXPLAINATION + DOC_VISITS + NS_CALLBELLRESPONSE + ## NS_NURSEPROACTIVENESS + OVS_OVERALLSTAFFPROMPTNESS + DP_DISCHARGEQUERIES,

```
library(randomForest)

## randomForest 4.6-14

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

p=0
for (p in 1:ncol(train_binary_final_Data1))
{
  if(is.character(train_binary_final_Data1[[p]]))
   train_binary_final_Data1[[p]]=as.factor(train_binary_final_Data1[[p]])
}

l=0
for (l in 1:ncol(test_binary_final_data1))
{
  if(is.character(test_binary_final_data1[[l]]))
   test_binary_final_data1[[l]]=as.factor(test_binary_final_data1[[l]])
}
accuracy=c() k
<- 10 nmethod
<- 1
folds <- cut(seq(1,nrow(train_binary_final_Data1)),breaks=k,labels=FALSE)
models.err <- matrix(-1,k,nmethod, dimnames=list(paste0("Fold", 1:k), c("rf")
)) i=0
for(i in 1:k)
{
  trainIndexes <- which(folds==i, arr.ind=TRUE)
  Validation <- train_binary_final_Data1[trainIndexes, ]
Train <- train_binary_final_Data1[-trainIndexes, ]
mtry_list= c(1:8)   pr.err <- c()
  s=0
  for(mt in mtry_list){
    rf <- randomForest(formula = Train$NPS_Status ~ HospitalNo2 +
```

```
    Department + Estimatedcost + CE_ACCESSIBILITY + CE_CSAT +
    CE_VALUEFORMONEY + EM_NURSING + EM_DOCTOR + AD_TARRIFFPACKAGESEXPLAINATIO
N +
    AD_STAFFATTITUDE + INR_ROOMCLEANLINESS + INR_ROOMAMBIENCE +
    FNB_FOODDELIVERYTIME + AE_PATIENTSTATUSINFO + AE_ATTENDEEFOOD +
    DOC_TREATMENTEXPLAINATION + DOC_VISITS + NS_CALLBELLRESPONSE +
    NS_NURSEPROACTIVENESS + OVS_OVERALLSTAFFPROMPTNESS +
DP_DISCHARGEQUERIES, data = Train, ntree = 100, mtry = mt,replace=T)
predicted <- predict(rf, newdata = Validation, type = "class")       pr.err
<- c(pr.err,mean(Validation$NPS_Status != predicted))   }
      bestmtry <-
which.min(pr.err)

  #test_binary_final_data1is the test data given in the case study

 rf <- randomForest(formula = Train$NPS_Status ~ HospitalNo2 +
    Department + Estimatedcost + CE_ACCESSIBILITY + CE_CSAT +
    CE_VALUEFORMONEY + EM_NURSING + EM_DOCTOR + AD_TARRIFFPACKAGESEXPLAINATIO
N +
    AD_STAFFATTITUDE + INR_ROOMCLEANLINESS + INR_ROOMAMBIENCE +
    FNB_FOODDELIVERYTIME + AE_PATIENTSTATUSINFO + AE_ATTENDEEFOOD +
    DOC_TREATMENTEXPLAINATION + DOC_VISITS + NS_CALLBELLRESPONSE +
    NS_NURSEPROACTIVENESS + OVS_OVERALLSTAFFPROMPTNESS + DP_DISCHARGEQUERIES,
data = Train, ntree = 200, mtry = bestmtry)
  rf.pred <- predict(rf, newdata = test_binary_final_data1, type = "class")
rf.conf=table(rf.pred,test_binary_final_data1$NPS_Status)
accuracy[i]=sum(diag(rf.conf))/sum(rf.conf)
 models.err[i] <- mean(test_binary_final_data1$NPS_Status != rf.pred)
}

accuracy_rf=1-mean(models.err) accuracy_rf ## [1] 0.7486111 cat("accuracy of
Random forest for binary classification :", mean(accuracy))

## accuracy of Random forest for binary classification : 0.7486111
```

##Random forest for multi classification

```
p=0
for (p in 1:ncol(train_multi_final_data1))
{
  if(is.character(train_multi_final_data1[[p]]))
   train_multi_final_data1[[p]]=as.factor(train_multi_final_data1[[p]])
}
```

```r
for (l in 1:ncol(test_multi_final_data1))
{
  if(is.character(test_multi_final_data1[[l]]))
   test_multi_final_data1[[l]]=as.factor(test_multi_final_data1[[l]])
}
accuracy_multi=c() k <- 10 nmethod <- 1 folds <-
cut(seq(1,nrow(train_multi_final_data1)),breaks=k,labels=FALSE)  models.err
<- matrix(-1,k,nmethod, dimnames=list(paste0("Fold", 1:k), c("rf")
)) i=0
for(i in 1:k)
{
  trainIndexes <- which(folds==i, arr.ind=TRUE)
  Validation <- train_multi_final_data1[trainIndexes,
]   Train <- train_multi_final_data1[-trainIndexes, ]
mtry_list= c(1:8)   pr.err <- c()
  s=0
  for(mt in mtry_list){
    rf <- randomForest(formula = Train$NPS_Status ~
    Department + Estimatedcost + CE_ACCESSIBILITY + CE_CSAT +
    CE_VALUEFORMONEY + EM_NURSING + EM_DOCTOR + AD_TARRIFFPACKAGESEXPLAINATIO
N +
    AD_STAFFATTITUDE + INR_ROOMCLEANLINESS + INR_ROOMAMBIENCE +
    FNB_FOODDELIVERYTIME + AE_PATIENTSTATUSINFO + AE_ATTENDEEFOOD +
    DOC_TREATMENTEXPLAINATION + DOC_VISITS + NS_CALLBELLRESPONSE +
    NS_NURSEPROACTIVENESS + OVS_OVERALLSTAFFPROMPTNESS +
DP_DISCHARGEQUERIES, data = Train, ntree = 200, mtry = mt,replace=T)
predicted <- predict(rf, newdata = Validation, type = "class")       pr.err
<- c(pr.err,mean(Validation$NPS_Status != predicted))   }
      bestmtry <-
which.min(pr.err)

    #test_binary_final_data1is the test data given in the case study

 rf <- randomForest(formula = Train$NPS_Status ~
    Department + Estimatedcost + CE_ACCESSIBILITY + CE_CSAT +
    CE_VALUEFORMONEY + EM_NURSING + EM_DOCTOR + AD_TARRIFFPACKAGESEXPLAINATIO
N +
    AD_STAFFATTITUDE + INR_ROOMCLEANLINESS + INR_ROOMAMBIENCE +
    FNB_FOODDELIVERYTIME + AE_PATIENTSTATUSINFO + AE_ATTENDEEFOOD +
    DOC_TREATMENTEXPLAINATION + DOC_VISITS + NS_CALLBELLRESPONSE +
    NS_NURSEPROACTIVENESS + OVS_OVERALLSTAFFPROMPTNESS + DP_DISCHARGEQUERIES,
data = Train, ntree = 100, mtry = bestmtry)
  rf.pred <- predict(rf, newdata = test_multi_final_data1, type = "class")
rf.conf=table(rf.pred,test_multi_final_data1$NPS_Status)
accuracy_multi[i]=sum(diag(rf.conf))/sum(rf.conf)
```

```
 models.err[i] <- mean(test_multi_final_data1$NPS_Status != rf.pred)
}

accuracy_rf_multi=1-mean(models.err)
accuracy_rf_multi ## [1] 0.6832418
cat("accuracy of Random forest for multi classification :", mean(accuracy_mul
ti))
## accuracy of Random forest for multi classification : 0.6832418
```

##1.8 effect of balancing method- undersampling -RF

```
library(caret)
```

```
## Loading required package: lattice ##
```

```
Loading required package: ggplot2
```

```
##
## Attaching package: 'ggplot2'
```

```
## The following object is masked from 'package:randomForest': ##
##     margin
```

```
set.seed(123)
down_train <- downSample(x = train_binary_final_Data1[, -ncol(train_binary_fi
nal_Data1)],                          y =
train_binary_final_Data1$NPS_Status)

table(train_binary_final_Data1$NPS_Status)
```

```
##
## Detractor  Promotor  ##
```

```
1849      3140
```

```
table(down_train$Class)
```

```
##
## Detractor  Promotor
##      1849      1849
```

```
##down_train is the undersampling data


accuracy=c() k
<- 10 nmethod
<- 1
folds <- cut(seq(1,nrow(down_train)),breaks=k,labels=FALSE)
models.err <- matrix(-1,k,nmethod, dimnames=list(paste0("Fold", 1:k), c("rf")
```

```
)) i=0
for(i in 1:k)
{
  trainIndexes <- which(folds==i, arr.ind=TRUE)
  Validation <- down_train[trainIndexes,
]   Train <- down_train[-trainIndexes, ]
mtry_list= c(1:8)  pr.err <- c()
  s=0
  for(mt in mtry_list){    rf <- randomForest(formula =
down_train$Class ~ HospitalNo2 +
    Department + Estimatedcost + CE_ACCESSIBILITY + CE_CSAT +
    CE_VALUEFORMONEY + EM_NURSING + EM_DOCTOR + AD_TARRIFFPACKAGESEXPLAINATIO
N +
    AD_STAFFATTITUDE + INR_ROOMCLEANLINESS + INR_ROOMAMBIENCE +
    FNB_FOODDELIVERYTIME + AE_PATIENTSTATUSINFO + AE_ATTENDEEFOOD +
    DOC_TREATMENTEXPLAINATION + DOC_VISITS + NS_CALLBELLRESPONSE +
    NS_NURSEPROACTIVENESS + OVS_OVERALLSTAFFPROMPTNESS +
DP_DISCHARGEQUERIES, data = down_train, ntree = 100, mtry = mt,replace=T)
predicted <- predict(rf, newdata = Validation, type = "class")       pr.err
<- c(pr.err,mean(Validation$Class != predicted))  }
      bestmtry <-
which.min(pr.err)
  #test_binary_final_data1is the test data given in the case study

 rf <- randomForest(formula = down_train$Class ~ HospitalNo2 +
    Department + Estimatedcost + CE_ACCESSIBILITY + CE_CSAT +
    CE_VALUEFORMONEY + EM_NURSING + EM_DOCTOR + AD_TARRIFFPACKAGESEXPLAINATIO
N +
    AD_STAFFATTITUDE + INR_ROOMCLEANLINESS + INR_ROOMAMBIENCE +
    FNB_FOODDELIVERYTIME + AE_PATIENTSTATUSINFO + AE_ATTENDEEFOOD +
    DOC_TREATMENTEXPLAINATION + DOC_VISITS + NS_CALLBELLRESPONSE +
    NS_NURSEPROACTIVENESS + OVS_OVERALLSTAFFPROMPTNESS + DP_DISCHARGEQUERIES,
data = down_train, ntree = 200, mtry = bestmtry)
  rf.pred <- predict(rf, newdata = test_binary_final_data1, type = "class")
rf.conf=table(rf.pred,test_binary_final_data1$NPS_Status)
  #accuracy[i]=sum(diag(rf.conf))/sum(rf.conf)
 models.err[i] <- mean(test_binary_final_data1$NPS_Status != rf.pred)
}

accuracy_rf_down=1-
mean(models.err) accuracy_rf_down ##
[1] 0.7108333
cat("accuracy of Random forest for binary classification with undersampled da
ta :", mean(accuracy_rf_down))
```

```
## accuracy of Random forest for binary classification with undersampled
data : 0.7108333
```

