HW1 Submission

1.

A. The pros and cons of the parametric approach are

Pros

- It assumes a functional form of the function which is linear and hence estimating the ‘f’ reduces to estimating ‘p’ coefficients where p is the number of dimensions in the data.
- Can be used when we have a limited number of samples.

Cons

- It can be very far from the true function. And hence the relationship between the variables may not be captured accurately.

The pros and cons of the non-parametric approach are:

Pros

- Since it doesn’t assume any functional form of the function, it can accurately fit a wide range of possible shapes for the function.
- Less risk of getting a function very far away from the true function.

Cons

- The problem of estimating the function is not reduced to just estimating coefficients like in parametric approaches.
- A large number of samples are required

B. If we have a small sample size or if we have a good understanding of the distribution of the data, the parametric approach should be used. In this case, we are assuming a functional form of the function. So, it's very important that we have a good idea about the actual distribution, or else the inferences or the predictions can be very wrong.

C. If we have a large number of samples and we have no idea of the distribution of the data, then a non-parametric approach can be used to get better accuracy. Since we are observing the actual data to make an estimate, we need to have a good sample size. In this case, we can get an estimate which is very close to the true function.

2.

A. In this setting, we can expect a more **inflexible** statistical machine learning model to perform better. We have less sample size and more predictors and hence, if we use a more flexible model in this case, it can make the model complex and it can cause overfitting. So, the accuracy of the new observations will be very low or the variance will be large.

B. In this setting, we can expect a **flexible** statistical machine learning model to perform better. If we use an inflexible machine learning model, then our model may be very simple and underfit data which can lead to low accuracy in both training and test data. A flexible model on a large sample size is less likely to overfit and the bias will also be low.

C. A **flexible** model will tend to perform better because the model needs to be flexible to capture the non-linear relationship or else the bias and variance both will be high.

D. **Inflexible** model will perform better since the variance term is large and a flexible model will probably overfit and capture too much noise.

3.

A. N=50,p=8.

It is a regression problem and the goal is to predict.

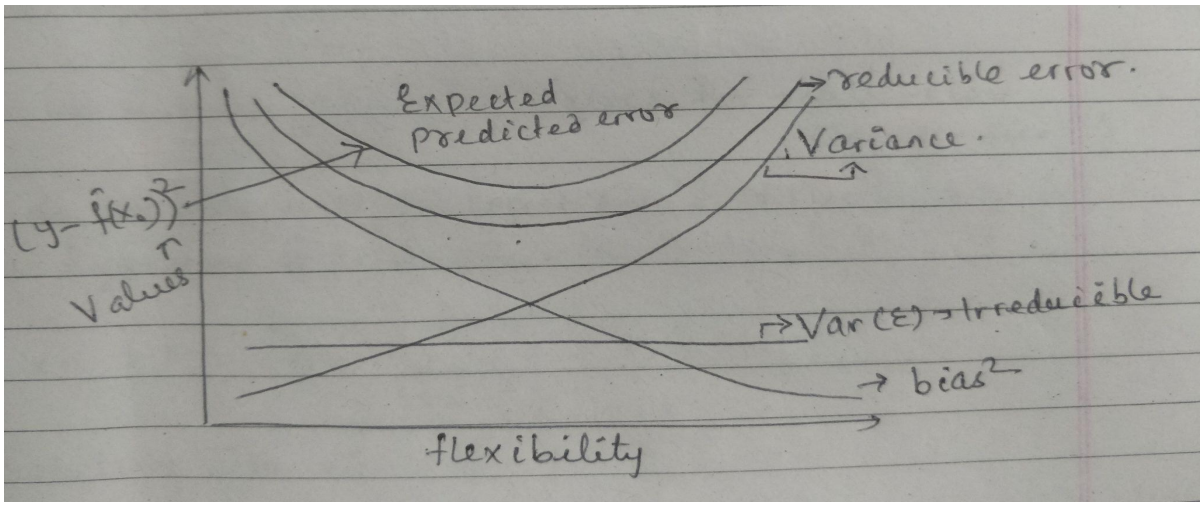
B. N=50, p=6

It is a classification problem and the goal is to infer.

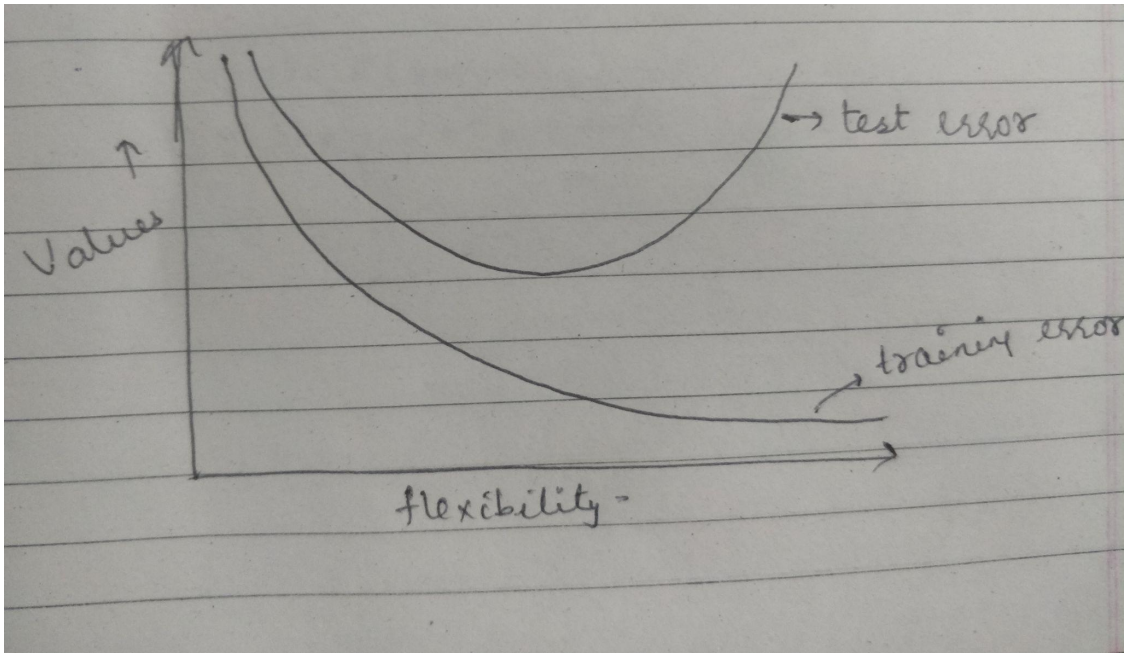
4.

