Data 558 - Homework 1

Shrusti Ghela

4/9/2022

1. Suppose that you are interested in performing regression on a particular dataset, in order to answer a particular scientific question. You need to decide whether to take a parametric or a non-parametric approach.

Here, we are interested in solving $Y = f(X) + \epsilon$, where

- X represents the input variables (a.k.a predictors/independent variables/features/variables)
- (For a regression problem) Y (output/dependent/response variable) is quantitative
- f is some unknown function (f represents the systematic information that X provides about Y)
- ϵ is a noise (random error) term $E(\epsilon) = 0$ and $\epsilon \perp\!\!\!\perp X$

a. In general, what are the pros and cons of taking a parametric versus a non-parametric approach?

Parametric approach

In Parametric approach, we first make an assumption about the functional form of $f(X)$ And then instead of solving for the entire $f(X)$, we just solve for the parameters (or coefficients) for our assumed $f(X)$ in a way that best fits the data at hand. (That is, $Y \approx \hat{f}(X)$.) Hence the name Parametric.

The problem is thus reduced to finding the coefficients of the pre-fixed functional form. Thus, instead of finding say some p-dimensional f, we find $p+1$ coefficients for f. This reduces the complexity of the problem, much significantly when p is large.

This approach has a major disadvantage when the real $f(X)$ is not close to our assumption of the functional form, we end up with a model that does not represent our data effectively.

Parametric approaches are not flexible enough but more interpretable.

Non-parametric approach

In non-parametric approach, we do not make any assumptions about the functional form of f. (Here, we try to fit a model in a way that it is "not too wiggly"). Since we are solving for $f(X)$ and not just the parameters and hence the problem is comparatively more complex. And because we don't make an assumption, we need a lot more observation to find a $f(X)$ that fits our data.

On the other side, we are not restricted by the assumption and we don't have to worry too much about whether or not our assumption of the functional form holds for the data at hand.

Non-parametric approaches are comparatively much more flexible and less interpretable. Because it is more flexible, this kind of approach tends to find patterns in data that doesn't exist (That is, it overfits the data)

b. What properties of the data or scientific question would lead you to take a parametric approach?

There are multiple scenarios when we prefer parametric approach:

- When we are trying to understand the association between the different variables, we are trying to estimate $f(X)$ for inference purposes. In such a situation, we would prefer a more interpretable approach rather than a more flexible but less interpretable approach. Parametric approach are generally more interpretable and much less flexible.
- In a situation, when our goal is both prediction and inference, we still generally prefer a more interpretable approach over more flexible ones. Because, Non-parametric approaches are still not very interpretable, but we can work with somewhat flexible parametric approaches.
- When we do not have a lot of observations, we would prefer Parametric approaches because it works well with a small number of observations.

c. What properties of the data or scientific question would lead you to take a non-parametric approach?

- When we are only interested in accurate prediction and not interested in the association between the variables, we would use a non-parametric approach.
- When the problem on hand is much more complex than that could be solved using simple parametric approaches and we have enough observations, we tend to prefer Non-parametric approaches.

2. In each setting, would you generally expect a flexible or an inflexible statistical machine learning method to perform better? Justify your answer.

a. Sample size n is very small, and number of predictors p is very large.

For flexible approaches to work accurately, they tend to require more observations than restrictive or inflexible approaches. Considering p to be very large, for flexible approach to work, we need n to be large enough, which is not the case. Hence, our best bet would be a restrictive or an inflexible approach.

b. Sample size n is very large, and number of predictors p is very small.

For a small number of predictors, we have a large sample size. This is a situation where we have enough amount of data for a flexible approach. So, deciding whether to pick a flexible approach or an inflexible approach then depends on what problem are we trying to solve. If our only goal is accurate prediction and we don't care about the association between the varaibles, we can use a flexible approach with a risk of overfitting. However, if we want to understand the association between the variables irrespective of the fact that we are interested in prediction or not, we use an inflexible approach because they are more interpretable.

c. Relationship between predictors and response is highly non-linear.

Flexible approaches are not restrictive in a sense that they can generate a more wider range of possible shapes to estimate f. However, this is not the case with inflexible approaches. Since our data has non-linear relationship between the predictors, flexible approaches might be able to solve this problem better than a restrictive approach. Using a flexible approach always comes with the risk of overfitting.

d. The variance of the error terms, i.e. $\sigma^2 = var(\epsilon)$, is extremely high.

The choice of a flexible or inflexible approach does not depend on the irreducible error and if irreducible error is high or low, because as the name suggests that whether we choose flexible or inflexible error, we could not get rid of extremely high variance of error terms i.e irreducible error.

3. For each scenario, determine whether it is a regression or a classification problem, determine whether the goal is inference or prediction, and state the values of n (sample size) and p (number of predictors).

a. I want to predict each student's final exam score based on his or her homework scores. There are 50 students enrolled in the course, and each student has completed 8 homeworks.

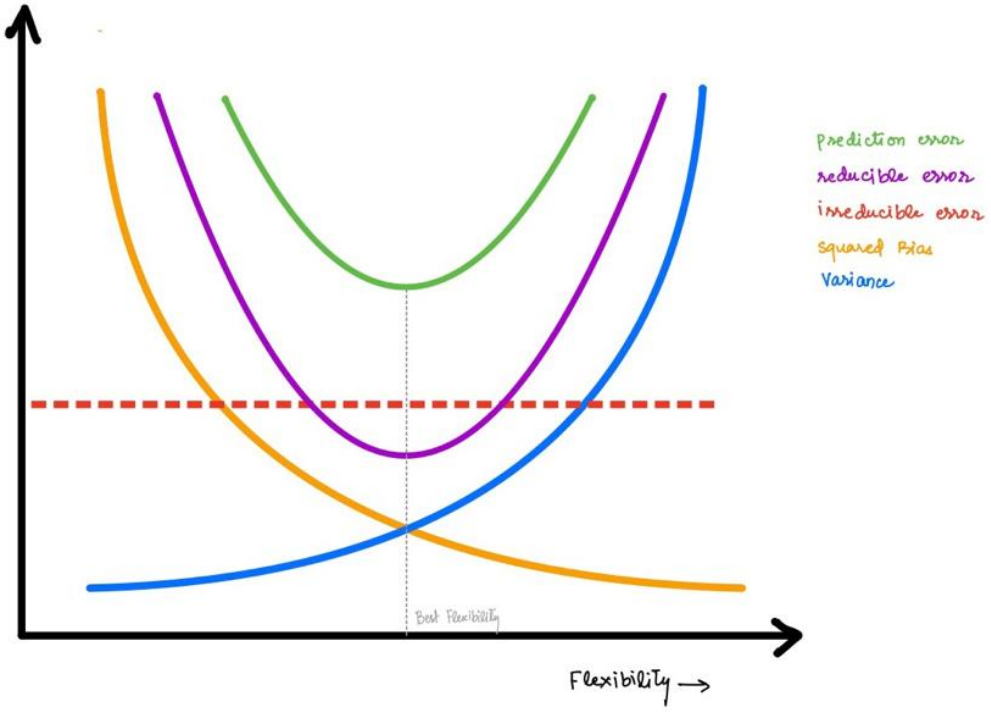
- Problem: Regression (output final exam score is quantitative)
- Goal: Prediction
- n: 50 (students)
- p: 8 (homeworks)

b. I want to understand the factors that contribute to whether or not a student passes this course. The factors that I consider are (i) whether or not the student has previous programming experience; (ii) whether or not the student has previously studied linear algebra; (iii) whether or not the student has taken a previous stats/probability course; (iv) whether or not the student attends office hours; (v) the student's overall GPA; (vi) the student's year (e.g. freshman, sophomore, junior, senior, or grad student). I have data for all 50 students enrolled in the course.

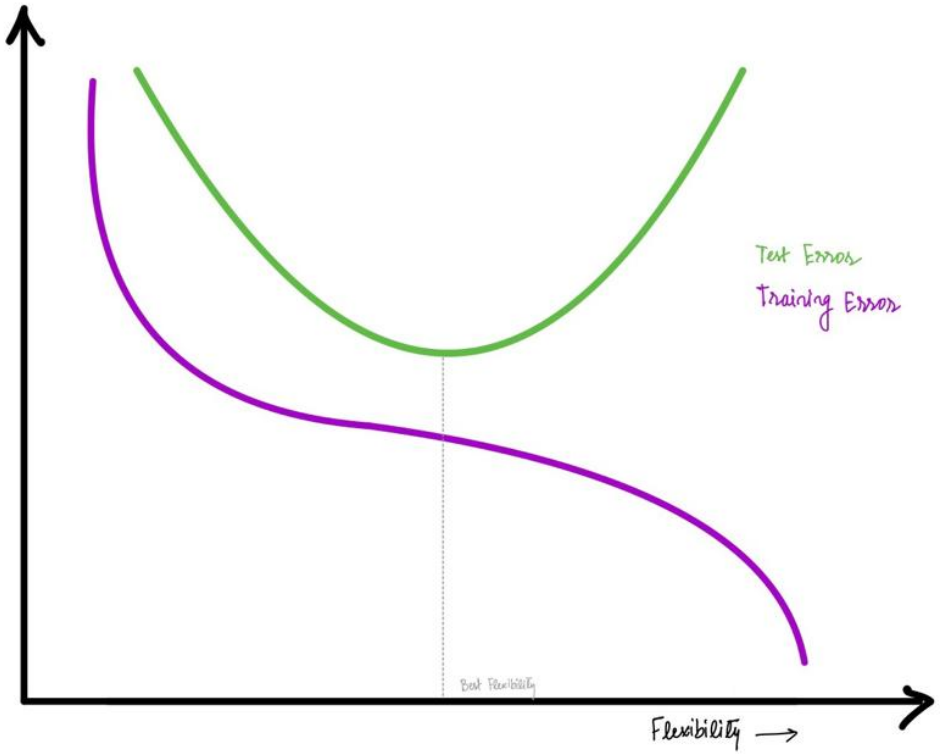
- Problem: Classification (pass/fail)
- Goal: Inference (Understand the factors that contribute towards pass/fail)
- n: 50 (students)
- p: 6 (previous programming experience, Linear Algebra, Statistics/Probability, Office hours, GPA, student's year)

4. This problem has to do with the bias-variance trade-off and related ideas, in the context of regression. For (a) and (b), it's okay to submit hand-sketched plots: you are not supposed to actually compute the quantities referred to below on data; instead, this is a thought exercise.

a. Make a plot, like the one we saw in class, with "flexibility" on the x- axis. Sketch the following curves: squared bias, variance, irreducible error, reducible error, expected prediction error. Be sure to label each curve. Indicate which level of flexibility is "best"

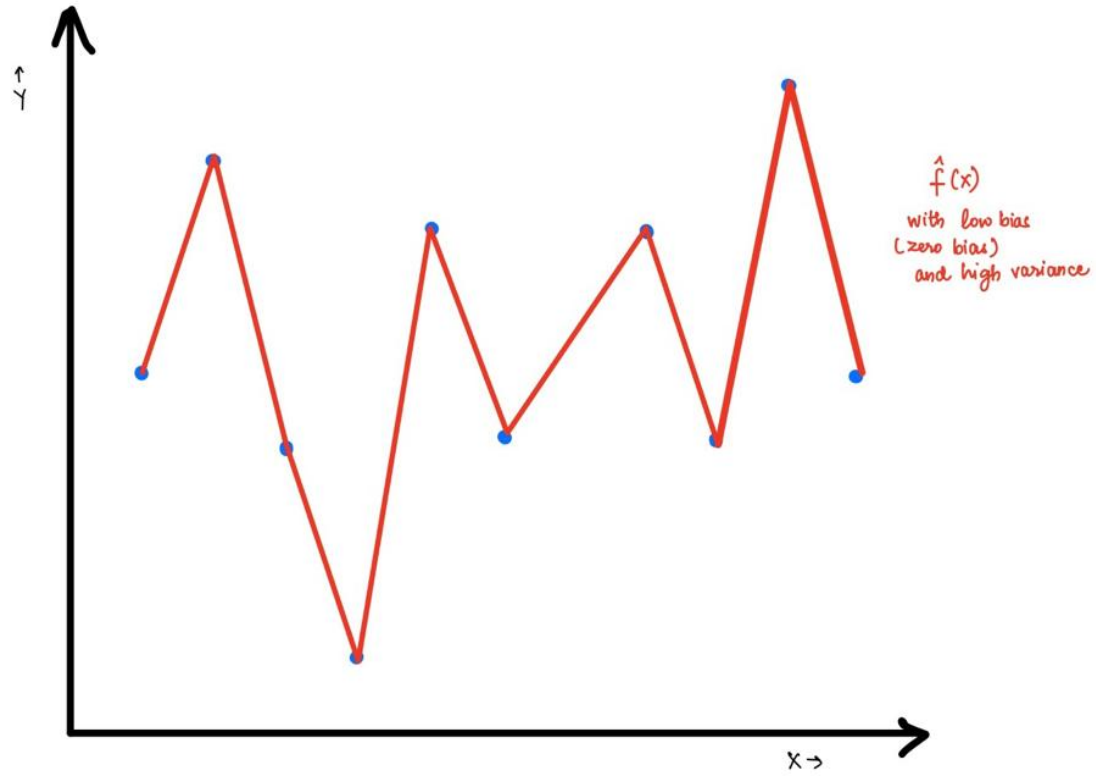


b. Make a plot with "flexibility" on the x-axis. Sketch curves corresponding to the training error and the test error. Be sure to label each curve. Indicate which level of flexibility is "best".



c. Describe an $\hat{f}$ that has extremely low bias, and extremely high variance. Explain your answer.

