APPENDIX: CODE

Objective 1:

Program 1:

Code to create average of PANSS_Score and individual categories per VisitDay.

The code was run with 1st pass setting TxGroup as “Control” and then setting it as “Treatment”.

```
dataABCD = read.csv("C:/Users/msingh13/Documents/STAT202/project/Study_ABCD.csv", header = T)
p_scores = dataABCD[9:15]
n_scores = dataABCD[16:22]
g_scores = dataABCD[23:38]
total_p_scores = rowSums(p_scores)
total_n_scores = rowSums(n_scores)
total_g_scores = rowSums(g_scores)
dataABCD[["TotalPScore"]] = total_p_scores
dataABCD[["TotalNScore"]] = total_n_scores
dataABCD[["TotalGScore"]] = total_g_scores
composite_score = dataABCD[["TotalPScore"]] - dataABCD[["TotalNScore"]]
dataABCD[["comp_score"]] = composite_score
sorted_data = data.frame()

for (i in 0:max(dataABCD$VisitDay)) {
  count = 0

p1=p2=p3=p4=p5=p6=p7=n1=n2=n3=n4=n5=n6=n7=g1=g2=g3=g4=g5=g6=g7=g8=g9=g10=g11=g12=g13=g14=g15=g16=
p_total=n_total=g_total=comp_total=panss_total = 0

  for (j in 1:nrow(dataABCD)) {
      if (dataABCD[j, "VisitDay"] == i ) {
         #count = count + 1
         if(dataABCD[j,"TxGroup"] == "Control") {


           p1 = p1 + dataABCD[j,"P1"]
           p2 = p2 + dataABCD[j,"P2"]
           p3 = p3 + dataABCD[j,"P3"]
           p4 = p4 + dataABCD[j,"P4"]
           p5 = p5 + dataABCD[j,"P5"]
           p6 = p6 + dataABCD[j,"P6"]
           p7 = p7 + dataABCD[j,"P7"]
```

```
            n1 = n1 + dataABCD[j,"N1"]
            n2 = n2 + dataABCD[j,"N2"]
            n3 = n3 + dataABCD[j,"N3"]
            n4 = n4 + dataABCD[j,"N4"]
            n5 = n5 + dataABCD[j,"N5"]
            n6 = n6 + dataABCD[j,"N6"]
            n7 = n7 + dataABCD[j,"N7"]


            g1 = g1 + dataABCD[j,"G1"]
            g2 = g2 + dataABCD[j,"G2"]
            g3 = g3 + dataABCD[j,"G3"]
            g4 = g4 + dataABCD[j,"G4"]
            g5 = g5 + dataABCD[j,"G5"]
            g6 = g6 + dataABCD[j,"G6"]
            g7 = g7 + dataABCD[j,"G7"]
            g8 = g8 + dataABCD[j,"G8"]
            g9 = g9 + dataABCD[j,"G9"]
            g10 = g10 + dataABCD[j,"G10"]
            g11 = g11 + dataABCD[j,"G11"]
            g12 = g12 + dataABCD[j,"G12"]
            g13 = g13 + dataABCD[j,"G13"]
            g14 = g14 + dataABCD[j,"G14"]
            g15 = g15 + dataABCD[j,"G15"]
            g16 = g16 + dataABCD[j,"G16"]

            panss_total = panss_total + dataABCD[j,"PANSS_Total"]
            p_total =p_total + dataABCD[j,"TotalPScore"]
            n_total =n_total + dataABCD[j,"TotalNScore"]
            g_total =g_total + dataABCD[j,"TotalGScore"]
            comp_total = comp_total + dataABCD[j,"comp_score"]
            count = count + 1
          }

      }
     # cat("Iterating for Control Group,", j,"\n" )
}
if (count == 0) {
   count = 1
}


sorted_data[i+1,"VisitDay"] = i


for (k in 1:7) {
```

```
                a = paste("p", toString(k), sep = "")
                b = eval(parse(text=eval(parse(text= "a"))))
                c = paste("P", toString(k), sep = "")
                sorted_data[i+1,c] =  b/count
               }
   for (k in 1:7) {
                a = paste("n", toString(k), sep = "")
                b = eval(parse(text=eval(parse(text= "a"))))
                c = paste("N", toString(k), sep = "")
                sorted_data[i+1,c] = b/count
               }

   for (k in 1:16) {
                a = paste("g", toString(k), sep = "")
                b = eval(parse(text=eval(parse(text= "a"))))
                c = paste("G", toString(k), sep = "")
                sorted_data[i+1,c] = b/count
               }
   sorted_data[i+1,"PANSS_Total"] = panss_total/count
   sorted_data[i+1,"TotalPScore"] = p_total/count
   sorted_data[i+1,"TotalNScore"] = n_total/count
   sorted_data[i+1,"TotalGScore"] = g_total/count
   sorted_data[i+1,"CompScore"] = comp_total/count

   cat("Iteration Done for day ", i,"\n")

}
write.csv(sorted_data, file = "C:/Users/msingh13/Documents/STAT202/project/output_object1_Control_data.csv")
```

PROGRAM 2

Create PANSS_Total density plot for the given data

```
library(ggplot2)
library(ggpubr)
theme_set(theme_pubr())

data =
read.csv("C:/Users/msingh13/Documents/STAT202/Class_Project/output_objective1_Treatment_and_Control_data.csv"
, header = T)
 ggdensity(data, x = "PANSS_Total",   add = "mean", rug = TRUE,   color = "TxGroup",    palette = c("#0073C2FF",
"#FC4E07"))
 ggdensity(data, x = "TotalPScore",   add = "mean", rug = TRUE,   color = "TxGroup",    palette = c("#0073C2FF",
"#FC4E07"))
```

```
 ggdensity(data, x = "TotalNScore",   add = "mean", rug = TRUE,   color = "TxGroup",    palette = c("#0073C2FF", "#FC4E07"))
 ggdensity(data, x = "TotalGScore",   add = "mean", rug = TRUE,   color = "TxGroup",    palette = c("#0073C2FF", "#FC4E07"))
 ggdensity(data, x = "CompScore",   add = "mean", rug = TRUE,   color = "TxGroup",    palette = c("#0073C2FF", "#FC4E07"))

#To Plot QQ
ggqqplot(data, x = "PANSS_Total",   color = "TxGroup",    palette = c("#0073C2FF", "#FC4E07"),   ggtheme = theme_pubclean())
```

PROGRAM 3

```
data_Treatment_Control = read.csv("C:/Users/msingh13/Documents/STAT202/Class_Project/Treatment_and_Control_PANSS_Total_data.csv",header = T)
lm_fit = lm(data_Treatment_Control$PANSS_Total.Treatment ~ ., data=data_Treatment_Control)
summary(lm_fit)
```

OBJECTIVE 2

PROGRAM 1

```
data_obj2 = read.csv("C:/Users/msingh13/Documents/STAT202/Class_Project/Study_ABCD_For_Objective2.csv", header = T)
p_scores = data_obj2[9:15]
n_scores = data_obj2[16:22]
g_scores = data_obj2[23:38]
total_p_scores = rowSums(p_scores)
total_n_scores = rowSums(n_scores)
total_g_scores = rowSums(g_scores)
data_obj2[["TotalPScore"]] = total_p_scores
data_obj2[["TotalNScore"]] = total_n_scores
data_obj2[["TotalGScore"]] = total_g_scores
#composite_score = data_obj2[["TotalPScore"]] - data_obj2[["TotalNScore"]]
data_obj2$Country = as.factor(data_obj2$Country)
data_obj2$PatientID = as.factor(data_obj2$PatientID)
data_obj2$TxGroup = as.factor(data_obj2$TxGroup)
data_obj2$LeadStatus = as.factor(data_obj2$LeadStatus)
#scale the Total P, N & G scores
library(factoextra)
library(NbClust)
res = NbClust(data_obj2[,c(41:43)], diss = NULL, distance = "euclidean", min.nc = 2, max.nc = 10, method = "kmeans")
# Run kmeans with 2 clusters and nstart as 20.
km.out =kmeans(data_obj2[,c(41:43)],2, nstart =20)
```

```
#Plot cluster output OBJECTIVE2 FIG1
plot3d(data_obj2[,c(41:43)], col =(km.out$cluster +1) , main="K-Means Clustering")
#Plot 2D Output
plot(data_obj2[,c(41:43)], col =(km.out$cluster +1) , main="K-Means Clustering")

#Iterations to identify ‘nstart’ value that gives total within cluster sum of squares.
 km.out$tot.withinss
#[1] 145001
 km.out =kmeans(data_obj2[,c(41:43)],2, nstart =50)
 km.out$tot.withinss
#[1] 145001
 km.out =kmeans(data_obj2[,c(41:43)],2, nstart =10)
 km.out$tot.withinss
#[1] 145001
 km.out =kmeans(data_obj2[,c(41:43)],2, nstart =1)
 km.out$tot.withinss
#[1] 145001
 km.out =kmeans(data_obj2[,c(41:43)],2, nstart =100)
 km.out$tot.withinss
#[1] 14500

# kmeans with cluster size of 3
km.out =kmeans(data_obj2[,c(41:43)],3, nstart =50)
#Plot 2D
plot(data_obj2[,c(41:43)], col =(km.out$cluster +1) , main="K-Means Clustering")


#Hierarchical clustering – Complete linkage
hc.complete =hclust (dist(data_obj2[,c(41:43)]), method ="complete")
plot(hc.complete , main=" Complete Linkage ", xlab="", sub ="",cex =.9)
```

OBJECTIVE 3

#PROGRAM 1

```
######### INPUT DATA TRANSFORMATION##############
#For recreation:
#Create a copy of Study_ABCD.csv which is merge of all study data from A through D.
#Keep it in working directory. Also have a copy of sample PANSS to get patient ids from it.
setwd("C:/Users/msingh13/Documents/STAT202/Class_Project/R_working_dir")
set.seed(1)
dataABCD = read.csv("Study_ABCD.csv", header = T)
```

```
data_to_analyze = dataABCD[,-c(1,4:6)]
#Eliminate entries with ERROR country
data_to_analyze = data_to_analyze[!data_to_analyze$Country=="ERROR",]

#For model creation we can omit entries with Lead Status that are not Passed
data_to_analyze = data_to_analyze[!data_to_analyze$LeadStatus=="Assign to CS",]
data_to_analyze = data_to_analyze[!data_to_analyze$LeadStatus=="Flagged",]
data_to_analyze = data_to_analyze[,-c(36)]
data_to_analyze = data_to_analyze[,-c(5:34)]

#*******************

new_abcd = data.frame()
pat_id_in_study =  unique(data_to_analyze$PatientID, incomparables = FALSE)
for (i in pat_id_in_study) {
  count = 0
  temp_array = data.frame()
  temp_array = data_to_analyze[data_to_analyze$PatientID == toString(i),]
  temp_array = temp_array[order(temp_array$VisitDay),]
  temp_array$VisitDay = as.numeric(as.character(temp_array$VisitDay))
  temp_array$PANSS_Total = as.numeric(as.character(temp_array$PANSS_Total))
  visit_number = 1
  visit_week = 1
  previous_visit_panss = 0
  for (j in 1:nrow(temp_array)) {

          new_abcd[nrow(new_abcd)+1,"VisitNumber"] = visit_number
          visit_number = visit_number + 1
          new_abcd[nrow(new_abcd),"VisitWeek"] = strtoi(temp_array[j,"VisitDay"])/7



          if(j>1){
          new_abcd[nrow(new_abcd),"PreviousVisitPANSS"] = temp_array[j-1,"PANSS_Total"]
          new_abcd[nrow(new_abcd),"PreviousVisitDay"] = temp_array[j-1,"VisitDay"]
          }
          if(j==1){
          new_abcd[nrow(new_abcd),"PreviousVisitPANSS"] = temp_array[j,"PANSS_Total"]
          new_abcd[nrow(new_abcd),"PreviousVisitDay"] = temp_array[j,"VisitDay"]
          }

          if(j<nrow(temp_array)){
          new_abcd[nrow(new_abcd),"NextVisitPANSS"] = temp_array[j+1,"PANSS_Total"]
          new_abcd[nrow(new_abcd),"NextVisitDay"] = temp_array[j+1,"VisitDay"]
          }
          if(j==nrow(temp_array)){
          new_abcd[nrow(new_abcd),"NextVisitPANSS"] = temp_array[j,"PANSS_Total"]
```

```
    new_abcd[nrow(new_abcd),"NextVisitDay"] = temp_array[j,"VisitDay"]
    }

    if(j<(nrow(temp_array)-1)){
    new_abcd[nrow(new_abcd),"Next2ndVisitPANSS"] = temp_array[j+2,"PANSS_Total"]
    new_abcd[nrow(new_abcd),"Next2ndVisitDay"] = temp_array[j+2,"VisitDay"]
    }else{
    new_abcd[nrow(new_abcd),"Next2ndVisitPANSS"] = temp_array[j,"PANSS_Total"]
    new_abcd[nrow(new_abcd),"Next2ndVisitDay"] = temp_array[j,"VisitDay"]
    }



    if(j>1){
    new_abcd[nrow(new_abcd),"DaysFromPreviousVisit"] = temp_array[j,"VisitDay"] -
new_abcd[nrow(new_abcd),"PreviousVisitDay"]
    new_abcd[nrow(new_abcd),"LastVisitPANSSDiff"] =  new_abcd[nrow(new_abcd),"PreviousVisitPANSS"] -
temp_array[j,"PANSS_Total"]
    }
    if(j==1){
      new_abcd[nrow(new_abcd),"DaysFromPreviousVisit"] = 0
      new_abcd[nrow(new_abcd),"LastVisitPANSSDiff"] = 0
    }

    if(j<nrow(temp_array)){
     new_abcd[nrow(new_abcd),"DaysToNextVisit"] = temp_array[j,"VisitDay"] - temp_array[j+1,"VisitDay"]
     new_abcd[nrow(new_abcd),"NextVisitPANSSDiff"] =  temp_array[j,"PANSS_Total"] -
temp_array[j+1,"PANSS_Total"]
    }
    if(j==nrow(temp_array)){
     new_abcd[nrow(new_abcd),"DaysToNextVisit"] = 0
     new_abcd[nrow(new_abcd),"NextVisitPANSSDiff"] =  0
    }
    new_abcd[nrow(new_abcd),"PatientMinPANSS"] = min(temp_array$PANSS_Total)
    new_abcd[nrow(new_abcd),"PatientMaxPANSS"] = max(temp_array$PANSS_Total)
    new_abcd[nrow(new_abcd),"PatientAveragePANSS"] = mean(temp_array$PANSS_Total)

    if(j>2){
    new_abcd[nrow(new_abcd),"SecondLastVisitPANSS"] = temp_array[j-2,"PANSS_Total"]
    new_abcd[nrow(new_abcd),"SecondLastVisitDay"] = temp_array[j-2,"VisitDay"]
    }
    if(j<=2){
    new_abcd[nrow(new_abcd),"SecondLastVisitPANSS"] = temp_array[j,"PANSS_Total"]
    new_abcd[nrow(new_abcd),"SecondLastVisitDay"] = temp_array[j,"VisitDay"]
    }
```

```
                new_abcd[nrow(new_abcd),"PANSSDiffWRTFirstDay"] = temp_array[1,"PANSS_Total"] -
temp_array[j,"PANSS_Total"]
                new_abcd[nrow(new_abcd),"TxGroup"]=temp_array[j,"TxGroup"]
                new_abcd[nrow(new_abcd),"Country"]=temp_array[j,"Country"]
                new_abcd[nrow(new_abcd),"PANSS_Total"]=temp_array[j,"PANSS_Total"]
                new_abcd[nrow(new_abcd),"VisitDay"]=temp_array[j,"VisitDay"]

   }

}

write.csv(new_abcd,file="transformed_data_ABCD_for_LM.csv", row.names=FALSE, na="")
```