A. If the model is too flexible and to the extreme right, it will have low bias but high variance i.e overfitting, and on the other hand, if the model is less flexible, it will have low variance but high bias i.e case of underfitting. And in both cases, the expected prediction error will be high.

Hence, a sweet spot would be the level of flexibility where the expected test prediction error is the minimum.



B. The level of flexibility where the test error is the minimum would be the best.



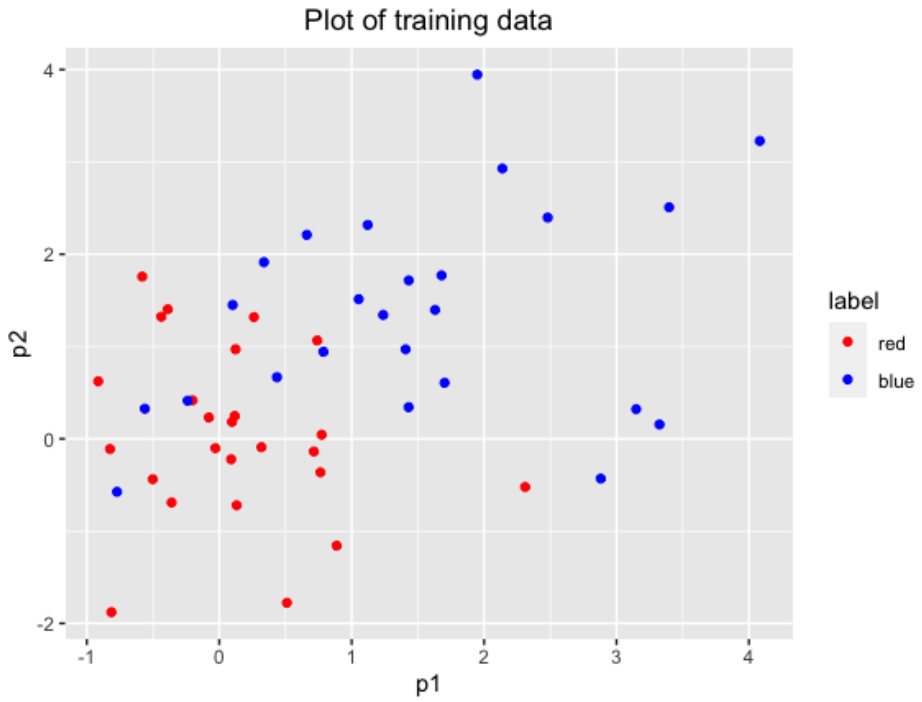
C. The f-hat, where the function is highly flexible i.e to the extreme right in the flexibility curve has an extremely low bias and extremely high variance.
A flexible non-linear model like k-nn with k=1 or 2 will have extremely low bias but high variance.

D. The f-hat, where the function is less flexible i.e to the extreme left in the flexibility curve has an extremely low variance and extremely high bias.
An example is a linear regression line that is parallel to the x-axis and, has zero variance and extremely high bias.

5.

A. The task is to generate a training set with 25 observations in each class -Red and Blue. rnorm function is used to generate random observations from a normal distribution. There are 2 features. So for each class, and each feature for that class, observations from normal distribution will be generated with mean and standard deviation as stated.
Below is the plot of training data.

```
set.seed(100)
#create training samples
p1_x <- rnorm(25,mean=0,sd=1)
p2_x <- rnorm(25,mean=0,sd=1)
x <- matrix(c(p1_x,p2_x),nrow=25,ncol=2)
colMeans(x)
df_x <- data.frame(x)
df_x$label <- 'red'
p1_y <- rnorm(25,mean=1.5,sd=1)
p2_y <- rnorm(25,mean=1.5,sd=1)
y <- matrix(c(p1_y,p2_y),nrow=25,ncol=2)
colMeans(y)
df_y <- data.frame(y)
df_y$label <- 'blue'
data <- matrix(c(x,y),nrow=50,ncol=2)

names(df_x)[1:2] <- c('p1','p2')
names(df_y)[1:2] <- c('p1','p2')
df_training <- rbind(df_x,df_y)
df_training$sample <- 'training'

#plot training data
plot1= ggplot(data=df_training,aes(p1,p2,color=label)) +
  geom_point()+scale_color_manual(values = c("red" = "red", "blue" = "blue"))
plot1 + ggtitle('Plot of training data') + theme(plot.title = element_text(hjust=0.5))
```



B. The task is to generate a similar dataset as generated above for the training set and plot training and test data in the same plot. For this task, I combined training and test data that was generated separately and created a new variable that identifies which data is training and which is the test. ggplot is used to generate plots. In the ‘aes’ argument, I have assigned ‘shape’ to the training/test sample identifier and ‘color’ to labels.

```
set.seed(200)
#create test samples
p1_x <- rnorm(25,mean=0,sd=1)
p2_x <- rnorm(25,mean=0,sd=1)
x <- matrix(c(p1_x,p2_x),nrow=25,ncol=2)
colMeans(x)
df_x <- data.frame(x)
df_x$label <- 'red'
p1_y <- rnorm(25,mean=1.5,sd=1)
p2_y <- rnorm(25,mean=1.5,sd=1)
y <- matrix(c(p1_y,p2_y),nrow=25,ncol=2)
colMeans(y)
df_y <- data.frame(y)
df_y$label <- 'blue'
data <- matrix(c(x,y),nrow=50,ncol=2)

names(df_x)[1:2] <- c('p1','p2')
names(df_y)[1:2] <- c('p1','p2')
df_test <- rbind(df_x,df_y)
df_test$sample <- 'test'
#plot test data
plot2 = ggplot(data=df_test,aes(p1,p2,color=label)) + geom_point()+
  scale_color_manual(values = c("red" = "red", "blue" = "blue"))
plot2 + ggtitle('Plot of test data') + theme(plot.title = element_text(hjust=0.5))
```

```
#combine training and test sample
data <- rbind(df_training,df_test)
plot3 = ggplot(data=data,aes(p1,p2,shape=sample,color=label)) +
  geom_point()+scale_color_manual(values = c("red" = "red", "blue" = "blue"))
plot3 + ggtitle('Plot of training vs test data') + theme(plot.title = element_text(hjust=0.5))

X_train <- subset(df_training,select=c('p1','p2'))
y_train <- df_training$label
X_test <- subset(df_test,select=c('p1','p2'))
y_test <- df_test$label
```