##on undersampling, the accuracy has reduced from 75 to 71.75 % for binary classification RF ensemble method

##effect of balancing method- oversampling -RF

```
set.seed(234)
over_train <- upSample(x = train_binary_final_Data1[, -ncol(train_binary_fina
l_Data1)],                         y = train_binary_final_Data1$NPS_Status)

table(train_binary_final_Data1$NPS_Status)

##
## Detractor  Promotor  ##
1849      3140
table(over_train$Class)

##
## Detractor  Promotor
##      3140      3140

##over_train is the undersampling data


accuracy=c() k
<- 10 nmethod
<- 1
folds <- cut(seq(1,nrow(over_train)),breaks=k,labels=FALSE)
models.err <- matrix(-1,k,nmethod, dimnames=list(paste0("Fold", 1:k), c("rf")
)) i=0
for(i in 1:k)
{
  trainIndexes <- which(folds==i, arr.ind=TRUE)
  Validation <- over_train[trainIndexes, ]
Train <- over_train[-trainIndexes, ]
mtry_list= c(1:8)   pr.err <- c()
  s=0
  for(mt in mtry_list){
    rf <- randomForest(formula = over_train$Class ~ HospitalNo2 +
    Department + Estimatedcost + CE_ACCESSIBILITY + CE_CSAT +
    CE_VALUEFORMONEY + EM_NURSING + EM_DOCTOR + AD_TARRIFFPACKAGESEXPLAINATIO
N +
    AD_STAFFATTITUDE + INR_ROOMCLEANLINESS + INR_ROOMAMBIENCE +
    FNB_FOODDELIVERYTIME + AE_PATIENTSTATUSINFO + AE_ATTENDEEFOOD +
    DOC_TREATMENTEXPLAINATION + DOC_VISITS + NS_CALLBELLRESPONSE +
```

```
    NS_NURSEPROACTIVENESS + OVS_OVERALLSTAFFPROMPTNESS +
DP_DISCHARGEQUERIES, data = over_train, ntree = 100, mtry = mt,replace=T)
predicted <- predict(rf, newdata = Validation, type = "class")       pr.err
<- c(pr.err,mean(Validation$Class != predicted))   }
      bestmtry <-
which.min(pr.err)
  #test_binary_final_data1is the test data given in the case study

 rf <- randomForest(formula = over_train$Class ~ HospitalNo2 +
    Department + Estimatedcost + CE_ACCESSIBILITY + CE_CSAT +
    CE_VALUEFORMONEY + EM_NURSING + EM_DOCTOR + AD_TARRIFFPACKAGESEXPLAINATIO
N +
    AD_STAFFATTITUDE + INR_ROOMCLEANLINESS + INR_ROOMAMBIENCE +
    FNB_FOODDELIVERYTIME + AE_PATIENTSTATUSINFO + AE_ATTENDEEFOOD +
    DOC_TREATMENTEXPLAINATION + DOC_VISITS + NS_CALLBELLRESPONSE +
    NS_NURSEPROACTIVENESS + OVS_OVERALLSTAFFPROMPTNESS + DP_DISCHARGEQUERIES,
data = over_train, ntree = 200, mtry = bestmtry)
  rf.pred <- predict(rf, newdata = test_binary_final_data1, type = "class")
rf.conf=table(rf.pred,test_binary_final_data1$NPS_Status)
  #accuracy[i]=sum(diag(rf.conf))/sum(rf.conf)
 models.err[i] <- mean(test_binary_final_data1$NPS_Status != rf.pred)
}

accuracy_rf_over=1-
mean(models.err) accuracy_rf_over ##
[1] 0.7458333
cat("accuracy of Random forest for binary classification with oversampled dat
a :", mean(accuracy_rf_over))
## accuracy of Random forest for binary classification with oversampled data
: 0.7458333
```

##effect of balancing method- SMOTE( under and over sampling) -RF

```
library(DMwR)
```

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

```
## Registered S3 method overwritten by 'xts':
##   method     from ##
as.zoo.xts zoo
```

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

```
var=c("HospitalNo2","Department","Estimatedcost", "CE_ACCESSIBILITY","CE_CSAT
","CE_VALUEFORMONEY","EM_NURSING","EM_DOCTOR","AD_TARRIFFPACKAGESEXPLAINATION
","AD_STAFFATTITUDE","INR_ROOMCLEANLINESS","INR_ROOMAMBIENCE","FNB_FOODDELIVE
```

```
RYTIME","AE_PATIENTSTATUSINFO","AE_ATTENDEEFOOD","DOC_TREATMENTEXPLAINATION",
"DOC_VISITS","NS_CALLBELLRESPONSE","NS_NURSEPROACTIVENESS","OVS_OVERALLSTAFFP
ROMPTNESS","DP_DISCHARGEQUERIES","NPS_Status")
training=train_binary_final_Data1[,var]
training=subset(training,training$EM_DOCTOR !=1)
testing=test_binary_final_data1[,var]
testing=subset(testing,testing$NS_NURSEPROACTIVENESS !=1)
table(testing$NS_NURSEPROACTIVENESS)

##
##   1   2   3   4 ##
0  19 129 212

testing$NS_NURSEPROACTIVENESS=droplevels(testing$NS_NURSEPROACTIVENESS,exclud
e="1")

##converting variables to not ordered as smote creates NA values with
ordered variables for (i in 1:ncol(training))
{
  if(is.factor(training[[i]]))
  training[[i]]= factor( training[[i]] , ordered = FALSE )
}
training[,"Department"]=as.factor(training[,"Department"])
testing[,"Department"]=as.factor(testing[,"Department"])


balanced_data=SMOTE(NPS_Status~., training, perc.over = 35,  perc.under = 400
)#k=5 control parameter


balanced_data1=balanced_data i=0
for (i in 1:(ncol(balanced_data)-1))
{
  if(class(balanced_data[[i]])=="factor" |
(is.numeric(balanced_data[[i]])))
{

   balanced_data1[[i]]=as.factor(balanced_data[[i]])
if (nlevels(balanced_data1[[i]])<5)
    balanced_data1[[i]]=as.ordered(balanced_data1[[i]])
else
     balanced_data1[[i]]=as.numeric(balanced_data[[i]])

}
}
balanced_data1[[22]]=as.factor(balanced_data[[22]])
balanced_data1[["Department"]]=as.factor(balanced_data[["Department"]])
testing[[22]]=as.factor(testing[[22]])
```

```
k <- 10 nmethod
<- 1
folds <- cut(seq(1,nrow(balanced_data1)),breaks=k,labels=FALSE)
models.err <- matrix(-1,k,nmethod, dimnames=list(paste0("Fold", 1:k), c("rf")
)) i=0
for(i in 1:k)
{
  trainIndexes <- which(folds==i, arr.ind=TRUE)
  Validation <- balanced_data1[trainIndexes, ]
Train <- balanced_data1[-trainIndexes, ]
mtry_list= c(1:8)   pr.err <- c()
  s=0
  for(mt in mtry_list){
    rf <- randomForest( balanced_data1$NPS_Status~., data = balanced_data1, n
tree = 100, mtry = mt,replace=T)     predicted <- predict(rf, newdata =
Validation, type = "class")        pr.err <-
c(pr.err,mean(Validation$NPS_Status != predicted))   }
      bestmtry <-
which.min(pr.err)
  #test_binary_final_data1is the test data given in the case study

 rf_x <- randomForest(balanced_data1$NPS_Status ~., data = balanced_data1, nt
ree = 200, mtry = bestmtry)
  rf.pred <- predict(rf_x, newdata = testing, type = "class")
rf.conf=table(rf.pred,testing$NPS_Status)
#accuracy[i]=sum(diag(rf.conf))/sum(rf.conf)  models.err[i] <-
mean(testing$NPS_Status != rf.pred)
}

accuracy_rf_smote=1-mean(models.err)

accuracy_rf_smote ## [1] 0.7008333

cat("accuracy of Random forest for binary classification with SMOTE data :",
mean(accuracy_rf_smote))

## accuracy of Random forest for binary classification with SMOTE data : 0.70
08333
```

#Problem 1 #load the train and test data

```
rm(list=ls(all =TRUE)) library(readxl)
Train <- read_excel("C:/R files/IMB651-XLS-ENG.xlsx", sheet = "Training Data
for Multi-Class M",
                    col_names = T)
Test <- read_excel("C:/R files/IMB651-XLS-ENG.xlsx", sheet = "Test Data for M
ulti-Class Model",

                    col_names = T)
```

#Function to convert the categorical variables to factors

```
factor_convert <- function(x){
for(i in 1:(length(x) ))
  {
     x[,i] <- as.factor(x[,i])

  }
 return(x)

}
```