#PROGRAM2

```
############Creating Model To Predict PANSS_Total###############
transformed_data = read.csv("transformed_data_ABCD_for_LM.csv", header = T)

transformed_data$VisitNumber = as.numeric(as.character(transformed_data$VisitNumber))
#transformed_data$VisitNumber = as.factor(as.character(transformed_data$VisitNumber))
transformed_data$TxGroup = as.factor(as.character(transformed_data$TxGroup))
transformed_data$Country = as.factor(as.character(transformed_data$Country))
transformed_data$VisitWeek = as.numeric(as.character(transformed_data$VisitWeek))
transformed_data$PatientAveragePANSS  = as.numeric(as.character(transformed_data$PatientAveragePANSS ))
transformed_data$PreviousVisitPANSS = as.numeric(as.character(transformed_data$PreviousVisitPANSS))
transformed_data$PreviousVisitDay = as.numeric(as.character(transformed_data$PreviousVisitDay))
transformed_data$NextVisitPANSS = as.numeric(as.character(transformed_data$NextVisitPANSS))
transformed_data$NextVisitDay = as.numeric(as.character(transformed_data$NextVisitDay))
transformed_data$DaysFromPreviousVisit= as.numeric(as.character(transformed_data$DaysFromPreviousVisit))
transformed_data$LastVisitPANSSDiff = as.numeric(as.character(transformed_data$LastVisitPANSSDiff))
transformed_data$DaysToNextVisit = as.numeric(as.character(transformed_data$DaysToNextVisit))
transformed_data$NextVisitPANSSDiff = as.numeric(as.character(transformed_data$NextVisitPANSSDiff))
transformed_data$PatientMinPANSS = as.numeric(as.character(transformed_data$PatientMinPANSS))
transformed_data$PatientMaxPANSS = as.numeric(as.character(transformed_data$PatientMaxPANSS))
transformed_data$SecondLastVisitPANSS = as.numeric(as.character(transformed_data$SecondLastVisitPANSS))
transformed_data$SecondLastVisitDay = as.numeric(as.character(transformed_data$SecondLastVisitDay))
transformed_data$PANSSDiffWRTFirstDay =as.numeric(as.character(transformed_data$PANSSDiffWRTFirstDay))
#levels(transformed_data$VisitNumber) = c(levels(transformed_data$VisitNumber),"23")
#levels(transformed_data$VisitNumber) = c(levels(transformed_data$VisitNumber),"24")
#levels(transformed_data$VisitNumber) = c(levels(transformed_data$VisitNumber),"25")

panss_total_train_mse = panss_total_test_mse = 0
train = 1:(round(nrow(transformed_data)*0.90))
#train = 1:(round(nrow(transformed_data)))
input_data.Train = data.frame()
input_data.Test = data.frame()
input_data.Train = transformed_data[train,]
input_data.Test = transformed_data[-train,]

#library(randomForest)
predictors_num = ncol(input_data.Train) - 1

#Bagging where the mtry is number of predictors
#model_for_panss_total = randomForest(PANSS_Total ~ .,data=input_data.Train,
mtry=predictors_num,ntree=400,importance=TRUE)

#For random forest the mtry is predictors/3
#model_for_panss_total = randomForest(PANSS_Total ~ .,data=input_data.Train, mtry=6,importance=TRUE)

#Linear Model
model_for_panss_total = lm(PANSS_Total ~ . + (VisitWeek:.) + (I(log(VisitWeek+1)):.) + (I((VisitWeek+1)^0.1):.),
data=input_data.Train)
save(model_for_panss_total,file="panss_total_LM_model.rda")

predicted_panss_Train = predict(model_for_panss_total,input_data.Train)
panss_total_train_mse = mean((input_data.Train$PANSS_Total-predicted_panss_Train)^2)
cat("TRAIN MSE",panss_total_train_mse,"\n")
predicted_panss_Test = predict(model_for_panss_total,input_data.Test)
panss_total_test_mse = mean((input_data.Test$PANSS_Total-predicted_panss_Test)^2)
cat("TEST MSE",panss_total_test_mse,"\n")

#plot(model_for_panss_total)
#detach(input_data.Train)

#TEST MSE 8.704427e-25
#TRAIN MSE 8.551817e-25


#PROGRAM3

############Study E Data Transformation###############

input_studyE = read.csv("Study_E.csv", header = T)

#Getting the patientIDs for interested patients
sample_sub_panss = read.csv("sample_submission_PANSS.csv", header = T)
pat_id_in_studyE =  unique(sample_sub_panss$PatientID, incomparables = FALSE)

#Interpreting UK as France in the given data.
#There is no training data for country UK so interpreting it as France as they are geographically close countries.
levels(input_studyE$Country) = c(levels(input_studyE$Country),"France")
input_studyE$Country[input_studyE$Country=="UK"] = "France"
```

```
new_e = data.frame()
for (i in pat_id_in_studyE) {
   temp_array = data.frame()
   temp_array = input_studyE[input_studyE$PatientID == toString(i),]
   temp_array = temp_array[order(temp_array$VisitDay),]
   temp_array$VisitDay = as.numeric(as.character(temp_array$VisitDay))
   temp_array$PANSS_Total = as.numeric(as.character(temp_array$PANSS_Total))
   visit_number = 1
   visit_week = 1
   previous_visit_panss = 0
   for (j in 1:nrow(temp_array)) {

            new_e[nrow(new_e)+1,"VisitNumber"] = visit_number
            visit_number = visit_number + 1
            new_e[nrow(new_e),"VisitWeek"] = strtoi(temp_array[j,"VisitDay"])/7

            if(j>1){
            new_e[nrow(new_e),"PreviousVisitPANSS"] = temp_array[j-1,"PANSS_Total"]
            new_e[nrow(new_e),"PreviousVisitDay"] = temp_array[j-1,"VisitDay"]
            }
            if(j==1){
            new_e[nrow(new_e),"PreviousVisitPANSS"] = temp_array[j,"PANSS_Total"]
            new_e[nrow(new_e),"PreviousVisitDay"] = temp_array[j,"VisitDay"]
            }

            if(j<nrow(temp_array)){
            new_e[nrow(new_e),"NextVisitPANSS"] = temp_array[j+1,"PANSS_Total"]
            new_e[nrow(new_e),"NextVisitDay"] = temp_array[j+1,"VisitDay"]
            }
            if(j==nrow(temp_array)){
            new_e[nrow(new_e),"NextVisitPANSS"] = temp_array[j,"PANSS_Total"]
            new_e[nrow(new_e),"NextVisitDay"] = temp_array[j,"VisitDay"]
            }

            if(j<(nrow(temp_array)-1)){
            new_e[nrow(new_e),"Next2ndVisitPANSS"] = temp_array[j+2,"PANSS_Total"]
            new_e[nrow(new_e),"Next2ndVisitDay"] = temp_array[j+2,"VisitDay"]
            }else{
            new_e[nrow(new_e),"Next2ndVisitPANSS"] = temp_array[j,"PANSS_Total"]
            new_e[nrow(new_e),"Next2ndVisitDay"] = temp_array[j,"VisitDay"]
            }


            if(j>1){
```

```
        new_e[nrow(new_e),"DaysFromPreviousVisit"] = temp_array[j,"VisitDay"] -
new_e[nrow(new_e),"PreviousVisitDay"]
        new_e[nrow(new_e),"LastVisitPANSSDiff"] =  new_e[nrow(new_e),"PreviousVisitPANSS"] -
temp_array[j,"PANSS_Total"]
        }
        if(j==1){
          new_e[nrow(new_e),"DaysFromPreviousVisit"] = 0
          new_e[nrow(new_e),"LastVisitPANSSDiff"] = 0
        }

        if(j<nrow(temp_array)){
         new_e[nrow(new_e),"DaysToNextVisit"] = temp_array[j,"VisitDay"] - temp_array[j+1,"VisitDay"]
         new_e[nrow(new_e),"NextVisitPANSSDiff"] =  temp_array[j,"PANSS_Total"] - temp_array[j+1,"PANSS_Total"]
        }
        if(j==nrow(temp_array)){
         new_e[nrow(new_e),"DaysToNextVisit"] = 0
         new_e[nrow(new_e),"NextVisitPANSSDiff"] =  0
        }
        new_e[nrow(new_e),"PatientMinPANSS"] = min(temp_array$PANSS_Total)
        new_e[nrow(new_e),"PatientMaxPANSS"] = max(temp_array$PANSS_Total)
        new_e[nrow(new_e),"PatientAveragePANSS"] = mean(temp_array$PANSS_Total)

        if(j>2){
        new_e[nrow(new_e),"SecondLastVisitPANSS"] = temp_array[j-2,"PANSS_Total"]
        new_e[nrow(new_e),"SecondLastVisitDay"] = temp_array[j-2,"VisitDay"]
        }
        if(j<=2){
        new_e[nrow(new_e),"SecondLastVisitPANSS"] = temp_array[j,"PANSS_Total"]
        new_e[nrow(new_e),"SecondLastVisitDay"] = temp_array[j,"VisitDay"]
        }

        new_e[nrow(new_e),"PANSSDiffWRTFirstDay"] = temp_array[1,"PANSS_Total"] -
temp_array[j,"PANSS_Total"]
        new_e[nrow(new_e),"TxGroup"]=temp_array[j,"TxGroup"]
        new_e[nrow(new_e),"Country"]=temp_array[j,"Country"]
        new_e[nrow(new_e),"PANSS_Total"]=temp_array[j,"PANSS_Total"]
        new_e[nrow(new_e),"PatientID"]=temp_array[j,"PatientID"]
        new_e[nrow(new_e),"VisitDay"]=temp_array[j,"VisitDay"]


   }
}

write.csv(new_e,file="transformed_data_E_for_LM.csv", row.names=FALSE, na="")

new_e = read.csv("transformed_data_E_for_LM.csv", header = T)
```

```
data_to_predict_on = data.frame()
for (i in pat_id_in_studyE) {
   temp_array = data.frame()
   temp_array = new_e[new_e$PatientID == toString(i),]
   temp_array = temp_array[order(temp_array$VisitWeek),]
   temp_array$VisitWeek = as.numeric(as.character(temp_array$VisitWeek))
   temp_array$PANSS_Total = as.numeric(as.character(temp_array$PANSS_Total))
   data_not_present_for_week_intersted_in = 1
   week_interested_in = week_neg = week_pos = 18
   while (data_not_present_for_week_intersted_in == 1) {
     #cat("in while loop ",i,data_not_present_for_week_intersted_in,week_neg,"\n")
     for (j in 1:nrow(temp_array)) {
          if ((week_interested_in == round(temp_array[j, "VisitWeek"])) &&
(data_not_present_for_week_intersted_in==1)) {
              #data_to_predict_on[nrow(data_to_predict_on)+1,] = temp_array[c(j),]
              data_to_predict_on = rbind(data_to_predict_on,temp_array[c(j),])
              data_not_present_for_week_intersted_in = 0
              #cat("in cond 1\n")
              break
          }
          if ((round(temp_array[j, "VisitWeek"]) == week_pos) && (data_not_present_for_week_intersted_in==1)) {
              #data_to_predict_on[nrow(data_to_predict_on)+1,] = temp_array[c(j),]
              data_to_predict_on = rbind(data_to_predict_on,temp_array[c(j),])
              data_not_present_for_week_intersted_in = 0
              #cat("in cond 2\n")
              break
          }
          if ((round(temp_array[j, "VisitWeek"]) == week_neg) && (data_not_present_for_week_intersted_in==1)) {
              #data_to_predict_on[nrow(data_to_predict_on)+1,] = temp_array[c(j),]
              data_to_predict_on = rbind(data_to_predict_on,temp_array[c(j),])
              data_not_present_for_week_intersted_in = 0
              break
          }

     }
     #data_to_predict_on
     week_neg = week_neg - 1
     week_pos = week_pos + 1
   }
   #cat("out of while loop\n")


}
#data_to_predict_on = data_to_predict_on[,-c(20)]
data_to_predict_on$VisitWeek = 18
```

```
data_to_predict_on$VisitNumber = as.factor(as.character(data_to_predict_on$VisitNumber))
#data_to_predict_on$VisitNumber = as.numeric(as.character(data_to_predict_on$VisitNumber))
#data_to_predict_on$TxGroup = as.factor(as.character(data_to_predict_on$TxGroup))
#data_to_predict_on$Country = as.factor(as.character(data_to_predict_on$Country))
#data_to_predict_on$VisitWeek = as.numeric(as.character(data_to_predict_on$VisitWeek))
#data_to_predict_on$PatientAveragePANSS  = as.numeric(as.character(data_to_predict_on$PatientAveragePANSS ))
#data_to_predict_on$PreviousVisitPANSS = as.numeric(as.character(data_to_predict_on$PreviousVisitPANSS))
#data_to_predict_on$PreviousVisitDay = as.numeric(as.character(data_to_predict_on$PreviousVisitDay))
#data_to_predict_on$NextVisitPANSS = as.numeric(as.character(data_to_predict_on$NextVisitPANSS))
#data_to_predict_on$NextVisitDay = as.numeric(as.character(data_to_predict_on$NextVisitDay))
#data_to_predict_on$DaysFromPreviousVisit= as.numeric(as.character(data_to_predict_on$DaysFromPreviousVisit))
#data_to_predict_on$LastVisitPANSSDiff = as.numeric(as.character(data_to_predict_on$LastVisitPANSSDiff))
#data_to_predict_on$DaysToNextVisit = as.numeric(as.character(data_to_predict_on$DaysToNextVisit))
#data_to_predict_on$NextVisitPANSSDiff = as.numeric(as.character(data_to_predict_on$NextVisitPANSSDiff))
#data_to_predict_on$PatientMinPANSS = as.numeric(as.character(data_to_predict_on$PatientMinPANSS))
#data_to_predict_on$PatientMaxPANSS = as.numeric(as.character(data_to_predict_on$PatientMaxPANSS))
#data_to_predict_on$SecondLastVisitPANSS = as.numeric(as.character(data_to_predict_on$SecondLastVisitPANSS))
#data_to_predict_on$SecondLastVisitDay = as.numeric(as.character(data_to_predict_on$SecondLastVisitDay))
#data_to_predict_on$PANSSDiffWRTFirstDay =as.numeric(as.character(data_to_predict_on$PANSSDiffWRTFirstDay))
#data_to_predict_on$PANSS_Total =as.numeric(as.character(data_to_predict_on$PANSS_Total))

levels(data_to_predict_on$Country) = levels(input_data.Train$Country)
levels(data_to_predict_on$TxGroup) = levels(input_data.Train$TxGroup)
levels(data_to_predict_on$VisitNumber) = levels(input_data.Train$VisitNumber)

predicted_panss = predict(model_for_panss_total,newdata=data_to_predict_on)
data_to_predict_on$PANSS_Total_Predicted  = predicted_panss
panss_total_predicted_mse = mean((data_to_predict_on$PANSS_Total-
data_to_predict_on$PANSS_Total_Predicted)^2)

write.csv(data_to_predict_on,"objective3_output_complete_data_with_LM.csv",row.names=FALSE, na="")

final_output=data.frame()
for (i in pat_id_in_studyE) {
  temp_array = data.frame()
  temp_array = data_to_predict_on[data_to_predict_on$PatientID == toString(i),]
  for (k in 1:nrow(temp_array)) {
    if (strtoi(temp_array[k, "VisitWeek"]) == week_interested_in) {
        final_output[nrow(final_output)+1,"PatientID"]=i
        final_output[nrow(final_output),"PANSS_total"]=temp_array[k, "PANSS_Total_Predicted"]
        break
    }
  }
}
write.csv(final_output,"objective3_output_with_LM.csv",row.names=FALSE, na="")
```

```
cat("TRAIN MSE",panss_total_train_mse,"\n")
cat("TEST MSE",panss_total_test_mse,"\n")
cat("MSE on the Data To be predicted",panss_total_predicted_mse,"\n")
```

```
#> cat("TRAIN MSE",panss_total_train_mse,"\n")
#TRAIN MSE 0.0007197515
#> cat("TEST MSE",panss_total_test_mse,"\n")
#TEST MSE 0.0008652924
#> cat("MSE on the Data To be predicted",panss_total_predicted_mse,"\n")
#MSE on the Data To be predicted 0.04878473
```

