Homework 1 Key (total 40 pts)

Hannah Director

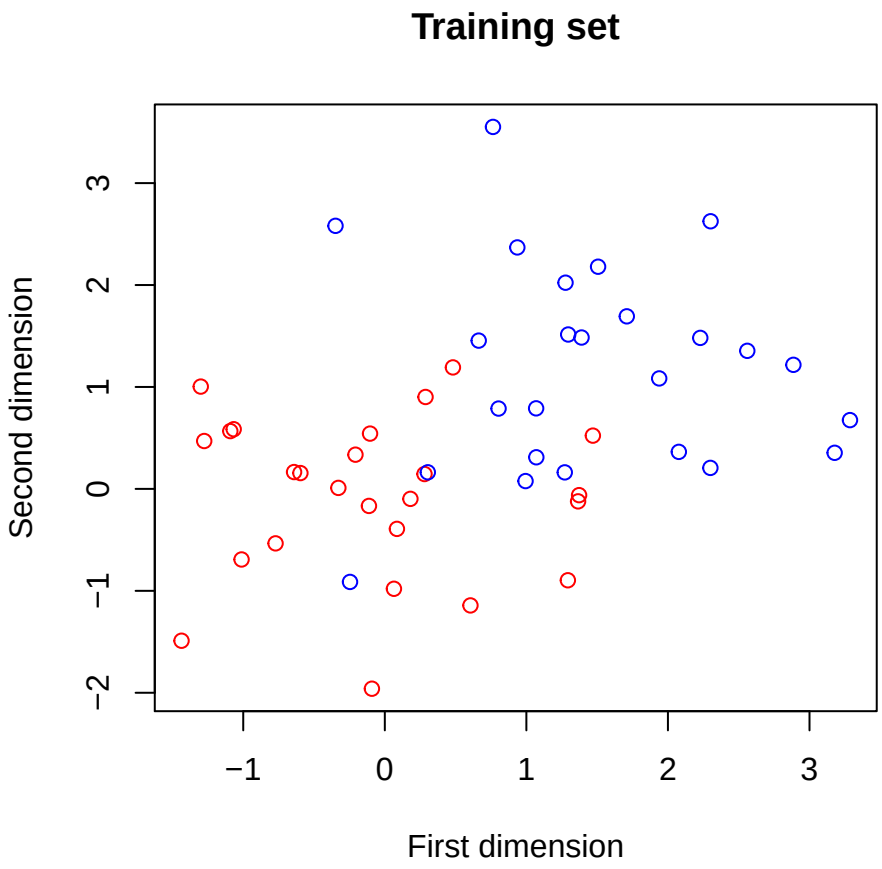
4/6/2018

For all the answers in this key, I kept the codes close to the problem so that it is easier for you to read. In homework submissions, you do not have to do that. In fact, it is usually visually better to leave most of the codes in the appendix at the end.

Problem 1 ($2+1+2+1+2=8$ pt)

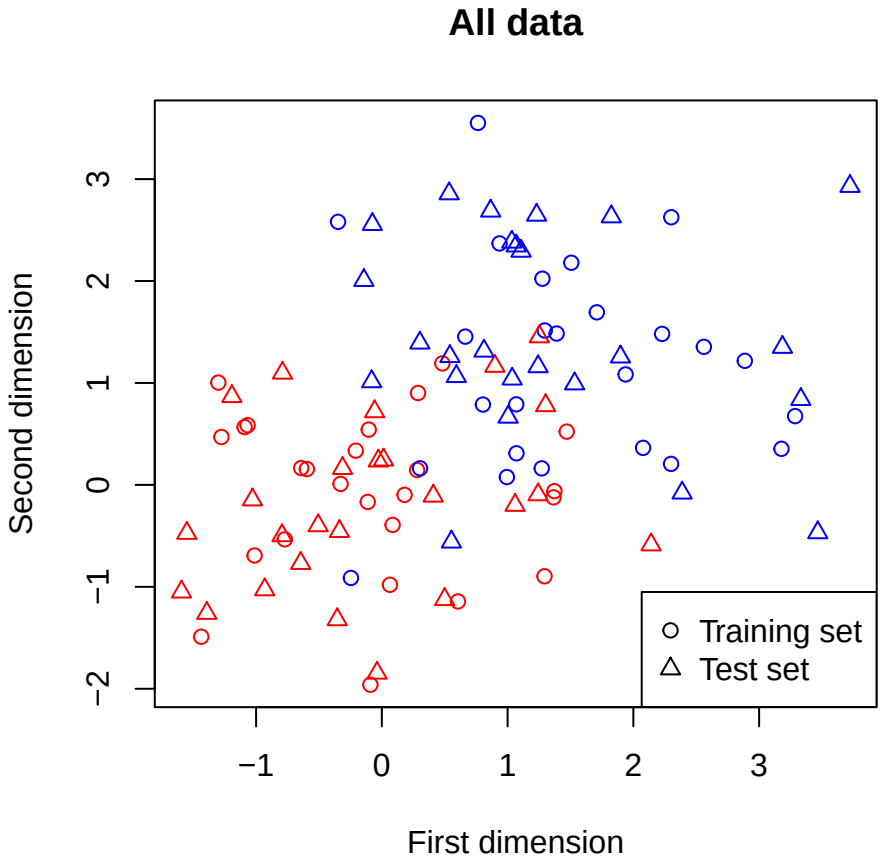
a. Notice there are no correlations between the two dimensions within each class, which means you can generate two dimensions separately.

```
set.seed(042017)
n <- 25 * 2
train <- matrix(NA, n, 2)
class <- rep("", n)
# red class
train[1:(n/2), 1] <- rnorm(n/2, 0, 1)
train[1:(n/2), 2] <- rnorm(n/2, 0, 1)
class[1:(n/2)] <- "red"

# blue class
train[(n/2+1):n, 1] <- rnorm(n/2, 1.5, 1)
train[(n/2+1):n, 2] <- rnorm(n/2, 1.5, 1)
class[(n/2+1):n] <- "blue"

plot(train[, 1], train[, 2], col = class,
     xlab = "First dimension", ylab = "Second dimension", main = "Training set")
```



