HW1 STAT 435

1a. Non-parametric method looks for f hat that gets as close as possible t the data without the graph being too wiggly. The pros of the non-parametric method is that it can avoid unnecessary assumptions, making it more flexible than the parametric method. However, the cons of it is that it needs a lot of data to obtain an accurate estimate for f. The non-parametric approach might also introduce a risk of over-fitting

For the parametric approach, we need to make an assumption about the form of f, then we use the data points to fit the model. As a result the assumption simplifies the problem of estimating f because it will be much easier to estimate the set of parameters in the linear model than fitting an entirely random function f. However, one of the cons is that the model we choose might not match the true unknown form of f, and if the chosen model is too far from the true f, then our estimate will not be accurate. This means that the parametric approach is less flexible.

1b. We can use the parametric approach if we have a smaller sample size. With the parametric approach, we want to see if the assumed model fits the data.

1c. You might want a large dataset to do the non-parametric approach because we really need a lot of data/observations to get an accurate estimate for f.

2a. If the sample size n is very small, and the number of predictors p is very large, we can use the inflexible statistical machine learning model because if we use a flexible model, it has a huge risk of overfitting the data because of the small sample size n.

2b. If the sample size n is very large, and the number of predictors p is very small, we can use a more flexible statistical machine learning because it can fit the data closer without the risk of overfitting as a result of the very large number of observations.

2c. When the relationship between the predictors and response is highly non-linear, we can use a more flexible statistical machine learning method since it gives more flexibility to fit better.

2d. When the variance of the error terms is extremely high, it is better to use an inflexible statistical machine learning because if we use a flexible statistical machine learning, it will try to fit the error term which we do not want.

3a. This is a regression problem and our goal is prediction because we want to predict each student's final exam score. The sample size n is 50, and the number of predictors p is 8.

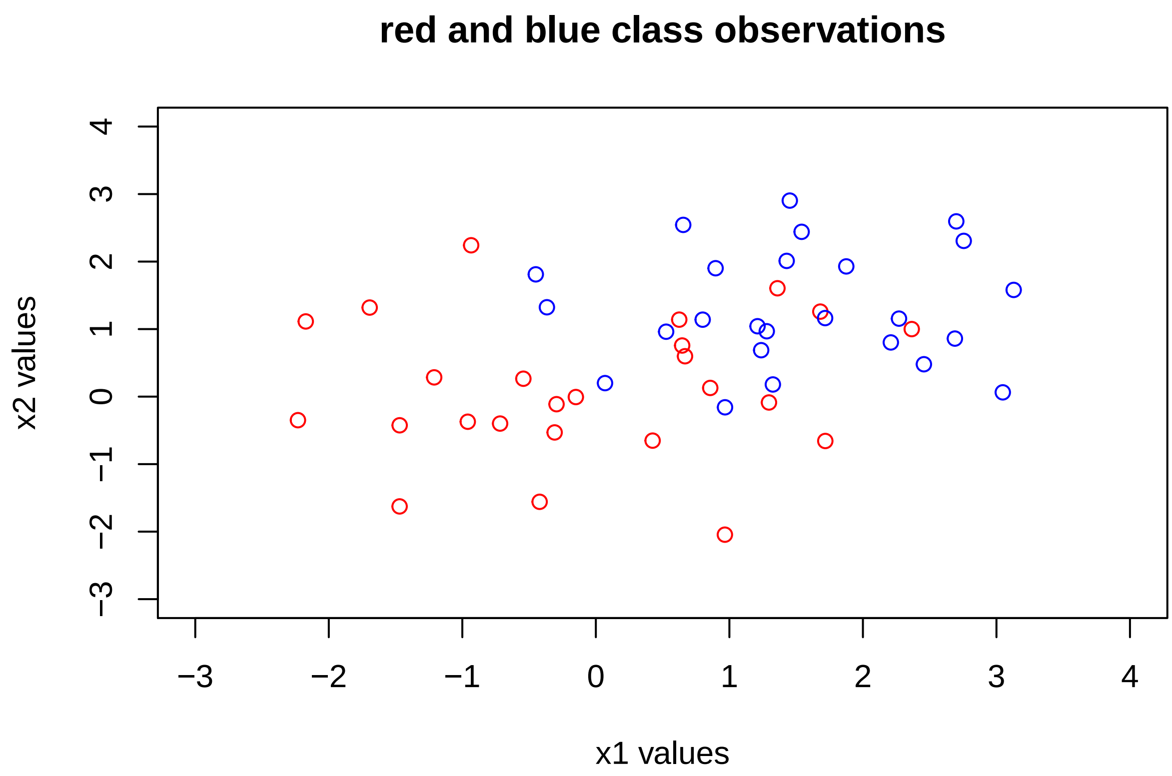
3b. This is a classification problem and our goal is inference because we want to understand the relationship between the different factors and whether or not a student passes the course. The sample size n is 50, and the number of predictors p is 6.

4c. An f hat that has a high variance but no bias is when we connect all the data points together by a single line. It has no bias because it perfectly follows the data points, but has a high variance because the value will differ so much.

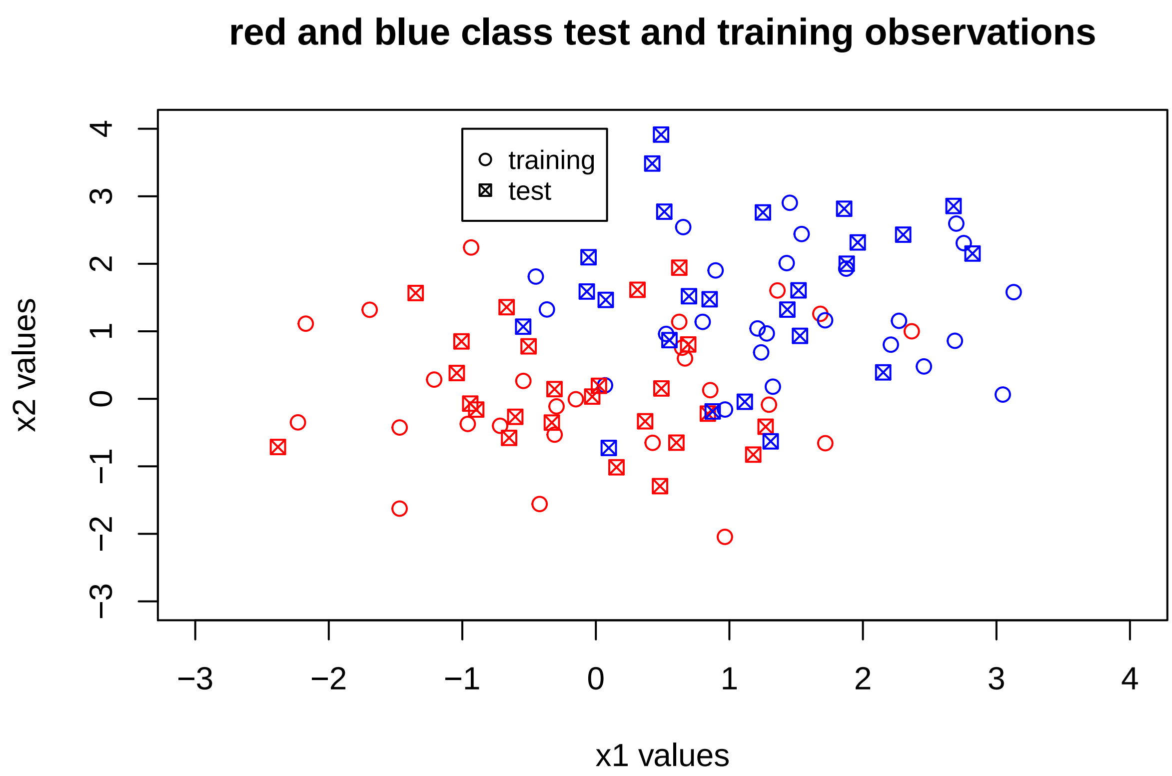
4d. An f hat that has no variance and a high bias is when we have a constant. This means that the value will remain the same, meaning that there is no variance. However, this will result in a high bias because it will not be close to our actual f.

```r
# part 5a
training <- data.frame(x = c(rnorm(n = 25, mean = 0, sd = 1),
                             rnorm(n = 25, mean = 1.5, sd = 1)),
```

