**ISYE -6501 Week 1 Homework**

**Answer: 2.1**

I work for the support team of a Product based software company which deals with Service Requests raised by customers.

Objective of Analysis: We have been building an analytical model that will predict the renewal of sold product based on the Service it got from us.

Train data: Persons who have already renewed the products.

Response variable: Whether the customer will renew the product or not. (1 or 0)

Variables/attributes/parameters:

1. CSAT Survey
2. Service request Time to Resolve.
3. Escalations.
4. No of time customer wrote a mail to us.
5. No of persons involved in resolving the issue.

#-------------------- Answer 2.2: 1st part----------------------

#installing packages

install.packages("kernlab")

#loading packages

library(kernlab)

#Reading Data

data1 <- read.csv("credit\_card\_data-headers.csv")

head(data1)

colnames(data1)[colnames(data1)=="ï..A1"] <- "A1"

#Building Model

model\_svm <- ksvm(as.matrix(data1[,1:10]),data1[,11],type="C-svc",kernel="vanilladot", C=500,scaled=TRUE)

#Predicting Values

pred <- predict(model\_svm,as.matrix(data1[,1:10]))

pred

#building Confusion Matrix

result <- table(predictions= pred,actual =data1[,11] )

result

#Calculating Accuracy

accuracy <- (result[1,1]+result[2,2])/(result[1,1]+result[2,2]+result[1,2]+result[2,1])

accuracy

#With C=100, accuracy is 0.8639144

#With C=1000, accuracy is 0.8623853

#With C=5000, accuracy is 0.8623853

#With C=500, accuracy is 0.8639144

#Finding Coefficients

a <- colSums(model\_svm@xmatrix[[1]] \* model\_svm@coef[[1]])

a

a0 <-model\_svm@b

a0

#-0.08158492

#---------------- Answer 2.2: 2nd part (Optional)--------------------

#Building Non-Linear Model

model\_svm\_nl <- ksvm(as.matrix(data1[,1:10]),data1[,11],type="C-svc",kernel="rbfdot", C=500,scaled=TRUE)

#Predicting Values

pred\_nl <- predict(model\_svm\_nl,as.matrix(data1[,1:10]))

pred\_nl

#building Confusion Matrix

result\_nl <- table(predictions= pred\_nl,actual =data1[,11] )

result\_nl

#Calculating Accuracy

accuracy\_nl <- (result\_nl[1,1]+result\_nl[2,2])/(result\_nl[1,1]+result\_nl[2,2]+result\_nl[1,2]+result\_nl[2,1])

accuracy\_nl

#0.9770642

#------------------- Answer 2.2: 3rd part-----------------

attach(data1)

install.packages("kknn")

library(kknn)

#Dividing into Train & Test

train\_index <- sample(seq\_len(nrow(data1)), size = floor(0.75 \* nrow(data1)))

train\_data <- data1[train\_index,]

test\_data <- data1[-train\_index,]

#Building Model

model\_kknn <- kknn(formula = formula(R1~.),train = train\_data ,test = test\_data ,k = 17,scale = TRUE)

fit <- fitted(model\_kknn)

#taking different cut-offs for defining 1/0

fit1 <- ifelse(fit>.6,1,0)

fit2 <- ifelse(fit>.7,1,0)

fit3 <- ifelse(fit>.8,1,0)

fit4 <- ifelse(fit>.9,1,0)

#Saving the 4 different cut-off results in 4 diff variables

result\_fit1 <- table(test\_data$R1, fit1)

result\_fit2 <- table(test\_data$R1, fit2)

result\_fit3 <- table(test\_data$R1, fit3)

result\_fit4 <- table(test\_data$R1, fit4)

#Calculating accuracy of all 4 cut-offs

accuracy\_fit1 <- (result\_fit1[1,1]+result\_fit1[2,2])/(result\_fit1[1,1]+result\_fit1[2,2]+result\_fit1[1,2]+result\_fit1[2,1])

accuracy\_fit2 <- (result\_fit2[1,1]+result\_fit2[2,2])/(result\_fit2[1,1]+result\_fit2[2,2]+result\_fit2[1,2]+result\_fit2[2,1])

accuracy\_fit3 <- (result\_fit3[1,1]+result\_fit3[2,2])/(result\_fit3[1,1]+result\_fit3[2,2]+result\_fit3[1,2]+result\_fit3[2,1])

accuracy\_fit4 <- (result\_fit4[1,1]+result\_fit4[2,2])/(result\_fit4[1,1]+result\_fit4[2,2]+result\_fit4[1,2]+result\_fit4[2,1])

accuracy\_fit1 #0.8414634 Maximum Accuracy is coming for cut-off as .6 or .7

accuracy\_fit2 #0.8414634

accuracy\_fit3 #0.7560976

accuracy\_fit4 #0.7073171

#Finding the optimal value of K

#At k = 5, Highest accuracy = 0.8170732

#At k = 11, Highest accuracy = 0.8353659

#At k = 17, Highest accuracy = 0.8414634

#At k = 21, Highest accuracy = 0.8414634

#At k = 25, Highest accuracy = 0.8353659

# As we can see highest accuracy is coming with k = 17 or 21.

#-----------------------Answer 3.1---------------------------

#--------------Part a: Cross Validation

#For KNN:

install.packages("class")

library(class)

knn\_cv <- knn.cv(data1[,1:10], data1[,11],constrain = nrow(data1), k = 11)

result\_knn\_cv <-table(predicted= knn\_cv,actual =data1[,11] )

accuracy\_knn\_cv <- (result\_knn\_cv[1,1]+result\_knn\_cv[2,2])/(result\_knn\_cv[1,1]+result\_knn\_cv[2,2]+result\_knn\_cv[1,2]+result\_knn\_cv[2,1])

accuracy\_knn\_cv

#-------------Part b: Splitting the Data

#Dividing into Train , Test & Validation

#Taking 50% data in train

train\_index <- sample(seq\_len(nrow(data1)), size = floor(0.50 \* nrow(data1)))

train\_data <- data1[train\_index,]

nrow(train\_data)

#taking 50% of remaining data in validation

remaining\_data <- data1[-train\_index,]

valid\_index <- sample(seq\_len(nrow(remaining\_data)), size = floor(0.50 \* nrow(remaining\_data)))

valid\_data <- data1[valid\_index,]

nrow(valid\_data)

#Taking remaining data as test data

test\_data <- remaining\_data[-valid\_index,]

nrow(test\_data)

#Building Model SVM

model\_svm\_split <- ksvm(as.matrix(train\_data[,1:10]),train\_data[,11],type="C-svc",kernel="vanilladot", C=500,scaled=TRUE)

#Predicting Values on Valid

pred\_valid <- predict(model\_svm,as.matrix(valid\_data[,1:10]))

pred\_valid

#Validation Confusion Matrix

result\_valid <- table(predictions= pred\_valid,actual =valid\_data[,11] )

result\_valid

#Calculating Accuracy

accuracy\_valid <- (result\_valid[1,1]+result\_valid[2,2])/(result\_valid[1,1]+result\_valid[2,2]+result\_valid[1,2]+result\_valid[2,1])

accuracy\_valid #0.8343558

#Predicting Values on Test

pred\_test <- predict(model\_svm,as.matrix(test\_data[,1:10]))

pred\_test

#Test Data Confusion Matrix

result\_test <- table(predictions= pred\_test,actual =test\_data[,11] )

result\_test

#Calculating Accuracy

accuracy\_test <- (result\_test[1,1]+result\_test[2,2])/(result\_test[1,1]+result\_test[2,2]+result\_test[1,2]+result\_test[2,1])

accuracy\_test #0.847561