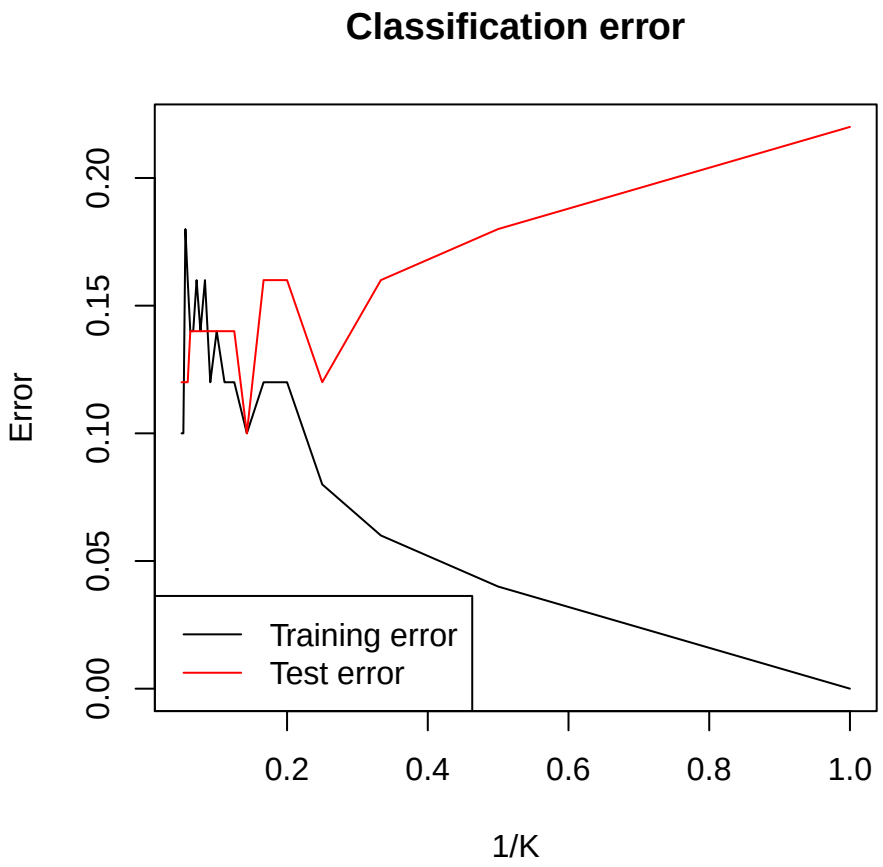
b. The plot is as follows.

```
test <- matrix(NA, n, 2)
class.test <- rep("", n)
# red class
test[1:(n/2), 1] <- rnorm(n/2, 0, 1)
test[1:(n/2), 2] <- rnorm(n/2, 0, 1)
class.test[1:(n/2)] <- "red"

# blue class
test[(n/2+1):n, 1] <- rnorm(n/2, 1.5, 1)
test[(n/2+1):n, 2] <- rnorm(n/2, 1.5, 1)
class.test[(n/2+1):n] <- "blue"

plot(train[, 1], train[, 2], col = class,
     xlab = "First dimension", ylab = "Second dimension", main = "All data",
     xlim = range(c(train[,1], test[,1])), ylim = range(c(train[,2], test[,2])))
points(test[, 1], test[, 2], col = class.test, pch = 2)
legend("bottomright", c("Training set", "Test set"), pch = c(1, 2))
```