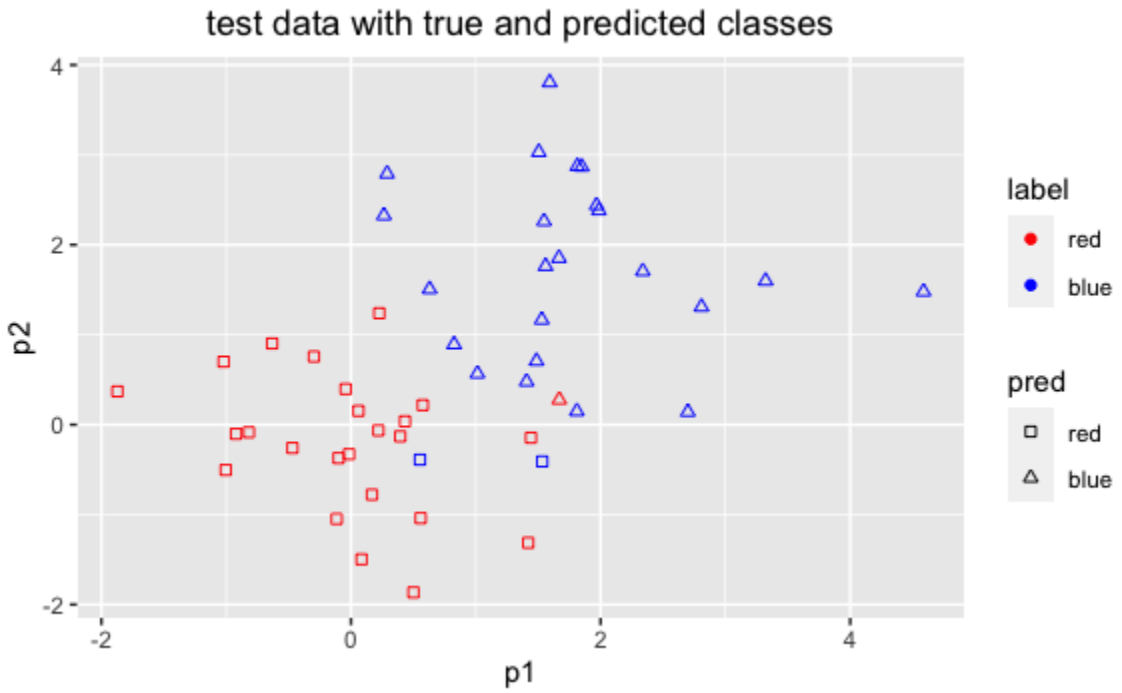
C. The knn is simulated 20 times and error is calculated in both training and test data. With an increase in k, the training error increased a lot initially till k=5 and then oscillated after a further increase in k. Similarly, test error decreased at a steady pace till k=5 and then oscillated with further increase.
This means after a certain value of k, there is no further significant improvement in the error rate.
From the plot, it was observed that the training and test error were both low for k=5 where the training error is 14% and the test error is 6%.
Below is the plot of 1/k vs error rate in training and test data. We can see that after k=5, all the values oscillate and are very close to each other which makes sense because we have very few observations and distributions are close due to sd=1 and closer means, so the error rate will not change significantly after a certain k value.

```
KNN_func2 <- function(X_train, y_train, X_test, y_test,n,obs)
{
  er.tr <- c(); er.ts <- c()
  for (k in 1:n) {
    set.seed(60)
    knnTr <- knn(X_train, X_train, y_train, k)
    set.seed(60)
    knnTs <- knn(X_train, X_test, y_train, k)
    trTable <- table(knnTr, y_train)
    tsTable <- table(knnTs, y_test)
    erTr <- (trTable[1,2] + trTable[2,1])/obs
    erTs <- (tsTable[1,2] + tsTable[2,1])/obs
    er.tr <- c(er.tr,erTr)
    er.ts <- c(er.ts,erTs)
  }
  error_rate <- data.frame(k=1:n, train_err=er.tr, test_err=er.ts)
  return(error_rate)
}

set.seed(60)
err2 <- KNN_func2(X_train, y_train, X_test, y_test,20,50)
```

```
set.seed(60)
err2 <- KNN_func2(X_train, y_train, X_test, y_test,20,50)



plot4 = ggplot(data=err2,aes(1/k)) + geom_point(aes(y=train_err,colour='train error')) +
  geom_line(aes(y=train_err,colour='train error'))
plot4= plot4 + geom_point(aes(y=test_err,colour='test error')) +
  geom_line(aes(y=test_err,colour='test error'))
plot4 + ggtitle('Plot of training vs test error rate') +
  theme(plot.title = element_text(hjust=0.5)) +labs(x='1/k',y='error rate')
```

D. The true classes are colored red and blue as per the classes. The predicted classes are given shapes of the square for red and triangle for blue.
So, the red squares and blue triangles are all correctly classified, and the remaining are misclassified.

```
set.seed(60)
df_test$pred = knn(train=X_train, test=X_test, cl=y_train, k=5)
plot5 = ggplot(data=df_test,aes(p1,p2, color=label, shape=pred)) +
  geom_point() +scale_color_manual(values = c("red" = "red", "blue" = "blue"))
plot5 + scale_shape_manual(values=c("red"=0,"blue"=2)) +
  ggtitle('test data with true and predicted classes') + theme(plot.title = element_text(hjust=0.5))
```



E. The Bayes error rate is 14.44% in this problem. The Bayes error rate is calculated computationally using simulations. 10000 observations were generated for class 'red' and 10000 for class 'blue'. The error rate is calculated by taking the square distance of observations from the true mean of both classes. If an observation is closer to the mean of class 'red' than class 'blue', it is classified as 'red'.

```r
set.seed(99)
p1_x <- rnorm(10000,mean=0,sd=1)
p2_x <- rnorm(10000,mean=0,sd=1)
x <- matrix(c(p1_x,p2_x),nrow=10000,ncol=2)
df_x <- data.frame(x)
df_x$label <- 'red'
p1_y <- rnorm(10000,mean=1.5,sd=1)
p2_y <- rnorm(10000,mean=1.5,sd=1)
y <- matrix(c(p1_y,p2_y),nrow=10000,ncol=2)
df_y <- data.frame(y)
df_y$label <- 'blue'
data <- matrix(c(x,y),nrow=20000,ncol=2)

names(df_x)[1:2] <- c('p1','p2')
names(df_y)[1:2] <- c('p1','p2')
df_training_naive <- rbind(df_x,df_y)

euclidean <- function(a, b,c,d) {return((a - b)^2 + (c - d)^2)}
data_naive <- rbind(df_training_naive,df_test_naive)
```