**Inputing data to the function except the quantitative variables and removing the SN column as it just ID and also CE_NPS as it conveys the same information as NPS_Status**

```
Train <- as.data.frame(Train) Test <- as.data.frame(Test) data1 <-
factor_convert(Train[,c(3,5,6,7,9,10,11,12,13,14,15,16,17,18,19,20,2
1,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,
47,52)])
Traindata<-cbind(data1,Train[,c(2,4,8,48,49,50)])

data1test <- factor_convert(Test[,c(3,5,6,7,9,10,11,12,13,14,15,16,17,18,19,2
0,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,
46,47,52)])

Testdata<-cbind(data1test,Test[,c(2,4,8,48,49,50)])
```

#checking for missing values

```
sum(is.na(Traindata))

## [1] 0
```

**Creating a 2 class variable for Detractors**

```
Traindata$NPS_Status <- ifelse(Traindata$NPS_Status=="Detractor"|Traindata$NPS_Status=="Passive",1,0) table(Traindata$NPS_Status)
```

```
##
##    0    1
## 3140 1849
```

```
Traindata$NPS_Status <- as.factor(Traindata$NPS_Status)
```

```
Testdata$NPS_Status <- ifelse(Testdata$NPS_Status=="Detractor"|Testdata$NPS_Status=="Passive",1,0) table(Testdata$NPS_Status)
```

```
##
##   0   1
## 203 161
```

```
Testdata$NPS_Status <- as.factor(Testdata$NPS_Status)
```

## Converting survey questions to ordinal variables

```
#Variables that are ordinal
Traindata2 <- Traindata [,c(9:43)]

#other variables
Traindata3 <- Traindata[,-c(9:43)]
```

```
library(dplyr)
```

```
##
## 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(tidyr)

#convert to ordinal
Traindata2 <- Traindata2 %>%
  mutate_if(is.factor,as.ordered)

#Combine the variables now
Traindata4 <- cbind(Traindata2,Traindata3)
```

```
Traindata5 <- Traindata4[,-
c(45,48,49)]#TEST DATA
Testdata2 <- Testdata [,c(9:43)]
Testdata3 <- Testdata[,-c(9:43)]
Testdata2 <- Testdata2 %>%
mutate_if(is.factor,as.ordered)
Testdata4 <-
cbind(Testdata2,Testdata3)
Testdata5 <- Testdata4[,-c(45,48,49)]
```

#Finding the variables leading to quasi-complete separation

```
library(brglm2)

quasi_fit=glm(Traindata5$NPS_Status~.,data=Traindata5, family=binomial("logit
"),method = "detect_separation",linear_program="dual")quasi_fit
```

```
## Separation: FALSE
## Existence of maximum likelihood estimates
##                          (Intercept)                 CE_ACCESSIBILITY.L
##                                  Inf                                  0
##                 CE_ACCESSIBILITY.Q                 CE_ACCESSIBILITY.C
##                                    0                                  0
##                           CE_CSAT.L                           CE_CSAT.Q
##                                    0                                  0
##                           CE_CSAT.C                 CE_VALUEFORMONEY.L
##                                    0                                  0
##                 CE_VALUEFORMONEY.Q                 CE_VALUEFORMONEY.C
##                                    0                                  0
##            EM_IMMEDIATEATTENTION.L            EM_IMMEDIATEATTENTION.Q
##                                    0                                  0
##            EM_IMMEDIATEATTENTION.C                       EM_NURSING.L
##                                    0                                  0
##                       EM_NURSING.Q                       EM_NURSING.C
##                                    0                                  0
##                        EM_DOCTOR.L                        EM_DOCTOR.Q
##                               -Inf                                Inf
##                        EM_DOCTOR.C                       EM_OVERALL.L
##                               -Inf                                  0
##                       EM_OVERALL.Q                       EM_OVERALL.C
##                                    0                                  0
##                          AD_TIME.L                          AD_TIME.Q
##                                    0                                  0
##                          AD_TIME.C   AD_TARRIFFPACKAGESEXPLAINATION.L
##                                    0                                  0
##   AD_TARRIFFPACKAGESEXPLAINATION.Q   AD_TARRIFFPACKAGESEXPLAINATION.C
##                                    0                                  0
```

```
##                 AD_STAFFATTITUDE.L                 AD_STAFFATTITUDE.Q
##                                  0                                  0
##                 AD_STAFFATTITUDE.C              INR_ROOMCLEANLINESS.L
##                                  0                                  0
##              INR_ROOMCLEANLINESS.Q              INR_ROOMCLEANLINESS.C
##                                  0                                  0
##                    INR_ROOMPEACE.L                    INR_ROOMPEACE.Q
##                                  0                                  0
##                    INR_ROOMPEACE.C                INR_ROOMEQUIPMENT.L
##                                  0                                  0
##                 INR_ROOMEQUIPMENT.Q                INR_ROOMEQUIPMENT.C
##                                  0                                  0
##                  INR_ROOMAMBIENCE.L                 INR_ROOMAMBIENCE.Q
##                                  0                                  0
##                  INR_ROOMAMBIENCE.C                  FNB_FOODQUALITY.L
##                                  0                                  0
##                   FNB_FOODQUALITY.Q                  FNB_FOODQUALITY.C
##                                  0                                  0
##             FNB_FOODDELIVERYTIME.L             FNB_FOODDELIVERYTIME.Q
##                                  0                                  0
##             FNB_FOODDELIVERYTIME.C                    FNB_DIETICIAN.L
##                                  0                                  0
##                    FNB_DIETICIAN.Q                    FNB_DIETICIAN.C
##                                  0                                  0
##                FNB_STAFFATTITUDE.L                FNB_STAFFATTITUDE.Q
##                                  0                                  0
##                FNB_STAFFATTITUDE.C                  AE_ATTENDEECARE.L
##                                  0                                  0
##                  AE_ATTENDEECARE.Q                  AE_ATTENDEECARE.C
##                                  0                                  0
##             AE_PATIENTSTATUSINFO.L             AE_PATIENTSTATUSINFO.Q
##                               -Inf                                Inf
```

```
##             AE_PATIENTSTATUSINFO.C                 AE_ATTENDEEFOOD.L
##                               -Inf                                 0
##                  AE_ATTENDEEFOOD.Q                 AE_ATTENDEEFOOD.C
##                                  0                                 0
##         DOC_TREATMENTEXPLAINATION.L         DOC_TREATMENTEXPLAINATION.Q
##                               -Inf                               Inf
##         DOC_TREATMENTEXPLAINATION.C                    DOC_ATTITUDE.L
##                               -Inf                                 0
##                     DOC_ATTITUDE.Q                    DOC_ATTITUDE.C
##                                  0                                 0
##                      DOC_VISITS.L                      DOC_VISITS.Q
##                                  0                                 0
##                      DOC_VISITS.C       DOC_TREATMENTEFFECTIVENESS.L
##                                  0                              -Inf
##       DOC_TREATMENTEFFECTIVENESS.Q       DOC_TREATMENTEFFECTIVENESS.C
##                               Inf                               -Inf
##              NS_CALLBELLRESPONSE.L              NS_CALLBELLRESPONSE.Q
##                                  0                                 0
##              NS_CALLBELLRESPONSE.C                NS_NURSESATTITUDE.L
##                                  0                              -Inf
##                NS_NURSESATTITUDE.Q                NS_NURSESATTITUDE.C
##                               Inf                               -Inf
##            NS_NURSEPROACTIVENESS.L            NS_NURSEPROACTIVENESS.Q
##                                  0                                 0
##                NS_NURSEPATIENCE.L                NS_NURSEPATIENCE.Q
##                               -Inf                               Inf
##                NS_NURSEPATIENCE.C          OVS_OVERALLSTAFFATTITUDE.L
##                               -Inf                               -Inf
##          OVS_OVERALLSTAFFATTITUDE.Q          OVS_OVERALLSTAFFATTITUDE.C
##                               Inf                               -Inf
##       OVS_OVERALLSTAFFPROMPTNESS.L       OVS_OVERALLSTAFFPROMPTNESS.Q
##                                  0                                 0
##       OVS_OVERALLSTAFFPROMPTNESS.C             OVS_SECURITYATTITUDE.L
##                                  0                                 0
##             OVS_SECURITYATTITUDE.Q             OVS_SECURITYATTITUDE.C
##                                  0                                 0
##                DP_DISCHARGETIME.L                DP_DISCHARGETIME.Q
##                                  0                                 0
##                DP_DISCHARGETIME.C              DP_DISCHARGEQUERIES.L
##                                  0                                 0
##              DP_DISCHARGEQUERIES.Q              DP_DISCHARGEQUERIES.C
##                                  0                                 0
##              DP_DISCHARGEPROCESS.L              DP_DISCHARGEPROCESS.Q
##                                  0                                 0
```

```
##              DP_DISCHARGEPROCESS.C                MaritalStatusMarried
##                                  0                                -Inf
##             MaritalStatusSeparated                 MaritalStatusSingle
##                               -Inf                                -Inf
##                MaritalStatusWidowed                                SexM
##                               -Inf                                   0
##                  BedCategoryDAYCARE                  BedCategoryGENERAL
##                               Inf                                 Inf
##              BedCategoryGENERAL HD                       BedCategoryITU
##                               Inf                                -Inf
##                BedCategoryRenal ICU              BedCategorySEMISPECIAL
##                               -Inf                                Inf
##           BedCategorySEMISPECIAL HD                  BedCategorySPECIAL
##                               Inf                                 Inf
##                BedCategoryULTRA DLX                BedCategoryULTRA SPL
##                               Inf                                 Inf
##                       DepartmentGEN                     DepartmentGYNAEC
##                                  0                                   0
##                     DepartmentORTHO                   DepartmentPEDIATRIC
##                                  0                                   0
##                     DepartmentRENAL                     DepartmentSPECIAL
##                                  0                                   0
##           InsPayorcategoryEXEMPTION           InsPayorcategoryINSURANCE
##                                  0                                   0
##       InsPayorcategoryINTERNATIONAL             InsPayorcategoryPATIENT
##                                  0                                   0
##            StateAndaman And Nicobar                  StateAndhra Pradesh
##                               -Inf                                 Inf
##                          StateAssam                      StateBangladesh
##                               Inf                                -Inf
##                    StateBhubaneshwar                         StateBihar
##                               -Inf                                 Inf
##                     StateChandigarh                    StateChhattisgarh
##                               -Inf                                 Inf
##                     StateDarjeeling                         StateDelhi
##                               -Inf                                 Inf
##                           StateDoha                       StateGermany
##                               -Inf                                -Inf
##                            StateGoa                       StateGujarat
##                               Inf                                -Inf
##                         StateHaryana                  StateInternational
##                               Inf                                -Inf
##                           StateIraq                       StateJharkand
##                               Inf                                -Inf
```

```
##                    StateJharkhand                    StateKarnataka
##                               Inf                               Inf
##                       StateKenya                       StateKerala
##                              -Inf                               Inf
##                     StateKolkata                      StateKolkatta
##                               Inf                               Inf
##                StateMadhya Pradesh                 StateMaharashtra
##                               Inf                               Inf
##                     StateMaldives                      StateManipur
##                               Inf                               Inf
##                    StateMauritius                     StateMeghalaya
##                               Inf                              -Inf
##                     StateMizoram                      StateMongolia
##                              -Inf                               Inf
##                      StateMumbai                       StateMuscat
##                              -Inf                              -Inf
##                       StateNepal                  StateNew Zealand
##                               Inf                               Inf
##                     StateNigeria                         StateOman
##                              -Inf                               Inf
##                     StateOntario                       StateOrissa
##                               Inf                               Inf
##                    StateRajasthan                       StateRanchi
##                               Inf                               Inf
##                      StateRWANDA                   StateSaudi Arabia
##                               Inf                              -Inf
##                     StateSikkim                     StateTamil Nadu
##                              -Inf                               Inf
##                     StateTanzania                      StateTripura
##                               Inf                               Inf
##                          StateUAE                           StateUK
##                              -Inf                               Inf
##                     StateUnknown                          StateUSA
##                               Inf                              -Inf
##                  StateUttar Pradesh                  StateUttarakhand
##                               Inf                               Inf
##                   StateWest Bengal                      StateZimbabwe
##                               Inf                               Inf
##                    CountryANGOLA                   CountryBANGLADESH
##                               Inf                              -Inf
##                    CountryCANADA                       CountryFIJI
##                                NA                              -Inf
##                   CountryGERMANY                       CountryINDIA
##                                NA                                 0
```

```
##                         CountryIRAQ    CountryISLAMIC REPUBLIC OF IRAN
##                                   0                                -Inf
##                       CountryKENYA                   CountryMALDIVES
##                                  NA                                NA
##                    CountryMAURITIUS                   CountryMONGOLIA
##                                   0                                NA
##                   CountryMOZAMBIQUE                      CountryNEPAL
##                                -Inf                                 0
##                  CountryNEW ZEALAND                     CountryNIGERIA
##                                  NA                                -Inf
##                        CountryOMAN                      CountryQATAR
##                                   0                                NA
##                 CountrySaudi Arabia                CountrySAUDI ARABIA
##                                -Inf                                NA
##                       CountrySUDAN                     CountryUGANDA
##                                 Inf                                NA
##         CountryUNITED ARAB EMIRATES              CountryUNITED KINGDOM
##                                  NA                                NA
```

```
## CountryUNITED REPUBLIC OF TANZANIA    CountryUNITED STATES OF AMERICA
##                                  NA                                   NA
##                         CountryYEMEN                      CountryZIMBABWE
##                                  NA                                   NA
##                        STATEZONEEAST              STATEZONEINTERNATIONAL
##                                  NA                                   NA
##                       STATEZONENORTH                     STATEZONESOUTH
##                                  NA                                   NA
##                    STATEZONEUnknown                      STATEZONEWEST
##                                  NA                                   NA
##                              AgeYrs                      Estimatedcost
##                                   0                                   0
##                         LengthofStay
##                                   0
## 0: finite value, Inf: infinity, -Inf: -infinity var
<- as.data.frame(quasi_fit$betas)
```