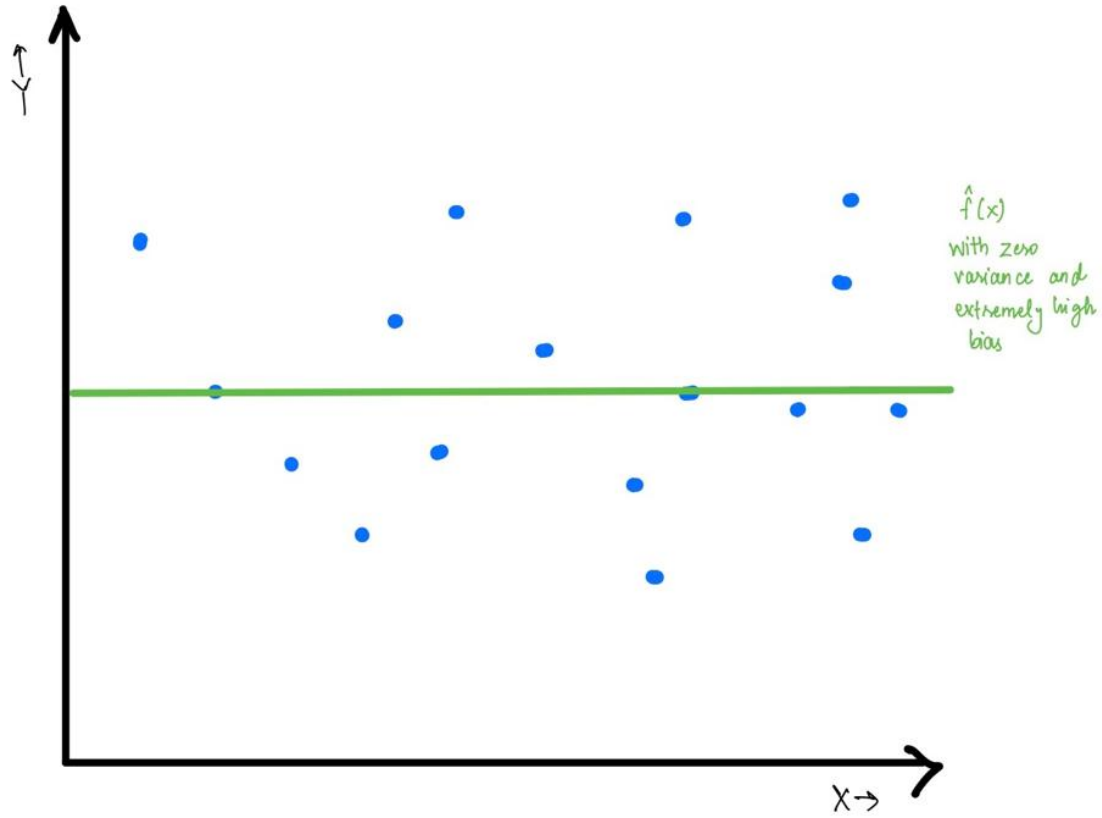
Let's define a model that tries to fit the data in such a way that a curve passes through every data point. It is the "connecting the dots" model. It tries to fit a function $\hat{f}(X)$ in such a way that the function passes through all the points in the training data.



In this case, the bias is extremely low, in fact zero. But the variance is extremely high. Why is this the case? Let us consider a situation wherein someone else is doing the same experiment, that is trying to answer the same question. And that person gets a slightly different dataset just due to random noise, then this model that tries to fit a curve that passes through every point in the training dataset results in a totally different curve. So, we say that this algorithm has a very high variance. There is a lot of variability in the predictions that this algorithm will make. However, this algorithm doesn't have any strong preconceptions that the data could be fit by any particular type of f. Hence, extremely low bias. Or for this case zero bias.

d. Describe an $\hat{f}$ that has extremely high bias, and zero variance. Explain your answer.

Let us define a model that gives out constant prediction, irrespective of the dataset. That is it tries to fit a constant line $Y = c$ to our dataset.



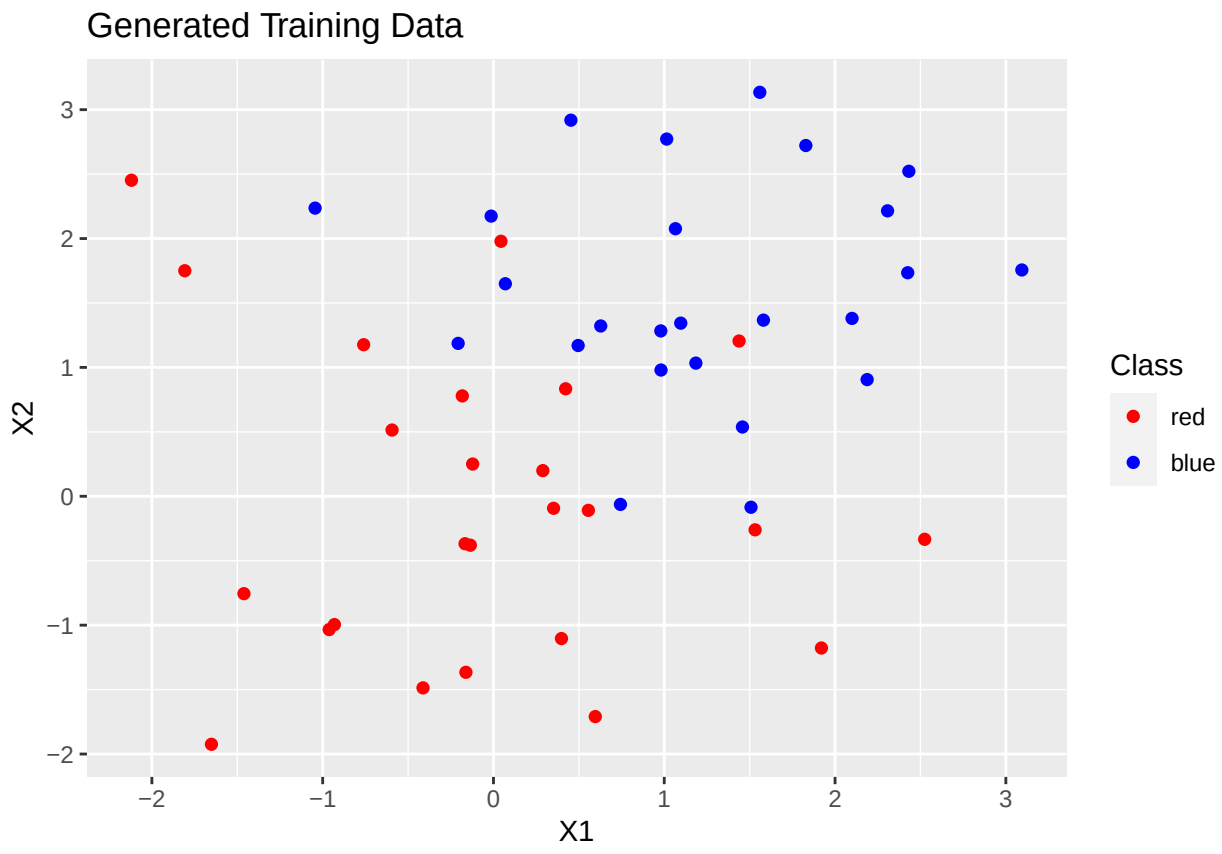
In this case, the bias is extremely high. But the variance is zero. Why? This is because irrespective of the fact that someone else doing this experiment might have a slightly different dataset, there would be no variance in the predictions because your model is giving out constant value. And doesn't really depend on the dataset. However, this learning algorithm has very strong preconceptions that the dataset could be fit using this constant line. This means it has extremely high bias. In cases like these, data is underfitted because such preconceptions are hardly ever true.

5. We now consider a classification problem. Suppose we have 2 classes (labels), 25 observations per class, and $p = 2$ features. We will call one class the "red" class and the other class the "blue" class. The observations in the red class are drawn i.i.d. from a $\mathcal{N}(\mu_r, I)$ distribution, and the observations in the blue class are drawn i.i.d. from a $\mathcal{N}(\mu_b, I)$ distribution, where $\mu_r = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ is the mean in the red class, and where $\mu_b = \begin{pmatrix} 1.5 \\ 1.5 \end{pmatrix}$ is the mean in the blue class.

a. Generate a training set, consisting of 25 observations from the red class and 25 observations from the blue class. (You will want to use the R function rnorm.) Plot the training set. Make sure that the axes are properly labeled, and that the observations are colored according to their class label.

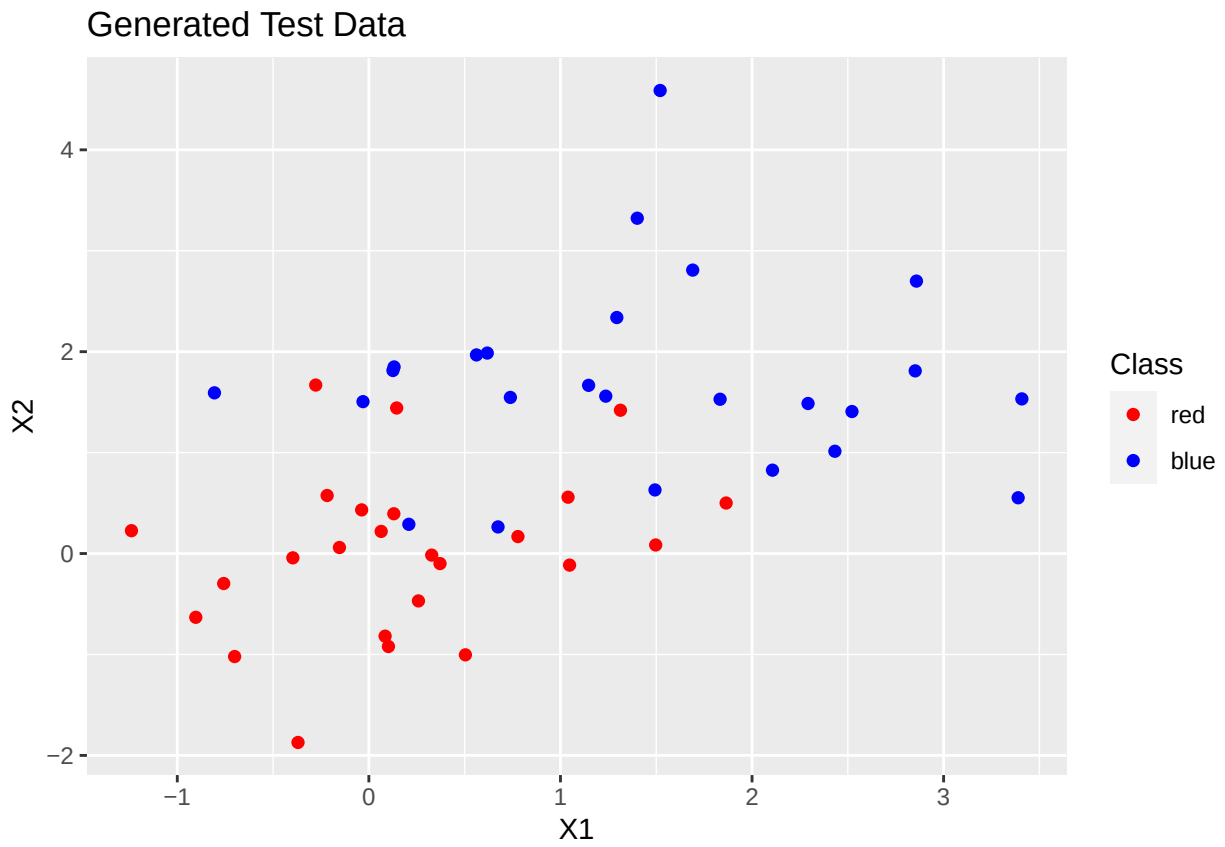
```
# loading the MASS library to use the mvrnorm function for generating the
# bivariate normal distribution

library(MASS)
```

```r
#loading the ggplot2 library for plots

library(ggplot2)

set.seed(209)

mu_1 <- c(0,0)
mu_2 <- c(1.5, 1.5)
sigma <- matrix(c(1,0,0,1), 2,2)

trainRed5 <- data.frame(mvrnorm(25, mu_1, sigma), Group ="red")
trainBlue5 <- data.frame(mvrnorm(25, mu_2, sigma), Group = "blue")

trainData5 <- rbind.data.frame(trainBlue5, trainRed5)

colors <- c("red" = "red", "blue"="blue")

ggplot(trainData5, aes(x=X1, y=X2))+
  geom_point(size=2, pch=16, aes(color=Group))+
  labs(x = "X1",
       y="X2",
       title = "Generated Training Data",
       color="Class")+
  scale_color_manual(values=colors)
```