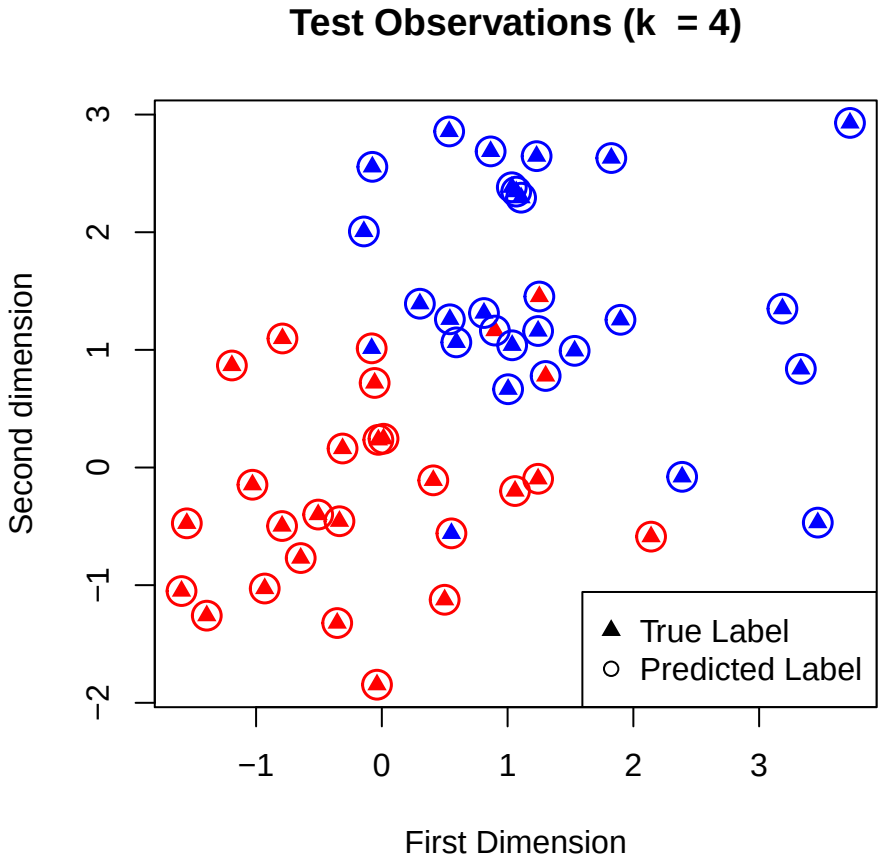
c. As we can see from the plot, as K decreases (i.e., $1/K$ increases), the training error goes down to 0, while the testing error starts to go up. Remember from the lecture we know that as K decreases, the model becomes more flexible. In the extreme case where K goes down to 1, training error can be eliminated. However, this tends to be overfitting for testing data and that is why we can observe the testing error going up on the right hand side.

```
library(class)
K <- 20
error <- matrix(NA, K, 2)
for(k in 1:K){
    knn1 <- knn(train=train, test=train, cl=class, k=k)
    error[k, 1] <- sum(knn1 != class) / n
    knn2 <- knn(train=train, test=test, cl=class, k=k)
    error[k, 2] <- sum(knn2 != class.test) / n
}
plot(1/seq(1, K), error[, 1], ylim = range(error), type = "l",
     xlab = "1/K", ylab = "Error", main = "Classification error")
lines(1/seq(1, K), error[, 2], col = "red")
legend("bottomleft", c("Training error", "Test error"),
        col = c("black", "red"), lty = c(1, 1))
```



d. In my simulation, $K = 4$ resulted in the smallest test error. The plot of the points and their true and predicted labels are given below.

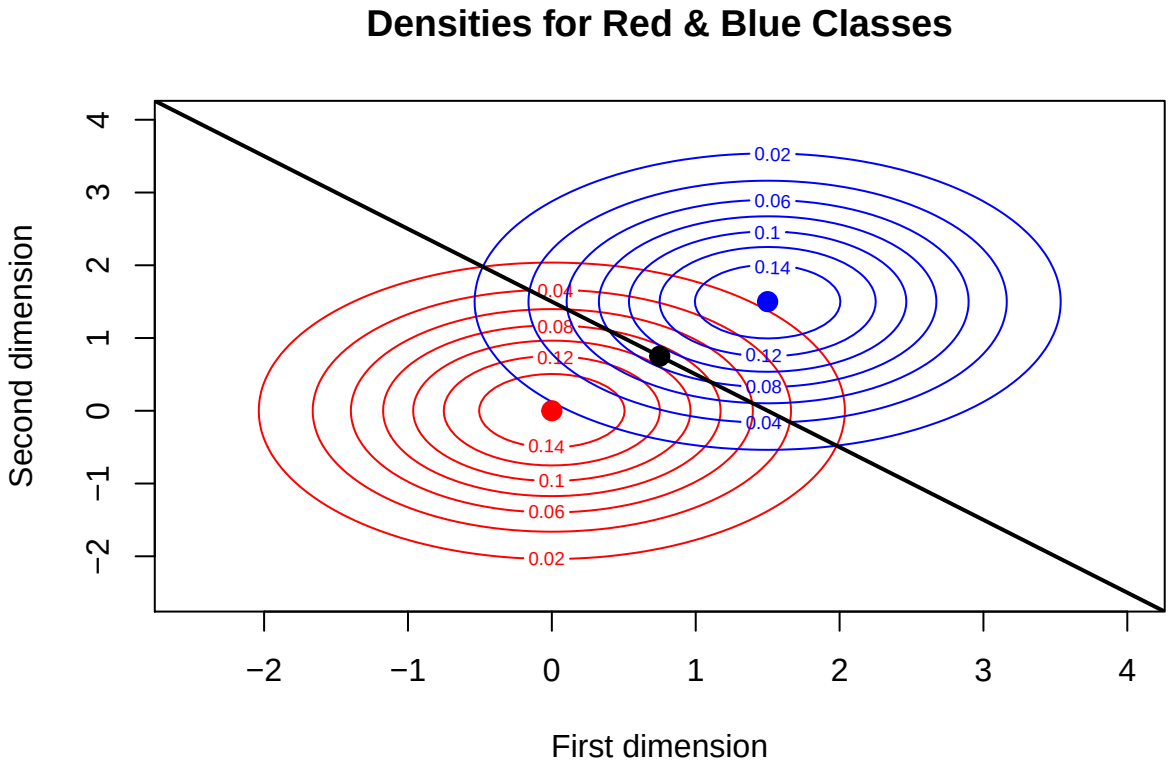
```
k <- which.min(error[, 2])[1]
pred <- knn(train=train, test=test, cl=class, k=k)
blue <- which(pred == "blue")
red <- which(pred == "red")
plot(test[, 1], test[, 2], col = class.test, pch = 17,
     xlab = "First Dimension", ylab = "Second dimension",
     main = "Test Observations (k  = 4)")
points(test[blue, 1], test[blue, 2], cex = 2, col = "blue", lwd=1.5)
points(test[red, 1], test[red, 2], cex = 2, col = "red", lwd=1.5)
legend("bottomright", c("True Label", "Predicted Label"),
          pch = c(17, 1))
```



