dir()

setwd("~/SchoolWork/Stat702/final")

churn.data = read.csv("churn\_data.csv", header = T)

#####################Exploratory Data Analysis############################

dim(churn.data)

names(churn.data)

# cc = credit card

# [1] "user"                    "churn" "age"                    "housing"

# [5] "credit\_score"            "deposits" "withdrawal"              "purchases\_partners"

# [9] "purchases"               "cc\_taken" "cc\_recommended"          "cc\_disliked"

# [13] "cc\_liked"                "cc\_application\_begin" "app\_downloaded"          "web\_user"

# [17] "app\_web\_user"            "ios\_user" "android\_user"            "registered\_phones"

# [21] "payment\_type"            "waiting\_4\_loan" "cancelled\_loan"          "received\_loan"

# [25] "rejected\_loan"           "zodiac\_sign" "left\_for\_two\_month\_plus" "left\_for\_one\_month"

# [29] "rewards\_earned"          "reward\_rate" "is\_referred"

sum(is.na(churn.data)) #11262

attach(churn.data)

housing = replace(housing, housing=='na', 'NA')

housing = as.factor(housing)

zodiac\_sign = replace(zodiac\_sign, zodiac\_sign == 'na', 'NA')

zodiac\_sign = as.factor(zodiac\_sign)

sum(is.na(zodiac\_sign)) #2159

sum(is.na(housing)) #13860

sum(is.na(age)) #4

sum(is.na(credit\_score)) #8031

sum(is.na(rewards\_earned)) #3227

sum(is.na(churn)) #0

dim(churn.data)

#response is churn

hist(churn, breaks = 2);table(churn)

# churn

# 0     1

# 15826 11174

sapply(churn.data, function(x) sum(is.na(x)))

#####replace na with NA in the variable housing

hist(age);table(age); #max = 91

table(housing)

# housing

# na     O R

# 13860  2171 10969

hist(credit\_score); table(credit\_score); #max =838

hist(deposits);max(deposits); #65

# deposits

# 0     1 2    3 4 5     6 7 8 9    10 11 12 13 14    15 16 17

# 18156  2461 1085   679 454 401   328 266 207 193   175 173 130 134 125   117 98 101

# 18    19 20    21 22 23    24 25 26 27    28 29 30 31 32    33 34 35

# 106    99 79   52 52 43    61 67 52 50    56 38 39 48 39    41 40 43

# 36    37 38    39 40 41    42 43 44 45    46 47 48 49 50    51 52 53

# 38    44 35    41 31 38    38 25 33 20    15 24 24 18 24    16 29 31

# 54    55 56    57 58 59    60 61 62 63    64 65

# 27    32 22    26 19 20    26 4 7 2     2 1

table(deposits);dim(table(deposits)) #66

hist(withdrawal);max(withdrawal)# 29

table(withdrawal)

# withdrawal

# 0     1 2    3 4 5     6 7 8 9    10 11 12 13 14    15 16 17

# 22995  2203 853   434 205 108    65 39 33 18     9 13 6 5 1     4 3 1

# 19    20 24    28 29

# 1     1 1    1 1

hist(purchases\_partners);max(purchases\_partners); #1067

table(purchases\_partners); dim(table(purchases\_partners))# 294

hist(purchases); max(purchases); # 63

table(purchases); dim(table(table(deposits)))# 52

# purchases

# 0     1 2    3 4 5     6 7 8 9    10 11 12 13 14    15 16 17

# 18295  2388 1059   678 445 403   327 262 210 196   179 168 129 134 122   119 94 109

# 18    19 20    21 22 23    24 25 26 27    28 29 30 31 32    33 34 35

# 110    90 71   47 64 45    57 64 48 59    51 38 54 37 42    38 42 39

# 36    37 38    39 40 41    42 43 44 45    46 47 48 49 50    51 52 53

# 38    42 43    40 29 31    40 18 31 25    17 29 16 23 30    26 20 31

# 54    55 56    57 58 59    60 61 62 63

# 21    43 19    19 23 13    15 3 1 1

hist(cc\_taken); max(cc\_taken);#29

table(cc\_taken); dim(table(cc\_taken)) # 12

# cc\_taken

# 0     1 2    3 4 5     6 7 8 10    11 29

# 25705   923 218   75 45 16    11 2 1 2     1 1

hist(cc\_recommended);max(cc\_recommended); #522

table(cc\_recommended); dim(table(cc\_recommended)) #325

hist(cc\_disliked); max(cc\_disliked); #65

table(cc\_disliked); dim(table(cc\_disliked)); #20 sum(cc\_disliked == 0)

# cc\_disliked

# 0     1 2    3 4 5     6 7 8 9    10 11 12 13 15    23 25 59

# 26444   363 78   34 36 11     9 2 3 3     2 3 2 1 3     1 1 2

# 62    65

# 1     1

hist(cc\_liked); max(cc\_liked); #27

table(cc\_liked); dim(table(cc\_liked)) #9

# cc\_liked

# 0     1 2    3 4 8     9 10 27

# 26770   182 26   11 6 1     2 1 1

hist(cc\_application\_begin); max(cc\_application\_begin) #263

table(cc\_application\_begin); dim(table(cc\_application\_begin)) #128

hist(app\_downloaded, breaks = 2); table(app\_downloaded);