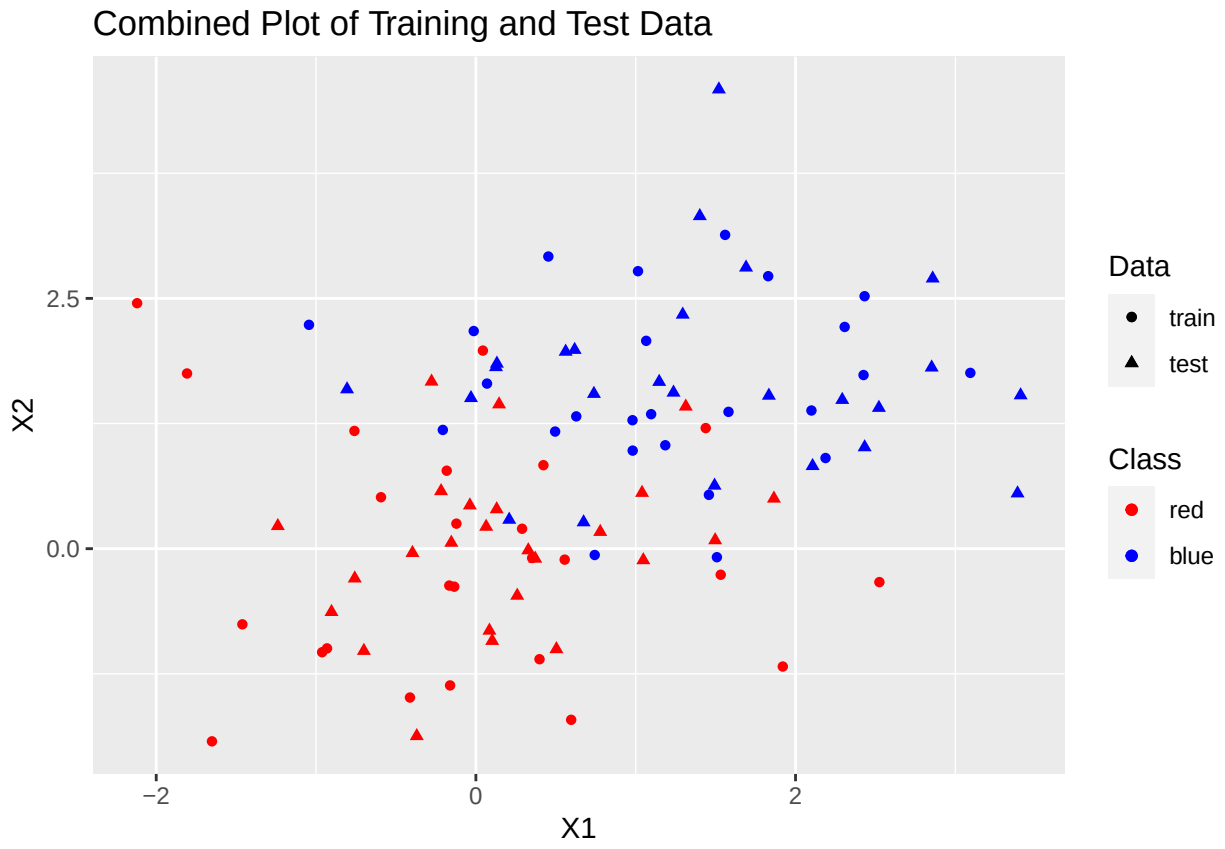


b. Now generate a test set consisting of 25 observations from the red class and 25 observations from the blue class. On a single plot, display both the training and test set, using one symbol to indicate training observations(e.g. circles) and another symbol to indicate the test observations (e.g. squares). Make sure that the axes are properly labeled, that the symbols for training and test observations are explained in a legend, and that the observations are colored according to their class labels.

```
set.seed(200)

testRed5 <- data.frame(mvrnorm(25, mu_1, sigma), Group ="red")
testBlue5 <- data.frame(mvrnorm(25, mu_2, sigma), Group= "blue")

testData5 <- rbind.data.frame(testRed5, testBlue5)

colors <- c("red" = "red", "blue"="blue")

ggplot(testData5, aes(x=X1, y=X2))+
  geom_point(size=2, pch=16, aes(color=Group))+
  labs(x = "X1",
       y="X2",
       title = "Generated Test Data",
       color="Class")+
  scale_color_manual(values=colors)
```



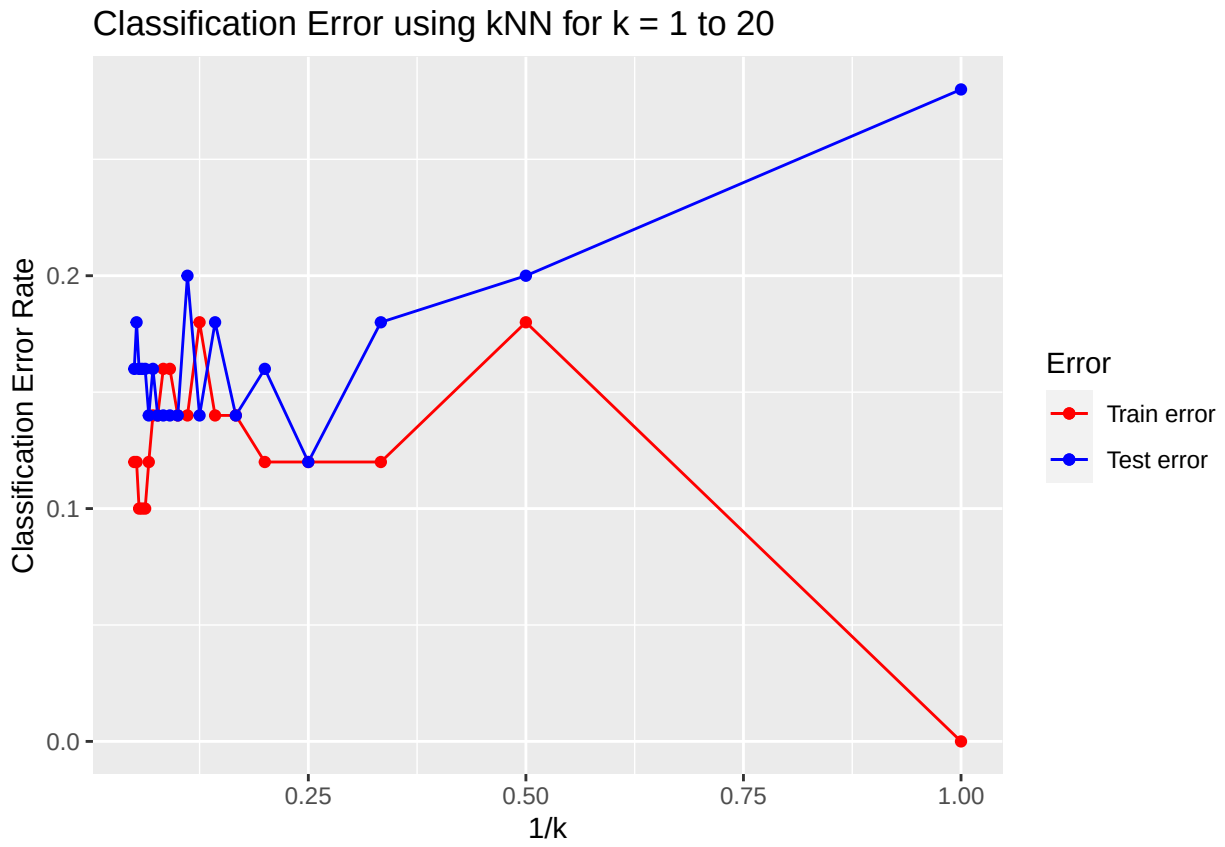
```
shapes <- c("train" = 16, "test" = 17)
ggplot() +
  geom_point(data = trainData5, aes(X1,X2, color=Group, shape="train"))+
  geom_point(data = testData5, aes(X1,X2, color=Group, shape="test"))+
  ggtitle("Combined Plot of Training and Test Data")+
  scale_color_manual(values=colors, name="Class")+
  scale_shape_manual(values=shapes, name="Data")
```



c. Using the knn function in the library class, fit a k-nearest neighbors model on the training set, for a range of values of k from 1 to 20. Make a plot that displays the value of 1/k on the x-axis, and classification error (both training error and test error) on the y-axis. Make sure all axes and curves are properly labeled. Explain your results.

```
set.seed(150)

library(class)

trainScale5 <- scale(trainData5[, 1:2])
testScale5 <- scale(testData5[, 1:2])
testError5<-c()
trainError5<-c()
for (i in 1:20){
  knnTrain5<-knn(trainScale5, trainScale5, cl=trainData5$Group, i)
```

```
  knnTest5<-knn(trainScale5, testScale5, cl=trainData5$Group, i)
  misClassErrortrain5 <- mean(knnTrain5 != trainData5$Group)
  trainError5[i] <- misClassErrortrain5
  misClassError5 <- mean(knnTest5 != testData5$Group)
  testError5[i] <- misClassError5
}

error5 = data.frame(trainError5, testError5, k=1:20)

cols <- c("Train error" = "red", "Test error" = "blue")

ggplot() +
  geom_line(data = error5, aes(1/k, trainError5, color="Train error"))+
  geom_line(data = error5, aes(1/k, testError5, color="Test error"))+
  geom_point(data = error5, aes(1/k, trainError5, color="Train error"))+
  geom_point(data = error5, aes(1/k, testError5, color="Test error"))+
  labs(x = "1/k", y = "Classification Error Rate")+
  scale_color_manual(values=cols, name="Error")+
  ggtitle("Classification Error using kNN for k = 1 to 20 ")
```



How do we interpret this graph? Let us understand it in the terminologies that we have defined earlier. k is the number of nearest neighbors used for the kNN. We know that as the k increases, the flexibility decreases, as the degree of freedom decreases. For our question, we can safely say that the kNN for $k = 20$ will be the least flexible while the kNN for $k = 1$ will be the most flexible. So if we go from $k = 20$ to $k = 1$, or say if $1/k$ goes from 1/20 to 1 to flexibility increases. Now as we saw in Question 4b, as flexibility increases, training error decreases, and test error first decreases and then increases.This is what we observe theoretically. This

is because of the fact that after a certain level of flexibility, the model starts overfitting the data. This is what we see in the graph above.

d. For the value of k that resulted in the smallest test error in part (c) above, make a plot displaying the test observations as well as their true

3 and predicted class labels. Make sure that all axes and points are clearly labeled.

```
which.min(error5$testError5)
```

```
## [1] 4
```

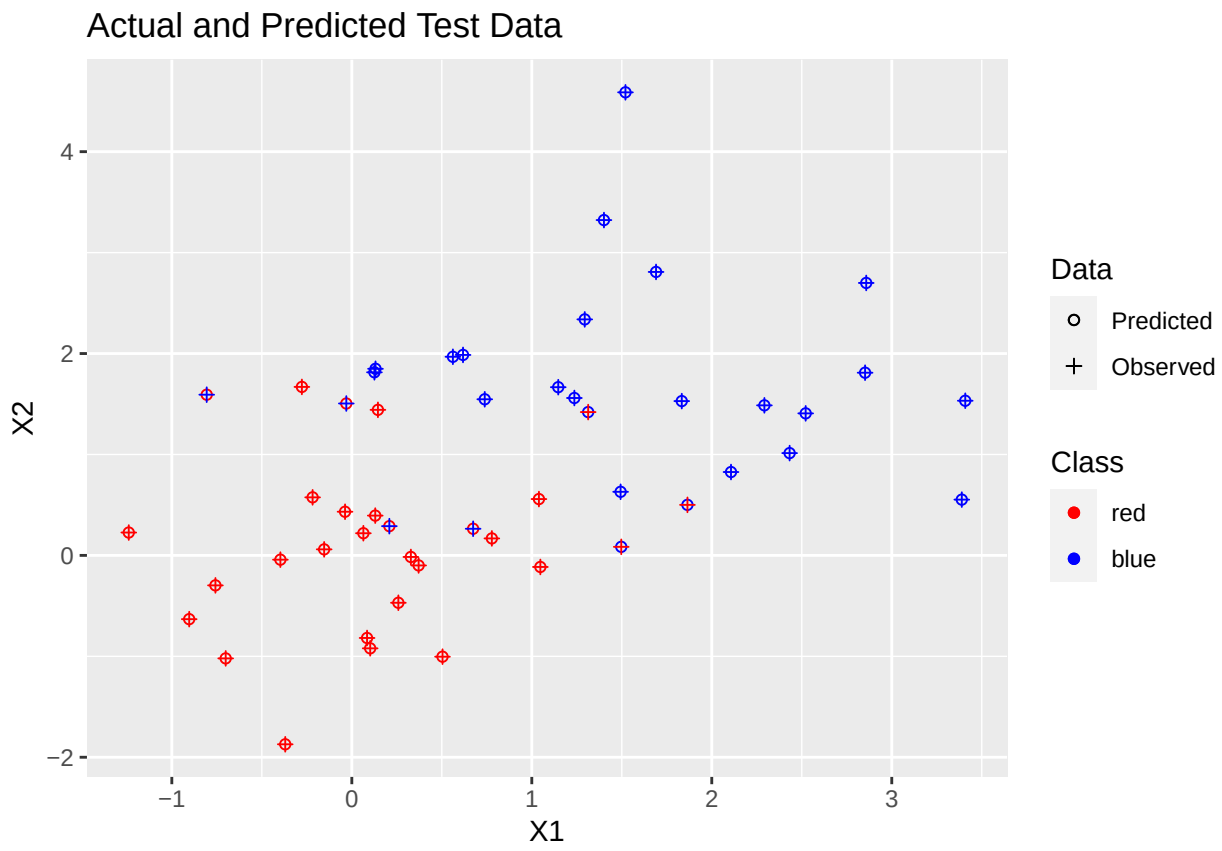
```
min(error5$testError5)
```

```
## [1] 0.12
```

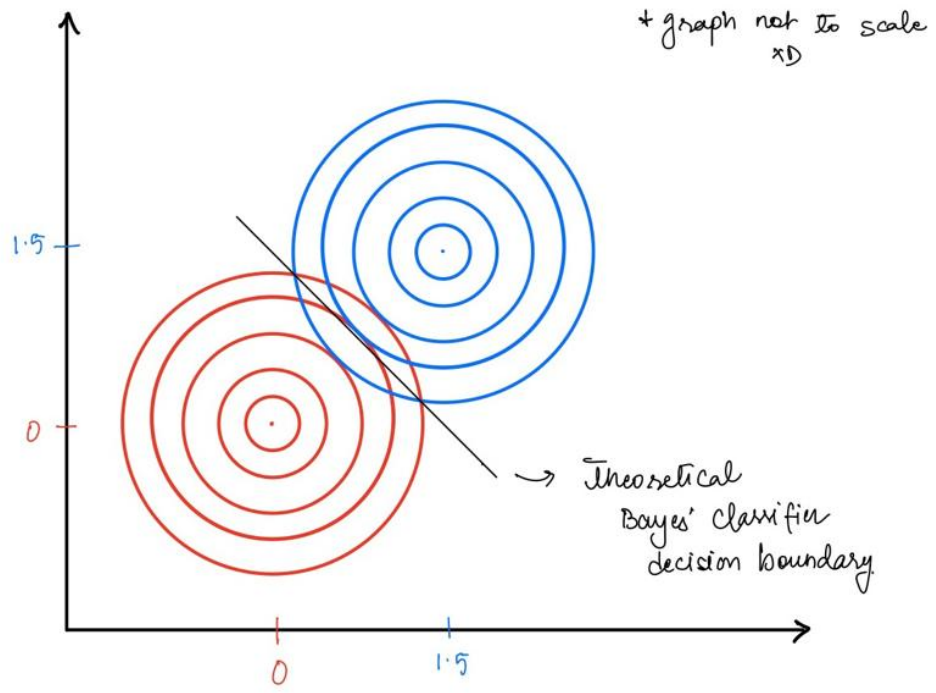
```
set.seed(150)
knnTest_4<-knn(trainScale5, testScale5, cl=trainData5$Group, 4)
knnPred_4<-data.frame(knnTest_4, testData5$X1, testData5$X2, testData5$Group)
confusionMatrix_4 <- table(testData5$Group, knnTest_4)
confusionMatrix_4
```

```
##       knnTest_4
##        blue red
##   blue   21   4
##   red     3  22
```

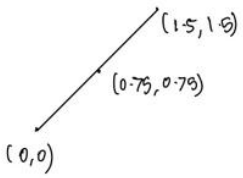
```
col <- c("red" = "red", "blue" = "blue")
sha <- c("Predicted" = 1, "Observed" = 3)
ggplot() +
  geom_point(data = knnPred_4, aes(testData5.X1,testData5.X2,color=knnTest_4, shape="Predicted"))+
  geom_point(data = knnPred_4, aes(testData5.X1,testData5.X2, color=testData5.Group, shape="Observed"))+
  labs(x = "X1", y = "X2")+
  ggtitle("Actual and Predicted Test Data")+
  scale_color_manual(values=col, name="Class")+
  scale_shape_manual(values=sha, name="Data")
```


Actual and Predicted Test Data
X2
X1
4
2
0
−2
−1
0
1
2
3
Data
Predicted
Observed
Class
red
blue

e. In this example, what is the Bayes error rate? Justify your answer, and explain how it relates to your findings in (c) and (d).