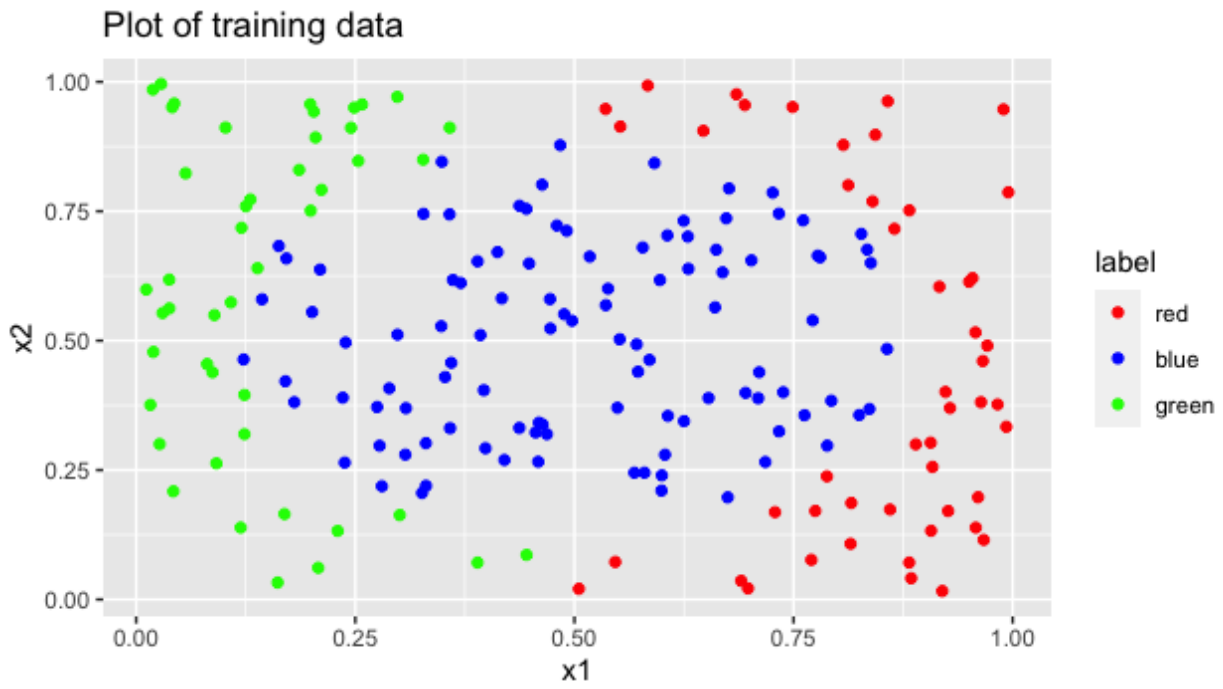
```r
naive_bayes_err <- function(data,obs){
  data$dR <- euclidean(data$p1,0,data$p2,0)
  data$dB <- euclidean(data$p1,1.5,data$p2,1.5)
  data$predict <- ifelse(data$dR < data$dB,'red','blue')
  trTable <- table(data$predict, data$label)
  print(trTable)
  erTr <- (trTable[1,2] + trTable[2,1])/obs
  return(erTr)
}

err_naive <- naive_bayes_err(df_training_naive,20000)
err_naive
```

```
       blue  red
  blue 8538 1426
  red  1462 8574
> err_naive
[1] 0.1444
```

6.

A. The task is to generate a training set with 200 observations with classes -Red and Blue and Green. runif function is used to generate random observations from a uniform distribution. There are 2 features. The classes are defined based on the conditions given in the question.
Below is the plot of training data.

```
#create training sample
set.seed(100)
x1 <- runif(200, 0,1)
x2 <- runif(200, 0, 1)

x <- matrix(c(x1,x2),nrow=200,ncol=2)
df_training<- data.frame(x)
names(df_training)[1:2]<- c('x1','x2')
df_training$label <- 'NA'
df_training$label <- ifelse(((((df_training$x1 -0.5)^2 + (df_training$x2-0.5)^2 ) >0.15 ) & df_training$x1 >0.5),'red','blue')
df_training$label <- ifelse((((df_training$x1 -0.5)^2 + (df_training$x2-0.5)^2 ) >0.15 ) & df_training$x1 <=0.5,'green',df_training$label )
print(table(df_training$label))

#sample name
df_training$sample <- 'training'

#plot training data
plot1= ggplot(data=df_training,aes(x1,x2,color=label)) +
  geom_point()+scale_color_manual(values = c("red" = "red", "blue" = "blue","green"="green"))
plot1 + ggtitle('Plot of training data')
```