# app\_downloaded

# 0     1

# 1283 25717

hist(web\_user, breaks = 2); table(web\_user)

# web\_user

# 0     1

# 10636 16364

hist(app\_web\_user, breaks = 2); table(app\_web\_user);

# app\_web\_user

# 0     1

# 11833 15167

hist(ios\_user, breaks = 2); table(ios\_user);

# ios\_user

# 0     1

# 16364 10636

hist(android\_user, breaks = 2); table(android\_user);

# android\_user

# 0     1

# 11144 15856

hist(registered\_phones, breaks = 2); table(registered\_phones);

# registered\_phones

# 0     2 3    4 5

# 21960  4048 754   183 55

table(payment\_type)

# payment\_type

# Bi-Weekly      Monthly na Semi-Monthly       Weekly

# 12716         2656 3899         2440 5289

hist(waiting\_4\_loan); table(waiting\_4\_loan)

# waiting\_4\_loan

# 0     1

# 26965    35

hist(cancelled\_loan, breaks = 2); table(cancelled\_loan);

# cancelled\_loan

# 0     1

# 26492   508

hist(received\_loan, breaks = 2); table(received\_loan);

# received\_loan

# 0     1

# 26509   491

hist(rejected\_loan, breaks = 2); table(rejected\_loan);

# rejected\_loan

# 0     1

# 26868   132

table(zodiac\_sign)

# zodiac\_sign

# Aquarius       Aries Cancer   Capricorn Gemini         Leo Libra na   Pisces

# 2117        2001 2424         682 2168 2374        2128 2159 2127

# Sagittarius     Scorpio Taurus       Virgo

# 2056        2118 2236        2410

hist(left\_for\_two\_month\_plus, breaks = 2); table(left\_for\_two\_month\_plus);

# left\_for\_two\_month\_plus

# 0     1

# 22317  4683

hist(left\_for\_one\_month, breaks = 2); table(left\_for\_one\_month);

# left\_for\_one\_month

# 0     1

# 26512   488

hist(rewards\_earned); table(rewards\_earned) #max = 114

hist(reward\_rate); table(reward\_rate) #max 4

hist(is\_referred, breaks = 2); table(is\_referred)

# is\_referred

# 0     1

# 18413  8587

###Correlation plot

numeric.var <- sapply(churn.data, is.numeric)

corr.matrix <- cor(churn.data[,numeric.var])

#library(corrplot)

corrplot(corr.matrix, main = '\n\nCorrelation Plot for Numerical Variables')

housing = replace(housing, housing== c('NA', 'O', 'R'), c(0,1,2))

                 table(housing)

#CORRELATION PLOT ON CAT VAR

# cat.var <- sapply(churn.data, is.factor)

# cat.matrix <- cor(churn.data[,cat.var])

# corrplot()

###Logistic Regression

dir()

setwd("~/SchoolWork/Stat702/final")

churn.data = read.csv("churn\_data.csv", header = T)

#####################Exploratory Data Analysis############################

dim(churn.data)

names(churn.data)

# cc = credit card

# [1] "user"                    "churn" "age"                    "housing"

# [5] "credit\_score"            "deposits" "withdrawal"              "purchases\_partners"

# [9] "purchases"               "cc\_taken" "cc\_recommended"          "cc\_disliked"

# [13] "cc\_liked"                "cc\_application\_begin" "app\_downloaded"          "web\_user"

# [17] "app\_web\_user"            "ios\_user" "android\_user"            "registered\_phones"

# [21] "payment\_type"            "waiting\_4\_loan" "cancelled\_loan"          "received\_loan"

# [25] "rejected\_loan"           "zodiac\_sign" "left\_for\_two\_month\_plus" "left\_for\_one\_month"

# [29] "rewards\_earned"          "reward\_rate" "is\_referred"

sum(is.na(churn.data)) #11262

attach(churn.data)

housing = replace(housing, housing=='na', 'NA')

housing = as.factor(housing)

zodiac\_sign = replace(zodiac\_sign, zodiac\_sign == 'na', 'NA')

zodiac\_sign = as.factor(zodiac\_sign)

sum(is.na(zodiac\_sign)) #2159

sum(is.na(housing)) #13860

sum(is.na(age)) #4

sum(is.na(credit\_score)) #8031

sum(is.na(rewards\_earned)) #3227

sum(is.na(churn)) #0

dim(churn.data)

#response is churn

hist(churn, breaks = 2);table(churn)

# churn

# 0     1

# 15826 11174

sapply(churn.data, function(x) sum(is.na(x)))

#####replace na with NA in the variable housing

hist(age);table(age); #max = 91

#hist(log(log(age))) looks better, but costs another log

hist(log(age)) # "good"

table(housing)

# housing

# na     O R

# 13860  2171 10969

hist(credit\_score); table(credit\_score); #max =838

hist(deposits);max(deposits); #65

hist(log(deposits))