PROGRAM4

```
#Complete program with bagging. Includes data transformation, model creation and data transformation before prediction and prediction.
########DATA TRANSFORMATION##############
#
setwd("C:/Users/msingh13/Documents/STAT202/Class_Project/R_working_dir")
set.seed(1)
dataABCD = read.csv("Study_ABCD.csv", header = T)

data_to_analyze = dataABCD[,-c(1,4:6)]
#Eliminate entries with ERROR country
data_to_analyze = data_to_analyze[!data_to_analyze$Country=="ERROR",]

#For model creation we can omit entries with Lead Status that are not Passed
data_to_analyze = data_to_analyze[!data_to_analyze$LeadStatus=="Assign to CS",]
data_to_analyze = data_to_analyze[!data_to_analyze$LeadStatus=="Flagged",]
data_to_analyze = data_to_analyze[,-c(36)]
data_to_analyze = data_to_analyze[,-c(5:34)]

#******************

new_abcd = data.frame()
pat_id_in_study =  unique(data_to_analyze$PatientID, incomparables = FALSE)
for (i in pat_id_in_study) {
   count = 0
   temp_array = data.frame()
   temp_array = data_to_analyze[data_to_analyze$PatientID == toString(i),]
   temp_array = temp_array[order(temp_array$VisitDay),]
   temp_array$VisitDay = as.numeric(as.character(temp_array$VisitDay))
   temp_array$PANSS_Total = as.numeric(as.character(temp_array$PANSS_Total))
   visit_number = 1
   visit_week = 1
   previous_visit_panss = 0
```

```
  for (j in 1:nrow(temp_array)) {

          new_abcd[nrow(new_abcd)+1,"VisitNumber"] = visit_number
          visit_number = visit_number + 1
          new_abcd[nrow(new_abcd),"VisitWeek"] = strtoi(temp_array[j,"VisitDay"])/7



          if(j>1){
          new_abcd[nrow(new_abcd),"PreviousVisitPANSS"] = temp_array[j-1,"PANSS_Total"]
          new_abcd[nrow(new_abcd),"PreviousVisitDay"] = temp_array[j-1,"VisitDay"]
          }
          if(j==1){
          new_abcd[nrow(new_abcd),"PreviousVisitPANSS"] = temp_array[j,"PANSS_Total"]
          new_abcd[nrow(new_abcd),"PreviousVisitDay"] = temp_array[j,"VisitDay"]
          }

          if(j<nrow(temp_array)){
          new_abcd[nrow(new_abcd),"NextVisitPANSS"] = temp_array[j+1,"PANSS_Total"]
          new_abcd[nrow(new_abcd),"NextVisitDay"] = temp_array[j+1,"VisitDay"]
          }
          if(j==nrow(temp_array)){
          new_abcd[nrow(new_abcd),"NextVisitPANSS"] = temp_array[j,"PANSS_Total"]
          new_abcd[nrow(new_abcd),"NextVisitDay"] = temp_array[j,"VisitDay"]
          }

          if(j>1){
          new_abcd[nrow(new_abcd),"DaysFromPreviousVisit"] = temp_array[j,"VisitDay"] -
new_abcd[nrow(new_abcd),"PreviousVisitDay"]
          new_abcd[nrow(new_abcd),"LastVisitPANSSDiff"] =  new_abcd[nrow(new_abcd),"PreviousVisitPANSS"] -
temp_array[j,"PANSS_Total"]
          }
          if(j==1){
            new_abcd[nrow(new_abcd),"DaysFromPreviousVisit"] = 0
            new_abcd[nrow(new_abcd),"LastVisitPANSSDiff"] = 0
          }

          if(j<nrow(temp_array)){
           new_abcd[nrow(new_abcd),"DaysToNextVisit"] = temp_array[j,"VisitDay"] - temp_array[j+1,"VisitDay"]
           new_abcd[nrow(new_abcd),"NextVisitPANSSDiff"] =  temp_array[j,"PANSS_Total"] -
temp_array[j+1,"PANSS_Total"]
          }
          if(j==nrow(temp_array)){
           new_abcd[nrow(new_abcd),"DaysToNextVisit"] = 0
           new_abcd[nrow(new_abcd),"NextVisitPANSSDiff"] =  0
          }
          new_abcd[nrow(new_abcd),"PatientMinPANSS"] = min(temp_array$PANSS_Total)
```

```
        new_abcd[nrow(new_abcd),"PatientMaxPANSS"] = max(temp_array$PANSS_Total)
        new_abcd[nrow(new_abcd),"PatientAveragePANSS"] = mean(temp_array$PANSS_Total)

        if(j>2){
        new_abcd[nrow(new_abcd),"SecondLastVisitPANSS"] = temp_array[j-2,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"SecondLastVisitDay"] = temp_array[j-2,"VisitDay"]
        }
        if(j<=2){
        new_abcd[nrow(new_abcd),"SecondLastVisitPANSS"] = temp_array[j,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"SecondLastVisitDay"] = temp_array[j,"VisitDay"]
        }

        new_abcd[nrow(new_abcd),"PANSSDiffWRTFirstDay"] = temp_array[1,"PANSS_Total"] -
temp_array[j,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"TxGroup"]=temp_array[j,"TxGroup"]
        new_abcd[nrow(new_abcd),"Country"]=temp_array[j,"Country"]
        new_abcd[nrow(new_abcd),"PANSS_Total"]=temp_array[j,"PANSS_Total"]
  }

}

write.csv(new_abcd,file="transformed_data_ABCD.csv", row.names=FALSE, na="")


############Creating Model To Predict PANSS_Total###############
transformed_data = read.csv("transformed_data_ABCD.csv", header = T)

#transformed_data$VisitNumber = as.numeric(as.character(transformed_data$VisitNumber))
transformed_data$VisitNumber = as.factor(as.character(transformed_data$VisitNumber))
transformed_data$TxGroup = as.factor(as.character(transformed_data$TxGroup))
transformed_data$Country = as.factor(as.character(transformed_data$Country))
transformed_data$VisitWeek = as.numeric(as.character(transformed_data$VisitWeek))
transformed_data$PatientAveragePANSS  = as.numeric(as.character(transformed_data$PatientAveragePANSS ))
transformed_data$PreviousVisitPANSS = as.numeric(as.character(transformed_data$PreviousVisitPANSS))
transformed_data$PreviousVisitDay = as.numeric(as.character(transformed_data$PreviousVisitDay))
transformed_data$NextVisitPANSS = as.numeric(as.character(transformed_data$NextVisitPANSS))
transformed_data$NextVisitDay = as.numeric(as.character(transformed_data$NextVisitDay))
transformed_data$DaysFromPreviousVisit= as.numeric(as.character(transformed_data$DaysFromPreviousVisit))
transformed_data$LastVisitPANSSDiff = as.numeric(as.character(transformed_data$LastVisitPANSSDiff))
transformed_data$DaysToNextVisit = as.numeric(as.character(transformed_data$DaysToNextVisit))
transformed_data$NextVisitPANSSDiff = as.numeric(as.character(transformed_data$NextVisitPANSSDiff))
transformed_data$PatientMinPANSS = as.numeric(as.character(transformed_data$PatientMinPANSS))
transformed_data$PatientMaxPANSS = as.numeric(as.character(transformed_data$PatientMaxPANSS))
transformed_data$SecondLastVisitPANSS = as.numeric(as.character(transformed_data$SecondLastVisitPANSS))
transformed_data$SecondLastVisitDay = as.numeric(as.character(transformed_data$SecondLastVisitDay))
transformed_data$PANSSDiffWRTFirstDay =as.numeric(as.character(transformed_data$PANSSDiffWRTFirstDay))
```

```
#transformed_data=transformed_data[,-c()]

panss_total_train_mse = panss_total_test_mse = 0
train = 1:(round(nrow(transformed_data)*0.90))
input_data.Train = data.frame()
input_data.Test = data.frame()
input_data.Train = transformed_data[train,]
input_data.Test = transformed_data[-train,]


library(randomForest)
predictors_num = ncol(input_data.Train) - 1

#Bagging where the mtry is number of predictors
model_for_panss_total = randomForest(PANSS_Total ~ .,data=input_data.Train,
mtry=predictors_num,ntree=400,importance=TRUE)

#For random forest the mtry is predictors/3
#model_for_panss_total = randomForest(PANSS_Total ~ .,data=input_data.Train, mtry=6,importance=TRUE)


predicted_panss_Train = predict(model_for_panss_total,input_data.Train)
panss_total_train_mse = mean((input_data.Train$PANSS_Total-predicted_panss_Train)^2)
cat("TRAIN MSE",panss_total_train_mse,"\n")
predicted_panss_Test = predict(model_for_panss_total,input_data.Test)
panss_total_test_mse = mean((input_data.Test$PANSS_Total-predicted_panss_Test)^2)
cat("TEST MSE",panss_total_test_mse,"\n")

#plots MSE wrt number of trees.
plot(model_for_panss_total)
#detach(input_data.Train)



############Study E Data Transformation##############

input_studyE = read.csv("Study_E.csv", header = T)

#Getting the patientIDs for interested patients
sample_sub_panss = read.csv("sample_submission_PANSS.csv", header = T)
pat_id_in_studyE =  unique(sample_sub_panss$PatientID, incomparables = FALSE)

#Interpreting UK as France in the given data.
#There is no training data for country UK so interpreting it as France as they are geographically close countries.
```

```
#levels(input_studyE$Country) = c(levels(input_studyE$Country),"France")
#input_studyE$Country[input_studyE$Country=="UK"] = "France"

new_e = data.frame()
for (i in pat_id_in_studyE) {
   temp_array = data.frame()
   temp_array = input_studyE[input_studyE$PatientID == toString(i),]
   temp_array = temp_array[order(temp_array$VisitDay),]
   temp_array$VisitDay = as.numeric(as.character(temp_array$VisitDay))
   temp_array$PANSS_Total = as.numeric(as.character(temp_array$PANSS_Total))
   visit_number = 1
   visit_week = 1
   previous_visit_panss = 0
   for (j in 1:nrow(temp_array)) {

          new_e[nrow(new_e)+1,"VisitNumber"] = visit_number
          visit_number = visit_number + 1
          new_e[nrow(new_e),"VisitWeek"] = strtoi(temp_array[j,"VisitDay"])/7

          if(j>1){
          new_e[nrow(new_e),"PreviousVisitPANSS"] = temp_array[j-1,"PANSS_Total"]
          new_e[nrow(new_e),"PreviousVisitDay"] = temp_array[j-1,"VisitDay"]
          }
          if(j==1){
          new_e[nrow(new_e),"PreviousVisitPANSS"] = temp_array[j,"PANSS_Total"]
          new_e[nrow(new_e),"PreviousVisitDay"] = temp_array[j,"VisitDay"]
          }

          if(j<nrow(temp_array)){
          new_e[nrow(new_e),"NextVisitPANSS"] = temp_array[j+1,"PANSS_Total"]
          new_e[nrow(new_e),"NextVisitDay"] = temp_array[j+1,"VisitDay"]
          }
          if(j==nrow(temp_array)){
          new_e[nrow(new_e),"NextVisitPANSS"] = temp_array[j,"PANSS_Total"]
          new_e[nrow(new_e),"NextVisitDay"] = temp_array[j,"VisitDay"]
          }

          if(j>1){
          new_e[nrow(new_e),"DaysFromPreviousVisit"] = temp_array[j,"VisitDay"] -
new_e[nrow(new_e),"PreviousVisitDay"]
          new_e[nrow(new_e),"LastVisitPANSSDiff"] =  new_e[nrow(new_e),"PreviousVisitPANSS"] -
temp_array[j,"PANSS_Total"]
          }
          if(j==1){
            new_e[nrow(new_e),"DaysFromPreviousVisit"] = 0
            new_e[nrow(new_e),"LastVisitPANSSDiff"] = 0
          }

          if(j<nrow(temp_array)){
           new_e[nrow(new_e),"DaysToNextVisit"] = temp_array[j,"VisitDay"] - temp_array[j+1,"VisitDay"]
           new_e[nrow(new_e),"NextVisitPANSSDiff"] =  temp_array[j,"PANSS_Total"] - temp_array[j+1,"PANSS_Total"]
          }
          if(j==nrow(temp_array)){
           new_e[nrow(new_e),"DaysToNextVisit"] = 0
           new_e[nrow(new_e),"NextVisitPANSSDiff"] =  0
          }
          new_e[nrow(new_e),"PatientMinPANSS"] = min(temp_array$PANSS_Total)
          new_e[nrow(new_e),"PatientMaxPANSS"] = max(temp_array$PANSS_Total)
          new_e[nrow(new_e),"PatientAveragePANSS"] = mean(temp_array$PANSS_Total)

          if(j>2){
          new_e[nrow(new_e),"SecondLastVisitPANSS"] = temp_array[j-2,"PANSS_Total"]
          new_e[nrow(new_e),"SecondLastVisitDay"] = temp_array[j-2,"VisitDay"]
          }
          if(j<=2){
          new_e[nrow(new_e),"SecondLastVisitPANSS"] = temp_array[j,"PANSS_Total"]
          new_e[nrow(new_e),"SecondLastVisitDay"] = temp_array[j,"VisitDay"]
          }

          new_e[nrow(new_e),"PANSSDiffWRTFirstDay"] = temp_array[1,"PANSS_Total"] -
temp_array[j,"PANSS_Total"]
          new_e[nrow(new_e),"TxGroup"]=temp_array[j,"TxGroup"]
          new_e[nrow(new_e),"Country"]=temp_array[j,"Country"]
          new_e[nrow(new_e),"PANSS_Total"]=temp_array[j,"PANSS_Total"]
          new_e[nrow(new_e),"PatientID"]=temp_array[j,"PatientID"]

   }
}

write.csv(new_e,file="transformed_data_E.csv", row.names=FALSE, na="")

new_e = read.csv("transformed_data_E.csv", header = T)

data_to_predict_on = data.frame()
for (i in pat_id_in_studyE) {
   temp_array = data.frame()
   temp_array = new_e[new_e$PatientID == toString(i),]
   temp_array = temp_array[order(temp_array$VisitWeek),]
   temp_array$VisitWeek = as.numeric(as.character(temp_array$VisitWeek))
   temp_array$PANSS_Total = as.numeric(as.character(temp_array$PANSS_Total))
   data_not_present_for_week_intersted_in = 1
   week_interested_in = week_neg = week_pos = 18
```

```
  while (data_not_present_for_week_intersted_in == 1) {
    #cat("in while loop ",i,data_not_present_for_week_intersted_in,week_neg,"\n")
    for (j in 1:nrow(temp_array)) {
        if ((week_interested_in == round(temp_array[j, "VisitWeek"])) &&
(data_not_present_for_week_intersted_in==1)) {
           #data_to_predict_on[nrow(data_to_predict_on)+1,] = temp_array[c(j),]
           data_to_predict_on = rbind(data_to_predict_on,temp_array[c(j),])
           data_not_present_for_week_intersted_in = 0
           #cat("in cond 1\n")
           break
        }
        if ((round(temp_array[j, "VisitWeek"]) == week_pos) && (data_not_present_for_week_intersted_in==1)) {
           #data_to_predict_on[nrow(data_to_predict_on)+1,] = temp_array[c(j),]
           data_to_predict_on = rbind(data_to_predict_on,temp_array[c(j),])
           data_not_present_for_week_intersted_in = 0
           #cat("in cond 2\n")
           break
        }
        if ((round(temp_array[j, "VisitWeek"]) == week_neg) && (data_not_present_for_week_intersted_in==1)) {
           #data_to_predict_on[nrow(data_to_predict_on)+1,] = temp_array[c(j),]
           data_to_predict_on = rbind(data_to_predict_on,temp_array[c(j),])
           data_not_present_for_week_intersted_in = 0
           break
        }

    }
    #data_to_predict_on
    week_neg = week_neg - 1
    week_pos = week_pos + 1
  }
  #cat("out of while loop\n")


}
#data_to_predict_on = data_to_predict_on[,-c(20)]
data_to_predict_on$VisitWeek = 18
data_to_predict_on$VisitNumber = as.factor(as.character(data_to_predict_on$VisitNumber))

levels(data_to_predict_on$Country) = levels(input_data.Train$Country)
levels(data_to_predict_on$TxGroup) = levels(input_data.Train$TxGroup)
levels(data_to_predict_on$VisitNumber) = levels(input_data.Train$VisitNumber)

predicted_panss = predict(model_for_panss_total,newdata=data_to_predict_on)
data_to_predict_on$PANSS_Total_Predicted  = predicted_panss
panss_total_predicted_mse = mean((data_to_predict_on$PANSS_Total-
data_to_predict_on$PANSS_Total_Predicted)^2)
```

```
write.csv(data_to_predict_on,"objective3_output_complete_data_with_bagging.csv",row.names=FALSE, na="")

final_output=data.frame()
for (i in pat_id_in_studyE) {
   temp_array = data.frame()
   temp_array = data_to_predict_on[data_to_predict_on$PatientID == toString(i),]
   for (k in 1:nrow(temp_array)) {
     if (strtoi(temp_array[k, "VisitWeek"]) == week_interested_in) {
          final_output[nrow(final_output)+1,"PatientID"]=i
          final_output[nrow(final_output),"PANSS_total"]=temp_array[k, "PANSS_Total_Predicted"]
          break
     }
   }
}
write.csv(final_output,"objective3_output_with_bagging.csv",row.names=FALSE, na="")
cat("TRAIN MSE with Bagging",panss_total_train_mse,"\n")
cat("TEST MSE with Bagging",panss_total_test_mse,"\n")
cat("MSE on the Data To be predicted on with Bagging",panss_total_predicted_mse,"\n")

#> cat("TRAIN MSE with Bagging",panss_total_train_mse,"\n")
#TRAIN MSE with Bagging 0.1182304
#> cat("TEST MSE with Bagging",panss_total_test_mse,"\n")
#TEST MSE with Bagging 1.34499
#> cat("MSE on the Data To be predicted on with Bagging",panss_total_predicted_mse,"\n")
#MSE on the Data To be predicted on with Bagging 2.323321
```