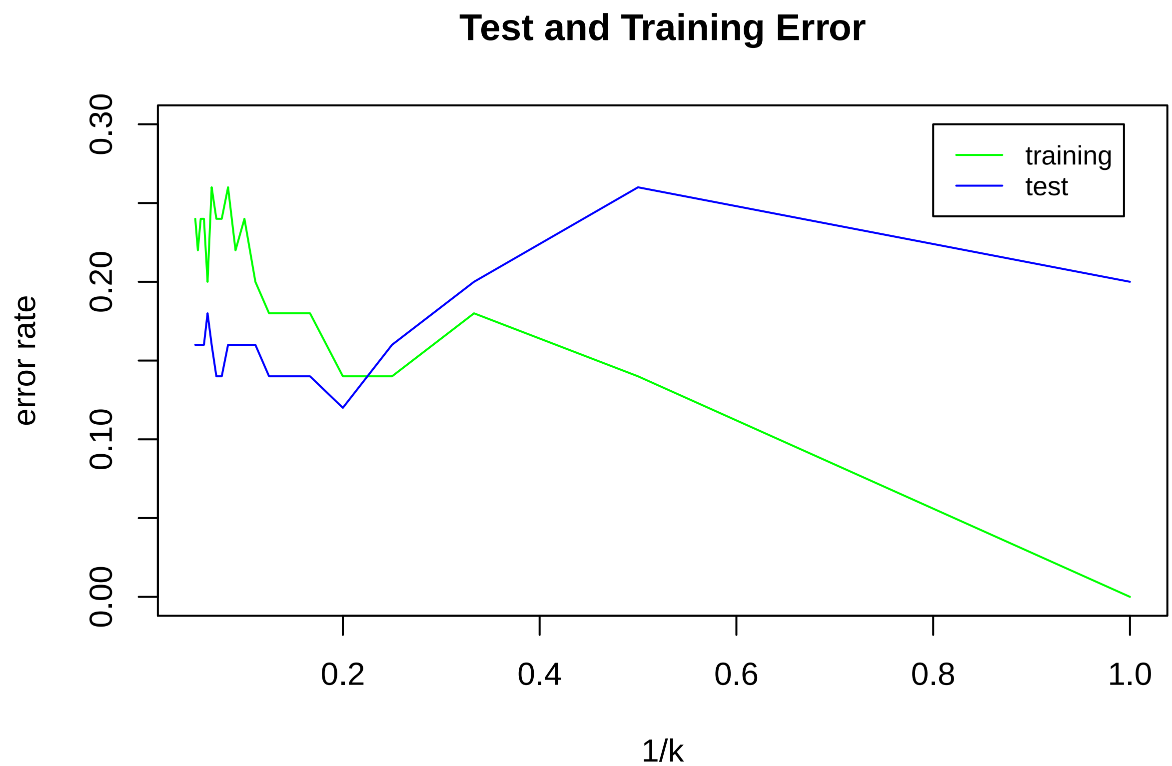
```
                         y = c(rnorm(n = 25, mean = 0, sd = 1),
                               rnorm(n = 25, mean = 1.5, sd = 1)))
plot(training[c(1:25),], col='red', xlim=c(-3,4), ylim=c(-3,4), xlab = "x1 values",
     ylab = "x2 values", main="red and blue class observations")
points(training[c(26:50),], col='blue')
```



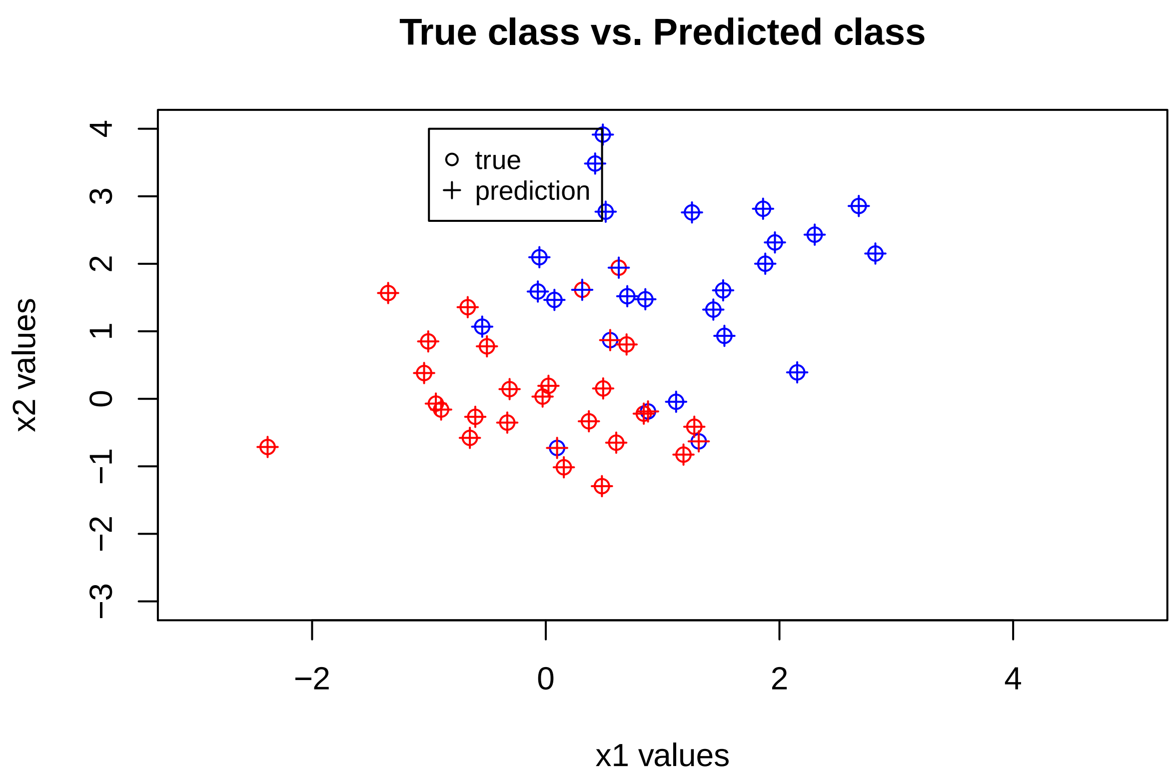

```
# part 5b
test <- data.frame(x = c(rnorm(n = 25, mean = 0, sd = 1),
                         rnorm(n = 25, mean = 1.5, sd = 1)),
                   y = c(rnorm(n = 25, mean = 0, sd = 1),
                         rnorm(n = 25, mean = 1.5, sd = 1)))
plot(training[c(1:25),], col='red', xlim=c(-3,4), ylim=c(-3,4), xlab = "x1 values",
     ylab = "x2 values", main="red and blue class test and training observations")
points(training[c(26:50),], col='blue')
points(test[c(1:25),], col='red', pch=7)
points(test[c(26:50),], col='blue', pch=7)
legend(-1, 4, legend=c("training", "test"), pch=c(1,7), cex=0.8)
```




```
# part 5c
cl <- c(rep("red",25),rep("blue",25))
error <- data.frame(k = rep(0, 20),
                    tr = rep(0, 20),
                    te = rep(0, 20))
knn <- for (i in 1:20) {
          training_val <- knn(training, training, cl=cl, k=i)
          training_error <- mean(training_val != cl)

          test_val <- knn(training, test, cl=cl, k=i)
          test_error <- mean(test_val != cl)

          error[i, 'k'] <- i
          error[i, 'tr'] <- training_error
          error[i, 'te'] <- test_error
}
plot(1/error$k, error$tr, type = "l", col = "green", xlab = "1/k",
     ylab = "error rate", ylim=c(0,0.3),
     main="Test and Training Error")
lines(1/error$k, col = "blue", error$te)
legend(0.8, 0.3, legend=c("training", "test"), lty = 1,
       col=c("green","blue"), cex=0.8)
```




```
# part 5d
min_k <- error$k[error$te == min(error$te)]
predictor_value <- knn(training, test, cl=cl, k=min_k[1])

plot(test[c(1:25),], col='red', ylim=c(-3,4), xlab = "x1 values",
     ylab = "x2 values", xlim=c(-3,5), main="True class vs. Predicted class")
points(test[c(26:50),], col='blue')
points(test, col=ifelse(predictor_value == "red", "red", "blue"), pch=3)
legend(-1, 4, legend=c("true", "prediction"), pch=c(1,3), cex=0.8)
```




```r
# part 5e
n <- 10000
bayes_err <- c(rep(0, 2*n))

red_df <- data.frame(x1 = rnorm(n=n), x2 = rnorm(n=n))
blue_df <- data.frame(x1 = rnorm(n=n, mean = 1.5), x2 = rnorm(n=n, mean = 1.5))

for(i in 1:n) {
  prob_red <- dnorm(red_df$x1[i]) * dnorm(red_df$x2[i])
  prob_blue <- dnorm(red_df$x1[i], mean=1.5) * dnorm(red_df$x2[i], mean=1.5)
  bayes_err[i] <- max(prob_red, prob_blue)/(prob_red+prob_blue)
}

for(i in 1:n) {
  prob_red <- dnorm(blue_df$x1[i]) * dnorm(blue_df$x2[i])
  prob_blue <- dnorm(blue_df$x1[i], mean=1.5) * dnorm(blue_df$x2[i], mean=1.5)
  bayes_err[i+n] <- max(prob_red, prob_blue)/(prob_red+prob_blue)
}

bayes_error_rate <- 1 - mean(bayes_err)
bayes_error_rate
```