# deposits

# 0     1 2    3 4 5     6 7 8 9    10 11 12 13 14    15 16 17

# 18156  2461 1085   679 454 401   328 266 207 193   175 173 130 134 125   117 98 101

# 18    19 20    21 22 23    24 25 26 27    28 29 30 31 32    33 34 35

# 106    99 79   52 52 43    61 67 52 50    56 38 39 48 39    41 40 43

# 36    37 38    39 40 41    42 43 44 45    46 47 48 49 50    51 52 53

# 38    44 35    41 31 38    38 25 33 20    15 24 24 18 24    16 29 31

# 54    55 56    57 58 59    60 61 62 63    64 65

# 27    32 22    26 19 20    26 4 7 2     2 1

table(deposits);dim(table(deposits)) #66

hist(withdrawal);max(withdrawal)# 29

hist(log(withdrawal))

table(withdrawal)

# withdrawal

# 0     1 2    3 4 5     6 7 8 9    10 11 12 13 14    15 16 17

# 22995  2203 853   434 205 108    65 39 33 18     9 13 6 5 1     4 3 1

# 19    20 24    28 29

# 1     1 1    1 1

hist(purchases\_partners);max(purchases\_partners); #1067

hist(log(purchases\_partners))

table(purchases\_partners); dim(table(purchases\_partners))# 294

hist(purchases); max(purchases); # 63

hist(log(purchases))

table(purchases); dim(table(table(deposits)))# 52

# purchases

# 0     1 2    3 4 5     6 7 8 9    10 11 12 13 14    15 16 17

# 18295  2388 1059   678 445 403   327 262 210 196   179 168 129 134 122   119 94 109

# 18    19 20    21 22 23    24 25 26 27    28 29 30 31 32    33 34 35

# 110    90 71   47 64 45    57 64 48 59    51 38 54 37 42    38 42 39

# 36    37 38    39 40 41    42 43 44 45    46 47 48 49 50    51 52 53

# 38    42 43    40 29 31    40 18 31 25    17 29 16 23 30    26 20 31

# 54    55 56    57 58 59    60 61 62 63

# 21    43 19    19 23 13    15 3 1 1

hist(cc\_taken); max(cc\_taken);#29

hist(log(cc\_taken))

table(cc\_taken); dim(table(cc\_taken)) # 12

# cc\_taken

# 0     1 2    3 4 5     6 7 8 10    11 29

# 25705   923 218   75 45 16    11 2 1 2     1 1

hist(cc\_recommended);max(cc\_recommended); #522

hist(log(cc\_recommended))

#####lgrowr<-ifelse(growr[set2]>0, log(growr[set2]+1), -log(-growr[set2]+1)) solve the log NaNs problem

table(cc\_recommended); dim(table(cc\_recommended)) #325

hist(cc\_disliked); max(cc\_disliked); #65

hist(log(cc\_disliked))

table(cc\_disliked); dim(table(cc\_disliked)); #20 sum(cc\_disliked == 0)

# cc\_disliked

# 0     1 2    3 4 5     6 7 8 9    10 11 12 13 15    23 25 59

# 26444   363 78   34 36 11     9 2 3 3     2 3 2 1 3     1 1 2

# 62    65

# 1     1

hist(cc\_liked); max(cc\_liked); #27

hist(log(cc\_liked))

table(cc\_liked); dim(table(cc\_liked)) #9

# cc\_liked

# 0     1 2    3 4 8     9 10 27

# 26770   182 26   11 6 1     2 1 1

hist(cc\_application\_begin); max(cc\_application\_begin)#263

table(cc\_application\_begin); dim(table(cc\_application\_begin)) #128

hist(app\_downloaded, breaks = 2); table(app\_downloaded);

# app\_downloaded

# 0     1

# 1283 25717

hist(web\_user, breaks = 2); table(web\_user)

# web\_user

# 0     1

# 10636 16364

hist(app\_web\_user, breaks = 2); table(app\_web\_user);

# app\_web\_user

# 0     1

# 11833 15167

hist(ios\_user, breaks = 2); table(ios\_user);

# ios\_user

# 0     1

# 16364 10636

hist(android\_user, breaks = 2); table(android\_user);

# android\_user

# 0     1

# 11144 15856

hist(registered\_phones, breaks = 2); table(registered\_phones);

# registered\_phones

# 0     2 3    4 5

# 21960  4048 754   183 55

table(payment\_type)

# payment\_type

# Bi-Weekly      Monthly na Semi-Monthly       Weekly

# 12716         2656 3899         2440 5289

hist(waiting\_4\_loan); table(waiting\_4\_loan)

# waiting\_4\_loan

# 0     1

# 26965    35

hist(cancelled\_loan, breaks = 2); table(cancelled\_loan);

# cancelled\_loan

# 0     1

# 26492   508

hist(received\_loan, breaks = 2); table(received\_loan);

# received\_loan

# 0     1

# 26509   491

hist(rejected\_loan, breaks = 2); table(rejected\_loan);

# rejected\_loan

# 0     1

# 26868   132

table(zodiac\_sign)

hist(as.numeric(zodiac\_sign), breaks = 20)

# zodiac\_sign

# Aquarius       Aries Cancer   Capricorn Gemini         Leo Libra na   Pisces

# 2117        2001 2424         682 2168 2374        2128 2159 2127

# Sagittarius     Scorpio Taurus       Virgo

# 2056        2118 2236        2410

hist(left\_for\_two\_month\_plus, breaks = 2); table(left\_for\_two\_month\_plus);

# left\_for\_two\_month\_plus

# 0     1

# 22317  4683

hist(left\_for\_one\_month, breaks = 2); table(left\_for\_one\_month);