Quasi- complete separation variables are "EM_DOCTOR","STATEZONE","Country","State","BedCategory","MaritalStatus","OVS_OVERALLSTAFFATTITUDE","EM_DOCTOR","NS_NURSESATTITUDE","DOC_TREATMENTEFFECTIVENESS","AE_PATIENTSTATUSINFO","NS_NURSEPATIENCE","DOC_TREATMENTEXPLAINATION"

## Stepwise Regression

```
cols_exclude1 <- c("EM_DOCTOR","STATEZONE","Country","State","BedCategory","MaritalStatus","OVS_OVERALLSTAFFATTITUDE","EM_DOCTOR","NS_NURSESATTITUDE","DOC_TREATMENTEFFECTIVENESS","AE_PATIENTSTATUSINFO","NS_NURSEPATIENCE","DOC_TREATMENTEXPLAINATION")

#Drop the quasi seperated variables
Traindata1 <- Traindata5[, !(colnames(Traindata5) %in% cols_exclude1), drop = FALSE]


library(MASS)

##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr': ##
##     select

model1 <- glm(NPS_Status ~., data = Traindata1, family = binomial("logit"))
model1

##
## Call:  glm(formula = NPS_Status ~ ., family = binomial("logit"), data = Traindata1)
```

```
##
## Coefficients:
##                       (Intercept)                CE_ACCESSIBILITY.L
##                         3.315e+00                         -1.523e+00
##                CE_ACCESSIBILITY.Q                CE_ACCESSIBILITY.C
##                         4.842e-01                          4.015e-01
##                         CE_CSAT.L                          CE_CSAT.Q
##                        -1.249e+00                         -9.829e-01
##                         CE_CSAT.C                CE_VALUEFORMONEY.L
##                         8.087e-01                         -1.171e+00
##                CE_VALUEFORMONEY.Q                CE_VALUEFORMONEY.C
##                        -3.039e-01                          5.274e-01
##           EM_IMMEDIATEATTENTION.L           EM_IMMEDIATEATTENTION.Q
##                        -5.857e-01                         -2.662e-02
##           EM_IMMEDIATEATTENTION.C                      EM_NURSING.L
##                         1.537e-01                          1.066e+00
##                      EM_NURSING.Q                      EM_NURSING.C
##                         -1.072e+00                         2.863e-01
##                      EM_OVERALL.L                      EM_OVERALL.Q
##                        -6.121e-01                         2.599e-01
##                      EM_OVERALL.C                         AD_TIME.L
##                         1.692e-01                        -1.786e-01
##                         AD_TIME.Q                         AD_TIME.C
##                         8.264e-02                        -5.539e-02
## AD_TARRIFFPACKAGESEXPLAINATION.L  AD_TARRIFFPACKAGESEXPLAINATION.Q
##                        -1.029e+00                         7.315e-04
## AD_TARRIFFPACKAGESEXPLAINATION.C               AD_STAFFATTITUDE.L
##                         2.293e-01                          7.960e-01
##                AD_STAFFATTITUDE.Q                AD_STAFFATTITUDE.C
##                         1.691e-01                         -2.833e-01
##             INR_ROOMCLEANLINESS.L             INR_ROOMCLEANLINESS.Q
##                         2.777e-01                          1.081e-01
##             INR_ROOMCLEANLINESS.C                   INR_ROOMPEACE.L
##                         1.890e-01                         -2.595e-02
##                   INR_ROOMPEACE.Q                   INR_ROOMPEACE.C
##                        -1.049e-01                         -8.273e-04
##               INR_ROOMEQUIPMENT.L               INR_ROOMEQUIPMENT.Q
##                         6.276e-01                         -3.628e-01
##               INR_ROOMEQUIPMENT.C                INR_ROOMAMBIENCE.L
##                         1.334e-02                         -5.582e-01
##                INR_ROOMAMBIENCE.Q                INR_ROOMAMBIENCE.C
##                        -4.608e-02                          2.809e-01
##                 FNB_FOODQUALITY.L                 FNB_FOODQUALITY.Q
##                        -3.567e-01                          1.802e-01
##                 FNB_FOODQUALITY.C             FNB_FOODDELIVERYTIME.L
##                        -2.726e-02                         -2.605e-01
```

```
##
## Degrees of Freedom: 4988 Total (i.e. Null);  4891 Residual
## Null Deviance:       6578
## Residual Deviance: 5045  AIC: 5241
```

```
# Stepwise regression model
stepwise_reg <- stepAIC(model1, direction = "both", trace = FALSE)
stepwise_reg$anova
```

```
## Stepwise Model Path
## Analysis of Deviance Table
##
## Initial Model:
## NPS_Status ~ CE_ACCESSIBILITY + CE_CSAT + CE_VALUEFORMONEY +
##     EM_IMMEDIATEATTENTION + EM_NURSING + EM_OVERALL + AD_TIME +
##     AD_TARRIFFPACKAGESEXPLAINATION + AD_STAFFATTITUDE + INR_ROOMCLEANLINES
S +
##     INR_ROOMPEACE + INR_ROOMEQUIPMENT + INR_ROOMAMBIENCE + FNB_FOODQUALITY
+
##     FNB_FOODDELIVERYTIME + FNB_DIETICIAN + FNB_STAFFATTITUDE +
##     AE_ATTENDEECARE + AE_ATTENDEEFOOD + DOC_ATTITUDE + DOC_VISITS +
##     NS_CALLBELLRESPONSE + NS_NURSEPROACTIVENESS + OVS_OVERALLSTAFFPROMPTNE
SS +
##     OVS_SECURITYATTITUDE + DP_DISCHARGETIME + DP_DISCHARGEQUERIES +
##     DP_DISCHARGEPROCESS + Sex + Department + InsPayorcategory +  ##
AgeYrs + Estimatedcost + LengthofStay
##
## Final Model:
## NPS_Status ~ CE_ACCESSIBILITY + CE_CSAT + CE_VALUEFORMONEY +
##     EM_NURSING + AD_TARRIFFPACKAGESEXPLAINATION + AD_STAFFATTITUDE +
##     INR_ROOMCLEANLINESS + INR_ROOMAMBIENCE + FNB_FOODDELIVERYTIME +
##     AE_ATTENDEEFOOD + DOC_VISITS + NS_CALLBELLRESPONSE + NS_NURSEPROACTIVE
NESS +
##     OVS_OVERALLSTAFFPROMPTNESS + DP_DISCHARGEQUERIES + Department +  ##
Estimatedcost + LengthofStay
##
##
##                       Step Df  Deviance Resid. Df Resid. Dev      AIC
## 1                                            4891   5044.994 5240.994
## 2                - AD_TIME  3 0.7509473      4894   5045.745 5235.745
## 3    - DP_DISCHARGEPROCESS  3 1.0345005      4897   5046.779 5230.779
## 4          - INR_ROOMPEACE  3 0.9789119      4900   5047.758 5225.758
## 5             - EM_OVERALL  3 1.6511287      4903   5049.409 5221.409
## 6          - FNB_DIETICIAN  3 1.9422239      4906   5051.352 5217.352
## 7        - AE_ATTENDEECARE  3 1.8028867      4909   5053.154 5213.154
## 8       - DP_DISCHARGETIME  3 2.0685256      4912   5055.223 5209.223
## 9      - FNB_STAFFATTITUDE  3 2.4763047      4915   5057.699 5205.699
## 10     - INR_ROOMEQUIPMENT  3 3.1908563      4918   5060.890 5202.890
```

```
## 11 - EM_IMMEDIATEATTENTION  3 4.0091293      4921   5064.899 5200.899
## 12      - InsPayorcategory  4 6.0716522      4925   5070.971 5198.971
```

```
## 13                - AgeYrs  1 0.4563331      4926   5071.427 5197.427
## 14                   - Sex  1 0.4843629      4927   5071.912 5195.912
## 15  - OVS_SECURITYATTITUDE  3 4.4552601      4930   5076.367 5194.367
## 16       - FNB_FOODQUALITY  3 5.3786448      4933   5081.745 5193.745 ## 17
- DOC_ATTITUDE  3 5.2503603      4936   5086.996 5192.996
```