- This theoretical boundary would be nothing but a line that is at equal distance from (0,0) and (1.5, 1.5) and is perpendicular to the line $y = x$ (equation of line that passes from (0,0) and (1.5, 1.5))



So it's $y = -x + c$

$0.75 = -0.75 + c$

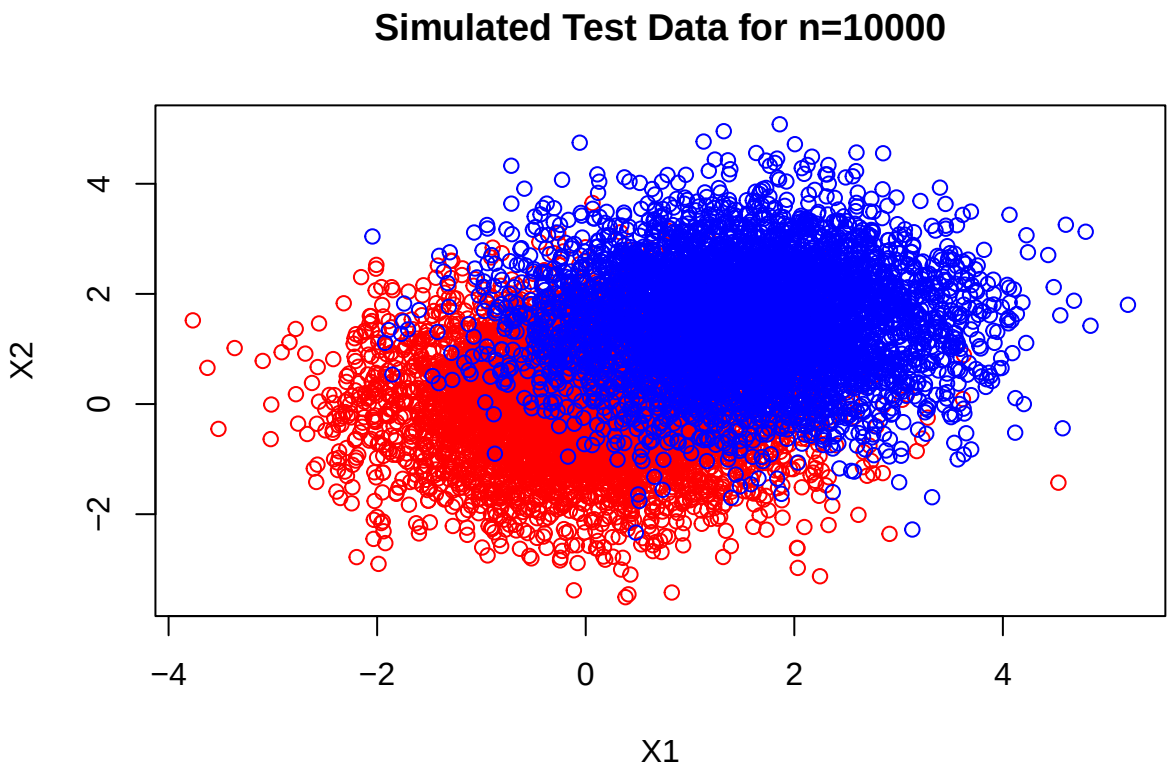
$\therefore c = 1.5$

→ Hence, equation of the line is $y = -x + 1.5$

Hence if x_0 is closer to (0,0)
↳ it would be classified as "red"

& if x_0 is closer to (1.5, 1.5)
↳ it would be classified as "blue"

```
set.seed(4203)

testRedBC <- data.frame(mvrnorm(5000, mu_1, sigma), y ="red")
testBlueBC <- data.frame(mvrnorm(5000, mu_2, sigma), y= "blue")

testDataBC<- rbind.data.frame(testRedBC, testBlueBC)

plot(testDataBC$X1, testDataBC$X2, col=testDataBC$y, xlab="X1", ylab="X2", main="Simulated Test Data fo
```




```
library(MASS)
X1 <- as.numeric(testDataBC$X1)
X2 <- as.numeric(testDataBC$X2)

predColor <- c()

for (i in 1:10000)
{

    if ( ( (X1[i])^2 + (X2[i])^2 ) < (X1[i] - 1.5)^2 + (X2[i]-1.5)^2)
    {
        predColor[i] <- "red"
    }

    else
    {
```

```
        predColor[i] <- "blue"
    }

}

testDataBC <- cbind.data.frame(testDataBC, predColor)

misClassErrorBC<- mean(testDataBC$y != testDataBC$predColor)
misClassErrorBC
```

```
## [1] 0.1387
```

This simulated plot looks similar to what we theoretically explored. And the Bayes error comes to be 0.1387. Bayes error is the best we can do for Classification problems. What it means is that Bayes Classifier is the optimal solution. There is no better case than this. The error of this optimal Bayes classifier is the unavoidable error for this learning task.

6. We will once again perform k-nearest-neighbors in a setting with p = 2 features. But this time, we'll generate the data differently: let $X1 \sim Unif[0,1]$ **and** $X2 \sim Unif[0,1]$**, i.e. the observations for each feature are i.i.d. from a uniform distribution. An observation belongs to class "red" if** $(X1-0.5)^2+(X2-0.5)^2 > 0.15$ **and** $X1 > 0.5$**; to class "green" if** $(X1-0.5)^2+(X2-0.5)^2 > 0.15$ **and** $X1 \leq 0.5$**; and to class "blue" otherwise.**

a. Generate a training set of $n = 200$ **observations. (You will want to use the R function runif.) Plot the training set. Make sure that the axes are**

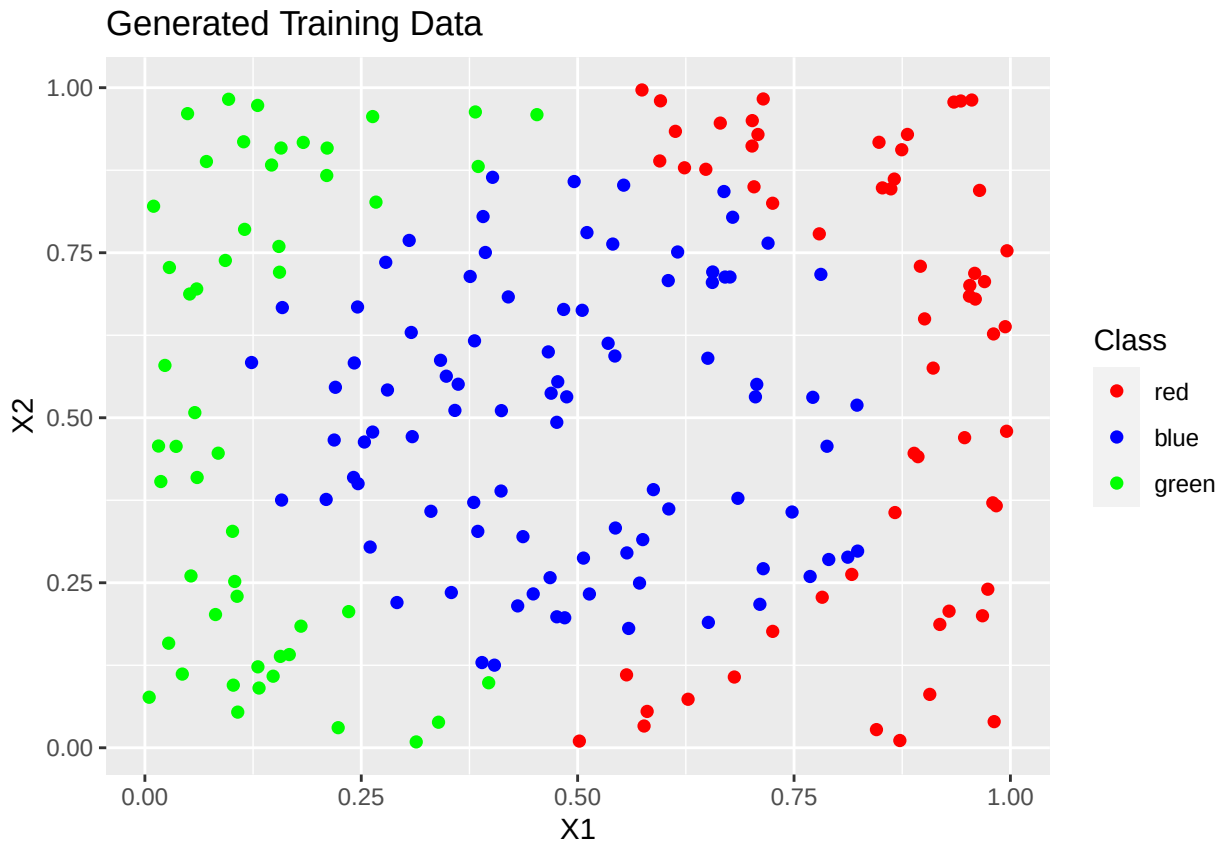
properly labeled, and that the observations are colored according to their class label.

```
set.seed(43)

library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following object is masked from 'package:MASS':
##
##     select

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

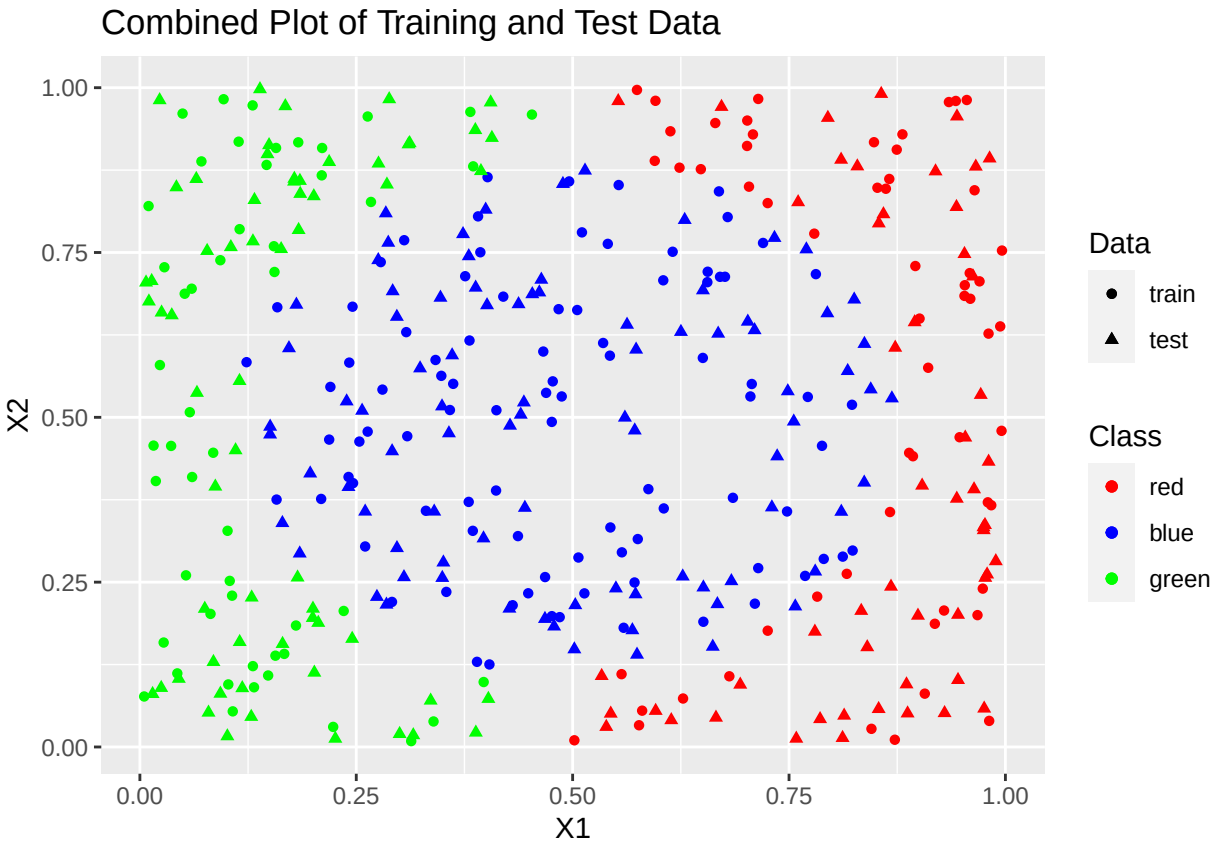
```
X1<- runif(200,0,1)
X2<- runif(200,0,1)
```