# left\_for\_one\_month

# 0     1

# 26512   488

hist(rewards\_earned); table(rewards\_earned) #max = 114

hist(reward\_rate); table(reward\_rate) #max 4

hist(is\_referred, breaks = 2); table(is\_referred)

# is\_referred

# 0     1

# 18413  8587

###Correlation plot

numeric.var <- sapply(churn.data, is.numeric)

corr.matrix <- cor(churn.data[,numeric.var])

#library(corrplot)

corrplot(corr.matrix, main = '\n\nCorrelation Plot for Numerical Variables')

housing = replace(housing, housing== c('NA', 'O', 'R'), c(0,1,2))

                 table(housing)

#CORRELATION PLOT ON CAT VAR

# cat.var <- sapply(churn.data, is.factor)

# cat.matrix <- cor(churn.data[,cat.var])

# corrplot()

###Logistic Regression

library(caret)

set.seed(2019)

intrain<- createDataPartition(churn,p=0.7,list=FALSE)

training<- churn.data[intrain,]

testing<- churn.data[-intrain,]

LogModel <- glm(churn ~ .,family=binomial(link="logit"),data=training)

print(summary(LogModel))

plot(LogModel)

anova(LogModel, test="Chisq")

LogModel1 <- glm(churn ~ user+age+housing + credit\_score + purchases\_partners+

                  cc\_taken + cc\_recommended + registered\_phones + payment\_type + cancelled\_loan + received\_loan +

                  rejected\_loan + zodiac\_sign + left\_for\_one\_month +

                  rewards\_earned + reward\_rate

                  ,family=binomial(link="logit"),data=training)

summary(LogModel1)

LogModel2 <- glm(churn ~ user+age+housing + credit\_score + purchases\_partners+

                  cc\_taken + cc\_recommended + registered\_phones + payment\_type + cancelled\_loan + received\_loan +

                  rejected\_loan + left\_for\_one\_month +

                  rewards\_earned + reward\_rate

                ,family=binomial(link="logit"),data=training)

summary(LogModel2)

LogTest <- glm(churn ~ user+age+housing + credit\_score + purchases\_partners+

                cc\_taken + cc\_recommended + registered\_phones + payment\_type + cancelled\_loan + received\_loan +

                rejected\_loan + left\_for\_one\_month +

                rewards\_earned + reward\_rate

              ,family=binomial(link="logit"),data=testing)

summary(LogModel2)

anova(LogModel, LogModel2)

anova(LogModel, LogModel1)

anova(LogModel1, LogModel2)

library(MASS)

exp(cbind(OR=coef(LogModel), confint(LogModel)))

#################################

# EDA of features

#################################

churn.data<-read.csv(file.choose(),header=TRUE)

names(churn.data)

set.seed(100)

#remove all missing values

churn.data <- na.omit(churn.data)

#Take log transform of age

churn.data$age<-log(churn.data$age)

#Set age as numerical

churn.data$age<-as.numeric(churn.data$age)

#Set response variable as a factor and add labels

churn.data$churn <- factor(churn.data$churn, levels=0:1, labels=c("!churn", "churn"))

#Set housing as factor

churn.data$housing <- factor(churn.data$housing)

#app\_web\_user correlated to web\_user

#remove app web\_user for now we will modify based on step wise model building

cor.test(~app\_web\_user+web\_user, data=churn.data)

churn.data$app\_web\_user<-NULL

#ios\_user correlated to android\_user

#remove android user for now, we will modify based on step wise model building

cor.test(~ios\_user+android\_user, data=churn.data)

churn.data$android\_user<-NULL

#remove user feature

churn.data$user<-NULL

#Set zodiac\_sign as factor

churn.data$zodiac\_sign<-as.factor(churn.data$zodiac\_sign)

#Change features to factor (5:9) and (11:13)

#Features 5:9 (deposits, withdrawal, purchase partners, purchases, cc taken)

churn.data[,5:9]=sapply(churn.data[,5:9],function(x) replace(x,x>0,1))

#Features 11:13 (cc\_disliked,cc\_liked,cc\_application begin)

churn.data[,11:13]=sapply(churn.data[,11:13],function(x) replace(x,x>0,1))

#set deposits as factor

churn.data$deposits<-as.factor(churn.data$deposits)

#set withdrawal as factor

churn.data$withdrawal<-as.factor(churn.data$withdrawal)

#set purchases as factor

churn.data$purchases<-as.factor(churn.data$purchases)

#set purchases\_partners as factor

churn.data$purchases\_partners<-as.factor(churn.data$purchases\_partners)

#set cc taken as factor

churn.data$cc\_taken<-as.factor(churn.data$cc\_taken)

#set cc\_disliked as factor

churn.data$cc\_disliked<-as.factor(churn.data$cc\_disliked)

#set cc\_liked as factor

churn.data$cc\_liked<-as.factor(churn.data$cc\_liked)

#set cc\_application\_begin as factor

churn.data$cc\_application\_begin<-as.factor(churn.data$cc\_application\_begin)

#set app downladed as factor

churn.data$app\_downloaded<-as.factor(churn.data$app\_downloaded)

#set web user as factor

churn.data$web\_user<-as.factor(churn.data$web\_user)

#Set ios user as factor

churn.data$ios\_user<-as.factor(churn.data$ios\_user)

