Significant Variables

1. Logistic Regression

Most Significant(\*\*\*)

CHK\_ACCT3

HISTORY4

Second Most Significant(\*\*)

HISTORY3

SAV\_ACCT3

DURATION

INSTALL\_RATE

1. Classification tree\

AMOUNT

CHK\_ACCT3

DURATION

1. Neural Networks

HISTORY4

OTHER\_INSTALL1

TELEPHONE1

Script

R-Code\_for\_German\_Credit

-------------------------------------------

install.packages("adabag")

#1. Preliminary Work

====================

getwd()

setwd("/Users/lovenishgaur/Documents/Predictive\_Analytics\_IIM\_Raipur/Assignment")

# Read in data

creditdata = read.csv("10-GermanCredit.csv")

str(creditdata)

table(creditdata$RESPONSE)

#Converting categorical variables into factors

names = c(2,4,12,13,20,24,28,30,31,32)

creditdata[,names] = lapply(creditdata[,names], as.factor)

str(creditdata)

creditdata = creditdata[,-1] #Removing OBS. variable from the dataset

#Converting qualitative variables into binary dummies

default1 = model.matrix(~ ., data=creditdata[,c(1,3,11:12,19,23,27,29:31)])

str(as.data.frame(default1))

default = as.data.frame(cbind(default1, creditdata))

str(default)

default = default[,-c(1,27,29,37:38,45,49,53,55:57)]

#Setting the baseline

table(default$RESPONSE1)

baseline = 700/(300+700) #70%

baseline

#Partitioning the dataset

library(caTools)

set.seed(1000)

split<- sample.split(default$RESPONSE,SplitRatio=0.7)

train= subset(default,split==TRUE)

valid= subset(default,split==FALSE)

str(train) #700 observantion of 46 variables

str(valid) #300 Observations of 46 variables

#2. Comparing Classification Models

=====================================

#Logistics Regression

train.log <- glm(RESPONSE1 ~., data = train, family = "binomial")

options(scipen = 999)

summary(train.log)

train.log$fitted.values

predictTrain.log = predict(train.log, type="response", data=train)

predictTrain.log

table(train$RESPONSE1, predictTrain.log>0.5)

(113+441)/(113+441+97+49) #79.14%

predictValid.log = predict(train.log, type="response", newdata = valid)

table(valid$RESPONSE, predictValid.log>0.5)

(46+184)/(46+184+44+26) #76.67%

#Classification Trees

library(rpart)

library(rpart.plot)

train.ct = rpart(RESPONSE1 ~ ., data=train, method ="class", control = rpart.control(minbucket=1))

summary(train.ct)

prp(train.ct)

predictTrain.ct = predict(train.ct, data = train, type="class")

table(train$RESPONSE1, predictTrain.ct)

(116+465)/(116+465+94+25) #83.00%

predictValid.ct = predict(train.ct, newdata = valid, type="class")

table(valid$RESPONSE1, predictValid.ct)

(35+184)/(35+184+55+26) #73.00%

#prune the tree

train.ct$cptable[which.min(train.ct$cptable[,"xerror"]),"CP"]

summary(train.ct)

pruned.ct<- prune(train.ct, cp=train.ct$cptable[which.min(train.ct$cptable[,"xerror"]),"CP"])

prp(pruned.ct)

predictTrain.ct.pruned = predict(pruned.ct, data = train, type="class")

table(train$RESPONSE1, predictTrain.ct.pruned)

(94+454)/(116+36+94+454) #78.28%

predictValid.ct = predict(pruned.ct, newdata = valid, type="class")

table(valid$RESPONSE1, predictValid.ct)

(31+180)/(31+180+59+30) #70.33%

#Neural Net Modeling

#Scale data / Normalize the data

maxs = apply(train, 2, max)

mins = apply(train, 2, min)

scaled = as.data.frame(scale(train, center = mins, scale = maxs - mins))

str(scaled)

train = scaled

str(train)

maxs = apply(valid, 2, max)

mins = apply(valid, 2, min)

scaled1 <- as.data.frame(scale(valid, center = mins, scale = maxs - mins))

str(scaled1)

valid = scaled1

str(valid)

#neuralnet analysis

library(neuralnet)

library(grid)

library(MASS)

library(caret)

str(default)

train$response = train$RESPONSE1

train = train[,-25]

str(train)

valid$response = valid$RESPONSE1

valid = valid[,-25]

str(valid)

#Converting categorical variables into factors

n = names(train)

f = as.formula(paste("response ~", paste(n[!n %in% "response"], collapse = " + ")))

train.nn = neuralnet(f, data=train, hidden=1, threshold = 0.01, linear.output=FALSE) #linear.output = False for classication and true for regression

summary(train.nn)

train.nn$result.matrix #summary of results: matrix containing the error, reached threshold, needed steps, AIC and BIC and estimated weights for each replication.

train.nn$startweights

plot(train.nn)

train.nnet = nnet(f, data=train, size =20, hidden=1, threshold = 0.01, linear.output=FALSE)

predict.train.nn = compute(train.nn, train[,1:45])

table(train$response, predict.train.nn$net.result>0.50)

(149+456)/(149+456+61+34) #86.42%

predict.valid.nn = compute(train.nn, valid[,1:45])

table(valid$response, predict.valid.nn$net.result>0.50)