```
trainData6 <- data.frame(X1, X2)
trainData6 <-trainData6 %>% mutate(Group =
                      case_when((X1-0.5)^2 +(X2-0.5)^2 >0.15 & X1>0.5 ~ "red",
                                (X1-0.5)^2 +(X2-0.5)^2 >0.15 & X1<0.5 ~ "green",
                                (X1-0.5)^2 +(X2-0.5)^2 <=0.15 ~ "blue"))

colors_6 <- c("red" = "red", "blue"="blue", "green" = "green")

ggplot(trainData6, aes(x=X1, y=X2))+
  geom_point(size=2, pch=16, aes(color=Group))+
  labs(x = "X1",
       y="X2",
       title = "Generated Training Data",
       color="Class")+
  scale_color_manual(values=colors_6)
```



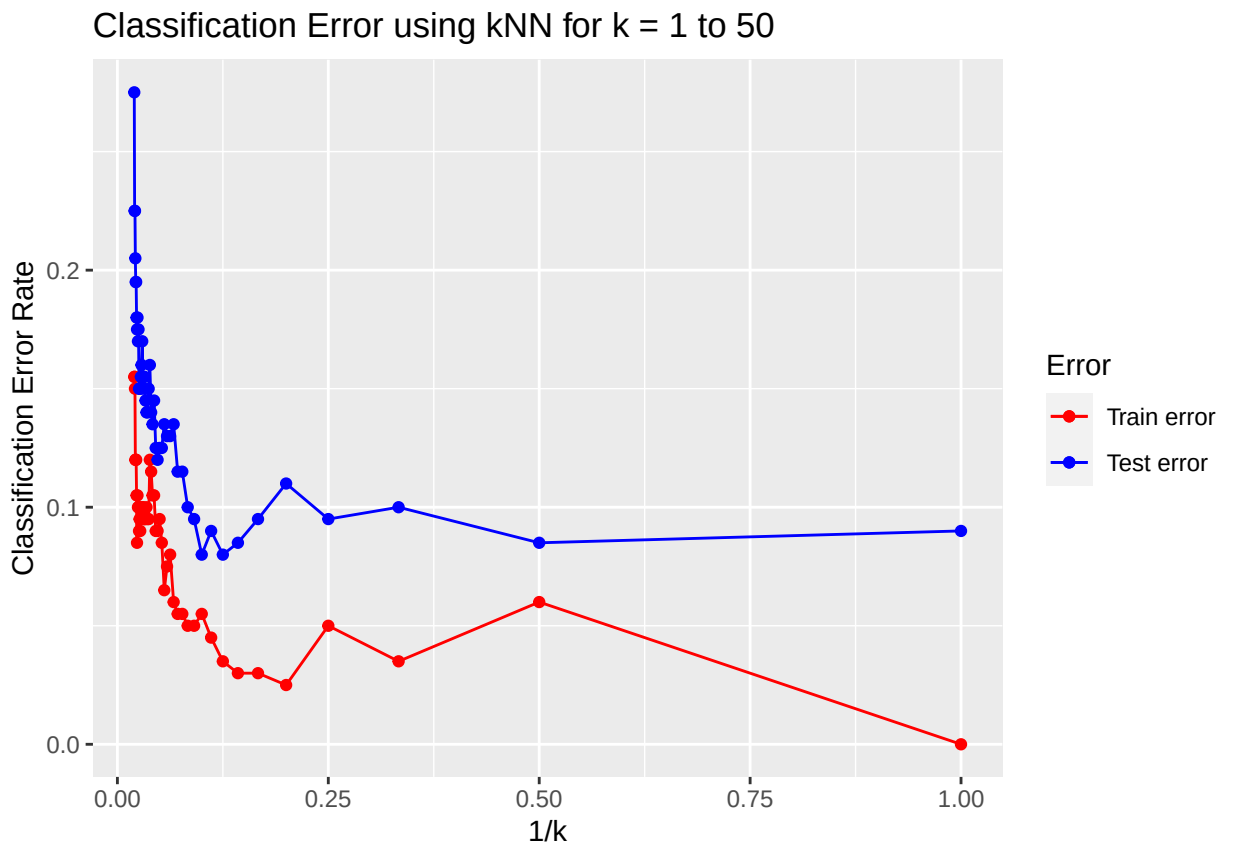
b. Now generate a test set consisting of another 200 observations. On a single plot, display both the training and test set, using one symbol to indicate training observations (e.g. circles) and another symbol to indicate the test observations (e.g. squares). Make sure that the axes are properly labeled, that the symbols for training and test observations are explained in a legend, and that the observations are colored according to their class label.

```
set.seed(2)
X1_t <- runif(200,0,1)
X2_t <- runif(200,0,1)
testData6 <- data.frame(X1_t, X2_t)
library(dplyr)
testData6<-testData6 %>% mutate(Group =
                      case_when((X1_t-0.5)^2 +(X2_t-0.5)^2 >0.15 & X1_t>0.5 ~ "red",
                                (X1_t-0.5)^2 +(X2_t-0.5)^2 >0.15 & X1_t<0.5 ~ "green",
                                (X1_t-0.5)^2 +(X2_t-0.5)^2 <=0.15 ~ "blue"))

shapes <- c("train" = 16, "test" = 17)
ggplot() +
  geom_point(data = trainData6, aes(X1,X2, color=Group, shape="train"))+
  geom_point(data = testData6, aes(X1_t,X2_t, color=Group, shape="test"))+
  ggtitle("Combined Plot of Training and Test Data")+
  scale_color_manual(values=colors_6, name="Class")+
  scale_shape_manual(values=shapes, name="Data")+
  labs(x="X1",y="X2")
```



c. Using the knn function in the library class, fit a k-nearest neighbors model on the training set, for a range of values of k from 1 to 50. Make a plot that displays the value of $1/k$ on the x-axis, and classification error (both training error and test error) on the y-axis. Make sure all axes and curves are properly labeled. Explain your results.

```r
set.seed(1)
trainScale6<- scale(trainData6[, 1:2])
testScale6 <- scale(testData6[, 1:2])
testError6<-c()
trainError6<-c()
for (i in 1:50){
  knnTrain6<-knn(trainScale6, trainScale6, cl=trainData6$Group, i)
  knnTest6<-knn(trainScale6, testScale6, cl=trainData6$Group, i)
  misClassErrorTrain6 <- mean(knnTrain6 != trainData6$Group)
  trainError6[i] <- misClassErrorTrain6
  misClassError6 <- mean(knnTest6 != testData6$Group)
  testError6[i] <- misClassError6
}

error6 = data.frame(trainError6, testError6, k=1:50)

cols <- c("Train error" = "red", "Test error" = "blue")

ggplot() +
  geom_line(data = error6, aes(1/k, trainError6, color="Train error"))+
  geom_line(data = error6, aes(1/k, testError6, color="Test error"))+
  geom_point(data = error6, aes(1/k, trainError6, color="Train error"))+
  geom_point(data = error6, aes(1/k, testError6, color="Test error"))+
  labs(x = "1/k", y = "Classification Error Rate")+
  scale_color_manual(values=cols, name="Error")+
  ggtitle("Classification Error using kNN for k = 1 to 50 ")
```



How do we interpret this graph? Let us understand it in the terminologies that we have defined earlier. k is the number of nearest neighbors used for the kNN. We know that as the k increases, the flexibility decreases, as the degree of freedom decreases. For our question, we can safely say that the kNN for $k = 50$ will be the least flexible while the kNN for $k = 1$ will be the most flexible. So if we go from $k = 50$ to $k = 1$, or say if $1/k$ goes from $1/50$ to 1 to flexibility increases. Now as we saw in Question 4b, as flexibility increases, training error decreases, and test error first decreases and then increases.This is what we observe theoretically. This is because of the fact that after a certain level of flexibility, the model starts overfitting the data. This is what we see in the graph above.

d. For the value of k that resulted in the smallest test error in part (c) above, make a plot displaying the test observations as well as their true and predicted class labels. Make sure that all axes and points are clearly labeled.

```
which.min(error6$testError6)
```

```
## [1] 8
```

```
min(error6$testError6)
```

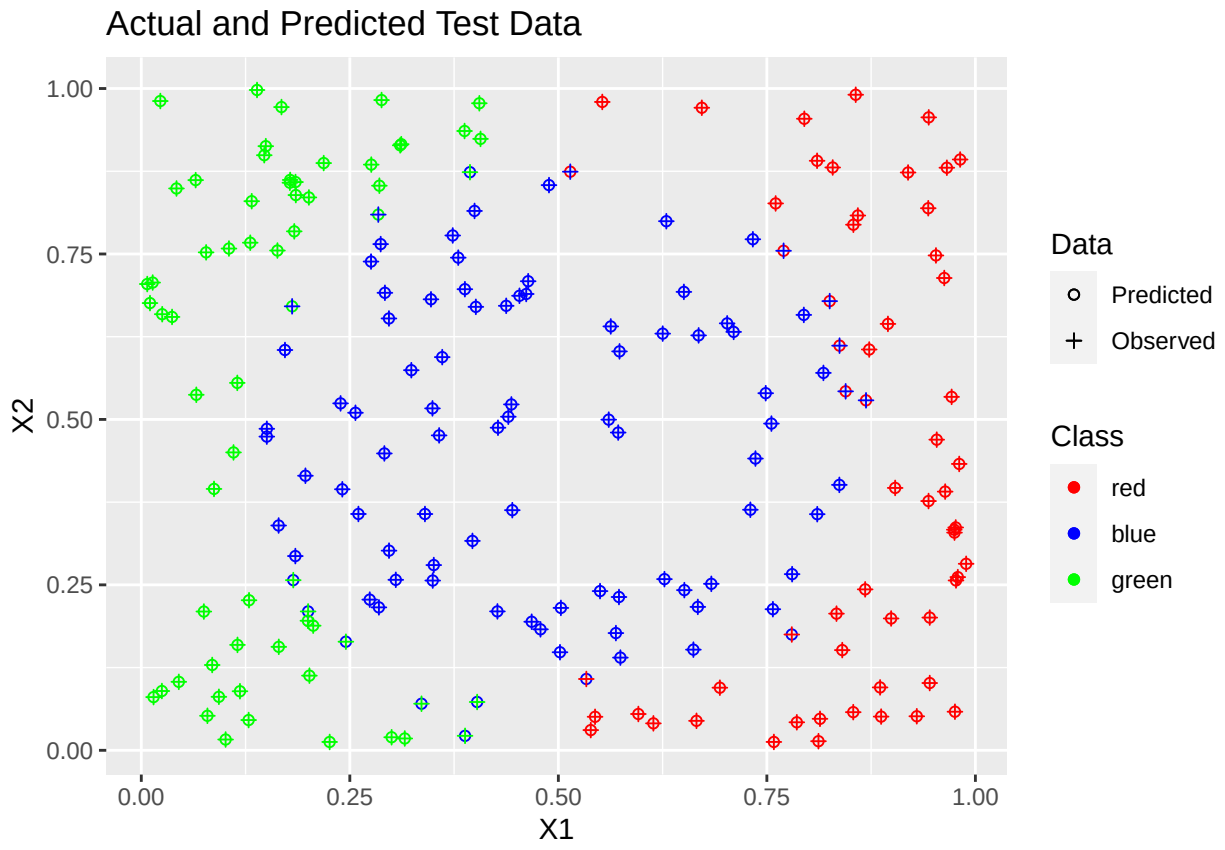
```
## [1] 0.08
```

```
set.seed(150)
knnTest8<-knn(trainScale6, testScale6, cl=trainData6$Group, 8)
```

```
knnPred8<-data.frame(knnTest8, testData6$X1_t, testData6$X2_t, testData6$Group)
confusionMatrix8 <- table(testData6$Group, knnTest8)
confusionMatrix8
```