PROGRAM5

```
#Complete program for RandomForest. Includes data transformation, model creation and data transformation before prediction and prediction. Only the model liner is different(mtry = 18/3) in this program wrt PROGRAM4
#########DATA TRANSFORMATION###############
#
setwd("C:/Users/msingh13/Documents/STAT202/Class_Project/R_working_dir")
set.seed(1)
dataABCD = read.csv("Study_ABCD.csv", header = T)

data_to_analyze = dataABCD[,-c(1,4:6)]
#Eliminate entries with ERROR country
data_to_analyze = data_to_analyze[!data_to_analyze$Country=="ERROR",]

#For model creation we can omit entries with Lead Status that are not Passed
data_to_analyze = data_to_analyze[!data_to_analyze$LeadStatus=="Assign to CS",]
data_to_analyze = data_to_analyze[!data_to_analyze$LeadStatus=="Flagged",]
data_to_analyze = data_to_analyze[,-c(36)]
data_to_analyze = data_to_analyze[,-c(5:34)]
```

```
#*******************

new_abcd = data.frame()
pat_id_in_study =  unique(data_to_analyze$PatientID, incomparables = FALSE)
for (i in pat_id_in_study) {
   count = 0
   temp_array = data.frame()
   temp_array = data_to_analyze[data_to_analyze$PatientID == toString(i),]
   temp_array = temp_array[order(temp_array$VisitDay),]
   temp_array$VisitDay = as.numeric(as.character(temp_array$VisitDay))
   temp_array$PANSS_Total = as.numeric(as.character(temp_array$PANSS_Total))
   visit_number = 1
   visit_week = 1
   previous_visit_panss = 0
   for (j in 1:nrow(temp_array)) {

              new_abcd[nrow(new_abcd)+1,"VisitNumber"] = visit_number
              visit_number = visit_number + 1
              new_abcd[nrow(new_abcd),"VisitWeek"] = strtoi(temp_array[j,"VisitDay"])/7


              if(j>1){
              new_abcd[nrow(new_abcd),"PreviousVisitPANSS"] = temp_array[j-1,"PANSS_Total"]
              new_abcd[nrow(new_abcd),"PreviousVisitDay"] = temp_array[j-1,"VisitDay"]
              }
              if(j==1){
              new_abcd[nrow(new_abcd),"PreviousVisitPANSS"] = temp_array[j,"PANSS_Total"]
              new_abcd[nrow(new_abcd),"PreviousVisitDay"] = temp_array[j,"VisitDay"]
              }

              if(j<nrow(temp_array)){
              new_abcd[nrow(new_abcd),"NextVisitPANSS"] = temp_array[j+1,"PANSS_Total"]
              new_abcd[nrow(new_abcd),"NextVisitDay"] = temp_array[j+1,"VisitDay"]
              }
              if(j==nrow(temp_array)){
              new_abcd[nrow(new_abcd),"NextVisitPANSS"] = temp_array[j,"PANSS_Total"]
              new_abcd[nrow(new_abcd),"NextVisitDay"] = temp_array[j,"VisitDay"]
              }

              if(j>1){
              new_abcd[nrow(new_abcd),"DaysFromPreviousVisit"] = temp_array[j,"VisitDay"] - new_abcd[nrow(new_abcd),"PreviousVisitDay"]
              new_abcd[nrow(new_abcd),"LastVisitPANSSDiff"] =  new_abcd[nrow(new_abcd),"PreviousVisitPANSS"] - temp_array[j,"PANSS_Total"]
              }
```

```
        if(j==1){
          new_abcd[nrow(new_abcd),"DaysFromPreviousVisit"] = 0
          new_abcd[nrow(new_abcd),"LastVisitPANSSDiff"] = 0
        }

        if(j<nrow(temp_array)){
         new_abcd[nrow(new_abcd),"DaysToNextVisit"] = temp_array[j,"VisitDay"] - temp_array[j+1,"VisitDay"]
         new_abcd[nrow(new_abcd),"NextVisitPANSSDiff"] =  temp_array[j,"PANSS_Total"] -
temp_array[j+1,"PANSS_Total"]
        }
        if(j==nrow(temp_array)){
         new_abcd[nrow(new_abcd),"DaysToNextVisit"] = 0
         new_abcd[nrow(new_abcd),"NextVisitPANSSDiff"] =  0
        }
        new_abcd[nrow(new_abcd),"PatientMinPANSS"] = min(temp_array$PANSS_Total)
        new_abcd[nrow(new_abcd),"PatientMaxPANSS"] = max(temp_array$PANSS_Total)
        new_abcd[nrow(new_abcd),"PatientAveragePANSS"] = mean(temp_array$PANSS_Total)

        if(j>2){
        new_abcd[nrow(new_abcd),"SecondLastVisitPANSS"] = temp_array[j-2,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"SecondLastVisitDay"] = temp_array[j-2,"VisitDay"]
        }
        if(j<=2){
        new_abcd[nrow(new_abcd),"SecondLastVisitPANSS"] = temp_array[j,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"SecondLastVisitDay"] = temp_array[j,"VisitDay"]
        }

        new_abcd[nrow(new_abcd),"PANSSDiffWRTFirstDay"] = temp_array[1,"PANSS_Total"] -
temp_array[j,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"TxGroup"]=temp_array[j,"TxGroup"]
        new_abcd[nrow(new_abcd),"Country"]=temp_array[j,"Country"]
        new_abcd[nrow(new_abcd),"PANSS_Total"]=temp_array[j,"PANSS_Total"]
  }

}

write.csv(new_abcd,file="transformed_data_ABCD.csv", row.names=FALSE, na="")


############Creating Model To Predict PANSS_Total###############
transformed_data = read.csv("transformed_data_ABCD.csv", header = T)

#transformed_data$VisitNumber = as.numeric(as.character(transformed_data$VisitNumber))
transformed_data$VisitNumber = as.factor(as.character(transformed_data$VisitNumber))
transformed_data$TxGroup = as.factor(as.character(transformed_data$TxGroup))
transformed_data$Country = as.factor(as.character(transformed_data$Country))
```

```
transformed_data$VisitWeek = as.numeric(as.character(transformed_data$VisitWeek))
transformed_data$PatientAveragePANSS  = as.numeric(as.character(transformed_data$PatientAveragePANSS ))
transformed_data$PreviousVisitPANSS = as.numeric(as.character(transformed_data$PreviousVisitPANSS))
transformed_data$PreviousVisitDay = as.numeric(as.character(transformed_data$PreviousVisitDay))
transformed_data$NextVisitPANSS = as.numeric(as.character(transformed_data$NextVisitPANSS))
transformed_data$NextVisitDay = as.numeric(as.character(transformed_data$NextVisitDay))
transformed_data$DaysFromPreviousVisit= as.numeric(as.character(transformed_data$DaysFromPreviousVisit))
transformed_data$LastVisitPANSSDiff = as.numeric(as.character(transformed_data$LastVisitPANSSDiff))
transformed_data$DaysToNextVisit = as.numeric(as.character(transformed_data$DaysToNextVisit))
transformed_data$NextVisitPANSSDiff = as.numeric(as.character(transformed_data$NextVisitPANSSDiff))
transformed_data$PatientMinPANSS = as.numeric(as.character(transformed_data$PatientMinPANSS))
transformed_data$PatientMaxPANSS = as.numeric(as.character(transformed_data$PatientMaxPANSS))
transformed_data$SecondLastVisitPANSS = as.numeric(as.character(transformed_data$SecondLastVisitPANSS))
transformed_data$SecondLastVisitDay = as.numeric(as.character(transformed_data$SecondLastVisitDay))
transformed_data$PANSSDiffWRTFirstDay =as.numeric(as.character(transformed_data$PANSSDiffWRTFirstDay))


#transformed_data=transformed_data[,-c()]

panss_total_train_mse = panss_total_test_mse = 0
train = 1:(round(nrow(transformed_data)*0.90))
input_data.Train = data.frame()
input_data.Test = data.frame()
input_data.Train = transformed_data[train,]
input_data.Test = transformed_data[-train,]



library(randomForest)
predictors_num = ncol(input_data.Train) - 1

#Bagging where the mtry is number of predictors
#model_for_panss_total = randomForest(PANSS_Total ~ .,data=input_data.Train,
mtry=predictors_num,ntree=400,importance=TRUE)

#For random forest the mtry is predictors/3
model_for_panss_total = randomForest(PANSS_Total ~ .,data=input_data.Train, mtry=6,importance=TRUE)


predicted_panss_Train = predict(model_for_panss_total,input_data.Train)
panss_total_train_mse = mean((input_data.Train$PANSS_Total-predicted_panss_Train)^2)
cat("TRAIN MSE",panss_total_train_mse,"\n")
predicted_panss_Test = predict(model_for_panss_total,input_data.Test)
panss_total_test_mse = mean((input_data.Test$PANSS_Total-predicted_panss_Test)^2)
cat("TEST MSE",panss_total_test_mse,"\n")

plot(model_for_panss_total)
```

```
#detach(input_data.Train)



###########Study E Data Transformation##############

input_studyE = read.csv("Study_E.csv", header = T)

#Getting the patientIDs for interested patients
sample_sub_panss = read.csv("sample_submission_PANSS.csv", header = T)
pat_id_in_studyE =  unique(sample_sub_panss$PatientID, incomparables = FALSE)

#Interpreting UK as France in the given data.
#There is no training data for country UK so interpreting it as France as they are geographically close countries.
#levels(input_studyE$Country) = c(levels(input_studyE$Country),"France")
#input_studyE$Country[input_studyE$Country=="UK"] = "France"

new_e = data.frame()
for (i in pat_id_in_studyE) {
  temp_array = data.frame()
  temp_array = input_studyE[input_studyE$PatientID == toString(i),]
  temp_array = temp_array[order(temp_array$VisitDay),]
  temp_array$VisitDay = as.numeric(as.character(temp_array$VisitDay))
  temp_array$PANSS_Total = as.numeric(as.character(temp_array$PANSS_Total))
  visit_number = 1
  visit_week = 1
  previous_visit_panss = 0
  for (j in 1:nrow(temp_array)) {

            new_e[nrow(new_e)+1,"VisitNumber"] = visit_number
            visit_number = visit_number + 1
            new_e[nrow(new_e),"VisitWeek"] = strtoi(temp_array[j,"VisitDay"])/7

            if(j>1){
            new_e[nrow(new_e),"PreviousVisitPANSS"] = temp_array[j-1,"PANSS_Total"]
            new_e[nrow(new_e),"PreviousVisitDay"] = temp_array[j-1,"VisitDay"]
            }
            if(j==1){
            new_e[nrow(new_e),"PreviousVisitPANSS"] = temp_array[j,"PANSS_Total"]
            new_e[nrow(new_e),"PreviousVisitDay"] = temp_array[j,"VisitDay"]
            }

            if(j<nrow(temp_array)){
            new_e[nrow(new_e),"NextVisitPANSS"] = temp_array[j+1,"PANSS_Total"]
            new_e[nrow(new_e),"NextVisitDay"] = temp_array[j+1,"VisitDay"]
            }
            if(j==nrow(temp_array)){
            new_e[nrow(new_e),"NextVisitPANSS"] = temp_array[j,"PANSS_Total"]
            new_e[nrow(new_e),"NextVisitDay"] = temp_array[j,"VisitDay"]
            }

            if(j>1){
            new_e[nrow(new_e),"DaysFromPreviousVisit"] = temp_array[j,"VisitDay"] -
new_e[nrow(new_e),"PreviousVisitDay"]
            new_e[nrow(new_e),"LastVisitPANSSDiff"] =  new_e[nrow(new_e),"PreviousVisitPANSS"] -
temp_array[j,"PANSS_Total"]
            }
            if(j==1){
              new_e[nrow(new_e),"DaysFromPreviousVisit"] = 0
              new_e[nrow(new_e),"LastVisitPANSSDiff"] = 0
            }

            if(j<nrow(temp_array)){
             new_e[nrow(new_e),"DaysToNextVisit"] = temp_array[j,"VisitDay"] - temp_array[j+1,"VisitDay"]
             new_e[nrow(new_e),"NextVisitPANSSDiff"] =  temp_array[j,"PANSS_Total"] - temp_array[j+1,"PANSS_Total"]
            }
            if(j==nrow(temp_array)){
             new_e[nrow(new_e),"DaysToNextVisit"] = 0
             new_e[nrow(new_e),"NextVisitPANSSDiff"] =  0
            }
            new_e[nrow(new_e),"PatientMinPANSS"] = min(temp_array$PANSS_Total)
            new_e[nrow(new_e),"PatientMaxPANSS"] = max(temp_array$PANSS_Total)
            new_e[nrow(new_e),"PatientAveragePANSS"] = mean(temp_array$PANSS_Total)

            if(j>2){
            new_e[nrow(new_e),"SecondLastVisitPANSS"] = temp_array[j-2,"PANSS_Total"]
            new_e[nrow(new_e),"SecondLastVisitDay"] = temp_array[j-2,"VisitDay"]
            }
            if(j<=2){
            new_e[nrow(new_e),"SecondLastVisitPANSS"] = temp_array[j,"PANSS_Total"]
            new_e[nrow(new_e),"SecondLastVisitDay"] = temp_array[j,"VisitDay"]
            }

            new_e[nrow(new_e),"PANSSDiffWRTFirstDay"] = temp_array[1,"PANSS_Total"] -
temp_array[j,"PANSS_Total"]
            new_e[nrow(new_e),"TxGroup"]=temp_array[j,"TxGroup"]
            new_e[nrow(new_e),"Country"]=temp_array[j,"Country"]
            new_e[nrow(new_e),"PANSS_Total"]=temp_array[j,"PANSS_Total"]
            new_e[nrow(new_e),"PatientID"]=temp_array[j,"PatientID"]

   }
}
```

```
write.csv(new_e,file="transformed_data_E.csv", row.names=FALSE, na="")

new_e = read.csv("transformed_data_E.csv", header = T)

data_to_predict_on = data.frame()
for (i in pat_id_in_studyE) {
   temp_array = data.frame()
   temp_array = new_e[new_e$PatientID == toString(i),]
   temp_array = temp_array[order(temp_array$VisitWeek),]
   temp_array$VisitWeek = as.numeric(as.character(temp_array$VisitWeek))
   temp_array$PANSS_Total = as.numeric(as.character(temp_array$PANSS_Total))
   data_not_present_for_week_intersted_in = 1
   week_interested_in = week_neg = week_pos = 18
   while (data_not_present_for_week_intersted_in == 1) {
     #cat("in while loop ",i,data_not_present_for_week_intersted_in,week_neg,"\n")
     for (j in 1:nrow(temp_array)) {
         if ((week_interested_in == round(temp_array[j, "VisitWeek"])) &&
(data_not_present_for_week_intersted_in==1)) {
             #data_to_predict_on[nrow(data_to_predict_on)+1,] = temp_array[c(j),]
             data_to_predict_on = rbind(data_to_predict_on,temp_array[c(j),])
             data_not_present_for_week_intersted_in = 0
             #cat("in cond 1\n")
             break
         }
         if ((round(temp_array[j, "VisitWeek"]) == week_pos) && (data_not_present_for_week_intersted_in==1)) {
             #data_to_predict_on[nrow(data_to_predict_on)+1,] = temp_array[c(j),]
             data_to_predict_on = rbind(data_to_predict_on,temp_array[c(j),])
             data_not_present_for_week_intersted_in = 0
             #cat("in cond 2\n")
             break
         }
         if ((round(temp_array[j, "VisitWeek"]) == week_neg) && (data_not_present_for_week_intersted_in==1)) {
             #data_to_predict_on[nrow(data_to_predict_on)+1,] = temp_array[c(j),]
             data_to_predict_on = rbind(data_to_predict_on,temp_array[c(j),])
             data_not_present_for_week_intersted_in = 0
             break
         }

     }
     #data_to_predict_on
     week_neg = week_neg - 1
     week_pos = week_pos + 1
   }
   #cat("out of while loop\n")
}
#data_to_predict_on = data_to_predict_on[,-c(20)]
data_to_predict_on$VisitWeek = 18
data_to_predict_on$VisitNumber = as.factor(as.character(data_to_predict_on$VisitNumber))


levels(data_to_predict_on$Country) = levels(input_data.Train$Country)
levels(data_to_predict_on$TxGroup) = levels(input_data.Train$TxGroup)
levels(data_to_predict_on$VisitNumber) = levels(input_data.Train$VisitNumber)

predicted_panss = predict(model_for_panss_total,newdata=data_to_predict_on)
data_to_predict_on$PANSS_Total_Predicted  = predicted_panss
panss_total_predicted_mse = mean((data_to_predict_on$PANSS_Total-
data_to_predict_on$PANSS_Total_Predicted)^2)

write.csv(data_to_predict_on,"objective3_output_complete_data_with_RF.csv",row.names=FALSE, na="")

final_output=data.frame()
for (i in pat_id_in_studyE) {
  temp_array = data.frame()
  temp_array = data_to_predict_on[data_to_predict_on$PatientID == toString(i),]
  for (k in 1:nrow(temp_array)) {
    if (strtoi(temp_array[k, "VisitWeek"]) == week_interested_in) {
        final_output[nrow(final_output)+1,"PatientID"]=i
        final_output[nrow(final_output),"PANSS_total"]=temp_array[k, "PANSS_Total_Predicted"]
        break
    }
  }
}
write.csv(final_output,"objective3_output_with_RF.csv",row.names=FALSE, na="")
cat("TRAIN MSE with Random Forest",panss_total_train_mse,"\n")
cat("TEST MSE with Random Forest",panss_total_test_mse,"\n")
cat("MSE on the Data To be predicted on with Random Forest",panss_total_predicted_mse,"\n")

# cat("TRAIN MSE with Random Forest",panss_total_train_mse,"\n")
#TRAIN MSE with Random Forest 0.2747294
# cat("TEST MSE with Random Forest",panss_total_test_mse,"\n")
#TEST MSE with Random Forest 2.088028
# cat("MSE on the Data To be predicted on with Random Forest",panss_total_predicted_mse,"\n")
#MSE on the Data To be predicted on with Random Forest 3.272701
```