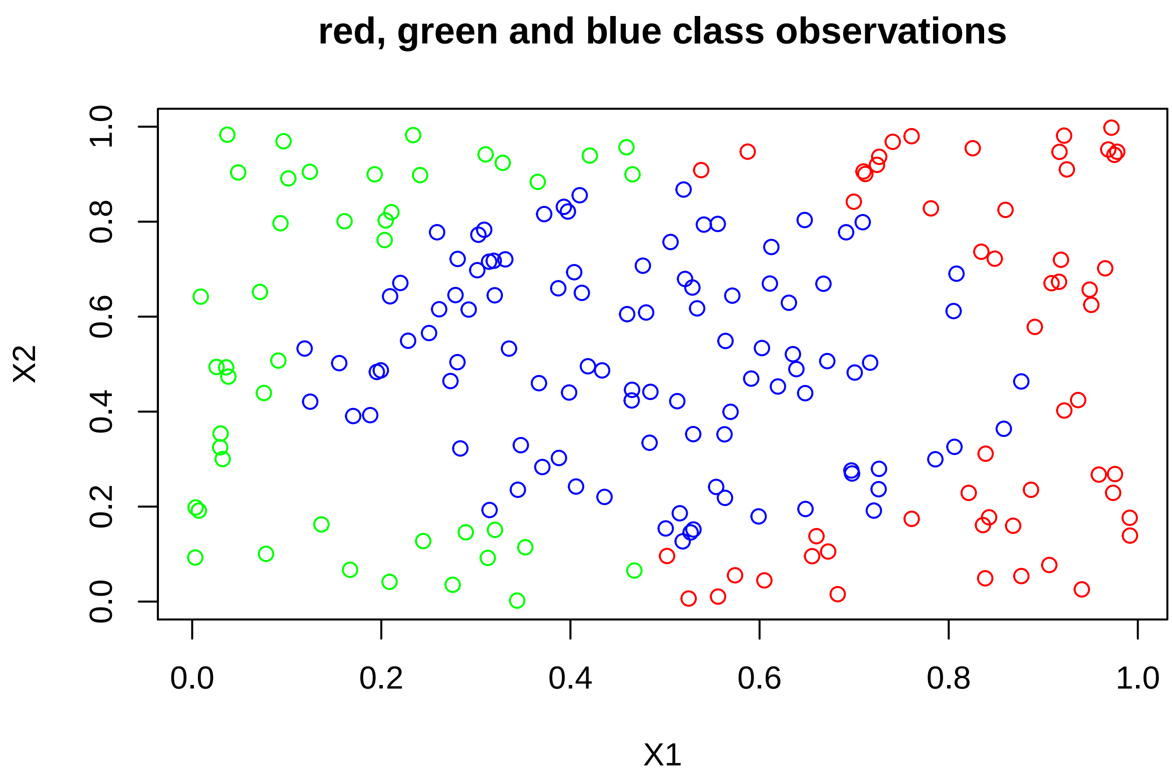
```
## [1] 0.1422521
```

5c. The graph shows that as 1/k increases, the method becomes more flexible. As we can see, the training error rate decreases as the flexibility increases (1/k increases). On the other hand, the testing error has a

U-shape characteristic, meaning that it will decline first before it increases again when the method becomes more flexible.

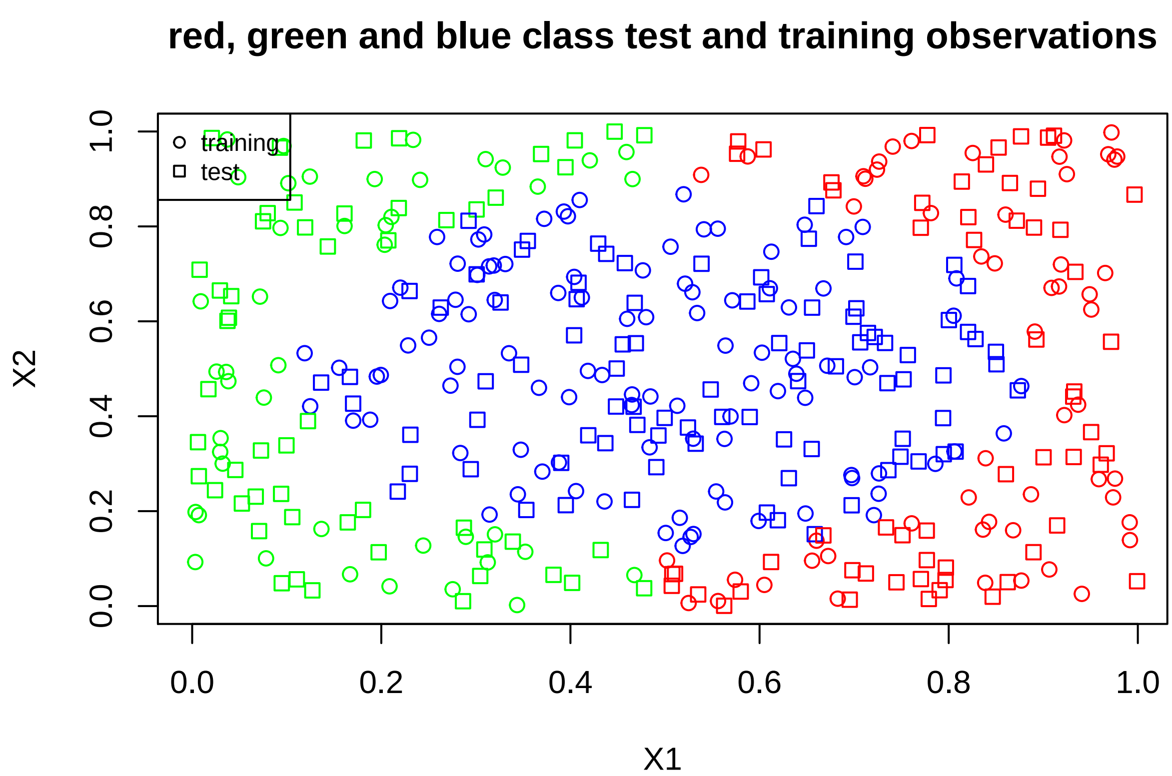
5e. The value of the bayes error rate is 0.1432. This means that 0.1432 is the lowest possible test error rate in classification which is produced by the Bayes classifier.

```
# part 6a
x1 <- runif(n=200, min = 0, max = 1)
x2 <- runif(n=200, min = 0, max = 1)

train <- data.frame(x1 = x1, x2 = x2, color = 0)
train$color <- ifelse((train$x1-0.5)^2 + (train$x2-0.5)^2 > 0.15,
                      ifelse(train$x1 > 0.5, "red", "green"), "blue")

plot(train$x1, train$x2, xlim = range(train$x1), ylim = range(train$x2),
     main="red, green and blue class observations",
     xlab = "X1", ylab ="X2",
     col = ifelse(train$color=="red", "red",
                  ifelse(train$color=="green", "green", "blue")))
```




```
# part 6b
test1 <- data.frame(x1 = runif(n=200, min = 0, max = 1),
                    x2 = runif(n=200, min = 0, max = 1),
                    color = 0)
test1$color <- ifelse((test1$x1-0.5)^2 + (test1$x2-0.5)^2 > 0.15,
                      ifelse(test1$x1 > 0.5, "red", "green"), "blue")
```

