```
# installing the packages needed for logistic regression

#install.packages(c("ROCR", "caTools", "MASS"))
library(dplyr)
library(ggplot2)
library(GGally)
library(caTools)
library(ROCR)
library(MASS)
library(car)

#Reading the file
framingham<- read.csv("framingham.csv")

#Setting seed value for the random function
set.seed(7)

#splitting dataset into test and training sets
split <- sample.split(framingham$TenYearCHD, SplitRatio = 0.7)
train <- filter(framingham, split == TRUE)
test <- filter(framingham, split == FALSE)

str(train)

#The number of people that have the disease in the training dataset
table(train$TenYearCHD)

#Creating the first logistic regression model using all the variables
logmodel <- glm(TenYearCHD ~ male + age + education + currentSmoker +
cigsPerDay + BPMeds + prevalentStroke + prevalentHyp +
                diabetes + sysBP + diaBP + BMI + heartRate + glucose +
totChol, data=train, family='binomial' )
summary(logmodel)

# removing heartRate
logmodel1 <- glm(TenYearCHD ~ male + age + education + currentSmoker +
cigsPerDay + BPMeds + prevalentStroke + prevalentHyp +
                  diabetes + sysBP + diaBP + BMI + glucose + totChol,
data=train, family='binomial' )
summary(logmodel1)

#removing current smoker

logmodel2 <- glm(TenYearCHD ~ male + age + education + cigsPerDay +
BPMeds + prevalentStroke + prevalentHyp +
                   diabetes + sysBP + diaBP + BMI + glucose + totChol,
data=train, family='binomial' )
summary(logmodel2)
```

```
#removing diabetes

logmodel3 <- glm(TenYearCHD ~ male + age + education + cigsPerDay +
BPMeds + prevalentStroke + prevalentHyp
                    + sysBP + diaBP + BMI + glucose + totChol,
data=train, family='binomial' )
summary(logmodel3)

#removing education

logmodel4 <- glm(TenYearCHD ~ male + age + cigsPerDay + BPMeds +
prevalentStroke + prevalentHyp
                  + sysBP + diaBP + BMI + glucose + totChol, data=train,
family='binomial' )
summary(logmodel4)


#removing BMI

logmodel5 <- glm(TenYearCHD ~ male + age + cigsPerDay + BPMeds +
prevalentStroke + prevalentHyp
                  + sysBP + diaBP + glucose + totChol, data=train,
family='binomial' )
summary(logmodel5)

#removing BPMeds

logmodel6 <- glm(TenYearCHD ~ male + age + cigsPerDay + prevalentStroke
+ prevalentHyp
                  + sysBP + diaBP + glucose + totChol, data=train,
family='binomial' )
summary(logmodel6)

#removing diaBP

logmodel7 <- glm(TenYearCHD ~ male + age + cigsPerDay + prevalentStroke
+ prevalentHyp
                  + sysBP + glucose + totChol, data=train,
family='binomial' )
summary(logmodel7)


#Predicting using final model
pred <- predict(logmodel7, newdata=test, type="response")
summary(pred)
str(pred)

#confusion matrix using the test dataset when outcome is not affected by
patient's decision
table(test$TenYearCHD, pred > 0.16)
```

```
#confusion matrix using the test dataset when outcome is affected by
patient's decision
#table(test$TenYearCHD, pred > 0.05)

# calculating accuracy
accuracy <- (675+97)/nrow(test)

# TPR
tpr <- 97/(97+70)

#FPR
fpr <- 237/(693+237)

# Baseline Test Set
table(test$TenYearCHD, pred >1)

#Baseline model accuracy, TPR, FPR
base_acc <- 930/nrow(test)
base_tpr <- 0/(930)

#predicting likelihood of new patient being at risk for CHD
new_obs <- data.frame(male = 0, age = 51, cigsPerDay = 20,
prevalentStroke = 0, prevalentHyp = 1,
                      sysBP = 140, glucose = 78, totChol =220, education
= 'College',currentSmoker = 1, BPMeds = 0,diabetes = 0, diaBP = 100
                      ,BMI = 31, heartRate = 59)
New_pred <- predict(logmodel, newdata=new_obs, type='response')
New_pred

#predicting likelihood of new patient being at risk for CHD for only
significant variables
new_obs1 <- data.frame(male = 0, age = 51, cigsPerDay = 20,
prevalentStroke = 0, prevalentHyp = 1,
                      sysBP = 140, glucose = 78, totChol =220)
New_pred1 <- predict(logmodel7, newdata=new_obs1, type='response')
New_pred1

#ROC Curves
rocr.log.pred <- prediction(pred, test$TenYearCHD)
logPerformance <- performance(rocr.log.pred, "tpr", "fpr")
plot(logPerformance, colorize = TRUE)
abline(0, 1)
as.numeric(performance(rocr.log.pred, "auc")@y.values)
```