#registed phone as binary and to factor

churn.data[,17]=sapply(churn.data[,17],function(x) replace(x,x>0,1))

churn.data$registered\_phones<-as.factor(churn.data$registered\_phones)

#waiting for loan as factor

churn.data$waiting\_4\_loan<-as.factor(churn.data$waiting\_4\_loan)

#cancelled loan as factor

churn.data$cancelled\_loan<-as.factor(churn.data$cancelled\_loan)

#Received loan as factor

churn.data$received\_loan<-as.factor(churn.data$received\_loan)

#set rejected loan as factor

churn.data$rejected\_loan<-as.factor(churn.data$rejected\_loan)

#set left for one month as factor

churn.data$left\_for\_one\_month<-as.factor(churn.data$left\_for\_one\_month)

#set left for more than twomonths to factor

churn.data$left\_for\_two\_month\_plus<-as.factor(churn.data$left\_for\_two\_month\_plus)

#set is\_referred as factor

churn.data$is\_referred<-as.factor(churn.data$is\_referred)

#############################################

Stratified sampling for training and test sets

#############################################

#Split the data into training and test set using stratified sampling

set1<-churn.data[churn.data$churn=="churn",]

set0<-churn.data[churn.data$churn=="!churn",]

dim(set1)  # 3214 28

dim(set0)  # 5239 28

3214\*2/3   # 2142.667

5239\*2/3   #3492

training1 <- sample(1:3214,2143)

test1 <- (1:3214)[-training1]

sum((1:3214) == sort(c(training1,test1)))

training0 <- sample(1:5239,3492)

test0 <- (1:5239)[-training0]

sum((1:5239 == sort(c(training0,test0)))) #2788

train <- rbind(set1[training1,], set0[training0,])

test <- rbind(set1[test1,], set0[test0,])

dim(train)

dim(test)

#################################################

CART tree

#################################################

library(rpart)

library(caret)

library("e1071")

set.seed(100)

#Build a large tree, using 10 fold cross validation, and complexity parameter set to 0

my.control= rpart.control(xval=10, cp=0)

churn.tree= rpart(churn~., data = train, method="class", control = my.control)

#Print Complexity Parameter Table

printcp(churn.tree)

#Find tree with smallest xerror

t.1<-churn.tree$cptable[which.min(churn.tree$cptable[,4]),]

t.1

#1 SE rule = xerror + xstd, from tree with smallest cross error

se.rule<-as.numeric(t.1[4]+t.1[5])

se.rule

#Find optimal tree using 1 SE rule

t.2<-churn.tree$cptable[min(which(churn.tree$cptable[,4]<se.rule)),]

t.2

#Obtain Complexity Parameter value for tree pruning

t.3<-churn.tree$cptable[max(which(churn.tree$cptable[,1]>as.numeric(t.2[1]))),]

t.3

alpha<-as.numeric(t.3[1]+t.2[1])/2

alpha

#Prune and obtain optimal tree

optimal.t<- prune.rpart(churn.tree, cp=(alpha))

optimal.t$cptable

#misclassification rate = (false positive + false negative)/total

pred\_val <- predict(optimal.t, newdata= test, type = "class")

table.1<-table(test$churn, pred\_val)

miss.rate<-(table(test$churn,pred\_val)[1,2]+table(test$churn,pred\_val)[2,1])/length(test$churn)

table.1

miss.rate #29% misclassification overall

#false positive and false negative rates

fp.rate<-(table(test$churn,pred\_val)[1,2])/(table(test$churn,pred\_val)[1,2]+table(test$churn,pred\_val)[1,1])

fp.rate #18% false positive

fn.rate<-(table(test$churn,pred\_val)[2,1])/((table(test$churn,pred\_val)[2,1])+table(test$churn,pred\_val)[2,2])

fn.rate #43% false negatives

############################################################################.

#using loss matrix to punish false negative

#Build a large tree using 10 fold cross validation, with loss matrix, and complexity paramter = 0

lmat <- matrix(c(0,1,4,0), nrow=2, byrow=T)

my.control2= rpart.control(xval=10,cp=0)

penalty.fit= rpart(churn~., data = train, control = my.control2, parms = list(loss=lmat), method = "class")

#Find tree with smallest xerror

pt.1<-penalty.fit$cptable[which.min(penalty.fit$cptable[,4]),]

pt.1

#1 SE rule = xerror + xstd, from tree with smallest cross error

se.rule2<-as.numeric(pt.1[4]+pt.1[5])

#Find optimal tree using 1 SE rule

pt.2<-penalty.fit$cptable[min(which(penalty.fit$cptable[,4]<se.rule2)),]

#Obtain Complexity Parameter value for tree pruning

pt.3<-penalty.fit$cptable[max(which(penalty.fit$cptable[,1]>as.numeric(pt.2[1]))),]

alpha3<-as.numeric(pt.3[1]+pt.2[1])/2

#Prune and obtain optimal tree

optimal.pt<- prune.rpart(penalty.fit, cp=(alpha3))