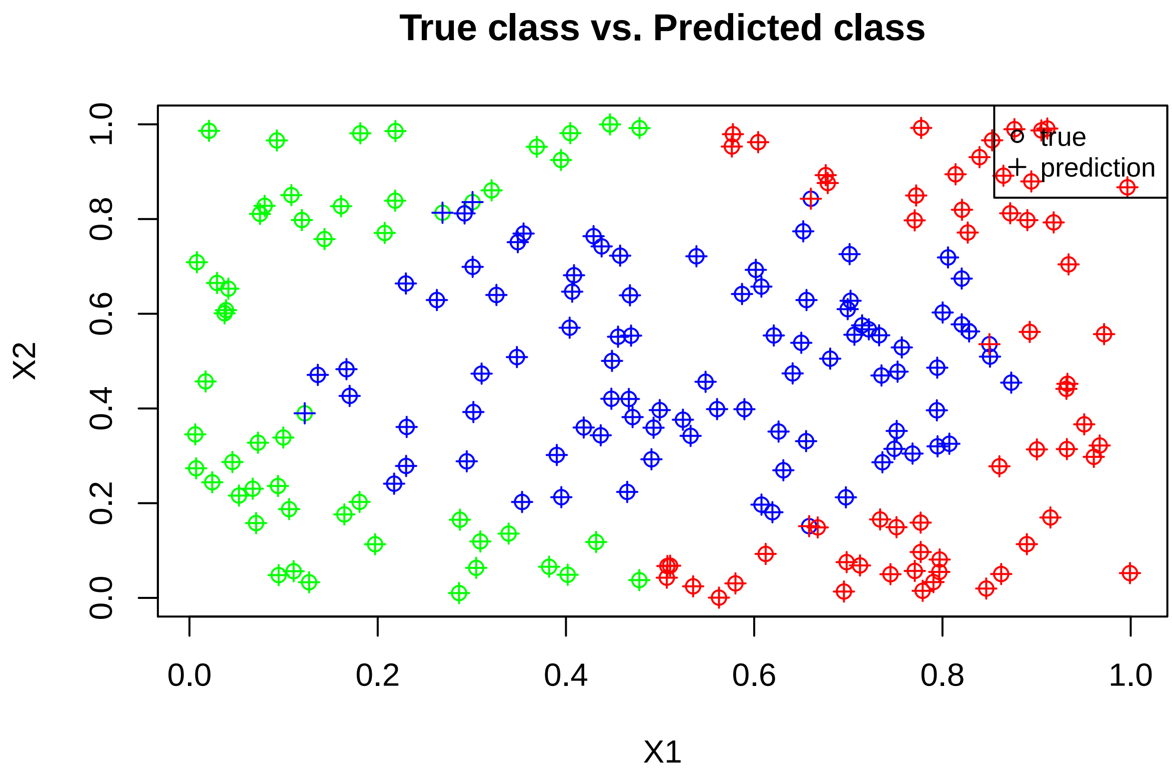
```r
plot(train$x1, train$x2, xlim = range(train$x1), ylim = range(train$x2),
     main="red, green and blue class test and training observations",
     xlab = "X1", ylab ="X2",
     col = ifelse(train$color=="red", "red",
                  ifelse(train$color=="green", "green", "blue")))
points(test1$x1, test1$x2,
       col = ifelse(test1$color=="red", "red",
                    ifelse(test1$color=="green", "green", "blue")),
       pch = 0)
legend("topleft", c("training", "test"), pch = c(1, 0), cex=.75)
```




```r
# part 6c
error1 <- data.frame(k = rep(0, 50),
                     tr = rep(0, 50),
                     te = rep(0, 50))
tr_select <- select(train, x1, x2)
te_select <- select(test1, x1, x2)

knn <- for (i in 1:50) {
          training_val1 <- knn(tr_select, tr_select, train$color, k=i)
          training_error1 <- mean(training_val1 != train$color)

          test_val1 <- knn(tr_select, te_select, train$color, k=i)
          test_error1 <- mean(test_val1 != test1$color)
```

```
        error1[i, 'k'] <- i
        error1[i, 'tr'] <- training_error1
        error1[i, 'te'] <- test_error1
}
plot(1/error1$k, error1$tr, xlim = c(0,1), ylim = range(error1$tr, error1$te),
     main="Test and Training Error",
xlab="1/k", ylab="error rate",
type="l")
lines(1/error1$k, error1$te, col="green")
legend("topright", legend=c("training", "test"), lty = 1,
       col=c("black","green"), cex=0.8)
```


```
# part 6d
min_k1 <- error1$k[error1$te == min(error1$te)]
predictor_value1 <- knn(tr_select, te_select, train$color, k=min_k1[1])

plot(test1$x1, test1$x2, xlim = range(test1$x1), ylim = range(test1$x2),
     xlab = "X1", ylab ="X2", main="True class vs. Predicted class",
     col = ifelse(test1$color=="red", "red",
                  ifelse(test1$color=="green", "green", "blue")))
points(test1$x1, test1$x2, col = ifelse(predictor_value1=="red", "red",
      ifelse(predictor_value1=="green", "green", "blue")), pch = 3)
legend("topright", c("true", "prediction"), pch=c(1,3), cex=0.8)
```



6c. We can see from the graph that the error rate decreases as the value of 1/k increases, making the method more flexible. As the method becomes more flexible, the error rate decreases.

6e. In this case, the Bayes error rate is 0 since the Bayes error rate is the same thing as the irreducible error. In this case, there is no overlap between the 3 classes since this dataset follows a function. As we can see from part c and d, the error rate did not reach the bayes error rate, and that the data points that are not near the boundary line has been accurately predicted.

