```
#vQuestion-1

install.packages("mlbench",dependencies = TRUE)
install.packages("caret",dependencies = TRUE)  # used for creating confusion matrix for
classification model
require(mlbench)

data(HouseVotes84)
HouseVotes84
class(HouseVotes84)
View(HouseVotes84)

# DATA IMPUTATION ( replace missing NAs )

# Using KNN Object to fill in the missing NAs in the columns

myData_knn <-
kNN(HouseVotes84,variable=c("V1","V2","V3","V4","V5","V6","V7","V8","V9","V10","V11","V12","V13","V14
","V15","V16"),k=5)
View(myData_knn)


require(e1071)
# In order to create unbiased training model, equal no. of Democrats and republicans shall be fed
into the model.

# Subsetting all Democrats
myData_Democrats <- subset(myData_knn,Class=="democrat")
View(myData_Democrats)
nrow(myData_Democrats)  # 267 democrats are there


#Subsetting all Republican
myData_Republican <- subset(myData_knn,Class=="republican")
View(myData_Republican)
nrow(myData_Republican)  # 168 republican are there


#  Creating training set

# taking 150 democrats from myData_Democrats and 150 republican from myData_Republican to create the
unbiased training set
# overall 300 data sets for training set

myData_Democrats_training <- head(myData_Democrats,150)
View(myData_Democrats_training)
length(myData_Democrats_training) # total 33 columns after kNN Imputation
#myData_Democrats_training <- myData_Democrats_training[,c((18:length(myData_Democrats_training)),1)]
myData_Democrats_training <- myData_Democrats_training[,c(2:17,1)]
myData_Democrats_training


myData_Republican_training <- head(myData_Republican,150)
length(myData_Republican_training) # total 33 columns after kNN Imputation
#myData_Republican_training <-
myData_Republican_training[,c((18:length(myData_Republican_training)),1)]
myData_Republican_training <- myData_Republican_training[,c(2:17,1)]
myData_Republican_training



my_training_set <- rbind(myData_Democrats_training,myData_Republican_training)
View(my_training_set)
```

```
nrow(my_training_set)


#  Creating Test Data

myData_Democrats_test <- tail(myData_Democrats,117)
nrow(myData_Democrats_test)
#myData_Democrats_test <- myData_Democrats_test[,c((18:length(myData_Democrats_test)),1)]
myData_Democrats_test <- myData_Democrats_test[,c(2:17,1)]

myData_Republican_test <- tail(myData_Republican,18)
nrow(myData_Republican_test)
#myData_Republican_test <- myData_Republican_test[,c((18:length(myData_Republican_test)),1)]
myData_Republican_test <- myData_Republican_test[,c(2:17,1)]



my_test_set <- rbind(myData_Democrats_test,myData_Republican_test)
View(my_test_set)
nrow(my_test_set)


# Creating the model


################ Naive Bayesian Classifier #########


# create the model on the training set first
classification_model_naiveB <-
naiveBayes(Class~V1+V2+V3+V4+V5+V6+V7+V8+V9+V10+V11+V12+V13+V14+V15+V16,my_training_set)


#Predict using above model which was created out of training data and use it for test data
prediction_results_naiveB <- predict(classification_model_naiveB,my_test_set)


#comparing test data ( actual ) with that of predicted result from the same test data
compare_predictionResult_with_testData_naiveB <-
as.data.frame(cbind(my_test_set$Class,prediction_results_naiveB))
colnames(compare_predictionResult_with_testData_naiveB) <- c("actual","predicted")
View(compare_predictionResult_with_testData_naiveB)


#`creating Confusion Matrix `( for above model )`````

# process 1 - manual creation of confusion matrix
table(prediction_results_naiveB,my_test_set$Class)

#Prediction   democrat republican
#democrat        105          1
#republican       12         17


# process 2 - Using Caret library to use its pre defined confusionMatrix ()
require(caret)
confusionMatrix(prediction_results_naiveB,my_test_set$Class)

#Confusion Matrix and Statistics

#Reference
#Prediction   democrat republican
#democrat        105          1
#republican       12         17
```

```
#Accuracy : 0.9037
#95% CI : (0.841, 0.9477)
#No Information Rate : 0.8667
#P-Value [Acc > NIR] : 0.124783

#Kappa : 0.6689
#Mcnemar's Test P-Value : 0.005546

#            Sensitivity : 0.8974
#            Specificity : 0.9444
#         Pos Pred Value : 0.9906
#         Neg Pred Value : 0.5862
#             Prevalence : 0.8667
#         Detection Rate : 0.7778
#   Detection Prevalence : 0.7852
#      Balanced Accuracy : 0.9209

#       'Positive' Class : democrat

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

  Question-2


data("glow500")
glow500
head(glow500, n = 10)
summary(glow500)

summary(mod2.2 <- glm(fracture ~ age + weight + priorfrac +
                        premeno + raterisk,
                         family = binomial,
                         data = glow500))

summary(mod2.3 <- update(mod2.2, . ~ . - weight - premeno))
vcov(mod2.3)
contrasts(glow500$raterisk)
contrasts(glow500$raterisk) <- matrix(c(-1,-1,1,0,0,1), byrow= TRUE, ncol = 2)
contrasts(glow500$raterisk)
rm(glow500)
rm


glow_rand.rda
data(glow_rand)
glow_rand

#install plotRoc package

plotROC::(glow_rand)
ggroc(rocdata, fpf_string = "FPF", tpf_string = "TPF", c_string = "c",
      ci = FALSE, label = NULL, label.adj.x = 0, label.adj.y = 0,
      label.angle = 45, plotmath = FALSE, xlabel = "False positive fraction",
      ylabel = "True positive fraction")
```