PROGRAM 6

#Boosting complete program with data transformation, model creation data transformation pre prediction and prediction.

```
#########DATA TRANSFORMATION###############
#
setwd("C:/Users/msingh13/Documents/STAT202/Class_Project/R_working_dir")
set.seed(1)
dataABCD = read.csv("Study_ABCD.csv", header = T)

data_to_analyze = dataABCD[,-c(1,4:6)]
#Eliminate entries with ERROR country
data_to_analyze = data_to_analyze[!data_to_analyze$Country=="ERROR",]

#For model creation we can omit entries with Lead Status that are not Passed
#data_to_analyze = data_to_analyze[!data_to_analyze$LeadStatus=="Assign to CS",]
#data_to_analyze = data_to_analyze[!data_to_analyze$LeadStatus=="Flagged",]
data_to_analyze = data_to_analyze[,-c(36)]
data_to_analyze = data_to_analyze[,-c(5:34)]

#******************

new_abcd = data.frame()
pat_id_in_study =  unique(data_to_analyze$PatientID, incomparables = FALSE)
for (i in pat_id_in_study) {
  count = 0
  temp_array = data.frame()
  temp_array = data_to_analyze[data_to_analyze$PatientID == toString(i),]
  temp_array = temp_array[order(temp_array$VisitDay),]
  temp_array$VisitDay = as.numeric(as.character(temp_array$VisitDay))
  temp_array$PANSS_Total = as.numeric(as.character(temp_array$PANSS_Total))
  visit_number = 1
  visit_week = 1
  previous_visit_panss = 0
  for (j in 1:nrow(temp_array)) {

         new_abcd[nrow(new_abcd)+1,"VisitNumber"] = visit_number
         visit_number = visit_number + 1
         new_abcd[nrow(new_abcd),"VisitWeek"] = strtoi(temp_array[j,"VisitDay"])/7


         if(j>1){
         new_abcd[nrow(new_abcd),"PreviousVisitPANSS"] = temp_array[j-1,"PANSS_Total"]
         new_abcd[nrow(new_abcd),"PreviousVisitDay"] = temp_array[j-1,"VisitDay"]
```

```
        }
        if(j==1){
        new_abcd[nrow(new_abcd),"PreviousVisitPANSS"] = temp_array[j,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"PreviousVisitDay"] = temp_array[j,"VisitDay"]
        }

        if(j<nrow(temp_array)){
        new_abcd[nrow(new_abcd),"NextVisitPANSS"] = temp_array[j+1,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"NextVisitDay"] = temp_array[j+1,"VisitDay"]
        }
        if(j==nrow(temp_array)){
        new_abcd[nrow(new_abcd),"NextVisitPANSS"] = temp_array[j,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"NextVisitDay"] = temp_array[j,"VisitDay"]
        }

        if(j<(nrow(temp_array)-1)){
        new_abcd[nrow(new_abcd),"Next2ndVisitPANSS"] = temp_array[j+2,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"Next2ndVisitDay"] = temp_array[j+2,"VisitDay"]
        }else{
        new_abcd[nrow(new_abcd),"Next2ndVisitPANSS"] = temp_array[j,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"Next2ndVisitDay"] = temp_array[j,"VisitDay"]
        }



        if(j>1){
        new_abcd[nrow(new_abcd),"DaysFromPreviousVisit"] = temp_array[j,"VisitDay"] -
new_abcd[nrow(new_abcd),"PreviousVisitDay"]
        new_abcd[nrow(new_abcd),"LastVisitPANSSDiff"] =  new_abcd[nrow(new_abcd),"PreviousVisitPANSS"] -
temp_array[j,"PANSS_Total"]
        }
        if(j==1){
          new_abcd[nrow(new_abcd),"DaysFromPreviousVisit"] = 0
          new_abcd[nrow(new_abcd),"LastVisitPANSSDiff"] = 0
        }

        if(j<nrow(temp_array)){
         new_abcd[nrow(new_abcd),"DaysToNextVisit"] = temp_array[j,"VisitDay"] - temp_array[j+1,"VisitDay"]
         new_abcd[nrow(new_abcd),"NextVisitPANSSDiff"] =  temp_array[j,"PANSS_Total"] -
temp_array[j+1,"PANSS_Total"]
        }
        if(j==nrow(temp_array)){
         new_abcd[nrow(new_abcd),"DaysToNextVisit"] = 0
         new_abcd[nrow(new_abcd),"NextVisitPANSSDiff"] =  0
        }
        new_abcd[nrow(new_abcd),"PatientMinPANSS"] = min(temp_array$PANSS_Total)
        new_abcd[nrow(new_abcd),"PatientMaxPANSS"] = max(temp_array$PANSS_Total)
        new_abcd[nrow(new_abcd),"PatientAveragePANSS"] = mean(temp_array$PANSS_Total)

        if(j>2){
        new_abcd[nrow(new_abcd),"SecondLastVisitPANSS"] = temp_array[j-2,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"SecondLastVisitDay"] = temp_array[j-2,"VisitDay"]
        }
        if(j<=2){
        new_abcd[nrow(new_abcd),"SecondLastVisitPANSS"] = temp_array[j,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"SecondLastVisitDay"] = temp_array[j,"VisitDay"]
        }

        new_abcd[nrow(new_abcd),"PANSSDiffWRTFirstDay"] = temp_array[1,"PANSS_Total"] -
temp_array[j,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"TxGroup"]=temp_array[j,"TxGroup"]
        new_abcd[nrow(new_abcd),"Country"]=temp_array[j,"Country"]
        new_abcd[nrow(new_abcd),"PANSS_Total"]=temp_array[j,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"VisitDay"]=temp_array[j,"VisitDay"]

   }

}

write.csv(new_abcd,file="transformed_data_ABCD.csv", row.names=FALSE, na="")



############Creating Model To Predict PANSS_Total###############
transformed_data = read.csv("transformed_data_ABCD.csv", header = T)

#transformed_data$VisitNumber = as.numeric(as.character(transformed_data$VisitNumber))
transformed_data$VisitNumber = as.factor(as.character(transformed_data$VisitNumber))
transformed_data$TxGroup = as.factor(as.character(transformed_data$TxGroup))
transformed_data$Country = as.factor(as.character(transformed_data$Country))
transformed_data$VisitWeek = as.numeric(as.character(transformed_data$VisitWeek))
transformed_data$PatientAveragePANSS  = as.numeric(as.character(transformed_data$PatientAveragePANSS ))
transformed_data$PreviousVisitPANSS = as.numeric(as.character(transformed_data$PreviousVisitPANSS))
transformed_data$PreviousVisitDay = as.numeric(as.character(transformed_data$PreviousVisitDay))
transformed_data$NextVisitPANSS = as.numeric(as.character(transformed_data$NextVisitPANSS))
transformed_data$NextVisitDay = as.numeric(as.character(transformed_data$NextVisitDay))
transformed_data$DaysFromPreviousVisit= as.numeric(as.character(transformed_data$DaysFromPreviousVisit))
transformed_data$LastVisitPANSSDiff = as.numeric(as.character(transformed_data$LastVisitPANSSDiff))
transformed_data$DaysToNextVisit = as.numeric(as.character(transformed_data$DaysToNextVisit))
transformed_data$NextVisitPANSSDiff = as.numeric(as.character(transformed_data$NextVisitPANSSDiff))
transformed_data$PatientMinPANSS = as.numeric(as.character(transformed_data$PatientMinPANSS))
transformed_data$PatientMaxPANSS = as.numeric(as.character(transformed_data$PatientMaxPANSS))
transformed_data$SecondLastVisitPANSS = as.numeric(as.character(transformed_data$SecondLastVisitPANSS))
transformed_data$SecondLastVisitDay = as.numeric(as.character(transformed_data$SecondLastVisitDay))
```

```
transformed_data$PANSSDiffWRTFirstDay =as.numeric(as.character(transformed_data$PANSSDiffWRTFirstDay))


#transformed_data=transformed_data[,-c()]

panss_total_train_mse = panss_total_test_mse = 0
train = 1:(round(nrow(transformed_data)*0.90))
input_data.Train = data.frame()
input_data.Test = data.frame()
input_data.Train = transformed_data[train,]
input_data.Test = transformed_data[-train,]



library(randomForest)
predictors_num = ncol(input_data.Train) - 1

#Bagging where the mtry is number of predictors
#model_for_panss_total = randomForest(PANSS_Total ~ .,data=input_data.Train,
mtry=predictors_num,ntree=400,importance=TRUE)

#For random forest the mtry is predictors/3
#model_for_panss_total = randomForest(PANSS_Total ~ .,data=input_data.Train, mtry=6,importance=TRUE)

library(gbm)
#Boosting
number_of_trees =4000
model_for_panss_total=gbm(PANSS_Total ~
.,data=input_data.Train,distribution="gaussian",n.trees=number_of_trees,interaction.depth=2,shrinkage=0.2511886)
#save(model_for_panss_total,file="panss_total_boosting_model.rda")

#Linear Model
#model_for_panss_total = lm(PANSS_Total ~ (VisitWeek:.) + (I(log(VisitWeek+1)):.) + (I((VisitWeek+1)^0.1):.),
data=input_data.Train)


predicted_panss_Train = predict(model_for_panss_total,input_data.Train,n.trees=number_of_trees)
panss_total_train_mse = mean((input_data.Train$PANSS_Total-predicted_panss_Train)^2)
cat("TRAIN MSE",panss_total_train_mse,"\n")
predicted_panss_Test = predict(model_for_panss_total,input_data.Test,n.trees=number_of_trees)
panss_total_test_mse = mean((input_data.Test$PANSS_Total-predicted_panss_Test)^2)
cat("TEST MSE",panss_total_test_mse,"\n")

#plot(model_for_panss_total)
#detach(input_data.Train)
```

```
############Study E Data Transformation###############

input_studyE = read.csv("Study_E.csv", header = T)

#Getting the patientIDs for interested patients
sample_sub_panss = read.csv("sample_submission_PANSS.csv", header = T)
pat_id_in_studyE =  unique(sample_sub_panss$PatientID, incomparables = FALSE)

#Interpreting UK as France in the given data.
#There is no training data for country UK so interpreting it as France as they are geographically close countries.
levels(input_studyE$Country) = c(levels(input_studyE$Country),"France")
input_studyE$Country[input_studyE$Country=="UK"] = "France"

new_e = data.frame()
for (i in pat_id_in_studyE) {
  temp_array = data.frame()
  temp_array = input_studyE[input_studyE$PatientID == toString(i),]
  temp_array = temp_array[order(temp_array$VisitDay),]
  temp_array$VisitDay = as.numeric(as.character(temp_array$VisitDay))
  temp_array$PANSS_Total = as.numeric(as.character(temp_array$PANSS_Total))
  visit_number = 1
  visit_week = 1
  previous_visit_panss = 0
  for (j in 1:nrow(temp_array)) {

          new_e[nrow(new_e)+1,"VisitNumber"] = visit_number
          visit_number = visit_number + 1
          new_e[nrow(new_e),"VisitWeek"] = strtoi(temp_array[j,"VisitDay"])/7

          if(j>1){
          new_e[nrow(new_e),"PreviousVisitPANSS"] = temp_array[j-1,"PANSS_Total"]
          new_e[nrow(new_e),"PreviousVisitDay"] = temp_array[j-1,"VisitDay"]
          }
          if(j==1){
          new_e[nrow(new_e),"PreviousVisitPANSS"] = temp_array[j,"PANSS_Total"]
          new_e[nrow(new_e),"PreviousVisitDay"] = temp_array[j,"VisitDay"]
          }

          if(j<nrow(temp_array)){
          new_e[nrow(new_e),"NextVisitPANSS"] = temp_array[j+1,"PANSS_Total"]
          new_e[nrow(new_e),"NextVisitDay"] = temp_array[j+1,"VisitDay"]
          }
          if(j==nrow(temp_array)){
          new_e[nrow(new_e),"NextVisitPANSS"] = temp_array[j,"PANSS_Total"]
          new_e[nrow(new_e),"NextVisitDay"] = temp_array[j,"VisitDay"]
          }

          if(j<(nrow(temp_array)-1)){
          new_e[nrow(new_e),"Next2ndVisitPANSS"] = temp_array[j+2,"PANSS_Total"]
          new_e[nrow(new_e),"Next2ndVisitDay"] = temp_array[j+2,"VisitDay"]
          }else{
          new_e[nrow(new_e),"Next2ndVisitPANSS"] = temp_array[j,"PANSS_Total"]
          new_e[nrow(new_e),"Next2ndVisitDay"] = temp_array[j,"VisitDay"]
          }



          if(j>1){
          new_e[nrow(new_e),"DaysFromPreviousVisit"] = temp_array[j,"VisitDay"] -
new_e[nrow(new_e),"PreviousVisitDay"]
          new_e[nrow(new_e),"LastVisitPANSSDiff"] =  new_e[nrow(new_e),"PreviousVisitPANSS"] -
temp_array[j,"PANSS_Total"]
          }
          if(j==1){
            new_e[nrow(new_e),"DaysFromPreviousVisit"] = 0
            new_e[nrow(new_e),"LastVisitPANSSDiff"] = 0
          }

          if(j<nrow(temp_array)){
           new_e[nrow(new_e),"DaysToNextVisit"] = temp_array[j,"VisitDay"] - temp_array[j+1,"VisitDay"]
           new_e[nrow(new_e),"NextVisitPANSSDiff"] =  temp_array[j,"PANSS_Total"] - temp_array[j+1,"PANSS_Total"]
          }
          if(j==nrow(temp_array)){
           new_e[nrow(new_e),"DaysToNextVisit"] = 0
           new_e[nrow(new_e),"NextVisitPANSSDiff"] =  0
          }
          new_e[nrow(new_e),"PatientMinPANSS"] = min(temp_array$PANSS_Total)
          new_e[nrow(new_e),"PatientMaxPANSS"] = max(temp_array$PANSS_Total)
          new_e[nrow(new_e),"PatientAveragePANSS"] = mean(temp_array$PANSS_Total)

          if(j>2){
          new_e[nrow(new_e),"SecondLastVisitPANSS"] = temp_array[j-2,"PANSS_Total"]
          new_e[nrow(new_e),"SecondLastVisitDay"] = temp_array[j-2,"VisitDay"]
          }
          if(j<=2){
          new_e[nrow(new_e),"SecondLastVisitPANSS"] = temp_array[j,"PANSS_Total"]
          new_e[nrow(new_e),"SecondLastVisitDay"] = temp_array[j,"VisitDay"]
          }

          new_e[nrow(new_e),"PANSSDiffWRTFirstDay"] = temp_array[1,"PANSS_Total"] -
temp_array[j,"PANSS_Total"]
```

```
              new_e[nrow(new_e),"TxGroup"]=temp_array[j,"TxGroup"]
              new_e[nrow(new_e),"Country"]=temp_array[j,"Country"]
              new_e[nrow(new_e),"PANSS_Total"]=temp_array[j,"PANSS_Total"]
              new_e[nrow(new_e),"PatientID"]=temp_array[j,"PatientID"]
              new_e[nrow(new_e),"VisitDay"]=temp_array[j,"VisitDay"]
   }
}

write.csv(new_e,file="transformed_data_E.csv", row.names=FALSE, na="")

new_e = read.csv("transformed_data_E.csv", header = T)

data_to_predict_on = data.frame()
for (i in pat_id_in_studyE) {
   temp_array = data.frame()
   temp_array = new_e[new_e$PatientID == toString(i),]
   temp_array = temp_array[order(temp_array$VisitWeek),]
   temp_array$VisitWeek = as.numeric(as.character(temp_array$VisitWeek))
   temp_array$PANSS_Total = as.numeric(as.character(temp_array$PANSS_Total))
   data_not_present_for_week_intersted_in = 1
   week_interested_in = week_neg = week_pos = 18
   while (data_not_present_for_week_intersted_in == 1) {
     #cat("in while loop ",i,data_not_present_for_week_intersted_in,week_neg,"\n")
     for (j in 1:nrow(temp_array)) {
          if ((week_interested_in == round(temp_array[j, "VisitWeek"])) &&
(data_not_present_for_week_intersted_in==1)) {
              #data_to_predict_on[nrow(data_to_predict_on)+1,] = temp_array[c(j),]
              data_to_predict_on = rbind(data_to_predict_on,temp_array[c(j),])
              data_not_present_for_week_intersted_in = 0
              #cat("in cond 1\n")
              break
          }
          if ((round(temp_array[j, "VisitWeek"]) == week_pos) && (data_not_present_for_week_intersted_in==1)) {
              #data_to_predict_on[nrow(data_to_predict_on)+1,] = temp_array[c(j),]
              data_to_predict_on = rbind(data_to_predict_on,temp_array[c(j),])
              data_not_present_for_week_intersted_in = 0
              #cat("in cond 2\n")
              break
          }
          if ((round(temp_array[j, "VisitWeek"]) == week_neg) && (data_not_present_for_week_intersted_in==1)) {
              #data_to_predict_on[nrow(data_to_predict_on)+1,] = temp_array[c(j),]
              data_to_predict_on = rbind(data_to_predict_on,temp_array[c(j),])
              data_not_present_for_week_intersted_in = 0
              break
          }
```

```
    }
    #data_to_predict_on
    week_neg = week_neg - 1
    week_pos = week_pos + 1
  }
  #cat("out of while loop\n")


}
#data_to_predict_on = data_to_predict_on[,-c(20)]
data_to_predict_on$VisitWeek = 18
data_to_predict_on$VisitNumber = as.factor(as.character(data_to_predict_on$VisitNumber))


levels(data_to_predict_on$Country) = levels(input_data.Train$Country)
levels(data_to_predict_on$TxGroup) = levels(input_data.Train$TxGroup)
levels(data_to_predict_on$VisitNumber) = levels(input_data.Train$VisitNumber)

predicted_panss = predict(model_for_panss_total,newdata=data_to_predict_on,n.trees=number_of_trees)
data_to_predict_on$PANSS_Total_Predicted  = predicted_panss
panss_total_predicted_mse = mean((data_to_predict_on$PANSS_Total-
data_to_predict_on$PANSS_Total_Predicted)^2)

write.csv(data_to_predict_on,"objective3_output_complete_data_with_boosting.csv",row.names=FALSE, na="")

final_output=data.frame()
for (i in pat_id_in_studyE) {
  temp_array = data.frame()
  temp_array = data_to_predict_on[data_to_predict_on$PatientID == toString(i),]
  for (k in 1:nrow(temp_array)) {
    if (strtoi(temp_array[k, "VisitWeek"]) == week_interested_in) {
        final_output[nrow(final_output)+1,"PatientID"]=i
        final_output[nrow(final_output),"PANSS_total"]=temp_array[k, "PANSS_Total_Predicted"]
        break
    }
  }
}
write.csv(final_output,"objective3_output_with_boosting.csv",row.names=FALSE, na="")
cat("TRAIN MSE with Boosting",panss_total_train_mse,"\n")
cat("TEST MSE with Boosting",panss_total_test_mse,"\n")
cat("MSE on the Data To be predicted on with Boosting",panss_total_predicted_mse,"\n")
```

PROGRAM7

```
#Boosting shrinkage value selection iterations.
########DATA TRANSFORMATION##############
```

```
#
setwd("C:/Users/msingh13/Documents/STAT202/Class_Project/R_working_dir")
set.seed(1)
dataABCD = read.csv("Study_ABCD.csv", header = T)

data_to_analyze = dataABCD[,-c(1,4:6)]
#Eliminate entries with ERROR country
data_to_analyze = data_to_analyze[!data_to_analyze$Country=="ERROR",]

#For model creation we can omit entries with Lead Status that are not Passed
data_to_analyze = data_to_analyze[!data_to_analyze$LeadStatus=="Assign to CS",]
data_to_analyze = data_to_analyze[!data_to_analyze$LeadStatus=="Flagged",]
data_to_analyze = data_to_analyze[,-c(36)]
data_to_analyze = data_to_analyze[,-c(5:34)]

#*******************

new_abcd = data.frame()
pat_id_in_study =  unique(data_to_analyze$PatientID, incomparables = FALSE)
for (i in pat_id_in_study) {
   count = 0
   temp_array = data.frame()
   temp_array = data_to_analyze[data_to_analyze$PatientID == toString(i),]
   temp_array = temp_array[order(temp_array$VisitDay),]
   temp_array$VisitDay = as.numeric(as.character(temp_array$VisitDay))
   temp_array$PANSS_Total = as.numeric(as.character(temp_array$PANSS_Total))
   visit_number = 1
   visit_week = 1
   previous_visit_panss = 0
   for (j in 1:nrow(temp_array)) {

               new_abcd[nrow(new_abcd)+1,"VisitNumber"] = visit_number
               visit_number = visit_number + 1
               new_abcd[nrow(new_abcd),"VisitWeek"] = strtoi(temp_array[j,"VisitDay"])/7



               if(j>1){
               new_abcd[nrow(new_abcd),"PreviousVisitPANSS"] = temp_array[j-1,"PANSS_Total"]
               new_abcd[nrow(new_abcd),"PreviousVisitDay"] = temp_array[j-1,"VisitDay"]
               }
               if(j==1){
               new_abcd[nrow(new_abcd),"PreviousVisitPANSS"] = temp_array[j,"PANSS_Total"]
               new_abcd[nrow(new_abcd),"PreviousVisitDay"] = temp_array[j,"VisitDay"]
               }

               if(j<nrow(temp_array)){
```

```
        new_abcd[nrow(new_abcd),"NextVisitPANSS"] = temp_array[j+1,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"NextVisitDay"] = temp_array[j+1,"VisitDay"]
        }
        if(j==nrow(temp_array)){
        new_abcd[nrow(new_abcd),"NextVisitPANSS"] = temp_array[j,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"NextVisitDay"] = temp_array[j,"VisitDay"]
        }

        if(j<(nrow(temp_array)-1)){
        new_abcd[nrow(new_abcd),"Next2ndVisitPANSS"] = temp_array[j+2,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"Next2ndVisitDay"] = temp_array[j+2,"VisitDay"]
        }else{
        new_abcd[nrow(new_abcd),"Next2ndVisitPANSS"] = temp_array[j,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"Next2ndVisitDay"] = temp_array[j,"VisitDay"]
        }



        if(j>1){
        new_abcd[nrow(new_abcd),"DaysFromPreviousVisit"] = temp_array[j,"VisitDay"] - 
new_abcd[nrow(new_abcd),"PreviousVisitDay"]
        new_abcd[nrow(new_abcd),"LastVisitPANSSDiff"] =  new_abcd[nrow(new_abcd),"PreviousVisitPANSS"] - 
temp_array[j,"PANSS_Total"]
        }
        if(j==1){
          new_abcd[nrow(new_abcd),"DaysFromPreviousVisit"] = 0
          new_abcd[nrow(new_abcd),"LastVisitPANSSDiff"] = 0
        }

        if(j<nrow(temp_array)){
         new_abcd[nrow(new_abcd),"DaysToNextVisit"] = temp_array[j,"VisitDay"] - temp_array[j+1,"VisitDay"]
         new_abcd[nrow(new_abcd),"NextVisitPANSSDiff"] =  temp_array[j,"PANSS_Total"] - 
temp_array[j+1,"PANSS_Total"]
        }
        if(j==nrow(temp_array)){
         new_abcd[nrow(new_abcd),"DaysToNextVisit"] = 0
         new_abcd[nrow(new_abcd),"NextVisitPANSSDiff"] =  0
        }
        new_abcd[nrow(new_abcd),"PatientMinPANSS"] = min(temp_array$PANSS_Total)
        new_abcd[nrow(new_abcd),"PatientMaxPANSS"] = max(temp_array$PANSS_Total)
        new_abcd[nrow(new_abcd),"PatientAveragePANSS"] = mean(temp_array$PANSS_Total)

        if(j>2){
        new_abcd[nrow(new_abcd),"SecondLastVisitPANSS"] = temp_array[j-2,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"SecondLastVisitDay"] = temp_array[j-2,"VisitDay"]
        }
        if(j<=2){
            new_abcd[nrow(new_abcd),"SecondLastVisitPANSS"] = temp_array[j,"PANSS_Total"]
            new_abcd[nrow(new_abcd),"SecondLastVisitDay"] = temp_array[j,"VisitDay"]
            }

            new_abcd[nrow(new_abcd),"PANSSDiffWRTFirstDay"] = temp_array[1,"PANSS_Total"] -
temp_array[j,"PANSS_Total"]
            new_abcd[nrow(new_abcd),"TxGroup"]=temp_array[j,"TxGroup"]
            new_abcd[nrow(new_abcd),"Country"]=temp_array[j,"Country"]
            new_abcd[nrow(new_abcd),"PANSS_Total"]=temp_array[j,"PANSS_Total"]
   }

}

write.csv(new_abcd,file="transformed_data_ABCD.csv", row.names=FALSE, na="")


############Creating Model To Predict PANSS_Total###############
transformed_data = read.csv("transformed_data_ABCD.csv", header = T)

#transformed_data$VisitNumber = as.numeric(as.character(transformed_data$VisitNumber))
transformed_data$VisitNumber = as.factor(as.character(transformed_data$VisitNumber))
transformed_data$TxGroup = as.factor(as.character(transformed_data$TxGroup))
transformed_data$Country = as.factor(as.character(transformed_data$Country))
transformed_data$VisitWeek = as.numeric(as.character(transformed_data$VisitWeek))
transformed_data$PatientAveragePANSS  = as.numeric(as.character(transformed_data$PatientAveragePANSS ))
transformed_data$PreviousVisitPANSS = as.numeric(as.character(transformed_data$PreviousVisitPANSS))
transformed_data$PreviousVisitDay = as.numeric(as.character(transformed_data$PreviousVisitDay))
transformed_data$NextVisitPANSS = as.numeric(as.character(transformed_data$NextVisitPANSS))
transformed_data$NextVisitDay = as.numeric(as.character(transformed_data$NextVisitDay))
transformed_data$DaysFromPreviousVisit= as.numeric(as.character(transformed_data$DaysFromPreviousVisit))
transformed_data$LastVisitPANSSDiff = as.numeric(as.character(transformed_data$LastVisitPANSSDiff))
transformed_data$DaysToNextVisit = as.numeric(as.character(transformed_data$DaysToNextVisit))
transformed_data$NextVisitPANSSDiff = as.numeric(as.character(transformed_data$NextVisitPANSSDiff))
transformed_data$PatientMinPANSS = as.numeric(as.character(transformed_data$PatientMinPANSS))
transformed_data$PatientMaxPANSS = as.numeric(as.character(transformed_data$PatientMaxPANSS))
transformed_data$SecondLastVisitPANSS = as.numeric(as.character(transformed_data$SecondLastVisitPANSS))
transformed_data$SecondLastVisitDay = as.numeric(as.character(transformed_data$SecondLastVisitDay))
transformed_data$PANSSDiffWRTFirstDay =as.numeric(as.character(transformed_data$PANSSDiffWRTFirstDay))


#transformed_data=transformed_data[,-c()]

panss_total_train_mse = panss_total_test_mse = 0
train = 1:(round(nrow(transformed_data)*0.90))
input_data.Train = data.frame()
input_data.Test = data.frame()
input_data.Train = transformed_data[train,]
input_data.Test = transformed_data[-train,]



library(randomForest)
predictors_num = ncol(input_data.Train) - 1

#Bagging where the mtry is number of predictors
#model_for_panss_total = randomForest(PANSS_Total ~ .,data=input_data.Train,
mtry=predictors_num,ntree=400,importance=TRUE)

#For random forest the mtry is predictors/3
#model_for_panss_total = randomForest(PANSS_Total ~ .,data=input_data.Train, mtry=6,importance=TRUE)

#Linear Model
#model_for_panss_total = lm(PANSS_Total ~ (VisitWeek:.) + (I(log(VisitWeek+1)):.) + (I((VisitWeek+1)^0.1):.),
data=input_data.Train)



library(gbm)
#Boosting

pows <- seq(-3, -0.2, by = 0.1)
lambdas <- 10^pows
panss_total_train_mse <- rep(NA, length(lambdas))
panss_total_test_mse <- rep(NA, length(lambdas))
number_of_trees =4000

for (i in 1:length(lambdas)) {
  #model_for_panss_total=gbm(PANSS_Total ~ (VisitWeek:.) + (I(log(VisitWeek+1)):.) +
(I((VisitWeek+1)^0.1):.),data=input_data.Train,distribution="gaussian",n.trees=number_of_trees,interaction.depth=3,sh
rinkage=0.2)
  model_for_panss_total=gbm(PANSS_Total ~ . + (VisitWeek:.) + (I(log(VisitWeek+1)):.) +
(I((VisitWeek+1)^0.1):.),data=input_data.Train,distribution="gaussian",n.trees=number_of_trees,interaction.depth=2,sh
rinkage=lambdas[i])

  predicted_panss_Train = predict(model_for_panss_total,input_data.Train,n.trees=number_of_trees)
  panss_total_train_mse[i] = mean((input_data.Train$PANSS_Total-predicted_panss_Train)^2)
  cat("TRAIN MSE",panss_total_train_mse,"\n")
  predicted_panss_Test = predict(model_for_panss_total,input_data.Test,n.trees=number_of_trees)
  panss_total_test_mse[i] = mean((input_data.Test$PANSS_Total-predicted_panss_Test)^2)
  cat("TEST MSE",panss_total_test_mse,"\n")
}
plot(lambdas, panss_total_test_mse, type = "b", xlab = "Shrinkage values", ylab = "Test MSE")
plot(lambdas, panss_total_train_mse, type = "b", xlab = "Shrinkage values", ylab = "Training MSE")
#
```