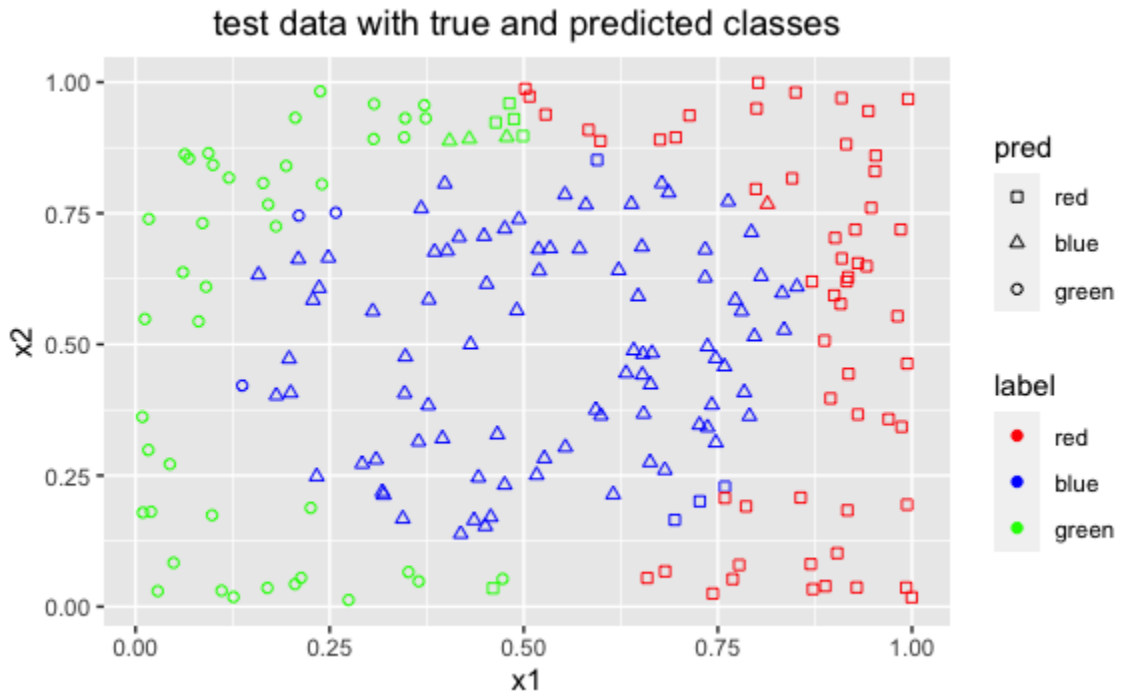
B. The task is to generate a similar dataset as generated above for the training set and plot training and test data in the same plot. For this task, I combined training and test data that was generated separately and created a new variable that identifies which data is training and which is the test. ggplot is used to generate plots. In the 'aes' argument, I have assigned 'shape' to the training/test sample identifier and 'color' to labels.

```
#create test sample
x1 <- runif(200, 0,1)
x2 <- runif(200, 0, 1)

x <- matrix(c(x1,x2),nrow=200,ncol=2)
df_test<- data.frame(x)
names(df_test)[1:2]<- c('x1','x2')
df_test$label <- 'NA'
# add labels
df_test$label <- ifelse((((df_test$x1 -0.5)^2 + (df_test$x2-0.5)^2 ) >0.15 ) & df_test$x1 >0.5,'red','blue')
df_test$label <- ifelse((((df_test$x1 -0.5)^2 + (df_test$x2-0.5)^2 ) >0.15 ) & df_test$x1 <=0.5,'green',df_test$label )
print(table(df_test$label))

#sample name
df_test$sample <- 'test'

#combine training and test sample
data <- rbind(df_training,df_test)
plot2 = ggplot(data=data,aes(x1,x2,shape=sample,color=label)) +
  geom_point()+scale_color_manual(values = c("red" = "red", "blue" = "blue","green"="green"))
plot2 + ggtitle('Plot of test data') + theme(plot.title = element_text(hjust=0.5))

X_train <- subset(df_training,select=c('x1','x2'))
y_train <- df_training$label
X_test <- subset(df_test,select=c('x1','x2'))
y_test <- df_test$label
```

C.The knn is simulated 50 times and error is calculated in both training and test data. With an increase in k, the test error decreased initially till k=4 and then increased after a further increase in k.
This means after a certain value of k, there is no further significant improvement in the error rate.
From the plot, it was observed that the training and test error were both low for k=4 where the training error is 2.5% and the test error is 3.0%.
Below is the plot of 1/k vs error rate in training and test data. We can see that after k=4, all the error values increase.

```
err <- KNN_func2(X_train, y_train, X_test, y_test,50,200 )

plot4 = ggplot(data=err,aes(1/k)) + geom_point(aes(y=train_err,colour='train error')) +
  geom_line(aes(y=train_err,colour='train error'))
plot4= plot4 + geom_point(aes(y=test_err,colour='test error')) +
  geom_line(aes(y=test_err,colour='test error'))
plot4 + ggtitle('Plot of training vs test error rate') +
  theme(plot.title = element_text(hjust=0.5)) + labs(x='1/k',y='error rate')
```

D. The true classes are colored red, green, and blue as per the classes. The predicted classes are given shapes of the square for red and triangle for blue and circle for green. So, the red squares,green circles, and blue triangles are all correctly classified, and the remaining are misclassified.