e.There are many ways you can compute the Bayes error rate, but in this solution, I will follow the hints Prof. Witten provided. I first need to determine to which group the Bayes classifier will assign a point. By definition, the Bayes classifier will assign a point to whichever class has higher probability. To determine which class is more likely, I think about the probability density functions (pdf's) for the red and blue classes. I know that the pdf's are both maximal at their mean point and decrease in value moving out from their mean point. Further, because both the blue and red class have an identity, I, covariance matrix, the value of the pdf's decrease by the same amount no matter what direction you move from the center. Also, the pdf's values are the same for the red and blue classes. They are just shifted in space. The figure below can be used to help visualize this. The red and blue points are the means of the distributions and the contours illustrate the values of the pdfs moving away from their centers.

Using the figure, it becomes clear that the pdf value at a point will be higher for the red class than the blue class when the point is closer to the mean of the red distribution than the mean of the blue distribution and vice versa. So, we know that the line dividing the classifier's assignment is actually just the line that splits the plane into points closer to the blue mean and points closer to the red mean. There are lots of ways you could come up with this line. One is to note that the slope of the dividing line will be perpendicular to the line connecting the red mean and blue mean and that it will go through the half-way point between the centers, $(0.75, 0.75)$. With a little algebra, this gives a line equation, $x_2' = -x_1' + 1.5$. Now, for any point (x_1, x_2) if $x_2' > x_2$ we know the Bayes classifier will assign (x_1, x_2) to the blue class and vice versa. Alternatively, we can just compute the value of the pdf of the red class and the value of the pdf of the blue class at any point (x_1, x_2) and assign (x_1, x_2) to whichever class gives the higher value.

```r
library("mvtnorm") #package for working with multivariate normal distributions
#layout a 100 x 100 grid of points to evalute the densities at
nGrid <- 100
x1 <- seq(-2.5, 4, length = nGrid)
x2 <- seq(-2.5, 4, length = nGrid)

#find the density values at all the points on the grid
muRed <- c(0, 0); muBlue <- c(1.5, 1.5)
zBlue <- zRed <- matrix(nrow = nGrid, ncol = nGrid)
for (i in 1:nGrid) {
  for (j in 1:nGrid) {
    zBlue[i, j] <- dmvnorm(c(x1[i], x2[j]), mean = muBlue, sigma = diag(2))
    zRed[i, j] <- dmvnorm(c(x1[i], x2[j]), mean = muRed, sigma = diag(2))
  }
}

#plot contours of the grid
contour(x1, x2, zRed, col = "red", xlab = "First dimension", ylab = "Second dimension",
        main = "Densities for Red & Blue Classes")
contour(x1, x2, zBlue, col = "blue", add = T)
points(0, 0, col = "red", pch = 20, cex = 2)
points(1.5, 1.5, col = "blue", pch = 20, cex = 2)

#plot the dividing line
x <- seq(-3, 4.5, length = 100)
points(x, -x + 1.5, type = "l", lwd = 2)
points(.75, .75, pch = 20, cex = 2)
```