OBJECTIVE 4

PROGRAM 1

```
### INPUT DATA TRANSFORMATION ###
#########DATA TRANSFORMATION###############
#
setwd("C:/Users/msingh13/Documents/STAT202/Class_Project/R_working_dir")
set.seed(1)
dataABCD = read.csv("Study_ABCD.csv", header = T)

data_to_analyze = dataABCD[,-c(1,4:6)]
#Eliminate entries with ERROR country
data_to_analyze = data_to_analyze[!data_to_analyze$Country=="ERROR",]

#For objective 4 we do need LeadStatus so commenting below lines
#For model creation we can omit entries with Lead Status that are not Passed
#data_to_analyze = data_to_analyze[!data_to_analyze$LeadStatus=="Assign to CS",]
#data_to_analyze = data_to_analyze[!data_to_analyze$LeadStatus=="Flagged",]
#data_to_analyze = data_to_analyze[,-c(36)]

data_to_analyze = data_to_analyze[,-c(5:34)]

#*******************

new_abcd = data.frame()
pat_id_in_study =  unique(data_to_analyze$PatientID, incomparables = FALSE)
for (i in pat_id_in_study) {
  count = 0
  temp_array = data.frame()
  temp_array = data_to_analyze[data_to_analyze$PatientID == toString(i),]
  temp_array = temp_array[order(temp_array$VisitDay),]
  temp_array$VisitDay = as.numeric(as.character(temp_array$VisitDay))
  temp_array$PANSS_Total = as.numeric(as.character(temp_array$PANSS_Total))
  visit_number = 1
  visit_week = 1
  previous_visit_panss = 0
  for (j in 1:nrow(temp_array)) {

          new_abcd[nrow(new_abcd)+1,"VisitNumber"] = visit_number
          visit_number = visit_number + 1
          new_abcd[nrow(new_abcd),"VisitWeek"] = strtoi(temp_array[j,"VisitDay"])/7
          new_abcd[nrow(new_abcd),"VisitDay"] = temp_array[j,"VisitDay"]

          if(j>1){
          new_abcd[nrow(new_abcd),"PreviousVisitPANSS"] = temp_array[j-1,"PANSS_Total"]
```

```
         new_abcd[nrow(new_abcd),"PreviousVisitDay"] = temp_array[j-1,"VisitDay"]
         }
         if(j==1){
         new_abcd[nrow(new_abcd),"PreviousVisitPANSS"] = temp_array[j,"PANSS_Total"]
         new_abcd[nrow(new_abcd),"PreviousVisitDay"] = temp_array[j,"VisitDay"]
         }

         if(j<nrow(temp_array)){
         new_abcd[nrow(new_abcd),"NextVisitPANSS"] = temp_array[j+1,"PANSS_Total"]
         new_abcd[nrow(new_abcd),"NextVisitDay"] = temp_array[j+1,"VisitDay"]
         }
         if(j==nrow(temp_array)){
         new_abcd[nrow(new_abcd),"NextVisitPANSS"] = temp_array[j,"PANSS_Total"]
         new_abcd[nrow(new_abcd),"NextVisitDay"] = temp_array[j,"VisitDay"]
         }

         if(j<(nrow(temp_array)-1)) {
          new_abcd[nrow(new_abcd),"Next2ndVisitPANSS"] = temp_array[j+2,"PANSS_Total"]
          new_abcd[nrow(new_abcd),"Next2ndVisitDay"] = temp_array[j+2,"VisitDay"]
         } else {
          #new_abcd[nrow(new_abcd),"Next2ndVisitPANSS"] = temp_array[j,"PANSS_Total"]
          #new_abcd[nrow(new_abcd,"Next2ndVisitDay"] = temp_array[j,"VisitDay"]
          new_abcd[nrow(new_abcd),"Next2ndVisitPANSS"] = temp_array[j,"PANSS_Total"]
          new_abcd[nrow(new_abcd),"Next2ndVisitDay"] = temp_array[j,"VisitDay"]

         }

         if(j>1){
         new_abcd[nrow(new_abcd),"DaysFromPreviousVisit"] = temp_array[j,"VisitDay"] -
new_abcd[nrow(new_abcd),"PreviousVisitDay"]
         new_abcd[nrow(new_abcd),"LastVisitPANSSDiff"] =  new_abcd[nrow(new_abcd),"PreviousVisitPANSS"] -
temp_array[j,"PANSS_Total"]
         }
         if(j==1){
           new_abcd[nrow(new_abcd),"DaysFromPreviousVisit"] = 0
           new_abcd[nrow(new_abcd),"LastVisitPANSSDiff"] = 0
         }

         if(j<nrow(temp_array)){
          new_abcd[nrow(new_abcd),"DaysToNextVisit"] = temp_array[j,"VisitDay"] - temp_array[j+1,"VisitDay"]
          new_abcd[nrow(new_abcd),"NextVisitPANSSDiff"] =  temp_array[j,"PANSS_Total"] -
temp_array[j+1,"PANSS_Total"]
         }
         if(j==nrow(temp_array)){
          new_abcd[nrow(new_abcd),"DaysToNextVisit"] = 0
          new_abcd[nrow(new_abcd),"NextVisitPANSSDiff"] =  0
```

```
        }
        new_abcd[nrow(new_abcd),"PatientMinPANSS"] = min(temp_array$PANSS_Total)
        new_abcd[nrow(new_abcd),"PatientMaxPANSS"] = max(temp_array$PANSS_Total)
        new_abcd[nrow(new_abcd),"PatientAveragePANSS"] = mean(temp_array$PANSS_Total)

        if(j>2){
        new_abcd[nrow(new_abcd),"SecondLastVisitPANSS"] = temp_array[j-2,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"SecondLastVisitDay"] = temp_array[j-2,"VisitDay"]
        }
        if(j<=2){
        new_abcd[nrow(new_abcd),"SecondLastVisitPANSS"] = temp_array[j,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"SecondLastVisitDay"] = temp_array[j,"VisitDay"]
        }

        new_abcd[nrow(new_abcd),"PANSSDiffWRTFirstDay"] = temp_array[1,"PANSS_Total"] -
temp_array[j,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"TxGroup"]=temp_array[j,"TxGroup"]
        new_abcd[nrow(new_abcd),"Country"]=temp_array[j,"Country"]
        new_abcd[nrow(new_abcd),"PANSS_Total"]=temp_array[j,"PANSS_Total"]

        if(j>1){
          if(temp_array[j-1,"LeadStatus"]=="Passed") {
           new_abcd[nrow(new_abcd),"PreviousLeadStatus"]=0
          }else {
           new_abcd[nrow(new_abcd),"PreviousLeadStatus"]=1
          }
        }
        if(j==1){
           if(temp_array[1,"LeadStatus"]=="Passed") {
            new_abcd[nrow(new_abcd),"PreviousLeadStatus"]=0
           } else {
            new_abcd[nrow(new_abcd),"PreviousLeadStatus"]=1
            }
        }
        if(j>2){
          if(temp_array[j-2,"LeadStatus"]=="Passed") {
           new_abcd[nrow(new_abcd),"Previous2DayLeadStatus"]=0
          } else {
           new_abcd[nrow(new_abcd),"Previous2DayLeadStatus"]=1
          }
        }
        if(j<=2){
          if(temp_array[1,"LeadStatus"]=="Passed") {
            new_abcd[nrow(new_abcd),"Previous2DayLeadStatus"]=0
          }else {
            new_abcd[nrow(new_abcd),"Previous2DayLeadStatus"]=1
```

```
               }
            }

            if(j<nrow(temp_array)){
               if(temp_array[j+1,"LeadStatus"]=="Passed") {
                 new_abcd[nrow(new_abcd),"NextVisitLeadStatus"]=0
               }else {
                 new_abcd[nrow(new_abcd),"NextVisitLeadStatus"]=1
               }
            }
            if(j==nrow(temp_array)){
               if(temp_array[j,"LeadStatus"]=="Passed") {
                 new_abcd[nrow(new_abcd),"NextVisitLeadStatus"]=0
               }else {
                 new_abcd[nrow(new_abcd),"NextVisitLeadStatus"]=1
               }
            }

            if(temp_array[j,"LeadStatus"]=="Passed") {
               new_abcd[nrow(new_abcd),"FlagOrACSLeadStatus"]=0
            }else {
               new_abcd[nrow(new_abcd),"FlagOrACSLeadStatus"]=1
            }
            #if(temp_array[j,"LeadStatus"]=="Passed") {
            #    new_abcd[nrow(new_abcd),"LeadStatusBinary"]=0
            #}else {
            #    new_abcd[nrow(new_abcd),"LeadStatusBinary"]=1
            #}
   }

}


new_abcd$VisitNumber = as.numeric(as.character(new_abcd$VisitNumber))
#library(gbm)
#number_of_trees=6000
#load("panss_total_boosting_model.rda")
#panss_total_boosting_model = model_for_panss_total
#predicted_panss_with_boosting = predict(panss_total_boosting_model,newdata=new_abcd,n.trees=number_of_trees)

#new_abcd$Predicted_PANSS_Boosting = predicted_panss_with_boosting


load("panss_total_LM_model.rda")
panss_total_lm_model = model_for_panss_total
predicted_panss_with_lm = predict(panss_total_lm_model,newdata=new_abcd)
```

```
new_abcd$Predicted_PANSS_LM = predicted_panss_with_lm

new_abcd$Predicted_vs_ActualPanssDiffLM = new_abcd$PANSS_Total - new_abcd$Predicted_PANSS_LM
#new_abcd$Predicted_vs_ActualPanssDiffBoost = new_abcd$PANSS_Total - new_abcd$Predicted_PANSS_Boosting

write.csv(new_abcd,file="transformed_data_ABCD_for_LeadStatus.csv", row.names=FALSE, na="")
```

PROGRAM2

```
#Fit the model and create ROC plot for Training and Test data.

############Creating Model To Predict PANSS_Total###############
transformed_data = read.csv("transformed_data_ABCD_for_LeadStatus.csv", header = T)

transformed_data$VisitNumber = as.numeric(as.character(transformed_data$VisitNumber))
#transformed_data$VisitNumber = as.factor(as.character(transformed_data$VisitNumber))
transformed_data$TxGroup = as.factor(as.character(transformed_data$TxGroup))
transformed_data$Country = as.factor(as.character(transformed_data$Country))
transformed_data$VisitWeek = as.numeric(as.character(transformed_data$VisitWeek))
transformed_data$PatientAveragePANSS  = as.numeric(as.character(transformed_data$PatientAveragePANSS ))
transformed_data$PreviousVisitPANSS = as.numeric(as.character(transformed_data$PreviousVisitPANSS))
transformed_data$PreviousVisitDay = as.numeric(as.character(transformed_data$PreviousVisitDay))
transformed_data$NextVisitPANSS = as.numeric(as.character(transformed_data$NextVisitPANSS))
transformed_data$NextVisitDay = as.numeric(as.character(transformed_data$NextVisitDay))
transformed_data$DaysFromPreviousVisit= as.numeric(as.character(transformed_data$DaysFromPreviousVisit))
transformed_data$LastVisitPANSSDiff = as.numeric(as.character(transformed_data$LastVisitPANSSDiff))
transformed_data$DaysToNextVisit = as.numeric(as.character(transformed_data$DaysToNextVisit))
transformed_data$NextVisitPANSSDiff = as.numeric(as.character(transformed_data$NextVisitPANSSDiff))
transformed_data$PatientMinPANSS = as.numeric(as.character(transformed_data$PatientMinPANSS))
transformed_data$PatientMaxPANSS = as.numeric(as.character(transformed_data$PatientMaxPANSS))
transformed_data$SecondLastVisitPANSS = as.numeric(as.character(transformed_data$SecondLastVisitPANSS))
transformed_data$SecondLastVisitDay = as.numeric(as.character(transformed_data$SecondLastVisitDay))
transformed_data$PANSSDiffWRTFirstDay =as.numeric(as.character(transformed_data$PANSSDiffWRTFirstDay))
transformed_data$FlagOrACSLeadStatus = as.factor(as.character(transformed_data$FlagOrACSLeadStatus))

#transformed_data$Predicted_vs_ActualPanssDiffLM = abs(transformed_data$Predicted_vs_ActualPanssDiffLM)

#transformed_data=transformed_data[,-c()]

panss_total_train_mse = panss_total_test_mse = 0
train = 1:(round(nrow(transformed_data)*0.90))
input_data.Train = data.frame()
input_data.Test = data.frame()
input_data.Train = transformed_data[train,]
input_data.Test = transformed_data[-train,]
```

```
#library(randomForest)
#predictors_num = ncol(input_data.Train) - 1

#Bagging where the mtry is number of predictors
#model_for_panss_total = randomForest(PANSS_Total ~ .,data=input_data.Train,
mtry=predictors_num,ntree=400,importance=TRUE)

#For random forest the mtry is predictors/3
#model_for_panss_total = randomForest(PANSS_Total ~ .,data=input_data.Train, mtry=6,importance=TRUE)

#Linear Model
#model_for_panss_total = lm(PANSS_Total ~ (VisitWeek:.) + (I(log(VisitWeek+1)):.) + (I((VisitWeek+1)^0.1):.),
data=input_data.Train)

###############Logistic Regression##################
#classification_model = glm(FlagOrACSLeadStatus ~ + PreviousVisitPANSS + NextVisitPANSS + PANSS_Total +
Next2ndVisitPANSS + SecondLastVisitPANSS + PatientMinPANSS + PatientMaxPANSS + PatientAveragePANSS,
data=input_data.Train, family=binomial)
#classification_model = glm(FlagOrACSLeadStatus ~ . + (Predicted_vs_ActualPanssDiffLM:.) + (PreviousVisitPANSS:.) +
(NextVisitPANSS:.) + (PANSS_Total:.) + (Next2ndVisitPANSS:.) + (SecondLastVisitPANSS:.) + (PatientMinPANSS:.) +
(PatientMaxPANSS:.) + (PatientAveragePANSS:.) , data=input_data.Train, family=binomial)
classification_model = glm(FlagOrACSLeadStatus ~ . + (Predicted_vs_ActualPanssDiffLM:.) +
(I(log(Predicted_vs_ActualPanssDiffLM+1)):.) + (I((Predicted_vs_ActualPanssDiffLM+1)^0.1):.) , data=input_data.Train,
family=binomial)



glm_probabilities = predict(classification_model,input_data.Train,type="response")
#Specifying prob level on Training dataset
#glm.pred=rep("0",1250)
#glm.pred$[glm_probabilities>0.5]="1"
#table(glm.pred,input_data.Train$FlagOrACSLeadStatus)

#glm.pred    0    1
#       0   65   24
#       1 3521 2326


#table(predict=glm_probabilities,truth=input_data.Train$FlagOrACSLeadStatus)

######ROC Curve#######
#library (ROCR)
# rocplot =function (pred,truth,...){
# predob = prediction(pred,truth)
# perf = performance(predob,"tpr","fpr")
# plot(perf,...)
#}

#fitted_train=attributes(predict(classification_model,input_data.Train,decision.values=TRUE))$decision.values
#par(mfrow=c(1,2))
#rocplot(fitted_train,input_data.Train$FlagOrACSLeadStatus,main="Training Data")
#fitted_test=attributes(predict(classification_model,input_data.Test,decision.values=TRUE))$decision.values
#rocplot(fitted_test,input_data.Test$FlagOrACSLeadStatus,main="Test Data")
######ROC Curve######



#####ROC CURVE 2 TRAIN#####
all_probs_train = predict(classification_model,input_data.Train,type=c("response"))
input_data.Train$FlagOrCSProb = all_probs_train
library(pROC)
g1=roc(FlagOrACSLeadStatus ~ FlagOrCSProb, data=input_data.Train)

#####ROC CURVE 2 TEST#####
all_probs_test = predict(classification_model,input_data.Test,type=c("response"))
input_data.Test$FlagOrCSProb = all_probs_test
library(pROC)
g2=roc(FlagOrACSLeadStatus ~ FlagOrCSProb, data=input_data.Test)
plot(g1)
lines(g2)
###############Logistic Regression###################
```