```
##        knnTest8
##         blue green red
##   blue    77     2   6
##   green    7    55   0
##   red      2     0  51
```

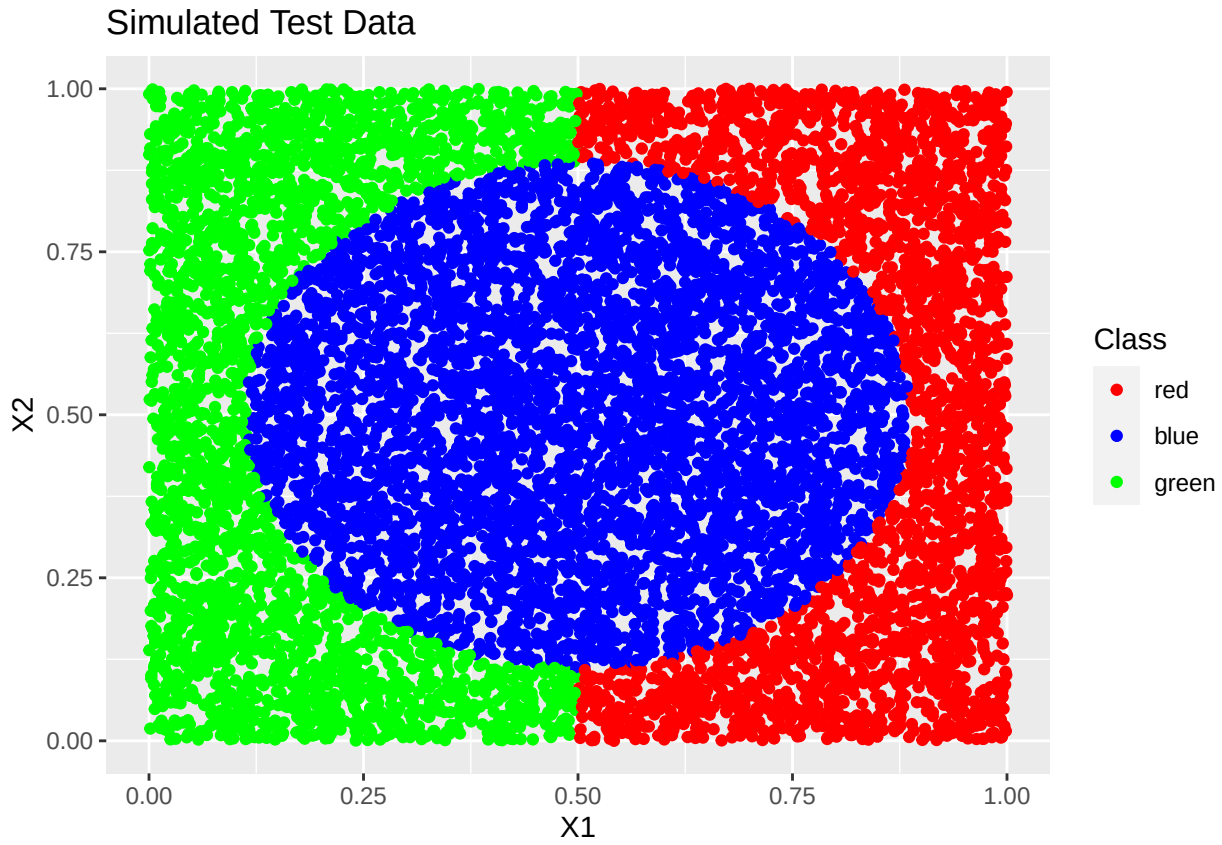
```
ggplot() +
  geom_point(data = knnPred8, aes(testData6.X1_t,testData6.X2_t,color=knnTest8, shape="Predicted"))+
  geom_point(data = knnPred8, aes(testData6.X1_t,testData6.X2_t, color=testData6.Group, shape="Observed
  labs(x = "X1", y = "X2")+
  ggtitle("Actual and Predicted Test Data")+
  scale_color_manual(values=colors_6, name="Class")+
  scale_shape_manual(values=sha, name="Data")
```



e. In this example, what is the Bayes error rate? Justify your answer, and explain how it relates to your findings in (c) and (d).

```
set.seed(5040)
X1s <- runif(10000,0,1)
```

```
X2s <- runif(10000,0,1)
testData6BC <- data.frame(X1s, X2s)
library(dplyr)
testData6BC<-testData6BC %>% mutate(Group =
                      case_when((X1s-0.5)^2 +(X2s-0.5)^2 >0.15 & X1s>0.5 ~ "red",
                                (X1s-0.5)^2 +(X2s-0.5)^2 >0.15 & X1s<0.5 ~ "green",
                                (X1s-0.5)^2 +(X2s-0.5)^2 <=0.15 ~ "blue"))

ggplot() +
  geom_point(data = testData6BC, aes(X1s,X2s, color=Group))+
  ggtitle("Simulated Test Data")+
  scale_color_manual(values=colors_6, name="Class")+
  labs(x="X1",y="X2")
```



Here, I simulated the data using the distribution given in the question for 20000 data points. We can clearly see that there is no overlapping between any class because our data is created in such a way. There is a specific boundary that separates each class. Since there is a uniform distribution and such rigidly defined boundaries, this is the optimal case. The boundaries as we defined while generating the data is the optimal case. That is the Bayes Classifier. In such a case, the best possible case could be achieved and hence, the Bayes error rate would essentially be 0.

In (c) and (d) of this we see this that the kNN does a pretty decent job of minimizing the test error to 0.08 which is much closer to 0. So, we try to achieve this best possible case, but can never get a value exactly as the Bayes error rate. Bayes error rate is a theoretical concept and we can't actually calculate it. We can just estimate it.

7. This exercise involves the Boston housing data set, which is part of the ISLR2

library.

```
library(ISLR2)
```

```
##
## Attaching package: 'ISLR2'
```

```
## The following object is masked from 'package:MASS':
##
##     Boston
```

a. How many rows are in this data set? How many columns? What do the rows and columns represent?

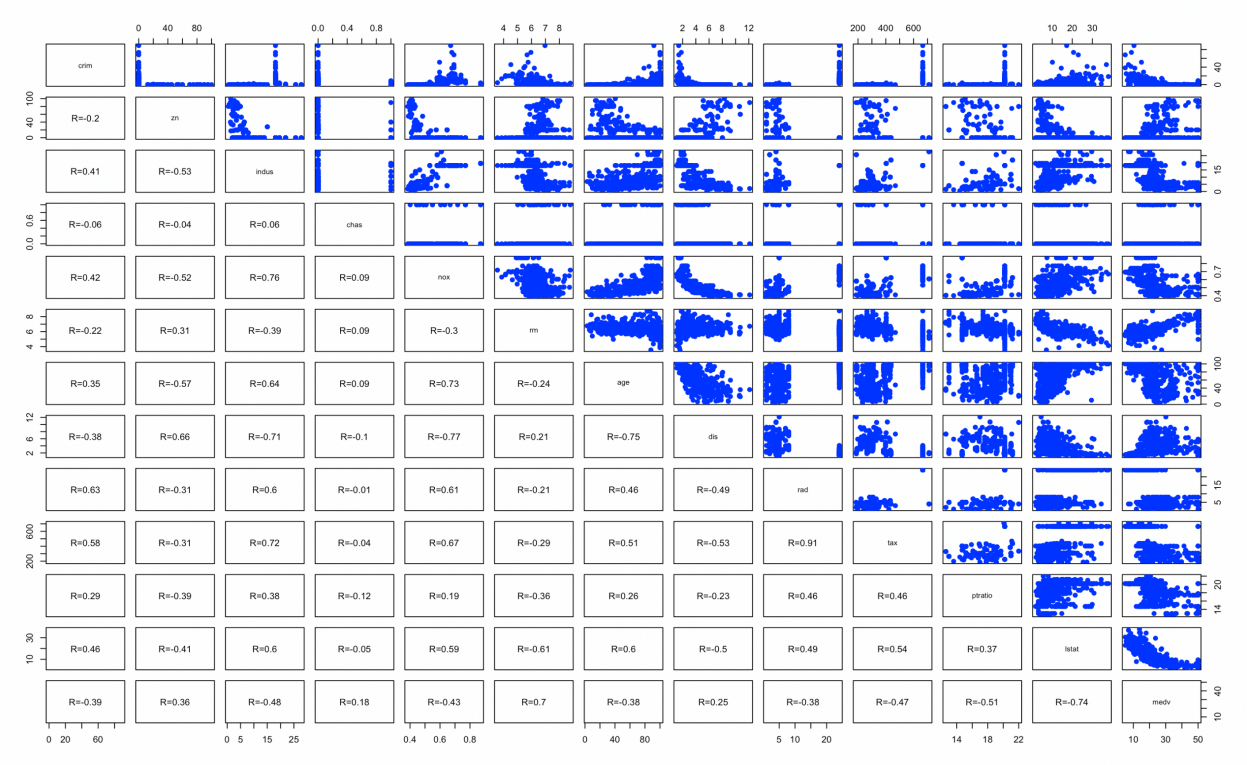
```
dim(Boston)
```

```
## [1] 506  13
```

The Boston housing dataset consists of hosing values of suburbs in Boston. Boston is dataframe with 506 rows and 13 columns. These 506 rows are observations that represent 506 suburbs for Boston and 13 columns are different variables that represent 13 different categories of information for these 506 suburbs.

b. Make some pairwise scatterplots of the predictors (columns) in this dataset. Describe your findings.

```
panel.cor<- function(x,y){
  usr<-par("usr"); on.exit(par(usr))
  par(usr = c(0,1,0,1))
  r<- round(cor(x,y), digits=2)
  txt<- paste0("R=", r)
  text(0.5, 0.5, txt)
}

upper.panel<- function(x,y){
  points(x,y, pch=19, col="blue")
}

pairs(crim~.,data=Boston, lower.panel=panel.cor, upper.panel=upper.panel)
```



I initially plotted just the pairwise scatter plot of all the variables. But it was pretty hard to read and understand what these scatter plots meant and so I removed the redundant lower panel and added correlations of two variables taken at a time. This made some of the relations pretty clear.

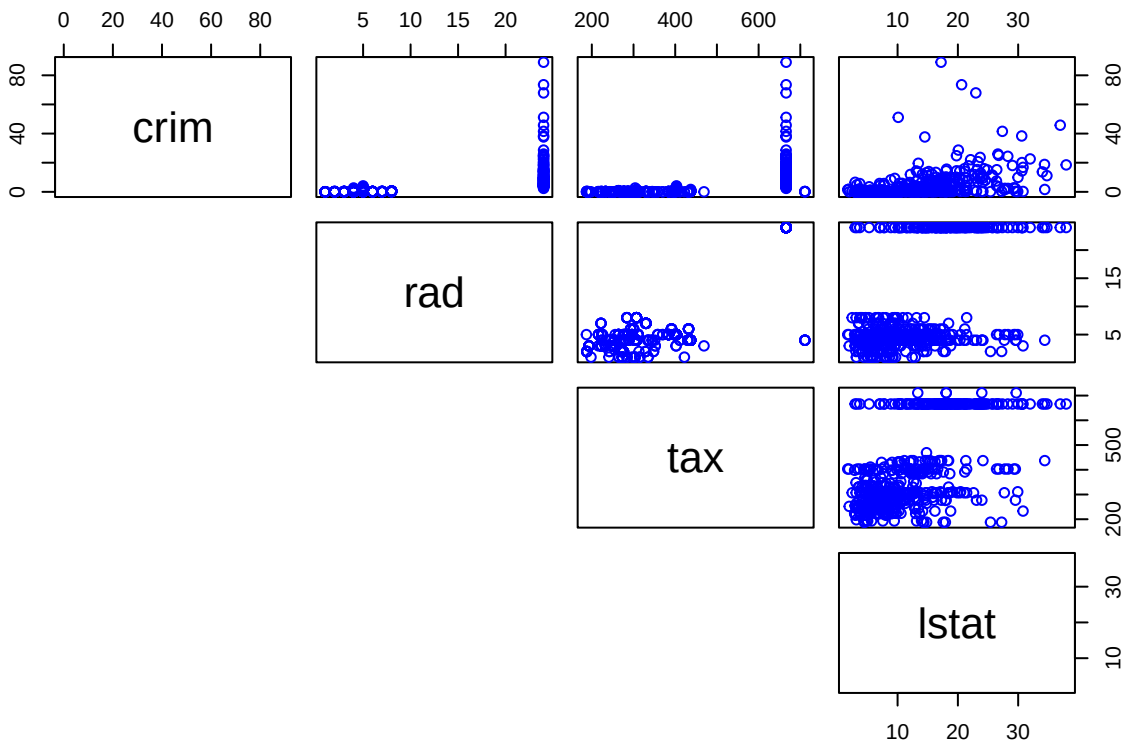
Some of the observations from this pairwise scatter plot:

- We can see that chas is a dummy variable. That is it has only two possible values: either 0 or 1. From the dataset, we know that chas=1 if tract bounds river and 0 otherwise.
- There is extremely strong correlation between tax and rad.
- There is strong negative correlation between lstat and medv.
- There is a strong correlation between age, dis, indus and nox.
- There is a positive correlation between rm and medv.
- We can also see that there are a large number of suburbs with crim value close to zero.
- There is a clear non-linear relationship between lstat and medv which is not accurately described by the correlation value.
- We observe that there are some suburbs take the same tax, rad, zn, radius and ptratio values.

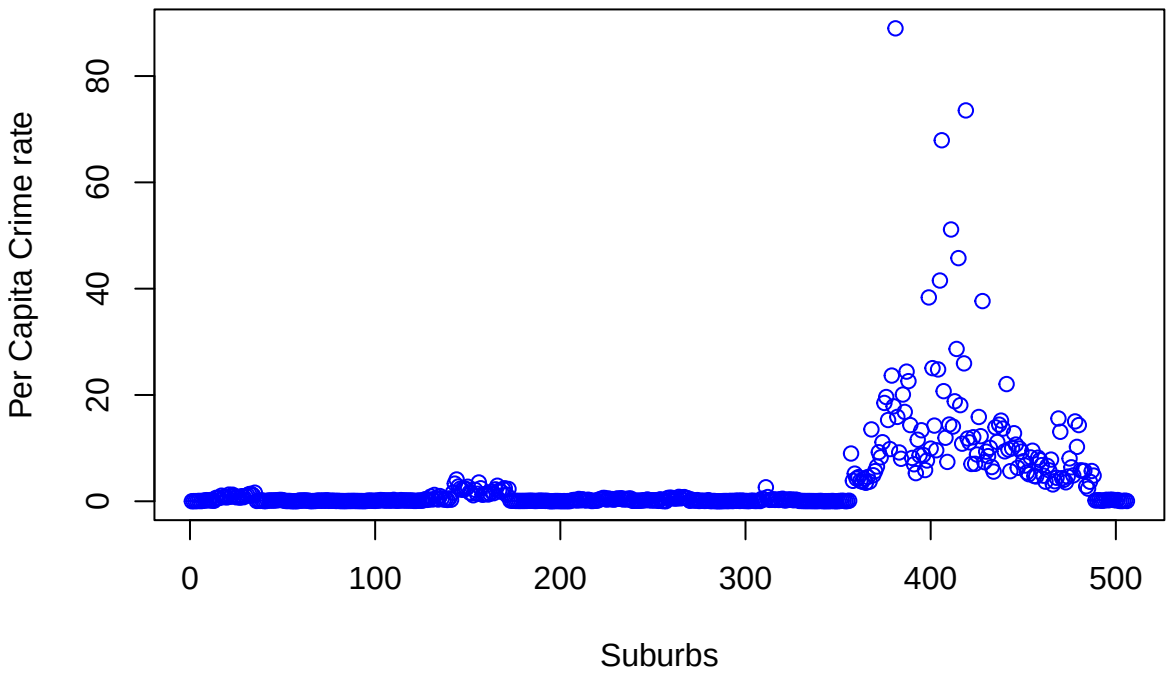
c. re any of the predictors associated with per capita crime rate? If so, explain the relationship.

Using the lower panel of the scatter plot above, we see that there is some sort of linear association between crim and rad, tax, lstat.