Now that I've established how any point will be classified with the Bayes classifier, I can compute the expected error in that classification via simulation. I do this by repeatedly generating points and checking if they were assigned correctly. Specifically, for every iteration, I draw with probability 0.5 whether the point is drawn from the red or blue class. Then I draw (x_1, x_2) from either the distribution corresponding to the red class or blue class accordingly. Next, I compute to what group the Bayes classifier would assign the point at (x_1, x_2). (In the first set of code, I use the line to find the classification of the points and in the second set of code I use the pdf's directly.) Finally, I record whether the assignment from the bayes classifier was correct. The proportion of iterations that were not assigned correctly gives the Bayes error rate. With both sets of code, I get an error of approximately 0.14 (Your exact answers may vary slightly depending on how many iterations you ran your simulation.)

```
nSim <- 1e5
errorInd <- rep(0, nSim) #vector that will indicate if an error was made on ith iteration
for (i in 1:nSim) {
  #with prob = 0.5, the point is in the red class (0) or the blue class (1)
  y <- sample(c(0, 1), 1)

  #given the class we can sample values for x1 and x2
  if (y == 0) {
    x1 <- rnorm(1, 0, 1); x2 <- rnorm(1, 0, 1)
  } else {
    x1 <- rnorm(1, 1.5, 1); x2 <- rnorm(1, 1.5, 1)
  }

  #If (x1, x2) is below the dividing line for the bayes classifier
  #the Bayes classified will predict red and blue otherwise
  lineX2 <- -x1 + 1.5
  if (x2 < lineX2) {
    bayesClass <- 0
  } else {
    bayesClass <- 1
  }

  #record if the prediction is in error
  if (bayesClass != y) {
    errorInd[i] <- 1
  }
}

#compute error rate
bayesErrorRate <- sum(errorInd)/nSim
bayesErrorRate
```

```
## [1] 0.14297
```

```
nSim <- 1e5
errorInd <- rep(0, nSim) #vector that will indicate if an error was made on ith iteration
for (i in 1:nSim) {
  #with prob = 0.5, the point is in the red class (0) or the blue class (1)
  y <- sample(c(0, 1), 1)

  #given the class we can sample values for x1 and x2
  if (y == 0) {
    x1 <- rnorm(1, 0, 1); x2 <- rnorm(1, 0, 1)
  } else {
```

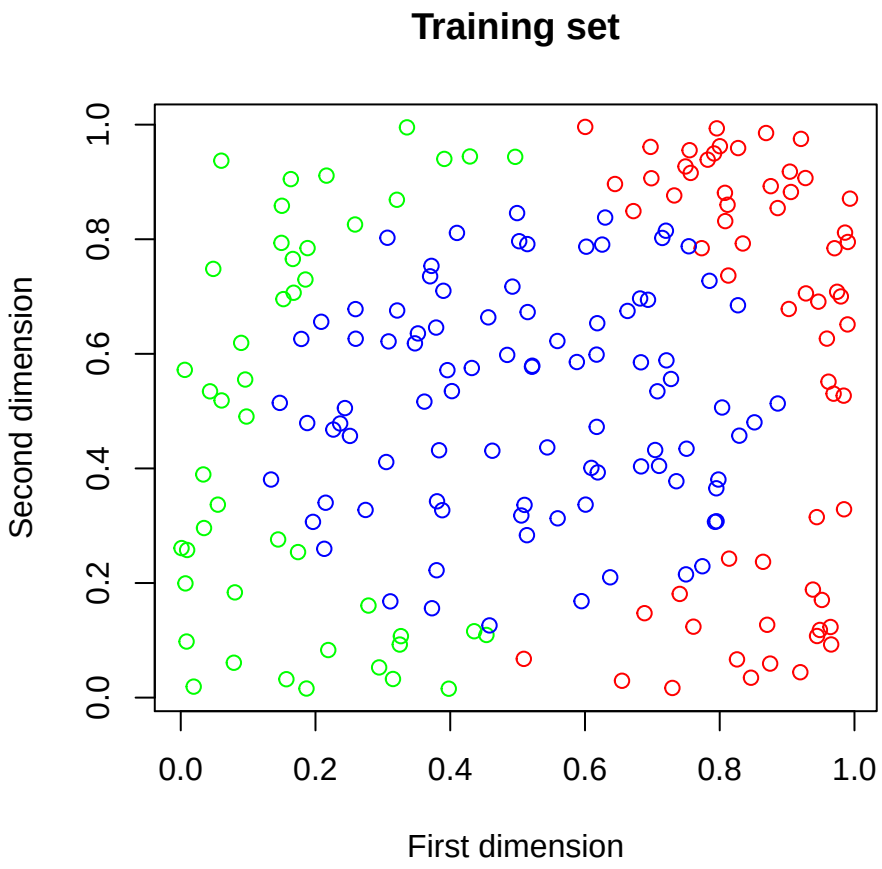
```
    x1 <- rnorm(1, 1.5, 1); x2 <- rnorm(1, 1.5, 1)
  }

  #Evalute the pdf for both classes at point (x1, x2); whichever pdf
  #has a higher value is more likely, so that will be the assigned class
  pBlue <- dmvnorm(c(x1, x2), mean = muBlue, sigma = diag(2))
  pRed <- dmvnorm((c(x1, x2)), mean = muRed, sigma = diag(2))
  if (pRed > pBlue) {
    bayesClass <- 0
  } else {
    bayesClass <- 1
  }

  #record if the prediction is in error
  if (bayesClass != y) {
    errorInd[i] <- 1
  }
}

#compute error rate
bayesErrorRate <- sum(errorInd)/nSim
bayesErrorRate
```

```
## [1] 0.14274
```