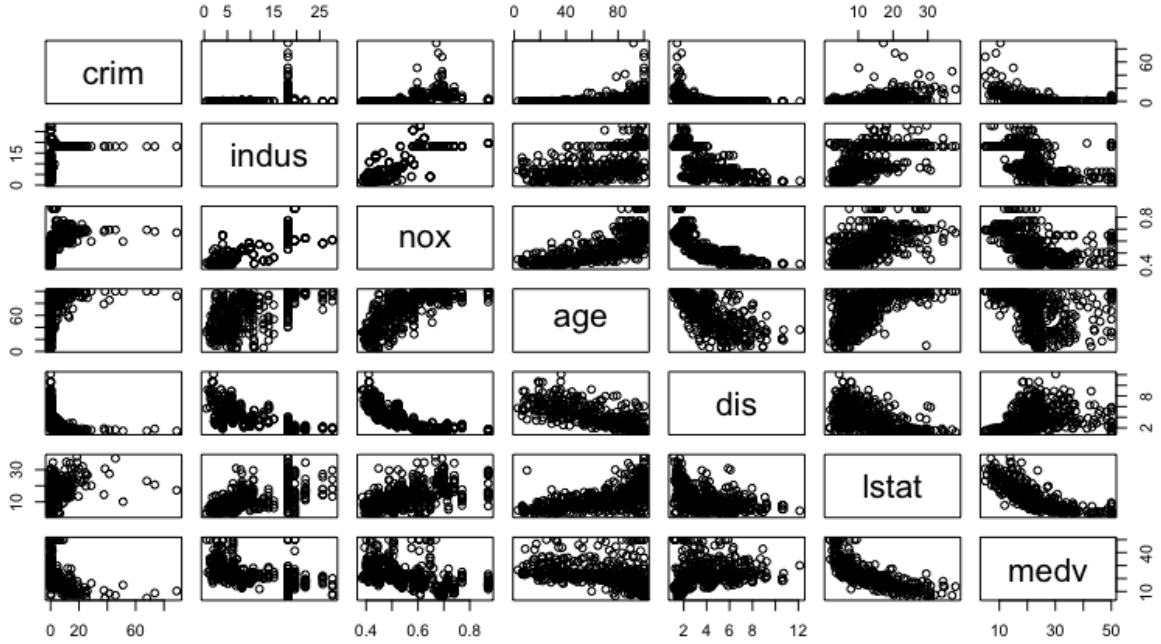


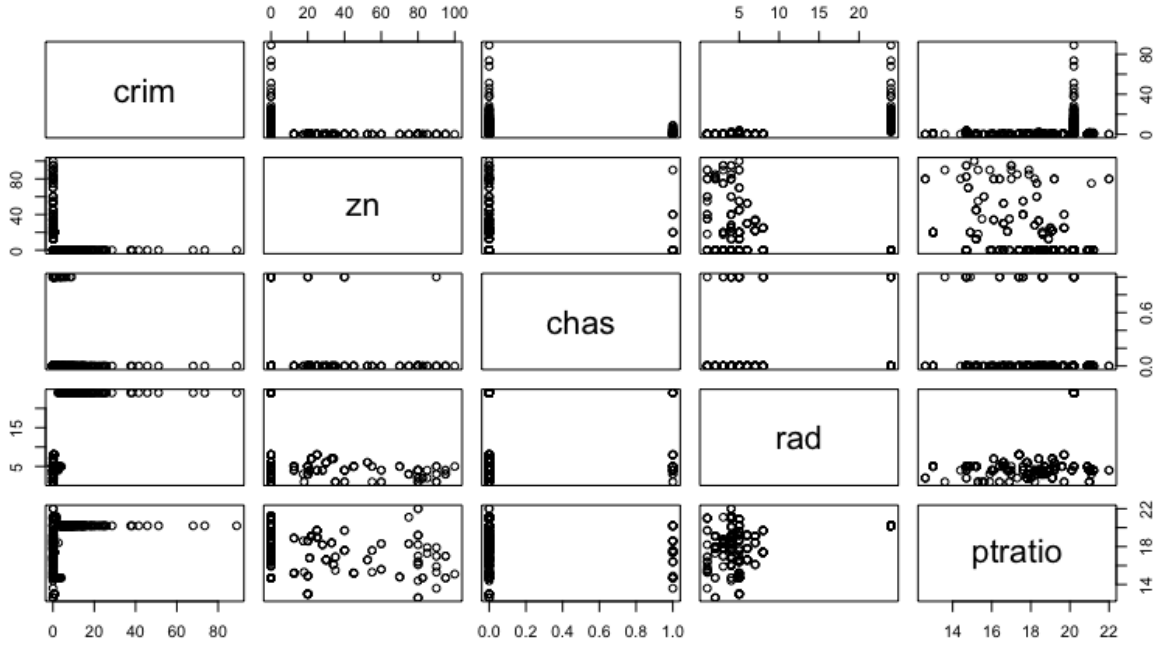
E. The Bayes error rate is zero. Since all the classes are separated from each other and there is no overlap, P(Y=class | X=x) for a class will always be higher than P(Y=class | X=x) for other classes, and hence, Baye's classifier will always classify correctly.

The error rate for training and test for k=4 was 2.5% and 3.0% which is higher than Baye's classifier error rate. It makes sense because ideally, Baye's error rate should be the lowest error rate due to the fact it is only the irreducible error.

7.

A. There is a total of 506 rows and 13 columns. nrow() and ncol() functions are used to find the number of rows and columns respectively. The rows represent the suburbs in the Boston area and columns indicate different metrics that provide information about each suburb like per capita crime rate and proportion of non-retail business per town.
B. pairs( ) function in R is used to plot pairwise scatter plots.





I. With the increase in the proportion of nonretail business acres per town, nitrogen oxide concentration also increases to some extent, and then it stays almost constant.

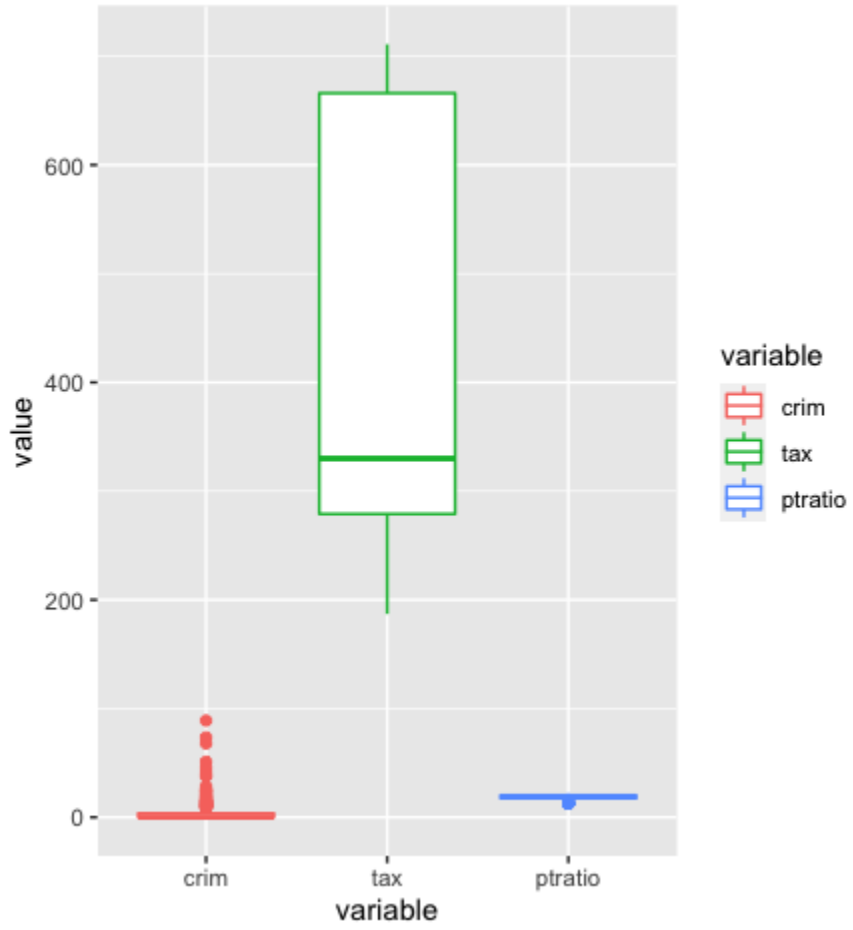
II. ‘age’ seems to be inversely proportional to ‘dis’ which implies that in the suburb where the proportion of owner-occupied units built prior to 1940 is higher, the suburb is closer to five Boston employment centers
III. ‘age’ is directly proportional to ‘nox’ which mean the suburb where the proportion of owner-occupied units built prior to 1940 is higher, nitrogen oxide concentration is also higher.
IV. The percentage of the lower status of the population is inversely proportional to the median value of owner-occupied homes in $1000s.