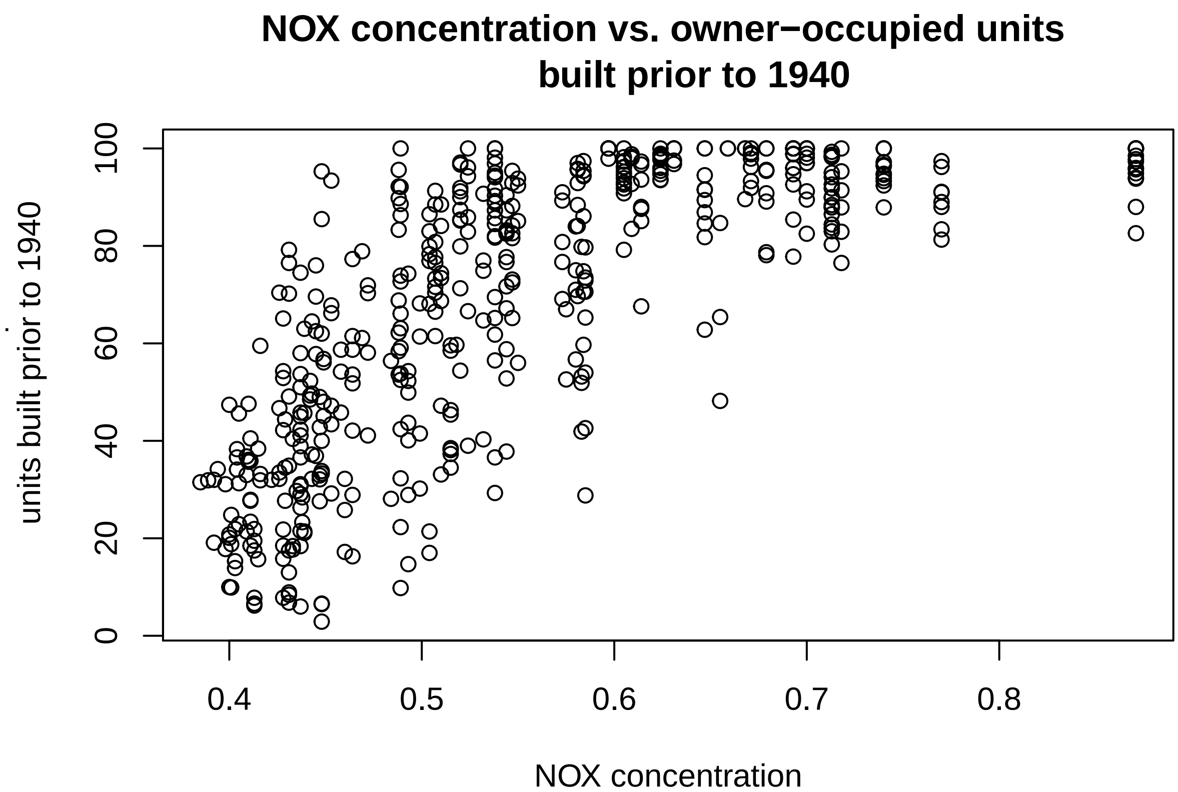
```
# part 7a
nrow(Boston)
```

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

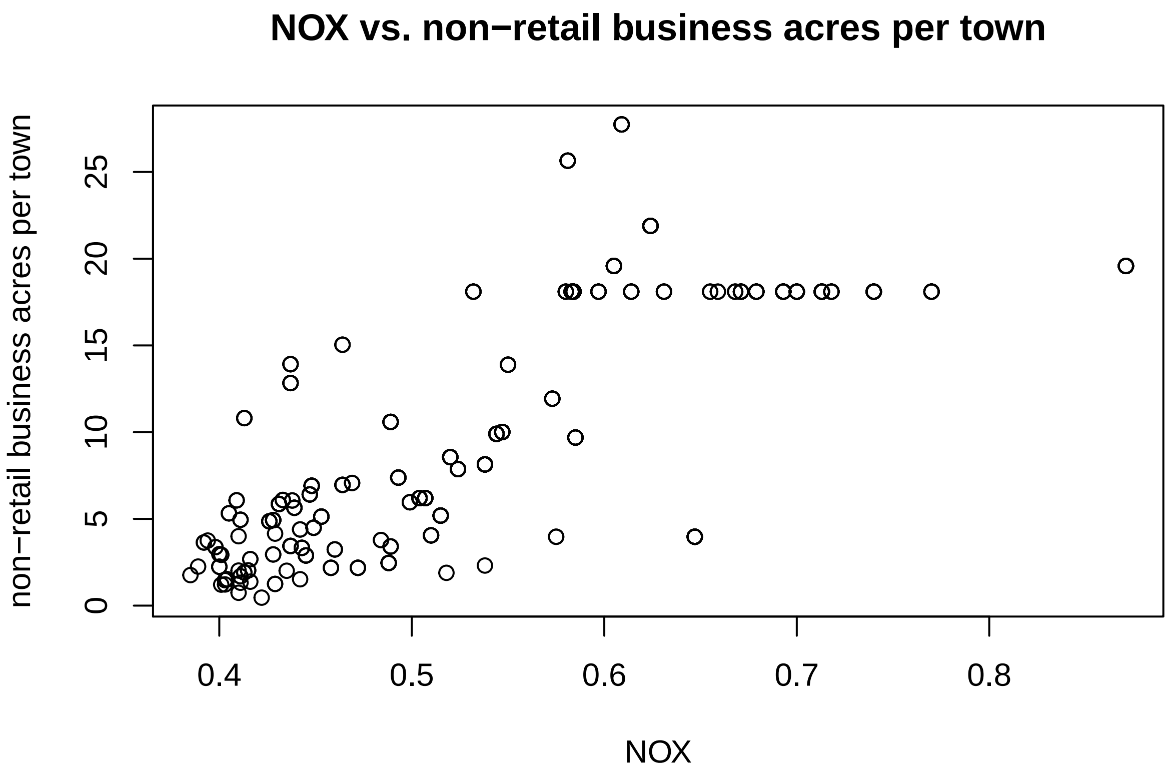
```
ncol(Boston)
```

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

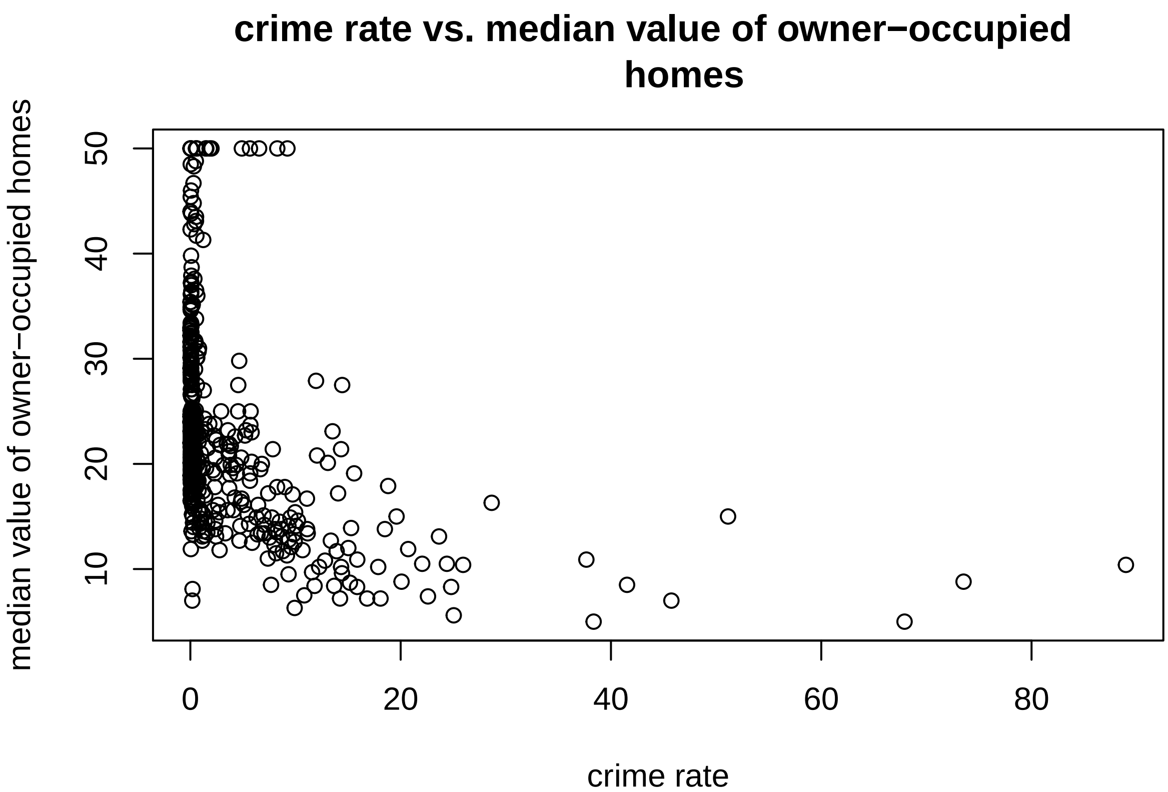
```
# part 7b
plot(Boston$nox, Boston$age, main="NOX concentration vs. owner-occupied units
     built prior to 1940", xlab = "NOX concentration", ylab = "owner-occupied
     units built prior to 1940")
```