#misclassification rate = (false positive + false negative)/total

pred\_val2 <- predict(optimal.pt, newdata= test, type = "class")

table.2<-table(test$churn, pred\_val2)

miss.rate2<-(table(test$churn,pred\_val2)[1,2]+table(test$churn,pred\_val2)[2,1])/length(test$churn)

table.2

miss.rate2

#false positive and false negative rates

fp.rate2<-(table(test$churn,pred\_val2)[1,2])/(table(test$churn,pred\_val2)[1,2]+table(test$churn,pred\_val2)[1,1])

fn.rate2<-(table(test$churn,pred\_val2)[2,1])/((table(test$churn,pred\_val2)[2,1])+table(test$churn,pred\_val2)[2,2])

fn.rate2 #Spam classified into good emails rate

#[1] 0.2561983

fp.rate2 #Emails getting classified into spam rate

#[1] 0.02473118

############################################################################

#looking at variable importance

varImp(optimal.t) #still need to understand the output and meaning of values

age                      274.345731

app\_downloaded            32.146192

cancelled\_loan            37.078302

cc\_application\_begin     105.394642

cc\_disliked                8.134225

cc\_recommended           954.414916

cc\_taken                  32.583068

credit\_score             396.748638

deposits                  30.427945

housing                   24.344878

ios\_user                   4.870708

is\_referred               11.857004

left\_for\_one\_month         5.290986

left\_for\_two\_month\_plus  184.480538

payment\_type             132.287610

purchases                248.143838

purchases\_partners       586.953656

registered\_phones         64.274117

rejected\_loan             22.610640

reward\_rate             1095.549456

rewards\_earned           911.592322

web\_user                 101.690276

withdrawal                 3.497298

zodiac\_sign              193.549556

cc\_liked                   0.000000

waiting\_4\_loan             0.000000

received\_loan              0.000000

> varImpPlot(optimal.t)

############################################################################Boosting

############################################################################

#cross validation errors 5.3.3 islr

lmat <- matrix(c(0,1,4,0), nrow=2, byrow=T)

library(ada)

#Discrete adaboost using 10 fold xval, cp=0

default=rpart.control(xval=10,cp=0)

fitdis<-ada(churn~.,data=train,iter=50,loss="e",type="discrete", control=default)

print(fitdis) #training error 0.087

#misclassification rate = (false positive + false negative)/total

pred\_val.2 <- predict(fitdis, newdata= test)

table.2<-table(test$churn, pred\_val.2)

miss.rate2<-(table(test$churn,pred\_val.2)[1,2]+table(test$churn,pred\_val.2)[2,1])/length(test$churn)

miss.rate2 #28.63% misclassification overall, discrete adaboost might overfit

#false positive and false negative rates

fp.rate2<-(table(test$churn,pred\_val.2)[1,2])/(table(test$churn,pred\_val.2)[1,2]+table(test$churn,pred\_val.2)[1,1])

fp.rate2 #16.77% false positive

fn.rate2<-(table(test$churn,pred\_val.2)[2,1])/((table(test$churn,pred\_val.2)[2,1])+table(test$churn,pred\_val.2)[2,2])

fn.rate2#47.03% false negatives

#variable importance plot

varplot(fitdis)

vip <- varplot(fitdis,plot.it=FALSE,type="scores")

round(vip,4)

#Real adaboost

fitreal<-ada(churn~.,data=train,iter=50,type="real",

   control=rpart.control(maxdepth=2,cp=-1,minsplit=0))

fitreal

#misclassification rate = (false positive + false negative)/total

pred\_val.3 <- predict(fitreal, newdata= test)

table.3<-table(test$churn, pred\_val.3)

miss.rate3<-(table(test$churn,pred\_val.3)[1,2]+table(test$churn,pred\_val.3)[2,1])/length(test$churn)

miss.rate3 #for real adaboost still at 29.84%

#false positive and false negative rates

fp.rate3<-(table(test$churn,pred\_val.3)[1,2])/(table(test$churn,pred\_val.3)[1,2]+table(test$churn,pred\_val.3)[1,1])

fp.rate3 #15.08% false positive

fn.rate3<-(table(test$churn,pred\_val.3)[2,1])/((table(test$churn,pred\_val.3)[2,1])+table(test$churn,pred\_val.3)[2,2])

fn.rate3#59.43% false negatives

#gentle adaboost

fitgen<-ada(churn~.,data=train,test.x=test[,-1],test.y=test[,1],iter=50,

   type="gentle",

   control=rpart.control(cp=-1,maxdepth=8))

(fitgen)

#misclassification rate = (false positive + false negative)/total

pred\_val.4 <- predict(fitgen, newdata= test)

table.4<-table(test$churn, pred\_val.4)

miss.rate4<-(table(test$churn,pred\_val.4)[1,2]+table(test$churn,pred\_val.4)[2,1])/length(test$churn)

miss.rate4 #still at 29.77%

#false positive and false negative rates

fp.rate4<-(table(test$churn,pred\_val.4)[1,2])/(table(test$churn,pred\_val.4)[1,2]+table(test$churn,pred\_val.4)[1,1])

fp.rate4 #18.17% false positive

fn.rate4<-(table(test$churn,pred\_val.4)[2,1])/((table(test$churn,pred\_val.4)[2,1])+table(test$churn,pred\_val.4)[2,2])

fn.rate4#46.20% false negatives

# training the model

model\_ada<-train(churn~., data=train,method='ada',tuneGrid=grid)

plot(model\_ada)