Problem 2 $(2+1+2+1+2=8$ pt)

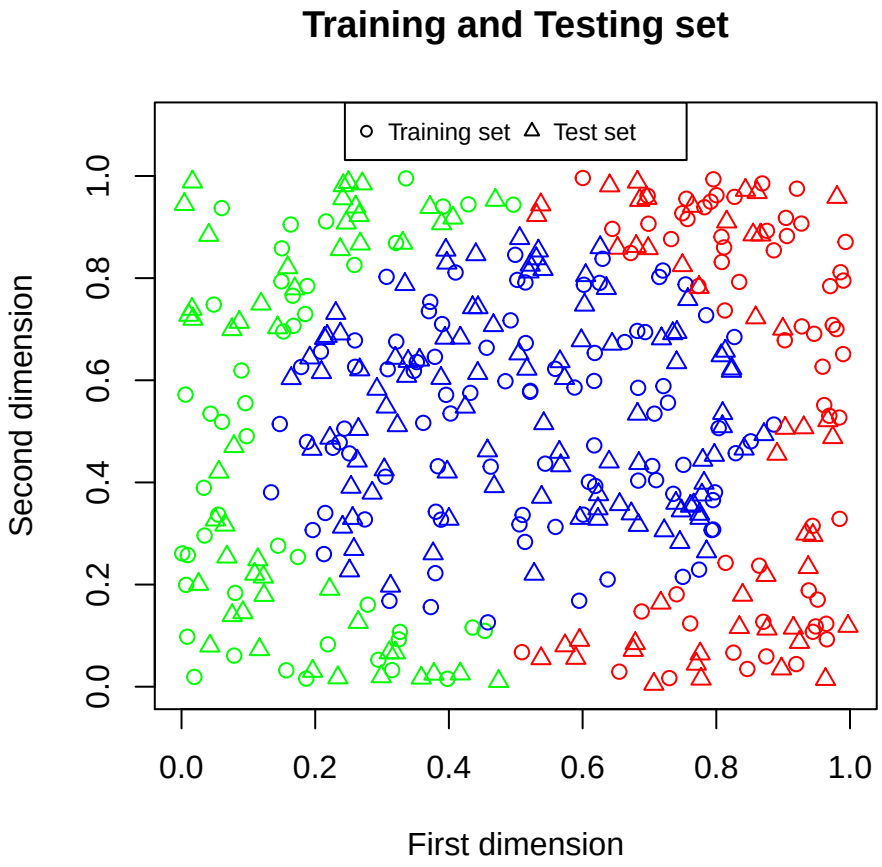
a. The generated training set plot is as follows.

```
set.seed(12345)
n <- 200
train <- cbind(runif(n), runif(n))
class <- rep("", n)
for(i in 1:n){
    if((train[i,1] - 0.5)^2 + (train[i,2] - 0.5)^2 > 0.15 && train[i,1] > 0.5){
        class[i] <- "red"
    }else if((train[i,1] - 0.5)^2 + (train[i,2] - 0.5)^2 > 0.15 && train[i,1] <= 0.5){
        class[i] <- "green"
    }else{
        class[i] <- "blue"
    }
}
plot(train[, 1], train[, 2], col = class,
     xlab = "First dimension", ylab = "Second dimension", main = "Training set")
```