```
pairs(crim ~ rad + tax+ lstat, data=Boston, col="blue", lower.panel=NULL)
```




```
plot(Boston$crim, col="blue", xlab="Suburbs", ylab="Per Capita Crime rate")
```



Here, we also observe that for some specific value of tax and rad, the crime rate is highly varying.

We can also see that there are a large number of suburbs with crim value close to zero.

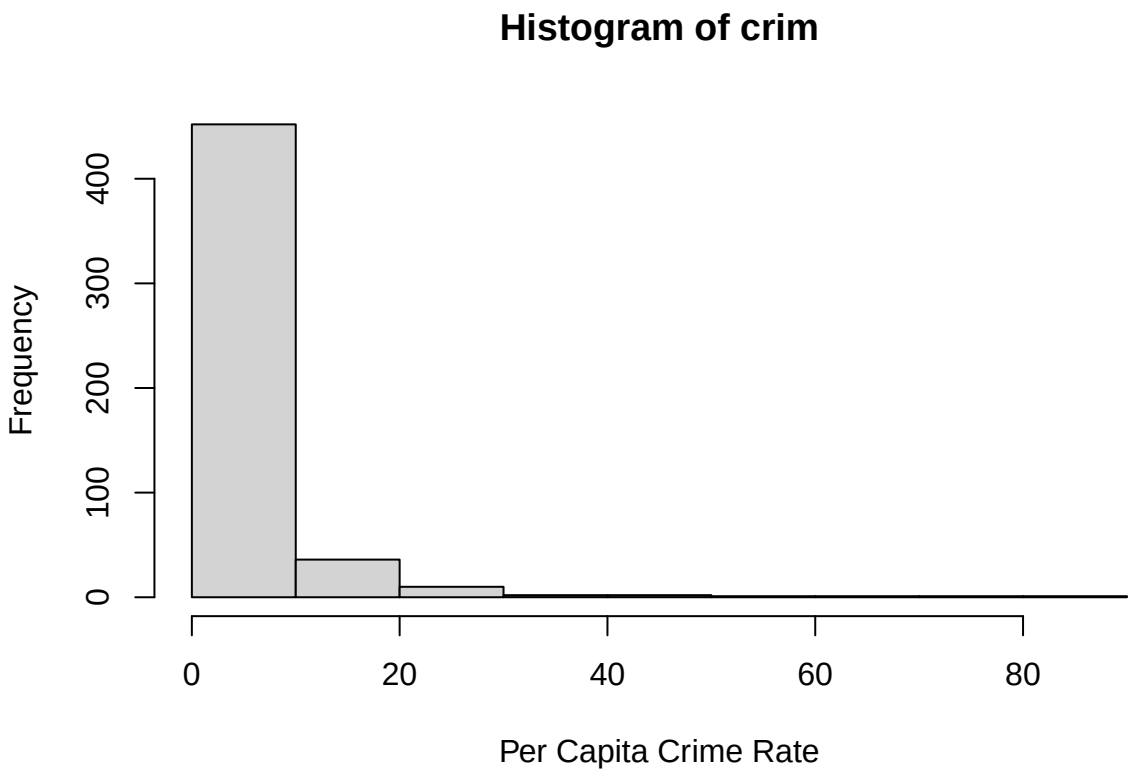
d. Do any of the suburbs of Boston appear to have particularly high crime rates? Tax rates? Pupil-teacher ratios? Comment on the range of each predictor.

```
boxplot(Boston$crim, ylab="Per Capita Crime Rate", main="Box plot of crim")
```


```
range(Boston$crim)
```

```
## [1]  0.00632 88.97620
```

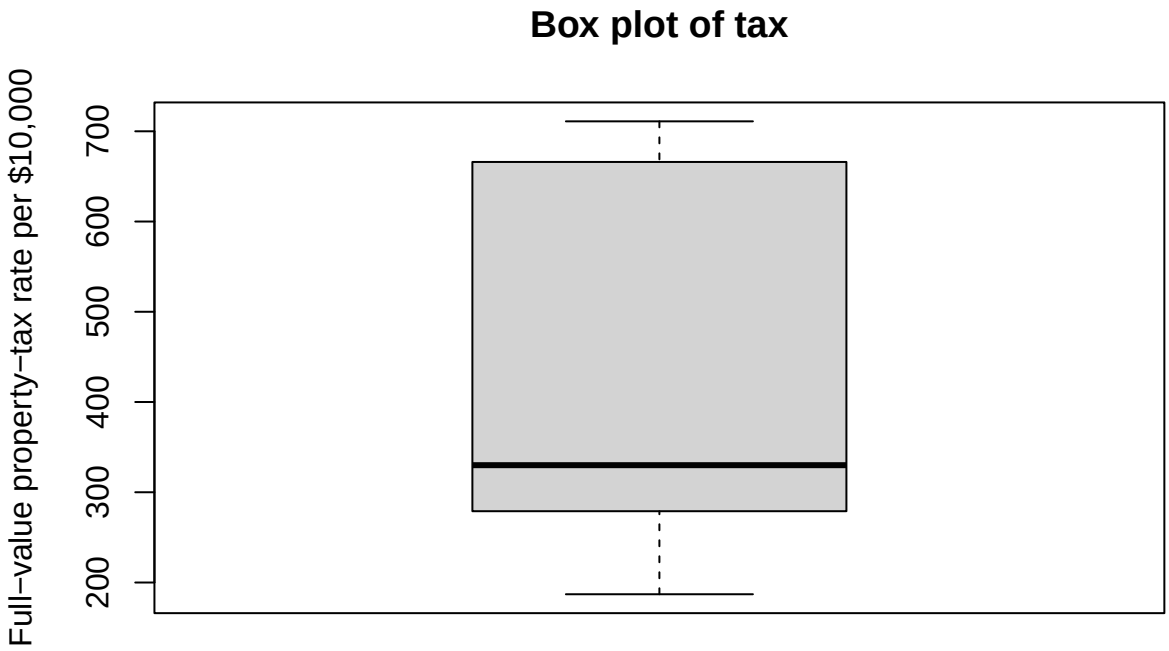
```
hist(Boston$crim, xlab="Per Capita Crime Rate", main="Histogram of crim")
```



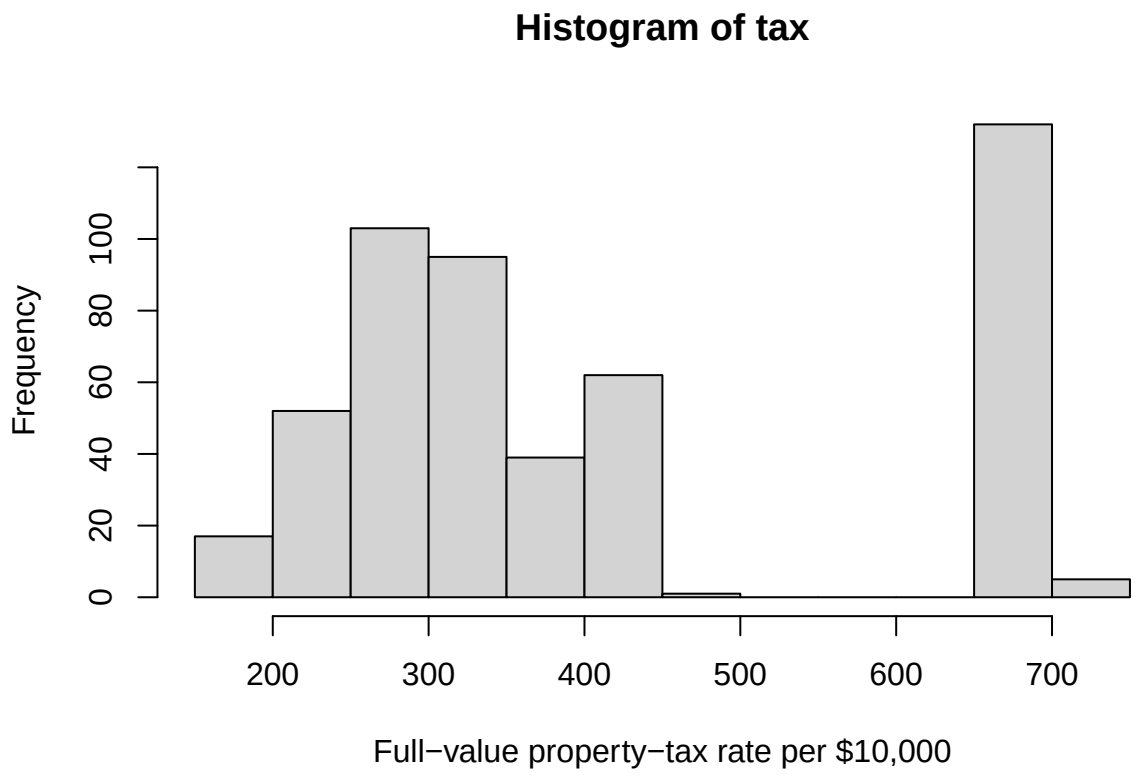

```
?hist
```

We can see that the Per Capita Crime Rate for suburbs of Boston vary from 0.00632 to 88.97620 From the boxplot and histogram, we observe that the majority of suburbs have values close to zero. There are however some suburbs that have particularly high crime rates.

```
boxplot(Boston$tax, ylab="Full-value property-tax rate per $10,000", main="Box plot of tax")
```




```
hist(Boston$tax, xlab="Full-value property-tax rate per $10,000", main="Histogram of tax")
```




```
range(Boston$tax)
```

```
## [1] 187 711
```

The value of the variable is ranging from 187 to 711. From the histogram, we see that there a huge number of suburbs taking the same tax values far from others. This could happen because they fall under the same tax rate.

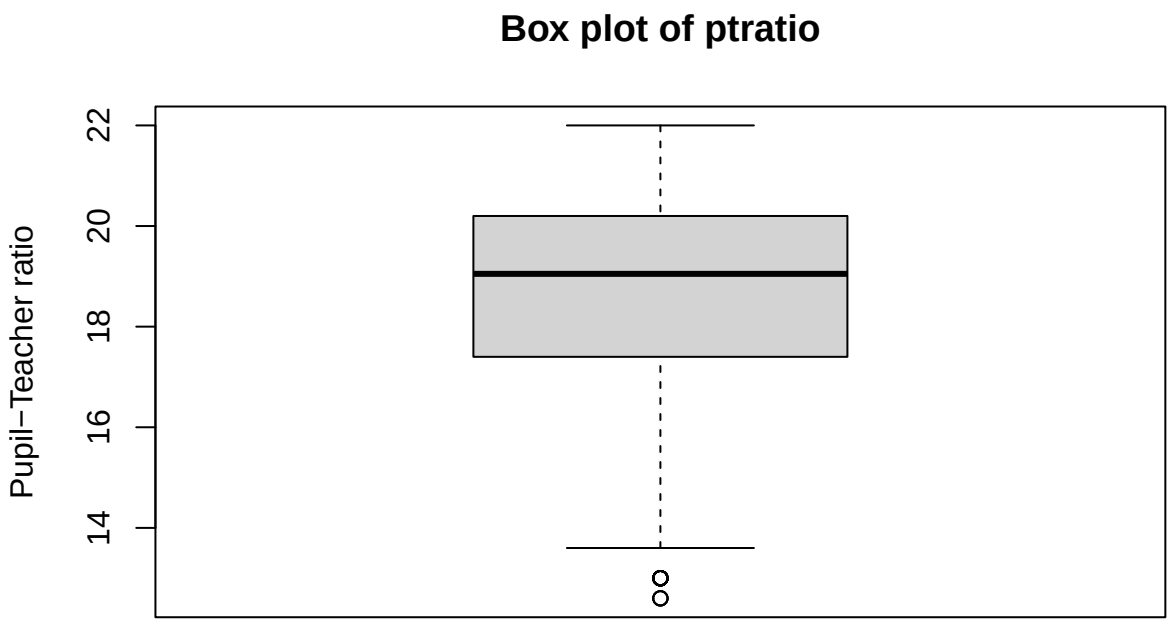
```
Boston %>%
  group_by(tax) %>%
  count() %>%
  arrange(desc(n))
```

```
## # A tibble: 66 x 2
## # Groups:   tax [66]
##      tax     n
##    <dbl> <int>
##  1   666   132
##  2   307    40
##  3   403    30
##  4   437    15
##  5   304    14
##  6   264    12
##  7   398    12
```

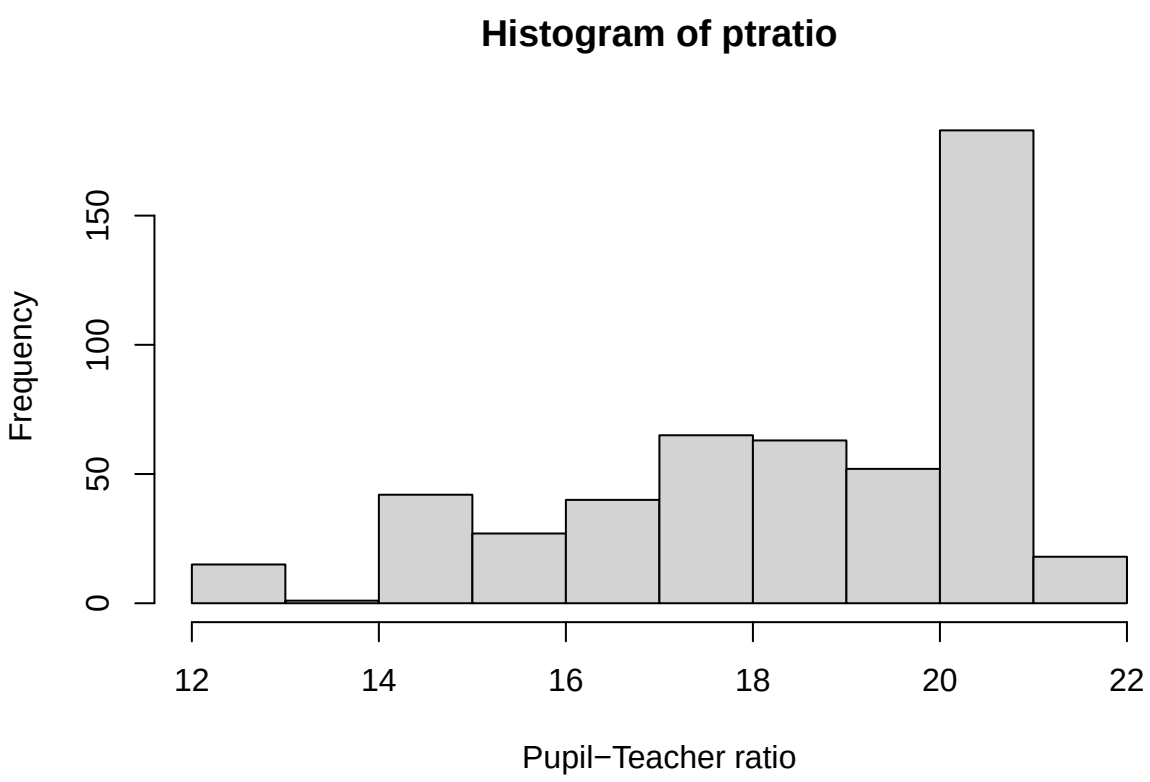
```
##  8   277    11
##  9   384    11
## 10   224    10
## # ... with 56 more rows
```

Here, we see that the value 666 for tac occurs 132 times.