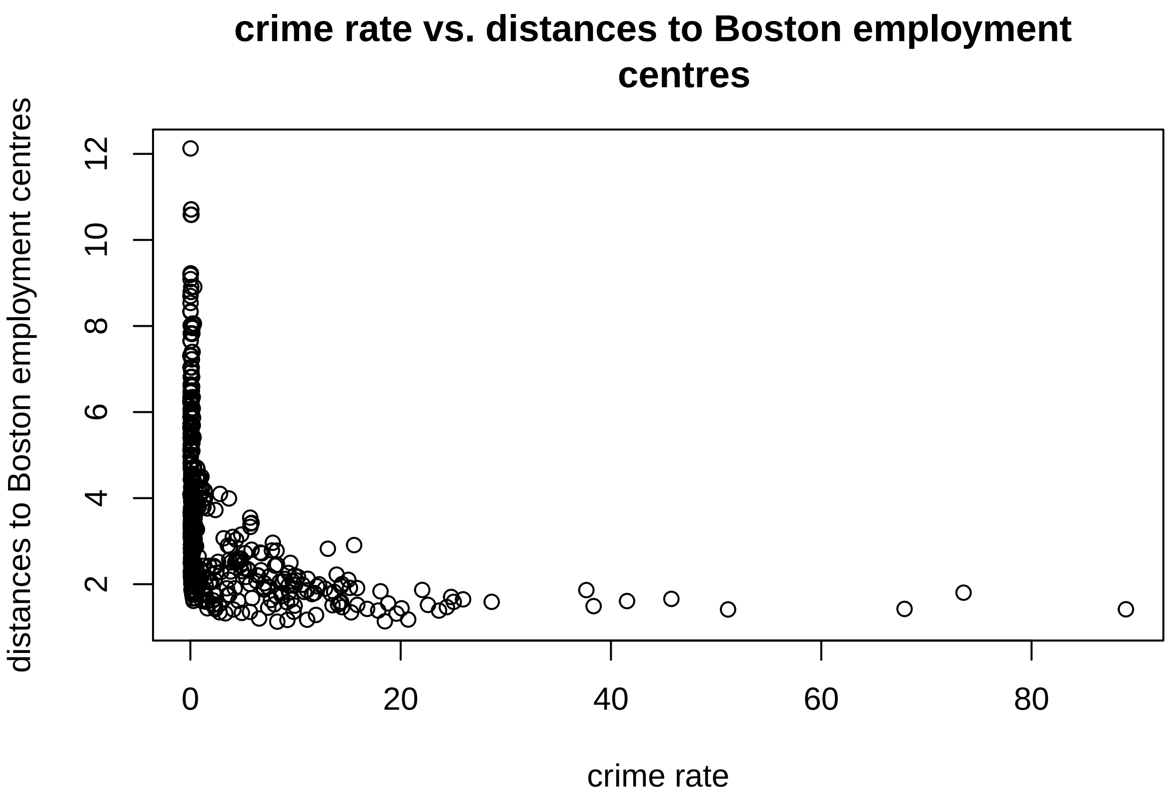

```r
plot(Boston$nox, Boston$indus, main="NOX vs. non-retail business acres per town"
     , xlab = "NOX", ylab = "non-retail business acres per town")
```




```
plot(Boston$crim, Boston$medv, main="crime rate vs. median value of owner-occupied
     homes", xlab = "crime rate", ylab = "median value of owner-occupied homes")
```




```r
plot(Boston$crim, Boston$dis, main="crime rate vs. distances to Boston employment
     centres", xlab = "crime rate", ylab = "distances to Boston employment centres")
```




```r
# part 7d
summary(Boston$crim)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##  0.00632  0.08204  0.25651  3.61352  3.67708 88.97620
```

```r
summary(Boston$tax)
```

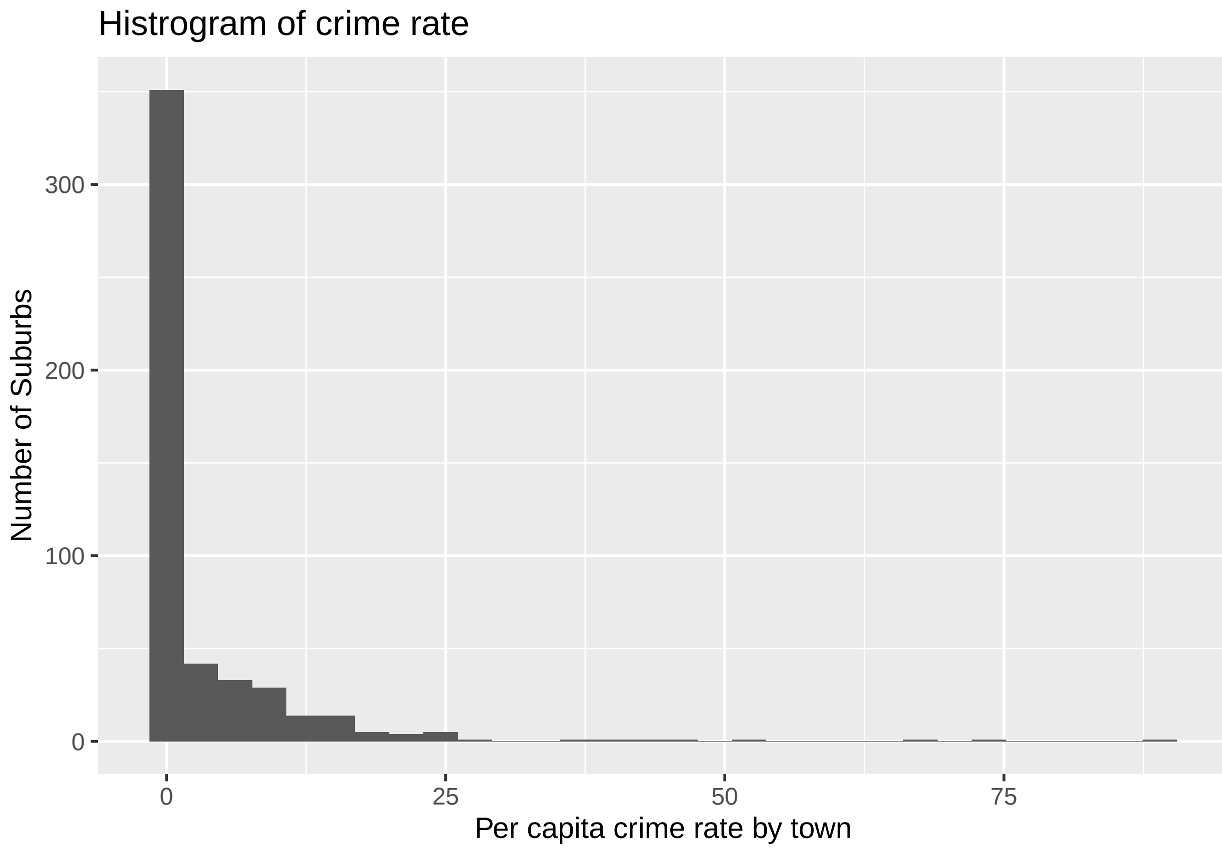
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   187.0   279.0   330.0   408.2   666.0   711.0
```