variables obtained after stepwise regression are CE_ACCESSIBILITY + CE_CSAT + CE_VALUEFORMONEY + EM_NURSING + AD_TARRIFFPACKAGESEXPLAINATION + AD_STAFFATTITUDE + INR_ROOMCLEANLINESS + INR_ROOMAMBIENCE + FNB_FOODDELIVERYTIME + AE_ATTENDEEFOOD + DOC_VISITS + NS_CALLBELLRESPONSE + NS_NURSEPROACTIVENESS + OVS_OVERALLSTAFFPROMPTNESS + DP_DISCHARGEQUERIES + Department + Estimatedcost + LengthofStay

##ADA boost for binary classification with parameter tuning

```
library(adabag)
```

```
## Loading required package: rpart
```

```
## Loading required package: caret
```

```
## Loading required package: lattice ##
```

```
Loading required package: ggplot2
```

```
##
## Attaching package: 'ggplot2'
```

```
## The following object is masked from 'package:randomForest': ##
##     margin
```

```
## Loading required package: foreach
```

```
## Loading required package: doParallel
```

```
## Loading required package: iterators ##
```

```
Loading required package: parallel
```

```
library(caret) library(ada)

#equating the levels of train and test data for the variable NS_NURSEPROACTIV
ENESS
Testdata$NS_NURSEPROACTIVENESS <- ordered(Testdata$NS_NURSEPROACTIVENESS, lev
els = levels(Traindata$NS_NURSEPROACTIVENESS))

#Tuning the hyperparameters #defining our parameters first params_ada =
expand.grid(iter=c(75),maxdepth=c(5,6,7,8),nu=c(0.01,0.1))

#We now build an AdaBoost model using Grid Search and fit it on the Train dat
```

```
aset
adab_gridsearch <- train(NPS_Status~.,data = Traindata,method="ada",tuneGrid=
params_ada) adab_gridsearch

## Boosted Classification Trees
##
## 4989 samples
##   18 predictor
##    2 classes: '0', '1'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 4989, 4989, 4989, 4989, 4989, 4989, ...
## Resampling results across tuning parameters: ##
##   nu    maxdepth  Accuracy   Kappa
##   0.01  5         0.7366746  0.3777369
##   0.01  6         0.7419377  0.3970632
##   0.01  7         0.7470097  0.4128051
##   0.01  8         0.7466794  0.4150242
##   0.10  5         0.7520811  0.4309352
##   0.10  6         0.7542786  0.4385603
##   0.10  7         0.7526516  0.4373799
##   0.10  8         0.7530929  0.4400564
##
## Tuning parameter 'iter' was held constant at a value of 75
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were iter = 75, maxdepth = 6 and
nu ##  = 0.1.

#best parameters
adab_gridsearch$bestTune

##   iter maxdepth  nu
## 6   75        6 0.1

#Best parameters obtained are for iterations 75, maxdepth of 6 and nu of 0.01
. Now we build another model with the best parameters.
model = boosting(NPS_Status~., data=Traindata, boos=TRUE, mfinal=75, control=
rpart.control(maxdepth=6, cp =0.01)) print(names(model))

## [1] "formula"    "trees"      "weights"    "votes"      "prob"
## [6] "class"      "importance" "terms"      "call" print(model$trees[1])

## [[1]]
## n= 4989
##
```

```
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node ##
##  1) root 4989 1881 0 (0.62297054 0.37702946)
##    2) CE_CSAT=4 2618  506 0 (0.80672269 0.19327731) *
##    3) CE_CSAT=1,2,3 2371  996 1 (0.42007592 0.57992408)
##      6) CE_VALUEFORMONEY=3,4 1965  951 1 (0.48396947 0.51603053)
##       12) NS_NURSEPROACTIVENESS=3,4 1882  935 0 (0.50318810 0.49681190)
##         24) INR_ROOMAMBIENCE=3,4 1770  851 0 (0.51920904 0.48079096)
##           48) CE_CSAT=3,4 1728  815 0 (0.52835648 0.47164352) *
##           49) CE_CSAT=1,2 42    6 1 (0.14285714 0.85714286) *
##         25) INR_ROOMAMBIENCE=1,2 112   28 1 (0.25000000 0.75000000) *
##       13) NS_NURSEPROACTIVENESS=2 83    4 1 (0.04819277 0.95180723) *
##      7) CE_VALUEFORMONEY=1,2 406   45 1 (0.11083744 0.88916256) *
```

```
#predict on test data pred =
predict(model, Testdata)
print(pred$confusion)
```

```
##                Observed Class
## Predicted Class   0   1
##               0 177  72 ##               1  26  89
print(pred$error) ## [1] 0.2692308 result =
data.frame(Testdata$NPS_Status, pred$prob, pred$class)

accuracy_ada=1- (pred$error)
accuracy_ada ## [1] 0.7308
cat("accuracy of Ada Boost for binary classification:", (accuracy_ada))
## accuracy of Ada Boost for binary classification: 0.7308
```

Recall of the model is 72%

#Ada boost cross validation

```
# cross-validataion method
cvmodel = boosting.cv(NPS_Status~., data=Traindata, boos=TRUE, mfinal=100, v=
5, control=rpart.control(maxdepth=6, cp =0.01))
```

```
## i:  1 Tue Nov 26 22:02:09 2019
## i:  2 Tue Nov 26 22:03:20 2019
## i:  3 Tue Nov 26 22:04:31 2019
## i:  4 Tue Nov 26 22:05:43 2019  ##
```

```
i:  5 Tue Nov 26 22:06:23 2019
```

```
print(cvmodel[-1])
```

```
## $confusion
##                Observed Class
## Predicted Class    0    1
##               0 2790  846 ##
1  350 1003
##
## $error
## [1] 0.2397274
```

```
cvmodel$error
```

```
## [1] 0.2397274
```

```
Accuracy of AdaBoost model is 76%
```

##Ada boost undersampling

```
undersampledata <- downSample(Traindata[,-17], Traindata$NPS_Status, list = F
, yname = "NPS_Status")
model = boosting(NPS_Status~., data=undersampledata, boos=TRUE, mfinal=100,co
ntrol=rpart.control(maxdepth=6, cp =0.01)) print(names(model))
```

```
## [1] "formula"    "trees"      "weights"    "votes"      "prob"
## [6] "class"      "importance" "terms"      "call" print(model$trees[1])
```

```
## [[1]]
## n= 3698
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 3698 1804 1 (0.4878313 0.5121687)
##   2) CE_CSAT=4 1811  553 0 (0.6946438 0.3053562) *
##   3) CE_CSAT=1,2,3 1887  546 1 (0.2893482 0.7106518) *
```

```
#predict on test data pred =
predict(model, Testdata)
print(pred$confusion)
```

```
##                Observed Class
## Predicted Class   0   1
##               0 146
49 ##               1  57
112 print(pred$error) ##
[1] 0.2912088
```

```
accuracy_ada=1- (pred$error)
accuracy_ada ## [1] 0.709
```

```
cat("accuracy of Ada Boost for undersample data:", (accuracy_ada))
```

```
## accuracy of Ada Boost
for undersample data:
0.709
```

Accuracy of the model was around 70.9% and overall recall is 72%

##Ada boost oversampling

```
oversampledata <- upSample(Traindata[,-17], Traindata$NPS_Status, list = F, y
name = "NPS_Status")
model = boosting(NPS_Status~., data=oversampledata, boos=TRUE, mfinal=100,con
trol=rpart.control(maxdepth=6, cp =0.01)) print(names(model))

## [1] "formula"    "trees"      "weights"    "votes"      "prob"
## [6] "class"      "importance" "terms"      "call" print(model$trees[1])



## [[1]]
## n= 6280
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node ##
## 1) root 6280 3134 0 (0.5009554 0.4990446)
##   2) CE_CSAT=4 3062  873 0 (0.7148922 0.2851078) *
##   3) CE_CSAT=1,2,3 3218  957 1 (0.2973897 0.7026103) *

#predict on test data pred = predict(model,
Testdata) print(pred$confusion)

##                Observed Class
## Predicted Class   0   1
##               0 151  56 ##               1
52 105
print(pred$error) ## [1] 0.2967033

accuracy_ada=1- (pred$error)

accuracy_ada ## [1] 0.704

cat("accuracy of Ada Boost for oversample data:", (accuracy_ada))

## accuracy of Ada Boost for oversample data:
0.704
```