PROGRAM3
#Data transformation for study E pre probablity prediction and
# Predicting probability.

```
############Study E Data Transformation###############

input_studyE = read.csv("Study_E.csv", header = T)

#Getting the patientIDs for interested patients
sample_sub_assesmentid = read.csv("sample_submission_status.csv", header = T)
assesment_id_in_studyE =  unique(sample_sub_assesmentid$AssessmentID, incomparables = FALSE)

patientIDs_for_given_assesmet_ids = data.frame()
for(i in assesment_id_in_studyE) {
        temp_array = data.frame()
        temp_array = input_studyE[input_studyE$AssessmentiD == toString(i),]
        for (j in 1:nrow(temp_array)) {
                patientIDs_for_given_assesmet_ids[nrow(patientIDs_for_given_assesmet_ids)+1,"PatientID"] =
temp_array[j,"PatientID"]
        }
}
```

```
patientIDs_for_given_assesmet_ids = unique(patientIDs_for_given_assesmet_ids$PatientID, incomparables = FALSE)



#Interpreting UK as France in the given data.
#There is no training data for country UK so interpreting it as France as they are geographically close countries.
levels(input_studyE$Country) = c(levels(input_studyE$Country),"France")
input_studyE$Country[input_studyE$Country=="UK"] = "France"

new_e = data.frame()
for (i in patientIDs_for_given_assesmet_ids) {
   temp_array = data.frame()
   temp_array = input_studyE[input_studyE$PatientID == toString(i),]
   temp_array = temp_array[order(temp_array$VisitDay),]
   temp_array[,"LeadStatus"] = "Passed"
   temp_array$VisitDay = as.numeric(as.character(temp_array$VisitDay))
   temp_array$PANSS_Total = as.numeric(as.character(temp_array$PANSS_Total))
   visit_number = 1
   visit_week = 1
   previous_visit_panss = 0
   for (j in 1:nrow(temp_array)) {

            new_e[nrow(new_e)+1,"VisitNumber"] = visit_number
            visit_number = visit_number + 1

            #temp_array[j+1,"LeadStatus"] =  temp_array[j,"LeadStatus"]
            new_e[nrow(new_e),"VisitWeek"] = strtoi(temp_array[j,"VisitDay"])/7
            new_e[nrow(new_e),"VisitDay"] = temp_array[j,"VisitDay"]

            if(j>1){
            new_e[nrow(new_e),"PreviousVisitPANSS"] = temp_array[j-1,"PANSS_Total"]
            new_e[nrow(new_e),"PreviousVisitDay"] = temp_array[j-1,"VisitDay"]
            }
            if(j==1){
            new_e[nrow(new_e),"PreviousVisitPANSS"] = temp_array[j,"PANSS_Total"]
            new_e[nrow(new_e),"PreviousVisitDay"] = temp_array[j,"VisitDay"]
            }

            if(j<nrow(temp_array)){
            new_e[nrow(new_e),"NextVisitPANSS"] = temp_array[j+1,"PANSS_Total"]
            new_e[nrow(new_e),"NextVisitDay"] = temp_array[j+1,"VisitDay"]
            }
            if(j==nrow(temp_array)){
            new_e[nrow(new_e),"NextVisitPANSS"] = temp_array[j,"PANSS_Total"]
            new_e[nrow(new_e),"NextVisitDay"] = temp_array[j,"VisitDay"]
```

```
                }

                 if(j<(nrow(temp_array)-1)) {
                 new_e[nrow(new_e),"Next2ndVisitPANSS"] = temp_array[j+2,"PANSS_Total"]
                 new_e[nrow(new_e),"Next2ndVisitDay"] = temp_array[j+2,"VisitDay"]
                } else {
                 #new_abcd[nrow(new_abcd),"Next2ndVisitPANSS"] = temp_array[j,"PANSS_Total"]
                 #new_abcd[nrow(new_abcd,"Next2ndVisitDay"] = temp_array[j,"VisitDay"]
                 new_e[nrow(new_e),"Next2ndVisitPANSS"] = temp_array[j,"PANSS_Total"]
                 new_e[nrow(new_e),"Next2ndVisitDay"] = temp_array[j,"VisitDay"]

                }



                if(j>1){
                new_e[nrow(new_e),"DaysFromPreviousVisit"] = temp_array[j,"VisitDay"] -
new_e[nrow(new_e),"PreviousVisitDay"]
                new_e[nrow(new_e),"LastVisitPANSSDiff"] =  new_e[nrow(new_e),"PreviousVisitPANSS"] -
temp_array[j,"PANSS_Total"]
                }
                if(j==1){
                   new_e[nrow(new_e),"DaysFromPreviousVisit"] = 0
                   new_e[nrow(new_e),"LastVisitPANSSDiff"] = 0
                }

                if(j<nrow(temp_array)){
                 new_e[nrow(new_e),"DaysToNextVisit"] = temp_array[j,"VisitDay"] - temp_array[j+1,"VisitDay"]
                 new_e[nrow(new_e),"NextVisitPANSSDiff"] =  temp_array[j,"PANSS_Total"] - temp_array[j+1,"PANSS_Total"]
                }
                if(j==nrow(temp_array)){
                 new_e[nrow(new_e),"DaysToNextVisit"] = 0
                 new_e[nrow(new_e),"NextVisitPANSSDiff"] =  0
                }
                new_e[nrow(new_e),"PatientMinPANSS"] = min(temp_array$PANSS_Total)
                new_e[nrow(new_e),"PatientMaxPANSS"] = max(temp_array$PANSS_Total)
                new_e[nrow(new_e),"PatientAveragePANSS"] = mean(temp_array$PANSS_Total)

                if(j>2){
                new_e[nrow(new_e),"SecondLastVisitPANSS"] = temp_array[j-2,"PANSS_Total"]
                new_e[nrow(new_e),"SecondLastVisitDay"] = temp_array[j-2,"VisitDay"]
                }
                if(j<=2){
                new_e[nrow(new_e),"SecondLastVisitPANSS"] = temp_array[j,"PANSS_Total"]
                new_e[nrow(new_e),"SecondLastVisitDay"] = temp_array[j,"VisitDay"]
                }
```

```
if(j>1){
    if(temp_array[j-1,"LeadStatus"]=="Passed") {
      new_e[nrow(new_e),"PreviousLeadStatus"]=(sample(2)-1)[1]
    }else {
      new_e[nrow(new_e),"PreviousLeadStatus"]=(sample(2)-1)[1]
    }
}
if(j==1){
     if(temp_array[1,"LeadStatus"]=="Passed") {
      new_e[nrow(new_e),"PreviousLeadStatus"]=(sample(2)-1)[1]
     } else {
      new_e[nrow(new_e),"PreviousLeadStatus"]=(sample(2)-1)[1]
      }
}
if(j>2){
    if(temp_array[j-2,"LeadStatus"]=="Passed") {
     new_e[nrow(new_e),"Previous2DayLeadStatus"]=(sample(2)-1)[1]
    } else {
     new_e[nrow(new_e),"Previous2DayLeadStatus"]=(sample(2)-1)[1]
    }
}
if(j<=2){
    if(temp_array[1,"LeadStatus"]=="Passed") {
       new_e[nrow(new_e),"Previous2DayLeadStatus"]=(sample(2)-1)[1]
    }else {
       new_e[nrow(new_e),"Previous2DayLeadStatus"]=(sample(2)-1)[1]
    }
}

if(j<nrow(temp_array)){
    if(temp_array[j+1,"LeadStatus"]=="Passed") {
      new_e[nrow(new_e),"NextVisitLeadStatus"]=(sample(2)-1)[1]
    }else {
      new_e[nrow(new_e),"NextVisitLeadStatus"]=(sample(2)-1)[1]
    }
}
if(j==nrow(temp_array)){
    if(temp_array[j,"LeadStatus"]=="Passed") {
      new_e[nrow(new_e),"NextVisitLeadStatus"]=(sample(2)-1)[1]
    }else {
      new_e[nrow(new_e),"NextVisitLeadStatus"]=(sample(2)-1)[1]
    }
}
```

```
            new_e[nrow(new_e),"PANSSDiffWRTFirstDay"] = temp_array[1,"PANSS_Total"] -
temp_array[j,"PANSS_Total"]
            new_e[nrow(new_e),"TxGroup"]=temp_array[j,"TxGroup"]
            new_e[nrow(new_e),"Country"]=temp_array[j,"Country"]
            new_e[nrow(new_e),"PANSS_Total"]=temp_array[j,"PANSS_Total"]
            new_e[nrow(new_e),"PatientID"]=temp_array[j,"PatientID"]
            new_e[nrow(new_e),"AssessmentiD"]=temp_array[j,"AssessmentiD"]

  }
}

predicted_panss_with_lm = predict(panss_total_lm_model,newdata=new_e)
new_e$Predicted_PANSS_LM = predicted_panss_with_lm

new_e$Predicted_vs_ActualPanssDiffLM = new_e$PANSS_Total - new_e$Predicted_PANSS_LM



write.csv(new_e,file="transformed_data_E_to_classify_on.csv", row.names=FALSE, na="")

data_to_predict_on = read.csv("transformed_data_E_to_classify_on.csv", header = T)
levels(data_to_predict_on$Country) = levels(input_data.Train$Country)
levels(data_to_predict_on$TxGroup) = levels(input_data.Train$TxGroup)



predicted_probability = predict(classification_model,newdata=data_to_predict_on,type="response")

data_to_predict_on$LeadStatus = predicted_probability

final_output=data.frame()
for (i in assesment_id_in_studyE) {
  temp_array = data.frame()
  temp_array = data_to_predict_on[data_to_predict_on$AssessmentiD == toString(i),]
  for (k in 1:nrow(temp_array)) {
    if (strtoi(temp_array[k, "AssessmentiD"]) == i) {
        final_output[nrow(final_output)+1,"AssessmentID"]=i
        final_output[nrow(final_output),"LeadStatus"]=temp_array[k, "LeadStatus"]
        break
    }
  }
}
write.csv(final_output,"objective4_output_with_GLM_LAG_Vars.csv",row.names=FALSE, na="")
```