C. Yes, ‘lstat’ variable is directly proportional to the crime rate but the correlation seems weak. An increase in the proportion of non-retail business acres per town also leads to an increase in the crime rate. Similarly ’medv’ variable also exhibits a weak negative correlation with the crime rate.

D. There are no suburbs that have particularly high tax rates but there are suburbs with high crime rates and teacher-pupil ratios. I used boxplots to identify particularly high values of the predictors.

```
data_sub <- Boston[,c('crim','tax','ptratio')]
data_long <- melt(data_sub)
ggplot(data_long, aes(x = variable, y = value, color = variable)) +
  geom_boxplot()
```

```
boxplot.stats(Boston$ptratio)$out
summary(data_sub)
```



	Min	Median	Max	range
crim	0.00632	0.25	88.97	88.963
tax	187	330	711	524
ptratio	12.60	19.5	22.0	9.4

The range for the crime is large but mostly because there are outliers or high values and the table and boxplots above clearly depict that it has outliers. The median value of ‘crim’ in the table is very far away from the max value. In fact, if we use the IQR to find outliers, it has 66 outliers.

The range of the tax rate is also large with no outliers which implies greater dispersion of data.
The range of pupil-teacher ratios is low which implies less dispersion of data and it has less outliers as well.

E. There is a total of 35 suburbs bound by the Charles river

F. The mean and standard deviation of the pupil-teacher ratio in the town are 18.45 and 2.16 respectively.

G. Below is the list of suburbs with the values of the predictor variables which have the highest median value of owner-occupied homes. There is a total of 16 such suburbs.

```
> Boston[Boston$medv==max(Boston$medv),]
       crim zn indus chas    nox    rm   age    dis rad tax ptratio lstat medv
162 1.46336  0 19.58    0 0.6050 7.489  90.8 1.9709   5 403    14.7  1.73   50
163 1.83377  0 19.58    1 0.6050 7.802  98.2 2.0407   5 403    14.7  1.92   50
164 1.51902  0 19.58    1 0.6050 8.375  93.9 2.1620   5 403    14.7  3.32   50
167 2.01019  0 19.58    0 0.6050 7.929  96.2 2.0459   5 403    14.7  3.70   50
187 0.05602  0  2.46    0 0.4880 7.831  53.6 3.1992   3 193    17.8  4.45   50
196 0.01381 80  0.46    0 0.4220 7.875  32.0 5.6484   4 255    14.4  2.97   50
205 0.02009 95  2.68    0 0.4161 8.034  31.9 5.1180   4 224    14.7  2.88   50
226 0.52693  0  6.20    0 0.5040 8.725  83.0 2.8944   8 307    17.4  4.63   50
258 0.61154 20  3.97    0 0.6470 8.704  86.9 1.8010   5 264    13.0  5.12   50
268 0.57834 20  3.97    0 0.5750 8.297  67.0 2.4216   5 264    13.0  7.44   50
284 0.01501 90  1.21    1 0.4010 7.923  24.8 5.8850   1 198    13.6  3.16   50
369 4.89822  0 18.10    0 0.6310 4.970 100.0 1.3325  24 666    20.2  3.26   50
370 5.66998  0 18.10    1 0.6310 6.683  96.8 1.3567  24 666    20.2  3.73   50
371 6.53876  0 18.10    1 0.6310 7.016  97.5 1.2024  24 666    20.2  2.96   50
372 9.23230  0 18.10    0 0.6310 6.216 100.0 1.1691  24 666    20.2  9.53   50
373 8.26725  0 18.10    1 0.6680 5.875  89.6 1.1296  24 666    20.2  8.88   50
```

Summary of suburbs with the highest median value of owner-occupied homes:

```
      crim                zn             indus            chas              nox              rm       
 Min.   :0.01381   Min.   : 0.00   Min.   : 0.460   Min.   :0.000   Min.   :0.4010   Min.   :4.970  
 1st Qu.:0.40920   1st Qu.: 0.00   1st Qu.: 3.647   1st Qu.:0.000   1st Qu.:0.5000   1st Qu.:6.933  
 Median :1.49119   Median : 0.00   Median :18.100   Median :0.000   Median :0.6050   Median :7.853  
 Mean   :2.70341   Mean   :19.06   Mean   :11.861   Mean   :0.375   Mean   :0.5666   Mean   :7.484  
 3rd Qu.:5.09116   3rd Qu.:20.00   3rd Qu.:18.470   3rd Qu.:1.000   3rd Qu.:0.6310   3rd Qu.:8.100  
 Max.   :9.23230   Max.   :95.00   Max.   :19.580   Max.   :1.000   Max.   :0.6680   Max.   :8.725  
      age              dis              rad              tax            ptratio          lstat       
 Min.   : 24.80   Min.   :1.130   Min.   : 1.00   Min.   :193.0   Min.   :13.00   Min.   :1.730  
 1st Qu.: 63.65   1st Qu.:1.351   1st Qu.: 4.75   1st Qu.:261.8   1st Qu.:14.62   1st Qu.:2.967  
 Median : 90.20   Median :2.043   Median : 5.00   Median :403.0   Median :14.70   Median :3.510  
 Mean   : 77.64   Mean   :2.586   Mean   :10.62   Mean   :415.4   Mean   :16.48   Mean   :4.355  
 3rd Qu.: 96.97   3rd Qu.:2.971   3rd Qu.:24.00   3rd Qu.:666.0   3rd Qu.:20.20   3rd Qu.:4.753  
 Max.   :100.00   Max.   :5.885   Max.   :24.00   Max.   :666.0   Max.   :20.20   Max.   :9.530  
      medv    
 Min.   :50  
 1st Qu.:50  
 Median :50  
 Mean   :50  
 3rd Qu.:50  
 Max.   :50  
```