```r
summary(Boston$ptratio)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   12.60   17.40   19.05   18.46   20.20   22.00
```

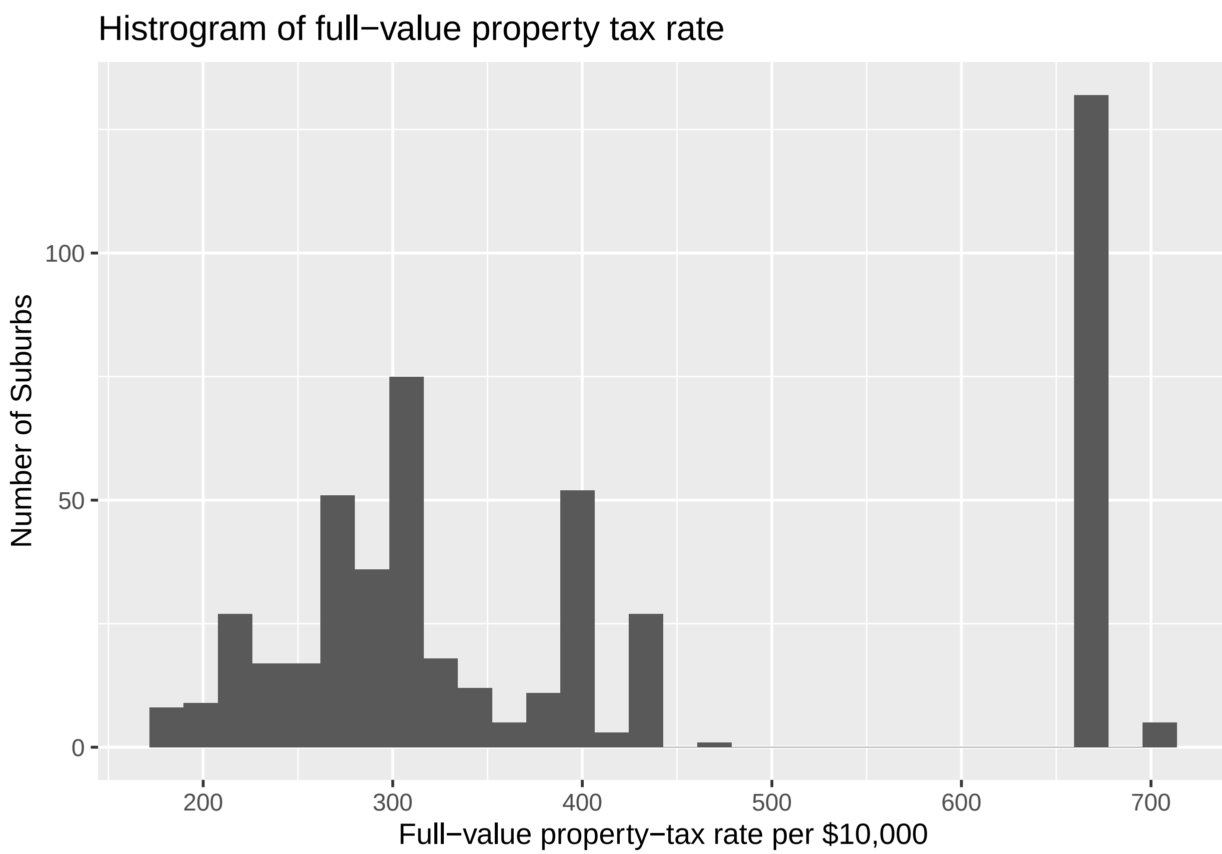
```r
qplot(Boston$crim, xlab = "Per capita crime rate by town", ylab="Number of Suburbs",
      main = "Histrogram of crime rate")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



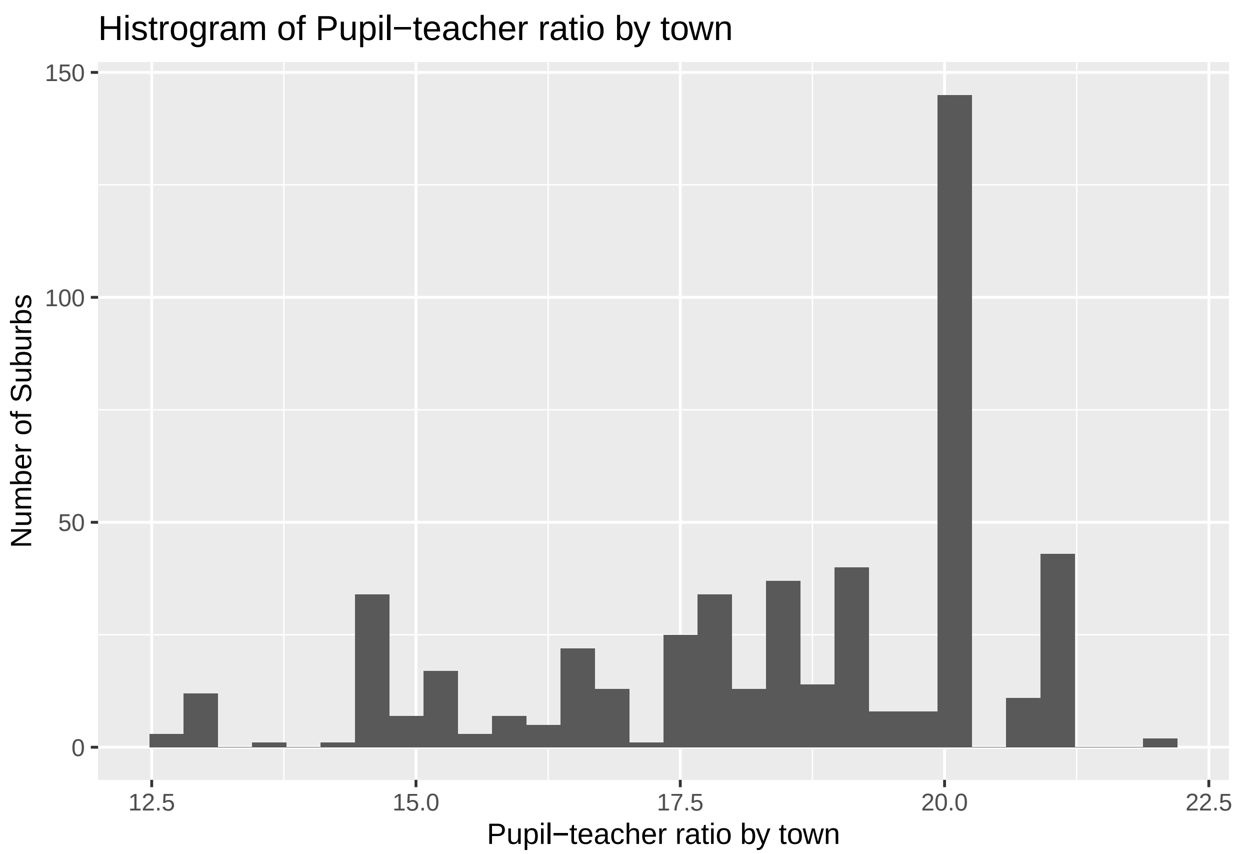

```
qplot(Boston$tax, xlab = "Full-value property-tax rate per $10,000",
      ylab="Number of Suburbs"
      , main = "Histrogram of full-value property tax rate")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```r
qplot(Boston$ptratio, xlab ="Pupil-teacher ratio by town", ylab="Number of Suburbs",
      main = "Histrogram of Pupil-teacher ratio by town")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```r
# part 7e
nrow(filter(Boston, chas ==1))
```

```
## [1] 35
```

```r
# part 7f
mean(Boston$ptratio)
```