Accuracy of Ada Boost model for oversample data is 70.4% and recall was 68%

##Ada boost SMOTE

```
library(DMwR)

## Loading required package: grid

## Registered S3 method overwritten by 'xts':
##   method     from ##
as.zoo.xts zoo

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

data_smote <- SMOTE(NPS_Status~., data = Traindata, perc.over = 100, perc.und
er = 100)

model = boosting(NPS_Status~., data=data_smote, boos=TRUE, mfinal=100,control
=rpart.control(maxdepth=6, cp =0.01)) print(names(model))

## [1] "formula"    "trees"      "weights"    "votes"      "prob"
## [6] "class"      "importance" "terms"      "call" print(model$trees[1])

## [[1]]
## n= 5547
##
## node), split, n, loss, yval, (yprob)
```

```
##       * denotes terminal node
##
##  1) root 5547 1873 1 (0.337659996 0.662340004)
##    2) NS_NURSEPROACTIVENESS=4 2496 1076 0 (0.568910256 0.431089744)
##      4) CE_VALUEFORMONEY=4 1138  258 0 (0.773286467 0.226713533) *
##      5) CE_VALUEFORMONEY=1,2,3 1358  540 1 (0.397643594 0.602356406)
##       10) DP_DISCHARGEQUERIES=4 695  342 0 (0.507913669 0.492086331)
##         20) CE_CSAT=4 251   96 0 (0.617529880 0.382470120) *
##         21) CE_CSAT=1,2,3 444  198 1 (0.445945946 0.554054054) *
##       11) DP_DISCHARGEQUERIES=1,2,3 663  187 1 (0.282051282 0.717948718) *
##    3) NS_NURSEPROACTIVENESS=2,3 3051  453 1 (0.148475910 0.851524090)
##      6) NS_CALLBELLRESPONSE=1,2,3 1630  384 1 (0.235582822 0.764417178)
##       12) NS_NURSEPROACTIVENESS=3,4 983  379 1 (0.385554425 0.614445575)
##         24) CE_CSAT=4 289   94 0 (0.674740484 0.325259516) *
##         25) CE_CSAT=1,2,3 694  184 1 (0.265129683 0.734870317) *
##       13) NS_NURSEPROACTIVENESS=2 647    5 1 (0.007727975 0.992272025) * ##
7) NS_CALLBELLRESPONSE=4 1421   69 1 (0.048557354 0.951442646) *
```

```
#predict on test data pred =
predict(model, Testdata)
print(pred$confusion)
```

```
##                Observed Class
## Predicted Class   0   1
##               0 134  48 ##
1  69 113 print(pred$error)
```

```
## [1] 0.3214286
```

```
 accuracy_ada=1- (pred$error)
accuracy_ada ## [1] 0.6788
```

```
cat("accuracy of Ada Boost for smote data:", (accuracy_ada))
```

```
## accuracy of Ada Boost for smote data: 0.6788
```

Accuracy of Ada Boost model for oversample data is 67.88% and overall recall was 68%

## Adaboost for Multiclassification

```
rm(list=ls())

setwd("C:\\Users\\thoma\\Documents\\UIC\\Courses\\IDS 572 Data Mining\\Asst4"
)

library(readxl) library(dplyr)

##
## 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(tidyr)
library(randomForest)

## randomForest 4.6-14

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

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:dplyr':
##
##     combine

Train_multiclass <- read_excel("IMB651-XLS-ENG.xlsx", sheet = "Training Data
for Multi-Class M",
                    col_names = T)

Test_multiclass <- read_excel("IMB651-XLS-ENG.xlsx", sheet = "Test Data for M
ulti-Class Model",                     col_names = T)
print(dim(Train_multiclass))

## [1] 4989   52

print(dim(Test_multiclass))

## [1] 364  52
```

```
##  $ NS_NURSEPATIENCE              : num  4 4 4 4 4 4 4 4 4 4 ...
##  $ OVS_OVERALLSTAFFATTITUDE      : num  4 4 4 4 4 4 3 4 4 4 ...
##  $ OVS_OVERALLSTAFFPROMPTNESS    : num  4 4 4 3 4 4 3 4 4 4 ...
##  $ OVS_SECURITYATTITUDE          : num  4 4 4 3 4 4 3 4 4 4 ...
##  $ DP_DISCHARGETIME              : num  4 4 3 3 4 4 3 3 4 2 ...
##  $ DP_DISCHARGEQUERIES           : num  4 4 4 4 4 4 3 4 4 3 ...
##  $ DP_DISCHARGEPROCESS           : num  4 4 3 3 4 4 4 3 4 2 ...
##  $ AdmissionDate                 : POSIXct, format: "2014-07-18" "2014-
0711" ...
##  $ DischargeDate                 : POSIXct, format: "2014-07-21" "2014-
0716" ...
##  $ LengthofStay                  : num  3 5 33 6 3 3 6 5 4 4 ...
##  $ CE_NPS                        : num  9 10 7 10 10 10 10 9 9 10 ... ##
$ NPS_Status                    : chr  "Promotor" "Promotor" "Passive" "P
romotor" ...
```

```
# types of variables in the data
table(unlist(lapply(Train_multiclass, class)))
```

```
##
## character   numeric   POSIXct    POSIXt
##         9        41         2         2
```

```
## Checking for missing values
sum(is.na(Train_multiclass))
```

```
## [1] 0
```

```
## Target Variables
# NPS_Status

table(Train_multiclass$NPS_Status)
```

```
##
## Detractor   Passive  Promotor
##       502      1347      3140
```

```
# dropping the 1st column
Train2_multiclass <- Train_multiclass[,-1]
```

```
# Q 6 Part 1
## Converting a 3 class problem to a Binary Class problem for Detractors

Train2_multiclass$Detractor_class <-ifelse(Train2_multiclass$NPS_Status=="Det
ractor",1,0) table(Train2_multiclass$Detractor_class)
```

```
##    0    1
## 4487  502
```

## Converting survey questions to ordinal variables

```
cols_exclude <- c("CE_NPS","NPS_Status","AdmissionDate","DischargeDate",
                  "Estimatedcost","AgeYrs","HospitalNo2",
                  "LengthofStay","MaritalStatus","Sex",
                   "BedCategory","Department", "InsPayorcategory",
                   "State","Country","STATEZONE","Detractor_class")


## subsetting for only survey variables
survey_vars <- Train2_multiclass[, !(colnames(Train2_multiclass) %in% cols_ex
clude), drop                 = FALSE]

colnames(survey_vars)
```

```
##  [1] "CE_ACCESSIBILITY"               "CE_CSAT"
##  [3] "CE_VALUEFORMONEY"               "EM_IMMEDIATEATTENTION"
##  [5] "EM_NURSING"                     "EM_DOCTOR"
##  [7] "EM_OVERALL"                     "AD_TIME"
##  [9] "AD_TARRIFFPACKAGESEXPLAINATION" "AD_STAFFATTITUDE"
## [11] "INR_ROOMCLEANLINESS"            "INR_ROOMPEACE"
## [13] "INR_ROOMEQUIPMENT"              "INR_ROOMAMBIENCE"
## [15] "FNB_FOODQUALITY"                "FNB_FOODDELIVERYTIME"
## [17] "FNB_DIETICIAN"                  "FNB_STAFFATTITUDE"
## [19] "AE_ATTENDEECARE"                "AE_PATIENTSTATUSINFO"
## [21] "AE_ATTENDEEFOOD"                "DOC_TREATMENTEXPLAINATION"
## [23] "DOC_ATTITUDE"                   "DOC_VISITS"
## [25] "DOC_TREATMENTEFFECTIVENESS"     "NS_CALLBELLRESPONSE"
## [27] "NS_NURSESATTITUDE"              "NS_NURSEPROACTIVENESS"
## [29] "NS_NURSEPATIENCE"               "OVS_OVERALLSTAFFATTITUDE"
## [31] "OVS_OVERALLSTAFFPROMPTNESS"     "OVS_SECURITYATTITUDE"          ##
[33] "DP_DISCHARGETIME"               "DP_DISCHARGEQUERIES"           ##
[35] "DP_DISCHARGEPROCESS"
```

```
## converting survey variables to ordinal variables
ordinal_vars <- survey_vars %>%
  mutate_if(is.numeric,as.ordered)



cols_to_keep <- c("LengthofStay","MaritalStatus","Sex",
                  "BedCategory","Department", "InsPayorcategory",
                  "Estimatedcost","AgeYrs")

Train3_multiclass <- cbind(Train2_multiclass[,cols_to_keep],ordinal_vars,
                           Train2_multiclass["NPS_Status"])
```