PROGRAM4

```
#SVM model
#Program to create ROC curve in SVM. This includes data transformations done as well.

########DATA TRANSFORMATION##############
#
setwd("C:/Users/msingh13/Documents/STAT202/Class_Project/R_working_dir")
set.seed(1)
dataABCD = read.csv("Study_ABCD.csv", header = T)

data_to_analyze = dataABCD[,-c(1,4:6)]
#Eliminate entries with ERROR country
data_to_analyze = data_to_analyze[!data_to_analyze$Country=="ERROR",]

#For objective 4 we do need LeadStatus so commenting below lines
#For model creation we can omit entries with Lead Status that are not Passed
#data_to_analyze = data_to_analyze[!data_to_analyze$LeadStatus=="Assign to CS",]
#data_to_analyze = data_to_analyze[!data_to_analyze$LeadStatus=="Flagged",]
#data_to_analyze = data_to_analyze[,-c(36)]

data_to_analyze = data_to_analyze[,-c(5:34)]

#*******************

new_abcd = data.frame()
pat_id_in_study =  unique(data_to_analyze$PatientID, incomparables = FALSE)
for (i in pat_id_in_study) {
   count = 0
   temp_array = data.frame()
   temp_array = data_to_analyze[data_to_analyze$PatientID == toString(i),]
   temp_array = temp_array[order(temp_array$VisitDay),]
   temp_array$VisitDay = as.numeric(as.character(temp_array$VisitDay))
   temp_array$PANSS_Total = as.numeric(as.character(temp_array$PANSS_Total))
   visit_number = 1
   visit_week = 1
   previous_visit_panss = 0
   for (j in 1:nrow(temp_array)) {

               new_abcd[nrow(new_abcd)+1,"VisitNumber"] = visit_number
               visit_number = visit_number + 1
               new_abcd[nrow(new_abcd),"VisitWeek"] = strtoi(temp_array[j,"VisitDay"])/7
               new_abcd[nrow(new_abcd),"VisitDay"] = temp_array[j,"VisitDay"]

               if(j>1){
               new_abcd[nrow(new_abcd),"PreviousVisitPANSS"] = temp_array[j-1,"PANSS_Total"]
```

```
        new_abcd[nrow(new_abcd),"PreviousVisitDay"] = temp_array[j-1,"VisitDay"]
        }
        if(j==1){
        new_abcd[nrow(new_abcd),"PreviousVisitPANSS"] = temp_array[j,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"PreviousVisitDay"] = temp_array[j,"VisitDay"]
        }

        if(j<nrow(temp_array)){
        new_abcd[nrow(new_abcd),"NextVisitPANSS"] = temp_array[j+1,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"NextVisitDay"] = temp_array[j+1,"VisitDay"]
        }
        if(j==nrow(temp_array)){
        new_abcd[nrow(new_abcd),"NextVisitPANSS"] = temp_array[j,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"NextVisitDay"] = temp_array[j,"VisitDay"]
        }

        if(j<(nrow(temp_array)-1)) {
         new_abcd[nrow(new_abcd),"Next2ndVisitPANSS"] = temp_array[j+2,"PANSS_Total"]
         new_abcd[nrow(new_abcd),"Next2ndVisitDay"] = temp_array[j+2,"VisitDay"]
        } else {
         #new_abcd[nrow(new_abcd),"Next2ndVisitPANSS"] = temp_array[j,"PANSS_Total"]
         #new_abcd[nrow(new_abcd,"Next2ndVisitDay"] = temp_array[j,"VisitDay"]
         new_abcd[nrow(new_abcd),"Next2ndVisitPANSS"] = temp_array[j,"PANSS_Total"]
         new_abcd[nrow(new_abcd),"Next2ndVisitDay"] = temp_array[j,"VisitDay"]

        }

        if(j>1){
        new_abcd[nrow(new_abcd),"DaysFromPreviousVisit"] = temp_array[j,"VisitDay"] -
new_abcd[nrow(new_abcd),"PreviousVisitDay"]
        new_abcd[nrow(new_abcd),"LastVisitPANSSDiff"] =  new_abcd[nrow(new_abcd),"PreviousVisitPANSS"] -
temp_array[j,"PANSS_Total"]
        }
        if(j==1){
           new_abcd[nrow(new_abcd),"DaysFromPreviousVisit"] = 0
           new_abcd[nrow(new_abcd),"LastVisitPANSSDiff"] = 0
        }

        if(j<nrow(temp_array)){
         new_abcd[nrow(new_abcd),"DaysToNextVisit"] = temp_array[j,"VisitDay"] - temp_array[j+1,"VisitDay"]
         new_abcd[nrow(new_abcd),"NextVisitPANSSDiff"] =  temp_array[j,"PANSS_Total"] -
temp_array[j+1,"PANSS_Total"]
        }
        if(j==nrow(temp_array)){
         new_abcd[nrow(new_abcd),"DaysToNextVisit"] = 0
         new_abcd[nrow(new_abcd),"NextVisitPANSSDiff"] =  0
        }
        new_abcd[nrow(new_abcd),"PatientMinPANSS"] = min(temp_array$PANSS_Total)
        new_abcd[nrow(new_abcd),"PatientMaxPANSS"] = max(temp_array$PANSS_Total)
        new_abcd[nrow(new_abcd),"PatientAveragePANSS"] = mean(temp_array$PANSS_Total)

        if(j>2){
        new_abcd[nrow(new_abcd),"SecondLastVisitPANSS"] = temp_array[j-2,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"SecondLastVisitDay"] = temp_array[j-2,"VisitDay"]
        }
        if(j<=2){
        new_abcd[nrow(new_abcd),"SecondLastVisitPANSS"] = temp_array[j,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"SecondLastVisitDay"] = temp_array[j,"VisitDay"]
        }

        new_abcd[nrow(new_abcd),"PANSSDiffWRTFirstDay"] = temp_array[1,"PANSS_Total"] -
temp_array[j,"PANSS_Total"]
        new_abcd[nrow(new_abcd),"TxGroup"]=temp_array[j,"TxGroup"]
        new_abcd[nrow(new_abcd),"Country"]=temp_array[j,"Country"]
        new_abcd[nrow(new_abcd),"PANSS_Total"]=temp_array[j,"PANSS_Total"]

        if(j>1){
          if(temp_array[j-1,"LeadStatus"]=="Passed") {
           new_abcd[nrow(new_abcd),"PreviousLeadStatus"]=0
          }else {
           new_abcd[nrow(new_abcd),"PreviousLeadStatus"]=1
          }
        }
        if(j==1){
           if(temp_array[1,"LeadStatus"]=="Passed") {
           new_abcd[nrow(new_abcd),"PreviousLeadStatus"]=0
           } else {
           new_abcd[nrow(new_abcd),"PreviousLeadStatus"]=1
            }
        }
        if(j>2){
          if(temp_array[j-2,"LeadStatus"]=="Passed") {
           new_abcd[nrow(new_abcd),"Previous2DayLeadStatus"]=0
          } else {
           new_abcd[nrow(new_abcd),"Previous2DayLeadStatus"]=1
          }
        }
        if(j<=2){
          if(temp_array[1,"LeadStatus"]=="Passed") {
            new_abcd[nrow(new_abcd),"Previous2DayLeadStatus"]=0
          }else {
            new_abcd[nrow(new_abcd),"Previous2DayLeadStatus"]=1
```

```
          }
        }

        if(j<nrow(temp_array)){
          if(temp_array[j+1,"LeadStatus"]=="Passed") {
           new_abcd[nrow(new_abcd),"NextVisitLeadStatus"]=0
          }else {
           new_abcd[nrow(new_abcd),"NextVisitLeadStatus"]=1
          }
        }
        if(j==nrow(temp_array)){
          if(temp_array[j,"LeadStatus"]=="Passed") {
           new_abcd[nrow(new_abcd),"NextVisitLeadStatus"]=0
          }else {
           new_abcd[nrow(new_abcd),"NextVisitLeadStatus"]=1
          }
        }

        if(temp_array[j,"LeadStatus"]=="Passed") {
          new_abcd[nrow(new_abcd),"FlagOrACSLeadStatus"]=0
        }else {
          new_abcd[nrow(new_abcd),"FlagOrACSLeadStatus"]=1
        }
        #if(temp_array[j,"LeadStatus"]=="Passed") {
        #    new_abcd[nrow(new_abcd),"LeadStatusBinary"]=0
        #}else {
        #    new_abcd[nrow(new_abcd),"LeadStatusBinary"]=1
        #}
  }

}
#Will now predict PANSS_Total for each entry with the model created in Objective3
new_abcd$VisitNumber = as.numeric(as.character(new_abcd$VisitNumber))
#library(gbm)
#number_of_trees=6000
#load("panss_total_boosting_model.rda")
#panss_total_boosting_model = model_for_panss_total
#predicted_panss_with_boosting = predict(panss_total_boosting_model,newdata=new_abcd,n.trees=number_of_trees)

#new_abcd$Predicted_PANSS_Boosting = predicted_panss_with_boosting



load("panss_total_LM_model.rda")
panss_total_lm_model = model_for_panss_total
predicted_panss_with_lm = predict(panss_total_lm_model,newdata=new_abcd)
new_abcd$Predicted_PANSS_LM = predicted_panss_with_lm
```

```
new_abcd$Predicted_vs_ActualPanssDiffLM = new_abcd$PANSS_Total - new_abcd$Predicted_PANSS_LM
#new_abcd$Predicted_vs_ActualPanssDiffBoost = new_abcd$PANSS_Total - new_abcd$Predicted_PANSS_Boosting

write.csv(new_abcd,file="transformed_data_ABCD_for_LeadStatus_SVM.csv", row.names=FALSE, na="")



############Creating Model To Predict PANSS_Total###############
transformed_data = read.csv("transformed_data_ABCD_for_LeadStatus_SVM.csv", header = T)

transformed_data$VisitNumber = as.numeric(as.character(transformed_data$VisitNumber))
#transformed_data$VisitNumber = as.factor(as.character(transformed_data$VisitNumber))
transformed_data$TxGroup = as.factor(as.character(transformed_data$TxGroup))
transformed_data$Country = as.factor(as.character(transformed_data$Country))
transformed_data$VisitWeek = as.numeric(as.character(transformed_data$VisitWeek))
transformed_data$PatientAveragePANSS  = as.numeric(as.character(transformed_data$PatientAveragePANSS ))
transformed_data$PreviousVisitPANSS = as.numeric(as.character(transformed_data$PreviousVisitPANSS))
transformed_data$PreviousVisitDay = as.numeric(as.character(transformed_data$PreviousVisitDay))
transformed_data$NextVisitPANSS = as.numeric(as.character(transformed_data$NextVisitPANSS))
transformed_data$NextVisitDay = as.numeric(as.character(transformed_data$NextVisitDay))
transformed_data$DaysFromPreviousVisit= as.numeric(as.character(transformed_data$DaysFromPreviousVisit))
transformed_data$LastVisitPANSSDiff = as.numeric(as.character(transformed_data$LastVisitPANSSDiff))
transformed_data$DaysToNextVisit = as.numeric(as.character(transformed_data$DaysToNextVisit))
transformed_data$NextVisitPANSSDiff = as.numeric(as.character(transformed_data$NextVisitPANSSDiff))
transformed_data$PatientMinPANSS = as.numeric(as.character(transformed_data$PatientMinPANSS))
transformed_data$PatientMaxPANSS = as.numeric(as.character(transformed_data$PatientMaxPANSS))
transformed_data$SecondLastVisitPANSS = as.numeric(as.character(transformed_data$SecondLastVisitPANSS))
transformed_data$SecondLastVisitDay = as.numeric(as.character(transformed_data$SecondLastVisitDay))
transformed_data$PANSSDiffWRTFirstDay =as.numeric(as.character(transformed_data$PANSSDiffWRTFirstDay))
transformed_data$FlagOrACSLeadStatus = as.factor(as.character(transformed_data$FlagOrACSLeadStatus))



panss_total_train_mse = panss_total_test_mse = 0
train = 1:(round(nrow(transformed_data)*0.90))
input_data.Train = data.frame()
input_data.Test = data.frame()
input_data.Train = transformed_data[train,]
input_data.Test = transformed_data[-train,]

#########SVM############
library(e1071)

#tune.out=tune(svm,FlagOrACSLeadStatus~.,
data=input_data.Train,kernel="linear",ranges=list(cost=c(0.01,0.1,1.5,10,20)),probability=TRUE)

svmfit=svm(FlagOrACSLeadStatus ~ .,data=input_data.Train, kernel="linear", cost=0.1,scale=FALSE, probability=TRUE)
```

```
save(svmfit,file="svmfit_model_for_probability_ver1_with_lag.rda")

##WE TUNE THE MODEL
#tune.out=tune(svm,FlagOrACSLeadStatus~.,
data=input_data.Train,kernel="linear",ranges=list(cost=c(0.001,0.01,0.1,1.5,10,100)))

#tune.out=tune(svm,FlagOrACSLeadStatus~.,
data=input_data.Train,kernel="linear",ranges=list(cost=c(0.01,0.1,1.5,10,20)),probability=TRUE)
#svmfit_model_fit=svm(FlagOrACSLeadStatus ~ ., data=input_data.Train, kernel="radial",
cost=1,gamma=2,decision.values=T,probability=TRUE)



######ROC Curve#######
library (ROCR)
 rocplot =function (pred,truth,...){
 predob = prediction(pred,truth)
 perf = performance(predob,"tpr","fpr")
 plot(perf,...)
}

fitted_train=attributes(predict(svmfit,input_data.Train,decision.values=TRUE))$decision.values
par(mfrow=c(1,2))
rocplot(fitted_train,input_data.Train$FlagOrACSLeadStatus,main="Training Data")
fitted_test=attributes(predict(svmfit,input_data.Test,decision.values=TRUE))$decision.values
rocplot(fitted_test,input_data.Test$FlagOrACSLeadStatus,main="Test Data")
######ROC Curve######

#To generate probabilities
#pred_train_prob=predict(svmfit,input_data.Train,probability=TRUE)


#pred_train=predict(svmfit,input_data.Train)
#table(predict=pred_train,truth=input_data.Train$FlagOrACSLeadStatus)
#       truth
#predict     0     1
#      0 15620  3216
#      1     0     0
#mean(pred_train==input_data.Train$FlagOrACSLeadStatus)
#[1] 0.8292631

#       truth
#predict     0     1
#      0 15619  3202
#      1     1    14
#> mean(pred_train==input_data.Train$FlagOrACSLeadStatus)
#[1] 0.8299533
```

```
#pred_test=predict(svmfit,input_data.Test)
#table(predict=pred_test,truth=input_data.Test$FlagOrACSLeadStatus)
#       truth
#predict    0    1
#      0  204 1889
#      1    0    0
#mean(pred_test==input_data.Test$FlagOrACSLeadStatus)
#[1] 0.09746775
#   truth
#predict    0    1
#      0  203 1839
#      1    1   50
#[1] 0.1208791

#attr(pred_train_prob,"prob")[1:10,]
```

PROGRAM5

```
#Probablity prediction using SVM
###########Study E Data Transformation##############

input_studyE = read.csv("Study_E.csv", header = T)

#Getting the patientIDs for interested patients
sample_sub_assesmentid = read.csv("sample_submission_status.csv", header = T)
assesment_id_in_studyE =  unique(sample_sub_assesmentid$AssessmentID, incomparables = FALSE)

patientIDs_for_given_assesmet_ids = data.frame()
for(i in assesment_id_in_studyE) {
        temp_array = data.frame()
        temp_array = input_studyE[input_studyE$AssessmentiD == toString(i),]
        for (j in 1:nrow(temp_array)) {
               patientIDs_for_given_assesmet_ids[nrow(patientIDs_for_given_assesmet_ids)+1,"PatientID"] =
temp_array[j,"PatientID"]
        }
}

patientIDs_for_given_assesmet_ids = unique(patientIDs_for_given_assesmet_ids$PatientID, incomparables = FALSE)



#Interpreting UK as France in the given data.
#There is no training data for country UK so interpreting it as France as they are geographically close countries.
levels(input_studyE$Country) = c(levels(input_studyE$Country),"France")
input_studyE$Country[input_studyE$Country=="UK"] = "France"

new_e = data.frame()
for (i in patientIDs_for_given_assesmet_ids) {
```

```
temp_array = data.frame()
temp_array = input_studyE[input_studyE$PatientID == toString(i),]
temp_array = temp_array[order(temp_array$VisitDay),]
temp_array[,"LeadStatus"] = "Passed"
temp_array$VisitDay = as.numeric(as.character(temp_array$VisitDay))
temp_array$PANSS_Total = as.numeric(as.character(temp_array$PANSS_Total))
visit_number = 1
visit_week = 1
previous_visit_panss = 0
for (j in 1:nrow(temp_array)) {

    new_e[nrow(new_e)+1,"VisitNumber"] = visit_number
    visit_number = visit_number + 1
    new_e[nrow(new_e),"VisitWeek"] = strtoi(temp_array[j,"VisitDay"])/7
    new_e[nrow(new_e),"VisitDay"] = temp_array[j,"VisitDay"]

    if(j>1){
    new_e[nrow(new_e),"PreviousVisitPANSS"] = temp_array[j-1,"PANSS_Total"]
    new_e[nrow(new_e),"PreviousVisitDay"] = temp_array[j-1,"VisitDay"]
    }
    if(j==1){
    new_e[nrow(new_e),"PreviousVisitPANSS"] = temp_array[j,"PANSS_Total"]
    new_e[nrow(new_e),"PreviousVisitDay"] = temp_array[j,"VisitDay"]
    }

    if(j<nrow(temp_array)){
    new_e[nrow(new_e),"NextVisitPANSS"] = temp_array[j+1,"PANSS_Total"]
    new_e[nrow(new_e),"NextVisitDay"] = temp_array[j+1,"VisitDay"]
    }
    if(j==nrow(temp_array)){
    new_e[nrow(new_e),"NextVisitPANSS"] = temp_array[j,"PANSS_Total"]
    new_e[nrow(new_e),"NextVisitDay"] = temp_array[j,"VisitDay"]
    }

     if(j<(nrow(temp_array)-1)) {
     new_e[nrow(new_e),"Next2ndVisitPANSS"] = temp_array[j+2,"PANSS_Total"]
     new_e[nrow(new_e),"Next2ndVisitDay"] = temp_array[j+2,"VisitDay"]
    } else {
     #new_abcd[nrow(new_abcd),"Next2ndVisitPANSS"] = temp_array[j,"PANSS_Total"]
     #new_abcd[nrow(new_abcd,"Next2ndVisitDay"] = temp_array[j,"VisitDay"]
     new_e[nrow(new_e),"Next2ndVisitPANSS"] = temp_array[j,"PANSS_Total"]
     new_e[nrow(new_e),"Next2ndVisitDay"] = temp_array[j,"VisitDay"]

    }
```

```
        if(j>1){
        new_e[nrow(new_e),"DaysFromPreviousVisit"] = temp_array[j,"VisitDay"] -
new_e[nrow(new_e),"PreviousVisitDay"]
        new_e[nrow(new_e),"LastVisitPANSSDiff"] =  new_e[nrow(new_e),"PreviousVisitPANSS"] -
temp_array[j,"PANSS_Total"]
        }
        if(j==1){
          new_e[nrow(new_e),"DaysFromPreviousVisit"] = 0
          new_e[nrow(new_e),"LastVisitPANSSDiff"] = 0
        }

        if(j<nrow(temp_array)){
         new_e[nrow(new_e),"DaysToNextVisit"] = temp_array[j,"VisitDay"] - temp_array[j+1,"VisitDay"]
         new_e[nrow(new_e),"NextVisitPANSSDiff"] =  temp_array[j,"PANSS_Total"] - temp_array[j+1,"PANSS_Total"]
        }
        if(j==nrow(temp_array)){
         new_e[nrow(new_e),"DaysToNextVisit"] = 0
         new_e[nrow(new_e),"NextVisitPANSSDiff"] =  0
        }
        new_e[nrow(new_e),"PatientMinPANSS"] = min(temp_array$PANSS_Total)
        new_e[nrow(new_e),"PatientMaxPANSS"] = max(temp_array$PANSS_Total)
        new_e[nrow(new_e),"PatientAveragePANSS"] = mean(temp_array$PANSS_Total)

        if(j>2){
        new_e[nrow(new_e),"SecondLastVisitPANSS"] = temp_array[j-2,"PANSS_Total"]
        new_e[nrow(new_e),"SecondLastVisitDay"] = temp_array[j-2,"VisitDay"]
        }
        if(j<=2){
        new_e[nrow(new_e),"SecondLastVisitPANSS"] = temp_array[j,"PANSS_Total"]
        new_e[nrow(new_e),"SecondLastVisitDay"] = temp_array[j,"VisitDay"]
        }

        if(j>1){
          if(temp_array[j-1,"LeadStatus"]=="Passed") {
            new_e[nrow(new_e),"PreviousLeadStatus"]=(sample(2)-1)[1]
          }else {
            new_e[nrow(new_e),"PreviousLeadStatus"]=(sample(2)-1)[1]
          }
        }
        if(j==1){
           if(temp_array[1,"LeadStatus"]=="Passed") {
            new_e[nrow(new_e),"PreviousLeadStatus"]=(sample(2)-1)[1]
           } else {
            new_e[nrow(new_e),"PreviousLeadStatus"]=(sample(2)-1)[1]
            }
```

```
                }
                if(j>2){
                    if(temp_array[j-2,"LeadStatus"]=="Passed") {
                     new_e[nrow(new_e),"Previous2DayLeadStatus"]=(sample(2)-1)[1]
                    } else {
                     new_e[nrow(new_e),"Previous2DayLeadStatus"]=(sample(2)-1)[1]
                    }
                }
                if(j<=2){
                    if(temp_array[1,"LeadStatus"]=="Passed") {
                      new_e[nrow(new_e),"Previous2DayLeadStatus"]=(sample(2)-1)[1]
                    }else {
                      new_e[nrow(new_e),"Previous2DayLeadStatus"]=(sample(2)-1)[1]
                    }
                }

                if(j<nrow(temp_array)){
                    if(temp_array[j+1,"LeadStatus"]=="Passed") {
                     new_e[nrow(new_e),"NextVisitLeadStatus"]=(sample(2)-1)[1]
                    }else {
                     new_e[nrow(new_e),"NextVisitLeadStatus"]=(sample(2)-1)[1]
                    }
                }
                if(j==nrow(temp_array)){
                    if(temp_array[j,"LeadStatus"]=="Passed") {
                     new_e[nrow(new_e),"NextVisitLeadStatus"]=(sample(2)-1)[1]
                    }else {
                     new_e[nrow(new_e),"NextVisitLeadStatus"]=(sample(2)-1)[1]
                    }
                }




                new_e[nrow(new_e),"PANSSDiffWRTFirstDay"] = temp_array[1,"PANSS_Total"] -
temp_array[j,"PANSS_Total"]
                new_e[nrow(new_e),"TxGroup"]=temp_array[j,"TxGroup"]
                new_e[nrow(new_e),"Country"]=temp_array[j,"Country"]
                new_e[nrow(new_e),"PANSS_Total"]=temp_array[j,"PANSS_Total"]
                new_e[nrow(new_e),"PatientID"]=temp_array[j,"PatientID"]
                new_e[nrow(new_e),"AssessmentiD"]=temp_array[j,"AssessmentiD"]


    }
}

predicted_panss_with_lm = predict(panss_total_lm_model,newdata=new_e)
```

```
new_e$Predicted_PANSS_LM = predicted_panss_with_lm

new_e$Predicted_vs_ActualPanssDiffLM = new_e$PANSS_Total - new_e$Predicted_PANSS_LM


write.csv(new_e,file="transformed_data_E_to_classify_on_SVM_lag.csv", row.names=FALSE, na="")

data_to_predict_on = read.csv("transformed_data_E_to_classify_on_SVM_lag.csv", header = T)
levels(data_to_predict_on$Country) = levels(input_data.Train$Country)
levels(data_to_predict_on$TxGroup) = levels(input_data.Train$TxGroup)

predict_prob=predict(svmfit,data_to_predict_on,probability=TRUE)
data_to_predict_on$LeadStatus = attr(predict_prob,"prob")[,c(1)]


final_output=data.frame()
for (i in assesment_id_in_studyE) {
  temp_array = data.frame()
  temp_array = data_to_predict_on[data_to_predict_on$AssessmentiD == toString(i),]
  for (k in 1:nrow(temp_array)) {
    if (strtoi(temp_array[k, "AssessmentiD"]) == i) {
        final_output[nrow(final_output)+1,"AssessmentID"]=i
        final_output[nrow(final_output),"LeadStatus"]=temp_array[k, "LeadStatus"]
        break
    }
  }
}
write.csv(final_output,"objective4_output_with_SVM_with_lag.csv",row.names=FALSE, na="")
```