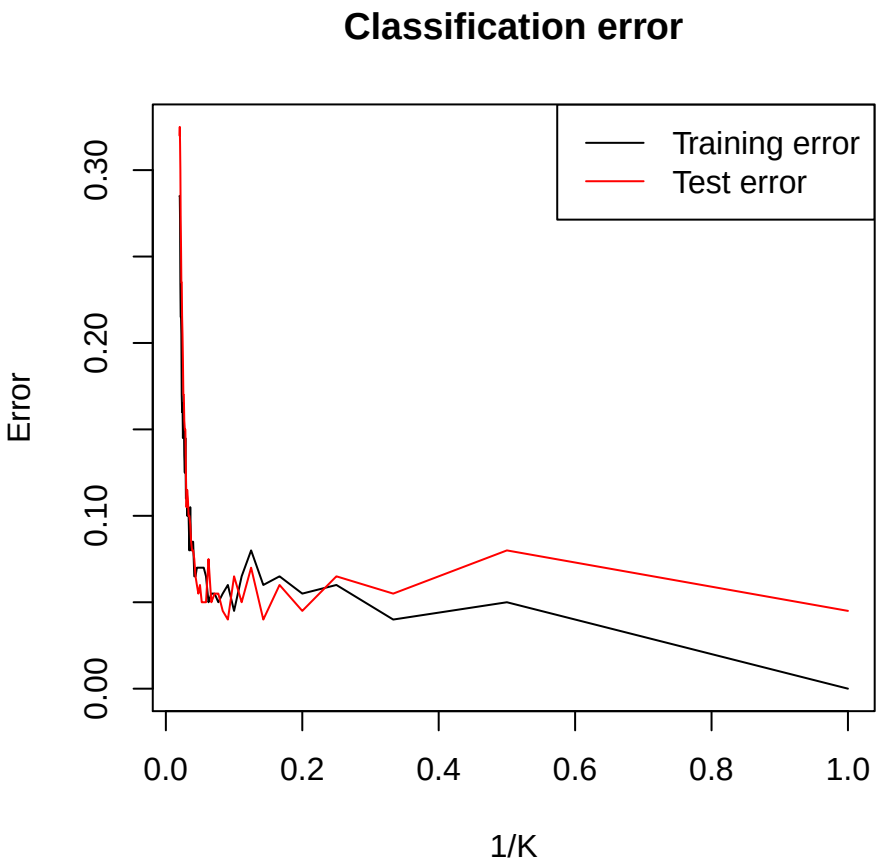
b. The generated observations are is plotted below.

```
n <- 200
test <- cbind(runif(n), runif(n))
class.test <- rep("", n)
for(i in 1:n){
    if((test[i,1] - 0.5)^2 + (test[i,2] - 0.5)^2 > 0.15 && test[i,1] > 0.5){
        class.test[i] <- "red"
    }else if((test[i,1] - 0.5)^2 + (test[i,2] - 0.5)^2 > 0.15 && test[i,1] <= 0.5){
        class.test[i] <- "green"
    }else{
        class.test[i] <- "blue"
    }
}
plot(train[, 1], train[, 2], col = class,
     xlim = c(0, 1), ylim = c(0, 1.1),
     xlab = "First dimension", ylab = "Second dimension", main = "Training and Testing set")
points(test[, 1], test[, 2], col = class.test, pch = 2)
legend("top", c("Training set", "Test set"),  h = T, pch = c(1, 2), cex = .75)
```

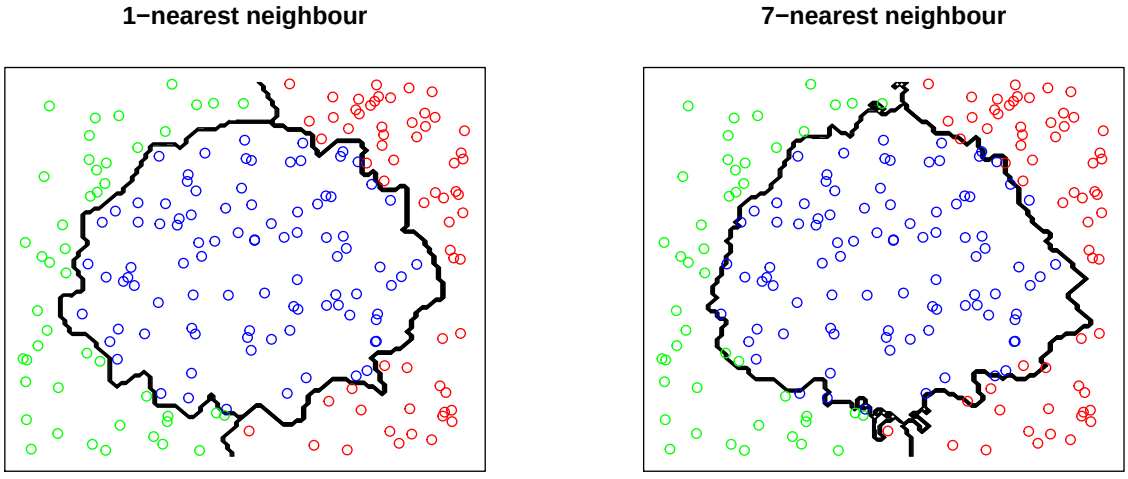


c. The classification error plot is displayed below. Unlike problem 1, although there is the testing error curve is till roughly U-shaped, the right end of the curve did not go up as quickly, indicating the 1-Nearest-Neighbor is not overfitting data as much as in the previous problem.

```
K <- 50
error <- matrix(NA, K, 2)
for(k in 1:K){
    knn1 <- knn(train=train, test=train, cl=class, k=k)
    error[k, 1] <- sum(knn1 != class) / n
    knn2 <- knn(train=train, test=test, cl=class, k=k)
    error[k, 2] <- sum(knn2 != class.test) / n
}
plot(1/seq(1, K), error[, 1], ylim = range(error), type = "l",
     xlab = "1/K", ylab = "Error", main = "Classification error")
lines(1/seq(1, K), error[, 2], col = "red")
legend("topright", c("Training error", "Test error"),
        col = c("black", "red"), lty = c(1, 1))
```



(FYI) In fact, we can check that from the decision boundary when different K values are chosen. Below shows the comparison of $K = 1$ and $K = 7$. Even when $K = 1$, since the regions of different groups do not overlap, the decision boundary is still reasonable.