## Prepping Test Data

```
Test2_multiclass <- Test_multiclass[,-1]

## Creating a 2 class variable for Detractors

Test2_multiclass$Detractor_class <- ifelse(Test2_multiclass$NPS_Status=="Detr
actor",1,0) table(Test2_multiclass$Detractor_class)

##
##   0   1
## 320  44

## subsetting for only survey variables survey_vars_test <-
Test2_multiclass[, !(colnames(Test2_multiclass) %in% cols _exclude),
drop = FALSE]

colnames(survey_vars_test)

##  [1] "CE_ACCESSIBILITY"                "CE_CSAT"
##  [3] "CE_VALUEFORMONEY"                "EM_IMMEDIATEATTENTION"
##  [5] "EM_NURSING"                      "EM_DOCTOR"
##  [7] "EM_OVERALL"                      "AD_TIME"
##  [9] "AD_TARRIFFPACKAGESEXPLAINATION" "AD_STAFFATTITUDE"
## [11] "INR_ROOMCLEANLINESS"             "INR_ROOMPEACE"
## [13] "INR_ROOMEQUIPMENT"               "INR_ROOMAMBIENCE"
## [15] "FNB_FOODQUALITY"                 "FNB_FOODDELIVERYTIME"
## [17] "FNB_DIETICIAN"                   "FNB_STAFFATTITUDE"
## [19] "AE_ATTENDEECARE"                 "AE_PATIENTSTATUSINFO"
## [21] "AE_ATTENDEEFOOD"                 "DOC_TREATMENTEXPLAINATION"
## [23] "DOC_ATTITUDE"                    "DOC_VISITS"
## [25] "DOC_TREATMENTEFFECTIVENESS"      "NS_CALLBELLRESPONSE"
## [27] "NS_NURSESATTITUDE"               "NS_NURSEPROACTIVENESS"
## [29] "NS_NURSEPATIENCE"                "OVS_OVERALLSTAFFATTITUDE"
## [31] "OVS_OVERALLSTAFFPROMPTNESS"      "OVS_SECURITYATTITUDE"          ##
[33] "DP_DISCHARGETIME"               "DP_DISCHARGEQUERIES"           ##
[35] "DP_DISCHARGEPROCESS"

## converting survey variables to ordinal variables
ordinal_vars_test <- survey_vars_test %>%
mutate_if(is.numeric,as.ordered)


dim(ordinal_vars)

## [1] 4989   35

## Multiclass dataset
Test3_multiclass <- cbind(Test2_multiclass[,cols_to_keep],ordinal_vars_test,
                          Test2_multiclass["NPS_Status"])
```

##Removing variables for Quasi_complete Seperation

```
colnames(Train3_multiclass)
##  [1] "LengthofStay"                    "MaritalStatus"
##  [3] "Sex"                             "BedCategory"
##  [5] "Department"                      "InsPayorcategory"
##  [7] "Estimatedcost"                   "AgeYrs"
##  [9] "CE_ACCESSIBILITY"                "CE_CSAT"
## [11] "CE_VALUEFORMONEY"                "EM_IMMEDIATEATTENTION"
## [13] "EM_NURSING"                      "EM_DOCTOR"
## [15] "EM_OVERALL"                      "AD_TIME"
## [17] "AD_TARRIFFPACKAGESEXPLAINATION" "AD_STAFFATTITUDE"
## [19] "INR_ROOMCLEANLINESS"             "INR_ROOMPEACE"
## [21] "INR_ROOMEQUIPMENT"               "INR_ROOMAMBIENCE"
## [23] "FNB_FOODQUALITY"                 "FNB_FOODDELIVERYTIME"
## [25] "FNB_DIETICIAN"                   "FNB_STAFFATTITUDE"
## [27] "AE_ATTENDEECARE"                 "AE_PATIENTSTATUSINFO"
## [29] "AE_ATTENDEEFOOD"                 "DOC_TREATMENTEXPLAINATION"
## [31] "DOC_ATTITUDE"                    "DOC_VISITS"
## [33] "DOC_TREATMENTEFFECTIVENESS"      "NS_CALLBELLRESPONSE"
## [35] "NS_NURSESATTITUDE"               "NS_NURSEPROACTIVENESS"
## [37] "NS_NURSEPATIENCE"                "OVS_OVERALLSTAFFATTITUDE"
## [39] "OVS_OVERALLSTAFFPROMPTNESS"      "OVS_SECURITYATTITUDE"          ##
[41] "DP_DISCHARGETIME"                "DP_DISCHARGEQUERIES"
## [43] "DP_DISCHARGEPROCESS"             "NPS_Status"

quasi_vars <- c("MaritalStatus","BedCategory","LengthofStay")

Train4_multiclass <- Train3_multiclass[, !(colnames(Train3_multiclass) %in%
quasi_vars),drop = FALSE]

Test4_multiclass <- Test3_multiclass[, !(colnames(Test3_multiclass) %in%
quasi_vars),drop = FALSE]
```

## Q 7

```
# Converting character columns to factor variables for Randomforest

class(Train4_multiclass$NPS_Status)

## [1] "character"

Train4_multiclass$NPS_Status <- as.factor(Train4_multiclass$NPS_Status)

Train5_multiclass <- Train4_multiclass %>%
mutate_if(is.character,as.factor)


# Converting character columns to factor variables for Randomforest
```

```r

Test4_multiclass$NPS_Status <- as.factor(Test4_multiclass$NPS_Status)

Test5_multiclass <- Test4_multiclass %>%
```

```
  mutate_if(is.character,as.factor)


## Adaboosting library(adabag)

## Loading required package: rpart

## Loading required package: caret

## Loading required package: lattice ##

Loading required package: ggplot2

##
## Attaching package: 'ggplot2'

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

## Loading required package: foreach

## Loading required package: doParallel

## Loading required package: iterators

## Loading required package: parallel

Test6_multiclass <- Test5_multiclass %>%
                    filter(NS_NURSEPROACTIVENESS!=1)


dim(Test5_multiclass)

## [1] 364  41

dim(Test6_multiclass)

## [1] 360  41

library(rpart)

maxdep <- c(1:5)
pr_val_err <- matrix()


for(i in maxdep){

    boost_model <- boosting(NPS_Status~., data = Train5_multiclass, boos = T,
mfinal = 100, coeflearn = "Breiman",
control=rpart.control(maxdepth=i))
```

```
    boost_pred <- predict.boosting(boost_model,newdata=Test6_multiclass)

    # pr_val_err[i] <- which.min(boost_pred$error)
pr_val_err[i] <- boost_pred$error
print(paste0(maxdep[i],",",pr_val_err[i]))


  }
```

```
## [1] "1,0.383333333333333"
## [1] "2,0.386111111111111"
## [1] "3,0.333333333333333"
## [1] "4,0.333333333333333" ##
[1] "5,0.336111111111111"
```

```
minsplit <- c(5,10,15)
pr_val_err2 <- matrix()

for(j in 1:length(minsplit)){

    boost_model <- boosting(NPS_Status~., data = Train5_multiclass, boos = T,
mfinal = 100, coeflearn = "Breiman",
                        control=rpart.control(maxdepth=4, minsplit = j))
        boost_pred <-
predict.boosting(boost_model,newdata=Test6_multiclass)

    # pr_val_err[i] <- which.min(boost_pred$error)
pr_val_err2[j] <- boost_pred$error
    print(paste0(minsplit[j],",",pr_val_err2[j]))


}
```

```
## [1] "5,0.336111111111111"
## [1] "10,0.330555555555556" ##
[1] "15,0.333333333333333"
```

```
trees<- c(100,200,300,400)
pr_val_err3 <- matrix()

for(k in 1:length(trees)){

    boost_model <- boosting(NPS_Status~., data = Train5_multiclass, boos = T,
mfinal = k, coeflearn = "Breiman",
                        control=rpart.control(maxdepth=4, minsplit = 10,cp =
0.01))
        boost_pred <-
predict.boosting(boost_model,newdata=Test6_multiclass)

    # pr_val_err[i] <- which.min(boost_pred$error)
```

```
    pr_val_err3[k] <- boost_pred$error
print(paste0(trees[k],",",pr_val_err3[k]))


  }
```

```
## [1] "100,0.341666666666667"
## [1] "200,0.363888888888889"
## [1] "300,0.333333333333333" ##
[1] "400,0.341666666666667"
```

```
set.seed(123)

boost_cv<- boosting.cv(NPS_Status~., data = Train5_multiclass, v = 5, boos =
TRUE,mfinal = 300, coeflearn = "Breiman",  control=rpart.control(maxdepth=4,
cp =0.01, minsplit =10), par=FALSE)
```

```
## i:  1 Wed Nov 27 02:46:28 2019
## i:  2 Wed Nov 27 02:50:22 2019
## i:  3 Wed Nov 27 02:53:45 2019
## i:  4 Wed Nov 27 02:57:15
```

```
2019  ## i:  5 Wed Nov 27
```

```
03:01:07 2019 boost_cv$error ##
```

```
[1] 0.3068751 boost_cv$confusion
```

```
##                Observed Class
## Predicted Class Detractor Passive Promotor
##       Detractor       215      97       23
##       Passive          80     282      156 ##
Promotor        207     968     2961
```

```
set.seed(123) boost2 <- boosting(NPS_Status~., data =
Train5_multiclass, boos = T,                        mfinal = 300,
coeflearn = "Breiman",
                  control=rpart.control(maxdepth=4, cp =0.01, minsplit =10))


adaboost_pred <- predict.boosting(boost2,newdata=Test6_multiclass) #
adaboost_pred

cf <- adaboost_pred$confusion cf
```

```
##                Observed Class
## Predicted Class Detractor Passive Promotor
##       Detractor        20      13        1
```

```
##       Passive           8      31       11 ##
Promotor         14      72      190

boost_err <- adaboost_pred$error

acc <-  1-boost_err acc

## [1] 0.6694444

sens_Class1 <- cf[1,1]/(cf[1,1]+cf[1,2]+cf[1,3])
sens_Class1 ## [1] 0.5882353

sens_Class2 <- cf[2,2]/(cf[2,1]+cf[2,2]+cf[2,3])
sens_Class2 ## [1] 0.62

sens_Class3 <- cf[3,3]/(cf[3,1]+cf[3,2]+cf[3,3]) sens_Class3

## [1] 0.6884058

## Undersampling library(caret)
under <- downSample(Train5_multiclass[,-41], Train5_multiclass$NPS_Status,

list = F, yname = "NPS_Status") class(under) ## [1] "data.frame"

dim(Train5_multiclass) ## [1] 4989   41 dim(under) ## [1] 1506   41

table(Train5_multiclass$NPS_Status)

##
## Detractor   Passive  Promotor  ##
502      1347      3140

table(under$NPS_Status)

##
## Detractor   Passive  Promotor
##       502       502       502
```

```
boost_under <- boosting(NPS_Status~., data = under, boos = T,
mfinal = 300, coeflearn = "Breiman",
                  control=rpart.control(maxdepth=4, cp =0.01, minsplit =10))


adaboost_pred_under <- predict.boosting(boost_under,newdata=Test6_multiclass)
# adaboost_pred

cf2 <- adaboost_pred_under$confusion cf2

##                Observed Class
## Predicted Class Detractor Passive Promotor
##       Detractor        22      14        2
##       Passive          15      65       65 ##
Promotor          5      37      135

boost_err2 <- adaboost_pred_under$error

acc2 <-  1-boost_err2 acc2

## [1] 0.6166667

sens_Class1 <- cf2[1,1]/(cf2[1,1]+cf2[1,2]+cf2[1,3])
sens_Class1 ## [1] 0.5789474

sens_Class2 <- cf2[2,2]/(cf2[2,1]+cf2[2,2]+cf2[2,3])
sens_Class2 ## [1] 0.4482759

sens_Class3 <- cf2[3,3]/(cf2[3,1]+cf2[3,2]+cf2[3,3]) sens_Class3

## [1] 0.7627119
```