###################################################################

#Looking at missing values in all the features

sapply(churn.data, function(x) sum(is.na(x)))

sum(is.na(churn.data))

numeric.var <- sapply(churn.data, is.numeric)

corr.matrix <- cor(churn.data[,numeric.var])

corrplot(corr.matrix, main = '\n\nCorrelation Plot for Numerical Variables')

#####redo by kelso#####

setwd("~/SchoolWork/Stat702/final")

churn.data = read.csv("churn.csv", header = T)

names(churn.data)

str(churn.data)

dim(churn.data)

sum(is.na(churn.data))

set.seed(702)

#Take log transform of age

churn.data$age<-log(churn.data$age)

#Set age as numerical

churn.data$age<-as.numeric(churn.data$age)

#Set response variable as a factor and add labels

churn.data$churn <- factor(churn.data$churn, levels=0:1, labels=c("!churn", "churn"))

#Set housing as factor

churn.data$housing <- factor(churn.data$housing)

#app\_web\_user correlated to web\_user

#remove app web\_user for now we will modify based on step wise model building

cor.test(~app\_web\_user+web\_user, data=churn.data)

churn.data$app\_web\_user<-NULL

#ios\_user correlated to android\_user

#remove android user for now, we will modify based on step wise model building

cor.test(~ios\_user+android\_user, data=churn.data)

churn.data$android\_user<-NULL

#remove user feature

churn.data$user<-NULL

#Set zodiac\_sign as factor

churn.data$zodiac\_sign<-as.factor(churn.data$zodiac\_sign)

#Change features to factor (5:9) and (11:13)

#Features 5:9 (deposits, withdrawal, purchase partners, purchases, cc taken)

churn.data[,5:9]=sapply(churn.data[,5:9],function(x) replace(x,x>0,1))

#Features 11:13 (cc\_disliked,cc\_liked,cc\_application begin)

churn.data[,11:13]=sapply(churn.data[,11:13],function(x) replace(x,x>0,1))

#set deposits as factor

churn.data$deposits<-as.factor(churn.data$deposits)

#set withdrawal as factor

churn.data$withdrawal<-as.factor(churn.data$withdrawal)

#set purchases as factor

churn.data$purchases<-as.factor(churn.data$purchases)

#set purchases\_partners as factor

churn.data$purchases\_partners<-as.factor(churn.data$purchases\_partners)

#set cc taken as factor

churn.data$cc\_taken<-as.factor(churn.data$cc\_taken)

#set cc\_disliked as factor

churn.data$cc\_disliked<-as.factor(churn.data$cc\_disliked)

#set cc\_liked as factor

churn.data$cc\_liked<-as.factor(churn.data$cc\_liked)

#set cc\_application\_begin as factor

churn.data$cc\_application\_begin<-as.factor(churn.data$cc\_application\_begin)

#set app downladed as factor

churn.data$app\_downloaded<-as.factor(churn.data$app\_downloaded)

#set web user as factor

churn.data$web\_user<-as.factor(churn.data$web\_user)

#Set ios user as factor

churn.data$ios\_user<-as.factor(churn.data$ios\_user)

#registered phone as binary and to factor

churn.data[,17]=sapply(churn.data[,17],function(x) replace(x,x>0,1))

churn.data$registered\_phones<-as.factor(churn.data$registered\_phones)

#waiting for loan as factor

churn.data$waiting\_4\_loan<-as.factor(churn.data$waiting\_4\_loan)

#cancelled loan as factor

churn.data$cancelled\_loan<-as.factor(churn.data$cancelled\_loan)

#Received loan as factor

churn.data$received\_loan<-as.factor(churn.data$received\_loan)

#set rejected loan as factor

churn.data$rejected\_loan<-as.factor(churn.data$rejected\_loan)

#set left for one month as factor

churn.data$left\_for\_one\_month<-as.factor(churn.data$left\_for\_one\_month)

#set left for more than twomonths to factor

churn.data$left\_for\_two\_month\_plus<-as.factor(churn.data$left\_for\_two\_month\_plus)

#set is\_referred as factor

churn.data$is\_referred<-as.factor(churn.data$is\_referred)

churn.data <- na.omit(churn.data)

dim(churn.data)

summary(churn.data)

sum(is.na(churn.data))

set1<-churn.data[churn.data$churn=="churn",]

set0<-churn.data[churn.data$churn=="!churn",]

dim(set1)  # 3214 28

dim(set0)  # 5239 28

3214\*2/3   # 2142.667

5239\*2/3   #3492

training1 <- sample(1:3214,2143)

test1 <- (1:3214)[-training1]

sum((1:3214) == sort(c(training1,test1)))

training0 <- sample(1:5239,3492)

test0 <- (1:5239)[-training0]

sum((1:5239 == sort(c(training0,test0)))) #2788

train <- rbind(set1[training1,], set0[training0,])

test <- rbind(set1[test1,], set0[test0,])

numeric.var <- sapply(churn.data, is.numeric)

corr.matrix <- cor(churn.data[,numeric.var])

library(corrplot)

corrplot(corr.matrix, main = '\n\nCorrelation Plot for Numerical Variables')

dim(train)

dim(test)

set.seed(100)

logmodel <- glm(churn ~.,family=binomial(link="logit"),data=train)

summary(logmodel)