(45+181)/(45+181+29+45) #75.33%

train.nnet = nnet(f, data=train, size =20, hidden=1, threshold = 0.01, linear.output=FALSE)

nnet

str(train.nnet)

summary(train.nnet)

topmodel = varImp(train.nnet) #Finding important predictors

head(topmodel,30)

predict.train.nnet = predict(train.nnet, data=train)

table(train$response, predict.train.nnet>0.50)

(203+489)/(203+489+7+1) #98.85%

predict.test.nnet = predict(train.nnet, newdata=valid, method="class", na.rm=TRUE)

table(valid$response, predict.test.nnet>0.50)

(47+168)/(47+168+42+43) #71.66%

#4. Plotting ROC Curve and Calculating AUC

library(ROCR)

predictROC = predict(train.nnet, newdata = valid)

predictROC

pred = prediction(predictROC, valid$response)

perf = performance(pred, "tpr", "fpr")

perf

plot(perf, col=rainbow(10), lty=2)

as.numeric(performance(pred, "auc")@y.values)

#3.Improving Predictive Accuracy using Voting based Ensembles

=============================================================

pred.log = as.data.frame(predictValid.log)

pred.log = ifelse(pred.log<0.5,0,1)

pred.ct = as.numeric(predictValid.ct)-1

pred.ct = as.data.frame(pred.ct)

pred.nn = as.data.frame(predict.valid.nn$net.result)

pred.nn = ifelse(pred.nn<0.5,0,1)

ensemble = cbind(actual = valid$response, log = pred.log, ct = pred.ct, nn = pred.nn)

str(ensemble)

names(ensemble)[1:4] = c("Actual","Logistic","ClassificationTree","NeuralNet")

ensemble$vote = ifelse((ensemble$Logistic+ensemble$ClassificationTree+ensemble$NeuralNet)>1,1,0)

table(ensemble$Actual,ensemble$vote)

(42+188)/(42+188+48+22) #76.67

table(ensemble$Actual, ensemble$Logistic)

(46+184)/(46+184+44+26) #76.67%

table(ensemble$Actual, ensemble$ClassificationTree)

(31+180)/(31+180+59+30) #70.33%

table(ensemble$Actual, ensemble$NeuralNet)

(45+181)/(45+181+29+45) #75.33%

#4. Improving Predictive Accuracy using Bagging and Boosting (For Tree Models)

==============================================================================

library(adabag)

library(caTools)

set.seed(1000)

split<- sample.split(creditdata$RESPONSE,SplitRatio=0.7)

train= subset(creditdata,split==TRUE)

valid= subset(creditdata,split==FALSE)

str(train) #700 observantion of 32 variables

str(valid) #300 Observations or 32 variables

#single tree

ensemble.ct = rpart(RESPONSE ~ ., data=train)

pred = predict(ensemble.ct, valid, type="class")

confusionMatrix(pred, valid$RESPONSE) #72.00%

# bagging (Bagged Trees)

bag = bagging(RESPONSE ~ ., data = train)

pred = predict(bag, valid, type = "class")

confusionMatrix(as.factor(pred$class), valid$RESPONSE) #75.67%

# boosting (Boosted Trees)

boost = boosting(RESPONSE ~ ., data = creditdata)

pred = predict(boost, valid, type = "class")

confusionMatrix(as.factor(pred$class), valid$RESPONSE) #100.00%

boost = boosting(RESPONSE ~ ., data = train)

pred = predict(boost, valid, type = "class")

confusionMatrix(as.factor(pred$class), valid$RESPONSE) #75.33%

#5. Improving Predictive Accuracy using Oversampling

====================================================

creditdata\_0 = subset(creditdata, creditdata$RESPONSE=='0')

str(creditdata\_0)

creditdata\_1 = subset(creditdata, creditdata$RESPONSE=='1')

str(creditdata\_1)

set.seed(1000)

split\_0 = sample.split(creditdata\_0$RESPONSE, SplitRatio = 0.50)

train = subset(creditdata\_0, split\_0==TRUE)

str(train)

split\_1 = sample.split(creditdata\_1$RESPONSE, SplitRatio=0.215)

train1 = subset(creditdata\_1, split\_1==TRUE)

str(train1)

train = rbind(train, train1)

str(train)

valid = subset(creditdata\_0, split\_0==FALSE)

str(valid)

valid1 = subset(creditdata\_1, split\_1==FALSE)

str(valid1)

split\_test = sample.split(valid1$RESPONSE, SplitRatio=0.63)

valid1 = subset(valid1, split\_test==TRUE)

str(valid1)

valid = rbind(valid, valid1)

str(valid)

str(valid)

# Logistics Regression Model

train.log.model2 <- glm(RESPONSE ~., data = train, family = "binomial")

summary(train.log.model2)

predictTrain = predict(train.log.model2, type="response", data=train)

predictTrain

table(train$RESPONSE, predictTrain>0.5)

(116+121)/(116+121+34+29) #80.66%

predictValid = predict(train.log.model2, type="response", newdata=valid)

predictValid

table(valid$RESPONSE, predictValid >0.5)

(106+228)/(106+228+44+118) #75.60%

library(ROCR)

ROCRpred = prediction(predictValid, valid$RESPONSE)

ROCRperf = performance(ROCRpred, "tpr","fpr")

ROCRperf

plot(ROCRperf, col="black", lty=2, lwd=1)

plot(ROCRperf, col=rainbow(1))

as.numeric(performance(ROCRpred, "auc")@y.values)