```
boxplot(Boston$ptratio, ylab="Pupil-Teacher ratio", main="Box plot of ptratio")
```




```
hist(Boston$ptratio, xlab="Pupil-Teacher ratio", main="Histogram of ptratio")
```




```
range(Boston$ptratio)
```

```
## [1] 12.6 22.0
```

Pupil-Teacher ratio does not seem to have any huge outliers or gaps and is over a smaller range. However, there are many observations with same value.

```
Boston %>%
  group_by(ptratio) %>%
  count() %>%
  arrange(desc(n))
```

```
## # A tibble: 46 x 2
## # Groups:   ptratio [46]
##    ptratio     n
##      <dbl> <int>
##  1    20.2   140
##  2    14.7    34
##  3    21      27
##  4    17.8    23
##  5    19.2    19
##  6    17.4    18
##  7    18.6    17
##  8    19.1    17
```

```
##  9    16.6    16
## 10    18.4    16
## # ... with 36 more rows
```

The observation 20.2 for ptratio occurs for 140 suburbs.

This results were also observed in the pairwise scatterplot in 7b. We observe that there are some suburbs take the same tax, rad, zn, indus and ptratio values.

```
Boston %>%
  group_by(ptratio, tax, indus, rad, zn) %>%
  count() %>%
  arrange(desc(n))
```

```
## # A tibble: 78 x 6
## # Groups:   ptratio, tax, indus, rad, zn [78]
##    ptratio   tax indus   rad    zn     n
##      <dbl> <dbl> <dbl> <int> <dbl> <int>
##  1    20.2   666 18.1     24     0   132
##  2    14.7   403 19.6      5     0    30
##  3    21     307  8.14     4     0    22
##  4    17.4   307  6.2      8     0    18
##  5    21.2   437 21.9      4     0    15
##  6    13     264  3.97     5    20    12
##  7    18.4   304  9.9      4     0    12
##  8    18.6   277 10.6      4     0    11
##  9    20.9   384  8.56     5     0    11
## 10    19.1   330  5.86     7    22    10
## # ... with 68 more rows
```

Here, we verify that 132 of these suburbs have the same values for ptratio, rad, tax, zn and indus.

```
Boston %>%
  group_by(ptratio, tax, indus, rad, zn, crim) %>%
  count() %>%
  arrange(desc(n))
```

```
## # A tibble: 505 x 7
## # Groups:   ptratio, tax, indus, rad, zn, crim [505]
##    ptratio   tax indus   rad    zn    crim     n
##      <dbl> <dbl> <dbl> <int> <dbl>   <dbl> <int>
##  1    20.2   666 18.1     24     0 14.3        2
##  2    12.6   329  1.52     2    80  0.0377     1
##  3    12.6   329  1.52     2    80  0.0401     1
##  4    12.6   329  1.52     2    80  0.0467     1
##  5    13     264  3.97     5    20  0.520      1
##  6    13     264  3.97     5    20  0.534      1
##  7    13     264  3.97     5    20  0.540      1
##  8    13     264  3.97     5    20  0.540      1
##  9    13     264  3.97     5    20  0.550      1
## 10    13     264  3.97     5    20  0.578      1
## # ... with 495 more rows
```

Here, when we take crim into consideration with these five variables that have some particular value that keeps on repeating, we don't see a larger cluster with the same value.

e. How many of the suburbs in this data set bound the Charles river?

```
Boston %>%
        group_by(chas) %>%
        tally()
```

```
## # A tibble: 2 x 2
##    chas     n
##   <int> <int>
## 1     0   471
## 2     1    35
```

There are 35 suburbs that are bound by the Charles river.

f. What are the mean and standard deviation of the pupil-teacher ratio among the towns in this data set?

```
mean(Boston$ptratio)
```

```
## [1] 18.45553
```

```
sd(Boston$ptratio)
```

```
## [1] 2.164946
```

g. Which suburb of Boston has highest median value of owner-occupied homes? What are the values of the other predictors for that suburb, and how do those values compare to the overall ranges for those predictors? Comment on your findings.

```
max(Boston$medv)
```

```
## [1] 50
```

```
maxMedv<-Boston[Boston$medv ==max(Boston$medv),]
```

It appears that there are 16 suburbs with the highest median value of owner-occupied homes (50).

```
percentile <- sapply(Boston[ ,-4], function(x) rank(x)/length(x)) %>%
  as.data.frame()

perg<-data.frame(percentile[c(rownames(maxMedv)), ])
perg
```

```
##           crim        zn       indus        nox         rm        age
## 162 0.68972332 0.3685771 0.917984190 0.68873518 0.94664032 0.67687747
## 163 0.70355731 0.3685771 0.917984190 0.68873518 0.96047431 0.88241107
## 164 0.69367589 0.3685771 0.917984190 0.68873518 0.99209486 0.74604743
## 167 0.70553360 0.3685771 0.917984190 0.68873518 0.97430830 0.82015810
## 187 0.17193676 0.3685771 0.080039526 0.31916996 0.96640316 0.31324111
## 196 0.01383399 0.9584980 0.001976285 0.09288538 0.97035573 0.14525692
## 205 0.03557312 0.9950593 0.089920949 0.08893281 0.97628458 0.14130435
## 226 0.58893281 0.3685771 0.340909091 0.38833992 0.99802372 0.55830040
## 258 0.61067194 0.7806324 0.170948617 0.77569170 0.99604743 0.61067194
## 268 0.60671937 0.7806324 0.170948617 0.59980237 0.98814229 0.41106719
## 284 0.02075099 0.9861660 0.005928854 0.02569170 0.97233202 0.09683794
## 369 0.79051383 0.3685771 0.757905138 0.76086957 0.02766798 0.95849802
## 370 0.80434783 0.3685771 0.757905138 0.76086957 0.77667984 0.83399209
## 371 0.82411067 0.3685771 0.757905138 0.76086957 0.87944664 0.85968379
## 372 0.87351779 0.3685771 0.757905138 0.76086957 0.51185771 0.95849802
## 373 0.85968379 0.3685771 0.757905138 0.79841897 0.23616601 0.66007905
##             dis        rad        tax    ptratio       lstat      medv
## 162 0.209486166 0.49407115 0.63932806 0.06818182 0.001976285 0.9851779
## 163 0.233201581 0.49407115 0.63932806 0.06818182 0.003952569 0.9851779
## 164 0.276679842 0.49407115 0.63932806 0.06818182 0.035573123 0.9851779
## 167 0.235177866 0.49407115 0.63932806 0.06818182 0.051383399 0.9851779
## 187 0.500000000 0.12549407 0.02470356 0.32213439 0.082015810 0.9851779
## 196 0.802371542 0.27173913 0.15217391 0.03359684 0.017786561 0.9851779
## 205 0.748023715 0.27173913 0.07806324 0.06818182 0.011857708 0.9851779
## 226 0.463438735 0.71640316 0.41600791 0.26778656 0.098814229 0.9851779
## 258 0.150197628 0.49407115 0.16699605 0.01877470 0.132411067 0.9851779
## 268 0.345849802 0.49407115 0.16699605 0.01877470 0.279644269 0.9851779
## 284 0.820158103 0.02075099 0.03359684 0.03162055 0.030632411 0.9851779
## 369 0.019762846 0.87055336 0.86067194 0.75197628 0.033596838 0.9851779
## 370 0.025691700 0.87055336 0.86067194 0.75197628 0.053359684 0.9851779
## 371 0.011857708 0.87055336 0.86067194 0.75197628 0.015810277 0.9851779
## 372 0.005928854 0.87055336 0.86067194 0.75197628 0.401185771 0.9851779
## 373 0.001976285 0.87055336 0.86067194 0.75197628 0.357707510 0.9851779
```

```
sapply(perg, mean)
```

```
##      crim        zn     indus       nox        rm       age       dis       rad
## 0.5620677 0.5347085 0.5201334 0.5554595 0.8233078 0.6045578 0.3031126 0.5452075
##       tax   ptratio     lstat      medv
## 0.4937006 0.2995924 0.1004817 0.9851779
```

I create a data frame percentile, which gives the percentile rank for every observation in the data frame, which I then filter for these 16 observations to see where they rank in the ranges of each variable. We can see that 16 observations with the highest medv take very similar values, and for many they take quite extreme values.

To simplify the result, I simply took the mean. Most of the variables are near the 50th percentile while others are more extreme, especially the lstat is in the bottom 10th percentile and the rm in the top 20th percentile.

h. In this data set, how many of the suburbs average more than six rooms per dwelling? More than eight rooms per dwelling? Comment on the suburbs that average more than eight rooms per dwelling.

```
length(which(Boston$rm>6))
```

```
## [1] 333
```

```
length(which(Boston$rm>8))
```

```
## [1] 13
```

```
boston8rooms<-Boston[Boston$rm >8,]
```

```
boston8rooms_per <-data.frame(percentile[c(rownames(boston8rooms)), ])
boston8rooms_per
```

```
##           crim        zn      indus        nox        rm        age       dis
## 98  0.34387352 0.3685771 0.09683794 0.21541502 0.9802372 0.48418972 0.5454545
## 164 0.69367589 0.3685771 0.91798419 0.68873518 0.9920949 0.74604743 0.2766798
## 205 0.03557312 0.9950593 0.08992095 0.08893281 0.9762846 0.14130435 0.7480237
## 225 0.52766798 0.3685771 0.34090909 0.38833992 0.9861660 0.50988142 0.4634387
## 226 0.58893281 0.3685771 0.34090909 0.38833992 0.9980237 0.55830040 0.4634387
## 227 0.56126482 0.3685771 0.34090909 0.38833992 0.9782609 0.60770751 0.5029644
## 233 0.60474308 0.3685771 0.34090909 0.40612648 0.9901186 0.46442688 0.5968379
## 234 0.53952569 0.3685771 0.34090909 0.40612648 0.9822134 0.43972332 0.5652174
## 254 0.55533597 0.8191700 0.28952569 0.13339921 0.9841897 0.01976285 0.9792490
## 258 0.61067194 0.7806324 0.17094862 0.77569170 0.9960474 0.61067194 0.1501976
## 263 0.58498024 0.7806324 0.17094862 0.77569170 0.9940711 0.69367589 0.3122530
## 268 0.60671937 0.7806324 0.17094862 0.59980237 0.9881423 0.41106719 0.3458498
## 365 0.74308300 0.3685771 0.75790514 0.92193676 1.0000000 0.55434783 0.1867589
##            rad        tax    ptratio       lstat      medv
## 98  0.06422925 0.21541502 0.37154150 0.075098814 0.9367589
## 164 0.49407115 0.63932806 0.06818182 0.035573123 0.9851779
## 205 0.27173913 0.07806324 0.06818182 0.011857708 0.9851779
## 225 0.71640316 0.41600791 0.26778656 0.071146245 0.9565217
## 226 0.71640316 0.41600791 0.26778656 0.098814229 0.9851779
## 227 0.71640316 0.41600791 0.26778656 0.027667984 0.9328063
## 233 0.71640316 0.41600791 0.26778656 0.007905138 0.9426877
## 234 0.71640316 0.41600791 0.26778656 0.064229249 0.9644269
## 254 0.67588933 0.49703557 0.51778656 0.043478261 0.9466403
## 258 0.49407115 0.16699605 0.01877470 0.132411067 0.9851779
## 263 0.49407115 0.16699605 0.01877470 0.179841897 0.9683794
## 268 0.49407115 0.16699605 0.01877470 0.279644269 0.9851779
## 365 0.87055336 0.86067194 0.75197628 0.143280632 0.5454545
```

```
sapply(boston8rooms_per, mean)
```

```
##       crim         zn      indus        nox         rm        age        dis
## 0.53815749 0.54651870 0.33612040 0.47514442 0.98814229 0.48008513 0.47202797
##        rad        tax    ptratio      lstat       medv
## 0.57236242 0.37473396 0.24407115 0.09007297 0.93227425
```

Similar to the previous question, most of the variables don't have extreme values and fall somewhere near the 50th percentile. The exception of lstat which lies in the bottom 10th percentile and medv in the top 10th percentile.

We could say that these suburbs are more well off.