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

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

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

```r
# part 7g
medv_max <- filter(Boston, Boston$medv == max(Boston$medv))
summary(Boston)
```

```
##       crim                zn             indus            chas        
##  Min.   : 0.00632   Min.   :  0.00   Min.   : 0.46   Min.   :0.00000  
##  1st Qu.: 0.08205   1st Qu.:  0.00   1st Qu.: 5.19   1st Qu.:0.00000  
##  Median : 0.25651   Median :  0.00   Median : 9.69   Median :0.00000  
##  Mean   : 3.61352   Mean   : 11.36   Mean   :11.14   Mean   :0.06917  
```

```
##  3rd Qu.: 3.67708   3rd Qu.: 12.50   3rd Qu.:18.10   3rd Qu.:0.00000
##  Max.   :88.97620   Max.   :100.00   Max.   :27.74   Max.   :1.00000
##       nox               rm             age              dis
##  Min.   :0.3850   Min.   :3.561   Min.   :  2.90   Min.   : 1.130
##  1st Qu.:0.4490   1st Qu.:5.886   1st Qu.: 45.02   1st Qu.: 2.100
##  Median :0.5380   Median :6.208   Median : 77.50   Median : 3.207
##  Mean   :0.5547   Mean   :6.285   Mean   : 68.57   Mean   : 3.795
##  3rd Qu.:0.6240   3rd Qu.:6.623   3rd Qu.: 94.08   3rd Qu.: 5.188
##  Max.   :0.8710   Max.   :8.780   Max.   :100.00   Max.   :12.127
##       rad              tax           ptratio          lstat
##  Min.   : 1.000   Min.   :187.0   Min.   :12.60   Min.   : 1.73
##  1st Qu.: 4.000   1st Qu.:279.0   1st Qu.:17.40   1st Qu.: 6.95
##  Median : 5.000   Median :330.0   Median :19.05   Median :11.36
##  Mean   : 9.549   Mean   :408.2   Mean   :18.46   Mean   :12.65
##  3rd Qu.:24.000   3rd Qu.:666.0   3rd Qu.:20.20   3rd Qu.:16.95
##  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
```

```r
# part 7h
rm_over6 <- filter(Boston, Boston$rm>6)
nrow(rm_over6)
```

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

```r
rm_over8 <- filter(Boston, Boston$rm>8)
nrow(rm_over8)
```

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

7a. There are a total of 506 rows and 13 columns in the Boston data set.

7b. I found out that the nitrogen oxides concentration is correlated with the proportion of owner-occupied units built prior to 1940. By looking at the scatter plot, we can see that as the nitrogen oxides concentration increases, the proportion of owner-occupied units built prior to 1940 also increases. Furthermore, I also think that the nitrogen oxides concentration is also correlated with the median value of owner-occupied homes in $1000s. These 2 predictors have a positive correlation because as the nitrogen oxides concentration increases, the median value of owner-occupied homes in $1000s also increases. There is also a correlation between the crime rate and the other predictors, and I will elaborate it further in part c.

7c. According to the scatterplot I designed in part b, there are 2 predictors that are associated with the per capita crime rate, and they are the median value of owner-occupied homes in $1000s and the weighted mean of distances to five Boston employment centers. We can see from the scatterplot, as the per capita crime rate increases, the median value of owner-occupied homes decreases. We can also see that as the per capita crime rate increases, the weighted mean of distances to five Boston employment centers decreases. These mean that the per capita crime rate and the other 2 predictors has a negative correlation.

7d. According to the histogram for the crime rate, it seems like there is a particularly high crime rate because we can see that there are several suburbs that has a very high crime rate, far from the mean and median. In addition, there is also a high full-value property-tax rate per $10,000 because as we can see from the histogram, there are a lot of suburbs that has a full-value property-tax rate above 600 dollars while the mean is only 408.2 dollars. Last but not least, it also seems like there is a particularly high pupil-teacher ratio because from the histogram, there are suburbs that has a pupil-teacher ratio of bigger than 20 while the mean is only 18.46.

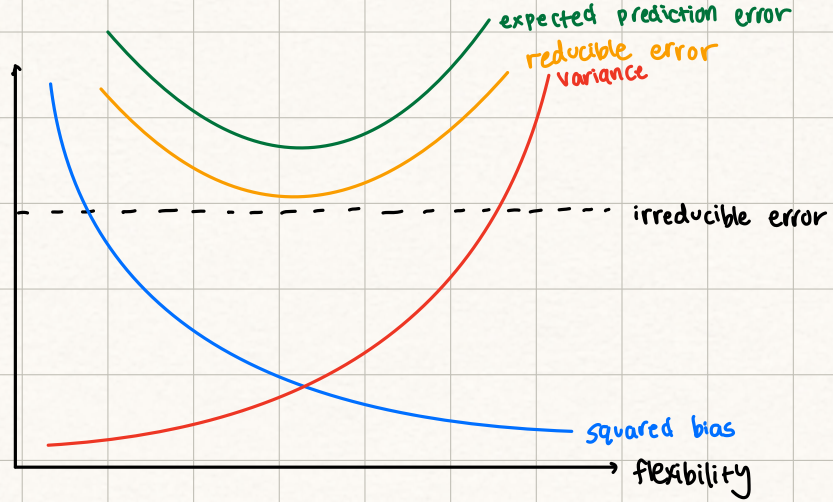
7e. There are 35 suburbs bound the Charles river.

7f. The mean of the pupil-teacher ratio among the towns is 18.456, and the standard deviation is 2.165

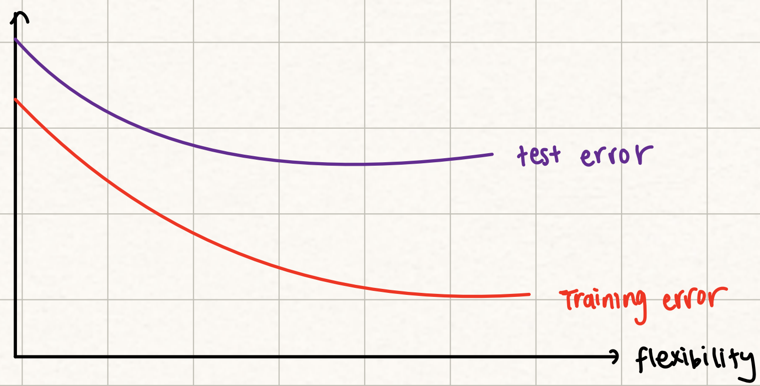
7g. There is a total of 16 suburbs that has the highest median value of owner-occupied homes. Let us focus on suburb #16. Compared to the mean of the crime rates, the crime rate of this suburb is really high as it is 8.26725 while the mean is only 3.61352. We can also see that, the proportion of non-retail business acres per town is 18.10, which is also above the mean. The nitrogen oxide concentration for this suburbs is also particularly high because it is 0.6680, which is above the mean. The proportion of owner-occupied units is 89.6, which is also considered a high value. However, the weighted mean of distances to five Boston employment centres is 1.1296, which is considered very low as it is far below the mean. The index of accessibility to radial highways is 1.1296, which is very high in comparison to the mean. Furthermore, the full-value property-tax rate and the pupil-teacher ratio by town also has a high value as they are all above the mean and median.

7h. There are 333 suburbs with average rooms per dwelling greater than 6, and 13 suburbs with average rooms per dwelling greater than 8.

4a.


expected prediction error
reducible error
variance
irreducible error
squared bias
flexibility

4b.


test error
Training error
flexibility