logmodel1 <- glm(churn~ housing+credit\_score+purchases\_partners+cc\_taken+

                  cc\_recommended+cc\_liked+cc\_application\_begin+web\_user+

                  ios\_user+registered\_phones+payment\_type+received\_loan+

                  rejected\_loan+zodiac\_sign+left\_for\_two\_month\_plus+rewards\_earned+

                  reward\_rate+is\_referred,

                family= binomial(link = "logit"), data = train)

summary(logmodel1)

logmodel2 <- glm(churn~ housing+credit\_score+purchases\_partners+cc\_taken+

                  cc\_recommended+cc\_liked+cc\_application\_begin+web\_user+

                  registered\_phones+received\_loan+

                  rejected\_loan+left\_for\_two\_month\_plus+rewards\_earned+

                  reward\_rate+is\_referred,

                family= binomial(link = "logit"), data = train)

summary(logmodel2)

testmodel  <- glm(churn~ housing+credit\_score+purchases\_partners+cc\_taken+

                   cc\_recommended+cc\_liked+cc\_application\_begin+web\_user+

                   registered\_phones+received\_loan+

                   rejected\_loan+left\_for\_two\_month\_plus+rewards\_earned+

                   reward\_rate+is\_referred,

                 family= binomial(link = "logit"), data = test)

summary(testmodel)

set.seed(702)

##random forest##

library(randomForest)

dim(train)

c.tune <- tuneRF(train[2:28], train$churn, ntreeTry=50, stepFactor=2,

                improve=0.05, trace=TRUE, plot=TRUE, dobest=FALSE, main = "mtry vs OOB error")

# mtry  OOBError

# 3.OOB     3 0.2917480

# 5.OOB     5 0.2857143

# 10.OOB   10 0.2897959

# 5 is the best mtry

c.tune

plot(c.tune)

train.rf <- randomForest(churn~.,data = train, mtry = 5, ntree = 501,norm.votes = F)

test.rf <- randomForest(churn~., data = test, mtry = 5, ntree = 501,norm.votes = F)

print(train.rf)

print(test.rf)

#test.rf OOB estimate of  error rate: 29.42% (misclass rate)

# OOB estimate of  error rate: 29.42%

# Confusion matrix:

#   !churn churn class.error

# !churn   1415 332  0.1900401

# churn     497 574 0.4640523

#varImpPlot(train.rf, main = "Variable Importance of Churn Rates")

varImpPlot(test.rf, main = "Variable Importance of Churn Rates")

#plot(train.rf, main = "MSE vs # of bootstrap Samples")

plot(test.rf, main="OOB error vs. # of boostrap Samples")

### prediction ###

ind<-sample(2,nrow(churn.data),replace=T,prob=c(.6, .4)) #1/2 (.6/.4): training/testing

table(ind)

churn.rf<-randomForest(churn~., data=churn.data[ind==1,])

print(churn.rf) #OOB estimate of  error rate: 27.7%

varImpPlot(churn.rf)

churn.pred <- randomForest(churn~., data = churn.data[ind == 2,])

print(churn.pred)

varImpPlot(churn.pred)

# OOB estimate of  error rate: 28.67%

# Confusion matrix:

#   !churn churn class.error

# !churn   1712 372  0.1785029

# churn     601 709 0.4587786

##rpart comparisons###

library(rpart)

my.control <- rpart.control(xval=10, cp=0)

tree <- rpart(churn ~.,

             data=churn.data, method="class", control=my.control)

plot(tree, margin = .1, uniform = T)

text(tree, use.n = T)

printcp(tree)

plotcp(tree)

0.76602 +0.012997 #0.779017 tree 16, 30 splits, 31 terminal nodes

besttree<-prune(tree,cp=0.0018)

plot(besttree,margin=.1)

text(besttree,use.n=T)

pred.tree<- predict(besttree,newdata=churn.data[ind==2,], type='class')

table(observed=churn.data[ind==2,"churn"],predicted=pred.tree)

nsplits= besttree$cptable[,2]

test\_error= besttree$cptable[,3]

xerror= besttree$cptable[,4]

xstd= besttree$cptable[,5]

plot(nsplits, test\_error, type = 'l', xlab = "Number of splits", ylab = "Test error",

    main = "Test error vs Number of splits")

lines(nsplits, xerror, lty=3, col = "red")

lines(nsplits, xerror+xstd, lty=4, col = "blue")

lines(nsplits, xerror-xstd, lty=4, col = "blue")

legend(17,1, c("test error", "xerror", "+/- 1 xstd"), lty = c(1,3,4))

#Plot of cross validation estimates of errors and training errors vs tree complexity

nsplits= cfit1$cptable[,2]

train\_error= cfit1$cptable[,3]

xerror= cfit1$cptable[,4]

xstd= cfit1$cptable[,5]

plot(nsplits, train\_error, type = 'l',

    xlab = "Number of Splits", ylab = "Error", main = "Training and CrossValidation Error Vs Tree Complexity")

lines(nsplits, xerror, lty=5, col = "red")

lines(nsplits, xerror+xstd, lty=4, col ="blue")

lines(nsplits, xerror-xstd, lty=4, col = "blue")

legend(30,1, c("Training Error", "Xerror", "+/- 1 xstd"), lty = c(1,5,4),col=c("black","red","blue"))