## Oversampling

```
over <- upSample(Train5_multiclass[,-41], Train5_multiclass$NPS_Status,
list = F, yname = "NPS_Status") class(under) ## [1] "data.frame"
dim(Train5_multiclass) ## [1] 4989   41 dim(over)
```

```
## [1] 9420   41
```

```
table(Train5_multiclass$NPS_Status)
```

```
##
## Detractor   Passive  Promotor  ##
502      1347      3140
```

```
table(over$NPS_Status)
```

```
##
## Detractor   Passive  Promotor  ##
3140      3140      3140
```

```
boost_over <- boosting(NPS_Status~., data = over, boos = T,
mfinal = 300, coeflearn = "Breiman",
                  control=rpart.control(maxdepth=4, cp =0.01, minsplit =10))


adaboost_pred_over <- predict.boosting(boost_over,newdata=Test6_multiclass) #
adaboost_pred

cf3 <- adaboost_pred_over$confusion cf3

##                Observed Class
## Predicted Class Detractor Passive Promotor
##       Detractor        21      13        1
##       Passive          17      48       45 ##
Promotor          4      55      156

boost_err3 <- adaboost_pred_over$error

acc3 <-  1-boost_err3 acc3

## [1] 0.625

sens_Class1 <- cf3[1,1]/(cf3[1,1]+cf3[1,2]+cf3[1,3])
sens_Class1 ## [1] 0.6

sens_Class2 <- cf3[2,2]/(cf3[2,1]+cf3[2,2]+cf3[2,3])
sens_Class2 ## [1] 0.4363636

sens_Class3 <- cf3[3,3]/(cf3[3,1]+cf3[3,2]+cf3[3,3]) sens_Class3

## [1] 0.7255814
```

```
## SMOTE
```

```r
library(DMwR)


Train5_multiclass$NPS_Status <- factor(Train5_multiclass$NPS_Status)

data_smote <- DMwR::SMOTE(NPS_Status~., data = Train5_multiclass
                    , perc.over = 500, k =10,
                    perc.under = 100)


data_smote2 <- data_smote[complete.cases(data_smote), ]
dim(data_smote2)
```

```
## [1] 3040   41
```

```r
boost_smote <- boosting(NPS_Status~., data = data_smote2, boos = T,
                        mfinal = 300, coeflearn = "Breiman",
                    control=rpart.control(maxdepth=4, cp =0.01, minsplit =10))


adaboost_pred_smote <- predict.boosting(boost_smote,newdata=Test6_multiclass)
# adaboost_pred

cf4 <- adaboost_pred_smote$confusion
cf4
```

```
##                Observed Class
## Predicted Class Detractor Passive Promotor
##       Detractor        21      14        1
##       Passive           5      16        8
##       Promotor         16      86      193
```

```r
boost_err2 <- adaboost_pred_smote$error

acc4 <-  1-boost_err2
acc4
```

```
## [1] 0.6388889
```

```r
sens_Class1 <- cf4[1,1]/(cf4[1,1]+cf4[1,2]+cf4[1,3])
sens_Class1
```

```
## [1] 0.5833333
```

```r
sens_Class2 <- cf4[2,2]/(cf4[2,1]+cf4[2,2]+cf4[2,3])
sens_Class2
```

```
## [1] 0.5517241
```

```r
sens_Class3 <- cf4[3,3]/(cf4[3,1]+cf4[3,2]+cf4[3,3])
```

```
## [1] 0.6542373
```

**Q 8.**

By undersampling, we solved the class imbalance issue, and increased the sensitivity of our models. However, results were very poor. A reason could indeed be that we trained our classifiers using few samples. Results are better for oversampling.

For B**inary classification** the results obtained are as follows:

**Random Forest:**

| | | | |
|---|---|---|---|
| Accuracy | : 74.08% | Sensitivity obtained | :72% |
| Accuracy obtained when Undersampling | : 71.08% | Sensitivity obtained | :73.3% |
| Accuracy obtained when Oversampling | :74.5% | Sensitivity obtained | :71% |
| Accuracy obtained when SMOTE | :70% | Sensitivity obtained | :69.9% |

**Ada Boost:**

| | | | |
|---|---|---|---|
| Accuracy obtained | : 73.08%, | Sensitivity obtained | : 72% |
| Accuracy obtained when Undersampling | :70.9%, | Sensitivity obtained | :72% |
| Accuracy obtained when Oversampling | :70.4%, | Sensitivity obtained | :68% |
| Accuracy obtained when SMOTE | :67.88%, | Sensitivity obtained | :68% |

**For Multi Class data**, the results obtained are as follows

| | |
|---|---|
| **Random Forest** Accuracy | : 68% |
| Sensitivity | : Sensitivity for Class 1- 69%, Class2 – 58% and Class3 -70% |
| **Ada Boost** Accuracy | : 66% |
| Sensitivity | : Sensitivity for Class 1- 58%, Class2 – 62% and Class3 -68% |

For **AdaBoost,**

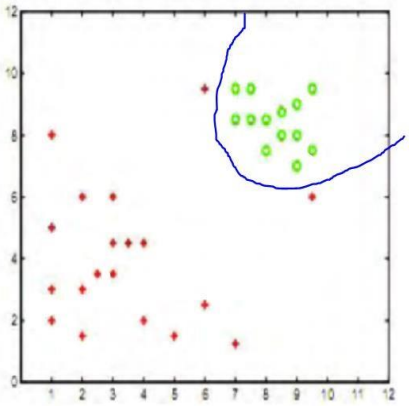
| | |
|---|---|
| Accuracy obtained when Undersampling | : 61% |
| Sensitivity obtained | : Sensitivity for Class 1- 61%, Class2 – 44% and Class3 -76% |
| Accuracy obtained when Oversampling | : 62.5% |
| Sensitivity obtained | : Sensitivity for Class 1- 60%, Class2 – 43% and Class3 -72% |
| Accuracy obtained when SMOTE | : 64% |
| Sensitivity obtained | : Sensitivity for Class 1- 58%, Class2 – 55% and Class3 -65% |

**Q 9.**

Identify the areas where Detractor customers were dissatisfied based on the scores provided in the survey and work toward improvement in those areas. If many Detractors unanimously provide negative feedback for a particular service, then maybe that service needs to be looked into. Offer promotions with highlights for the kind of Hospital services the Detractors are looking for.

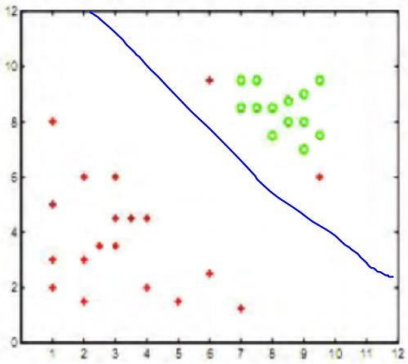
**PROBLEM 2:**

2.a)



Large values of C means cost of misclassification is high. So we try to reduce the errors by overfitting the data points. Hence the decision boundary looks like above.

2.b) The classifier can maximize the margin between most of the points, while misclassifying a few points, as the penalty is so low with C approximately equal to 0.



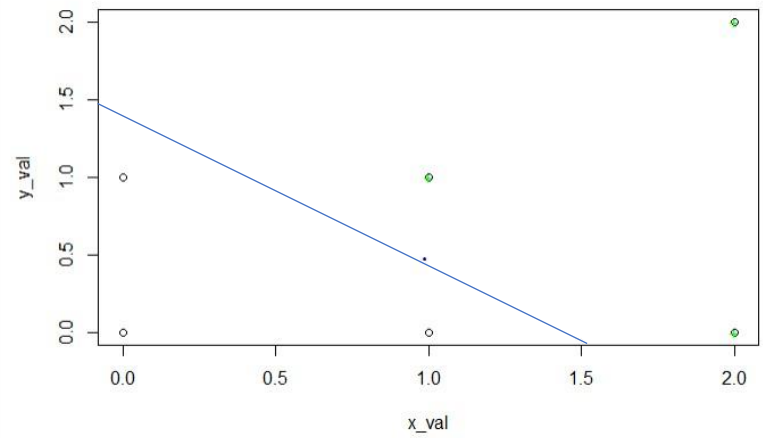
2.c)

The phenomenon is called overfitting. It fits properly for the train data and predictions on unseen data will can be highly varied. This overfitting model is high variance and low bias scenario which varies with each sample of data.

**PROBLEM 3:**

3.a) They are linearly separable.

```r
x_val=c(1,2,2,0,1,0)
y_val=c(1,2,0,0,0,1)
data=cbind(x_val,y_val)
plot(y_val ~x_val)
text(x = 1, y = 1, labels = c("+"), col = "Green")
text(x = 2, y = 2, labels = c("+"), col = "Green")
text(x = 2, y = 0, labels = c("+"), col = "Green")
```




3.b) (1,0), (0,1) , ( 1,1) , (2,0) are the support vectors.

yi (w1x1 + w2x2 + w3 ) = 1

This equation will pass through the support vectors

w1 (1) + w2(0) + w3 = -1

w1 (0) + w2(1) + w3= -1

w1(1) + w2(1) + w3 = 1

Solving these three equations, we get

W1 = 2, W2 = 2, w3 = -3

3.c) Removing the vectors, (1,0) and (1,1) will increase the optimal margin. It increases maximum distance of the support vector from the hyperplane. If we remove the support vectors (2,0) and (0,1) the optimal margin will remain same.