Summary of the median value of owner-occupied homes in overall data:

```
> summary(Boston)
      crim                zn             indus            chas              nox              rm       
 Min.   : 0.00632   Min.   :  0.00   Min.   : 0.46   Min.   :0.00000   Min.   :0.3850   Min.   :3.561  
 1st Qu.: 0.08205   1st Qu.:  0.00   1st Qu.: 5.19   1st Qu.:0.00000   1st Qu.:0.4490   1st Qu.:5.886  
 Median : 0.25651   Median :  0.00   Median : 9.69   Median :0.00000   Median :0.5380   Median :6.208  
 Mean   : 3.61352   Mean   : 11.36   Mean   :11.14   Mean   :0.06917   Mean   :0.5547   Mean   :6.285  
 3rd Qu.: 3.67708   3rd Qu.: 12.50   3rd Qu.:18.10   3rd Qu.:0.00000   3rd Qu.:0.6240   3rd Qu.:6.623  
 Max.   :88.97620   Max.   :100.00   Max.   :27.74   Max.   :1.00000   Max.   :0.8710   Max.   :8.780  
      age              dis              rad              tax            ptratio          lstat       
 Min.   :  2.90   Min.   : 1.130   Min.   : 1.000   Min.   :187.0   Min.   :12.60   Min.   : 1.73  
 1st Qu.: 45.02   1st Qu.: 2.100   1st Qu.: 4.000   1st Qu.:279.0   1st Qu.:17.40   1st Qu.: 6.95  
 Median : 77.50   Median : 3.207   Median : 5.000   Median :330.0   Median :19.05   Median :11.36  
 Mean   : 68.57   Mean   : 3.795   Mean   : 9.549   Mean   :408.2   Mean   :18.46   Mean   :12.65  
 3rd Qu.: 94.08   3rd Qu.: 5.188   3rd Qu.:24.000   3rd Qu.:666.0   3rd Qu.:20.20   3rd Qu.:16.95  
 Max.   :100.00   Max.   :12.127   Max.   :24.000   Max.   :711.0   Max.   :22.00   Max.   :37.97  
      medv      
 Min.   : 5.00  
 1st Qu.:17.02  
 Median :21.20  
 Mean   :22.53  
 3rd Qu.:25.00  
 Max.   :50.00  
```

Largely the values are comparable with the overall ranges of the predictor's variables except for crime, chas, age, dis, and lstat.
Let's denote the suburbs with the highest median value of owner-occupied homes by X.
Please find below the comments on the ranges and findings from the data:

I. **Crime**- the mean in X lies between the 50th and 75th percentile of overall data for the variable crime. The range in X is from 0.13 to 9.23 as compared to 0.00632 and 88.97 in overall data.
II. zn- the mean of X lies between the 75th and 100th percentile of overall data for variable zn but the range of values in X looks comparable with the overall data.
III. indus- the mean of X lies around the center of overall data for the variable indus.
IV. **chas** - Around 1% of X is bound to Charles river as compared to 7% in the overall data.
V. nox - The values in X range from 0.4 to 0.66 as compared to a range between 0.38 and 0.87 in the overall data.
IV. rm - Even for this variable there is not much difference in the range of value in X and overall data.
V. rad,tax ,ptratio - For these variables the range of values in X and overall data seems comparable.
VI.**age** - The range of values in X is very different from the range in overall data. The min value in X is 24 as compared to 2.9 in overall data.
VII. **dis**- The range of X is from 1.13 to ~6. So the spread is not much as compared to the overall range between 1.13 and 12.12
VIII. **lstat**- Here the mean of X is 4.35 as compared to the mean of 12.65 in overall data. The range is also very different.

H. There is a total of 333 suburbs with more than 6 dwellings per room and 13 suburbs with more than 8 dwellings per room.

```
Boston[Boston$rm>6,]
Boston[Boston$rm>8,]
summary(Boston[Boston$rm>8,])
```

Summary of suburbs with more than 8 dwellings per room

```
      crim                zn             indus            chas              nox              rm       
 Min.   :0.02009   Min.   : 0.00   Min.   : 2.680   Min.   :0.0000   Min.   :0.4161   Min.   :8.034  
 1st Qu.:0.33147   1st Qu.: 0.00   1st Qu.: 3.970   1st Qu.:0.0000   1st Qu.:0.5040   1st Qu.:8.247  
 Median :0.52014   Median : 0.00   Median : 6.200   Median :0.0000   Median :0.5070   Median :8.297  
 Mean   :0.71879   Mean   :13.62   Mean   : 7.078   Mean   :0.1538   Mean   :0.5392   Mean   :8.349  
 3rd Qu.:0.57834   3rd Qu.:20.00   3rd Qu.: 6.200   3rd Qu.:0.0000   3rd Qu.:0.6050   3rd Qu.:8.398  
 Max.   :3.47428   Max.   :95.00   Max.   :19.580   Max.   :1.0000   Max.   :0.7180   Max.   :8.780  
      age              dis              rad              tax           ptratio          lstat      
 Min.   : 8.40   Min.   :1.801   Min.   : 2.000   Min.   :224.0   Min.   :13.00   Min.   :2.47  
 1st Qu.:70.40   1st Qu.:2.288   1st Qu.: 5.000   1st Qu.:264.0   1st Qu.:14.70   1st Qu.:3.32  
 Median :78.30   Median :2.894   Median : 7.000   Median :307.0   Median :17.40   Median :4.14  
 Mean   :71.54   Mean   :3.430   Mean   : 7.462   Mean   :325.1   Mean   :16.36   Mean   :4.31  
 3rd Qu.:86.50   3rd Qu.:3.652   3rd Qu.: 8.000   3rd Qu.:307.0   3rd Qu.:17.40   3rd Qu.:5.12  
 Max.   :93.90   Max.   :8.907   Max.   :24.000   Max.   :666.0   Max.   :20.20   Max.   :7.44  
      medv      
 Min.   :21.9  
 1st Qu.:41.7  
 Median :48.3  
 Mean   :44.2  
 3rd Qu.:50.0  
 Max.   :50.0  
```

If we compare the summary of suburbs with more than 8 dwellings per room with the overall data we find that:

I. The median ‘lstat’ in suburbs with more than 8 dwellings is 4.14 which is lower than median ‘lstat’ of 11.36 in the overall data.
II. The median ‘medv’ is more than the median ‘medv’ of the overall data.
III. The median ‘indus’ is lower than median ‘indus’ of the overall data.
IV. The median crime rate in suburbs with more than 8 dwellings is 0.52% which is also higher than the median crime rate of the overall data which is 0.25%.