**1. Use the below given data set**

**DataSet**

Ans:

df <- read.csv("https://archive.ics.uci.edu/ml/machine-learning-databases/00273/Example\_WearableComputing\_weight\_lifting\_exercises\_biceps\_curl\_variations.csv",

header = FALSE)

data<-df[-1,]

head(data)

colnames(data) <- as.character(unlist(data[1,]))

data1 = data[-1, ]

data1<-as.data.frame(lapply(data1, as.integer))

table(data1$classe)

data1$classe<-as.factor(data1$classe)

head(data1)

replacement\_vec <- c("A", "B", "C", "D","E")

levels(data1$classe) <- replacement\_vec

**2. Perform the below given activities:**

**a. Create classification model using different classifiers**

Ans:

##naivebayes method

library(naivebayes)

model<-naive\_bayes(classe~.,data = data1)

plot(model)

#predictions

p<-predict(model,data1,type = "prob")

head(cbind(data1,p))

#confision matrix

p1<-predict(model,data1)

(tab1<-table(p1,data1$classe))

#accauracy

sum(diag(tab1))/sum(tab1)

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

#knn method

set.seed(1234)

pd<-sample(2,nrow(data1),replace = TRUE,prob = c(0.7,0.3))

train<-data1[pd==1,]

test<-data1[pd==2,]

train.def <- train$classe

test.def <- test$classe

library(class)

knn.1 <- knn(train, test, train.def, k=1)

knn.5 <- knn(train, test, train.def, k=5)

knn.20 <- knn(train, test, train.def, k=20)

tab1<-table(knn.1 ,test.def)

#accauracy

sum(diag(tab1))/sum(tab1)

tab2<-table(knn.5 ,test.def)

#accauracy

sum(diag(tab2))/sum(tab2)

tab3<-table(knn.20 ,test.def)

#accauracy

sum(diag(tab3))/sum(tab3)

**b. Verify model goodness of fit**

Ans:

#goodness of fit naivebayes method

actual<-table(data1$classe)

expected<-table(predict(model))

chisq.test(actual, p = expected/sum(expected))

**c. Apply all the model validation techniques.**

Ans:

#knn method

#libraries

library(caret)

library(pROC)

library(mlbench)

trControl<-trainControl(method = "repeatedcv",

number = 10,

repeats = 3)

knnreg(classe~.,data=data1)

fit<-train(classe~.,

data=data1,

method="knn",

tuneLength=20,

trControl=trControl,

preProc=c("center","scale"))

##naivebayes method

fit1<-train(classe~.,

data=data1,

method="naive\_bayes",

tuneLength=20,

trControl=trControl,

preProc=c("center","scale"))