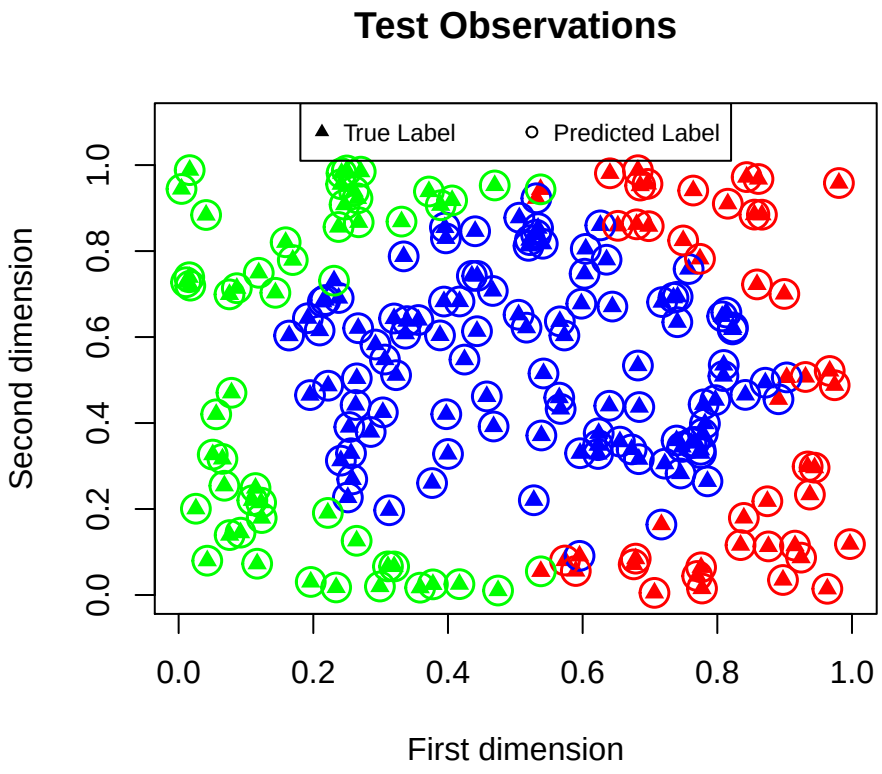


d. In my simulation, $k = 7$ resulted in the smallest test error. The test observations with the true and predicted labels are plotted below.

```
k <- which.min(error[, 2])[1]
print(k)
```

```
## [1] 7
```

```
pred <- knn(train=train, test=test, cl=class, k=k)
blue <- which(pred == "blue")
red <- which(pred == "red")
green <- which(pred == "green")
plot(test[, 1], test[, 2], col = class.test, pch = 17, ylim = c(0, 1.1),
     xlab = "First dimension", ylab = "Second dimension",
     main = "Test Observations")
points(test[blue, 1], test[blue, 2], cex = 2, col = "blue", lwd=1.5)
points(test[red, 1], test[red, 2], cex = 2, col = "red", lwd=1.5)
points(test[green, 1], test[green, 2], cex = 2, col = "green", lwd=1.5)
legend("top", c("True Label", "Predicted Label"), h = T, cex = .75,
        pch = c(17, 1))
```



e. In this problem, $Pr(Y = j|x)$ is piece-wise constant. That is, given any $x \in [0, 1]^2$, we know what the value of Y is. So, $Pr(Y = j|x)$ is 1 for one class, and 0 for the other two classes. Therefore $max_j Pr(Y = j|x) = 1, \forall x$. Thus Bayes error rate is 0. This seemingly surprise calculation is also related to the fact that even with $K = 1$, the model does not overfit the data very much. The data generating mechanism in this problem ensures that each class has a non-overlapping support (i.e., the domains of x which have positive probability of being in each classes do not overlap). Therefore, intuitively, even if we use $K = 1$, which is the most complex model, given large training samples, any testing data point will still have a higher chance of being closer to a training data point in the same class than any other points in a different class. Thus we will not expect 1-nearest-neighbor to overfit data very much.

Problem 3 ($1 + 1 = 2$ pts)

a. This is a regression problem since the final exam scores are *quantitative*. The goal is prediction, since we want to *predict* students' scores. In this case, $n = 50$ and $p = 8$.

(caveat: one might argue the final exam scores are categorical/qualitative, if only certain points are given. For example, in the (unlikely) scenario where there are only 10 True/False questions on the final, the score a student can get only takes value in $\{0, 1, 2, ..., 10\}$*. I did not deduct points for this type of answer as long as reasonable explanations are provided.)*

b. This is a classification problem since the response variable we are modeling is *qualitative/two-class/binary/categorical*. The goal is inference since we want to *understand* the influence of the various factors. In this case, $n = 50$ and $p = 6$.

(caveat: one might also argue that $p = 9$ *or* $p = 10$ *by recoding the student's year as 5 factors or 4 factors plus an implicit baseline. These are both reasonable answers if explanations are provided.)*

Problem 4 ($1 + 1 + 1 + 1 = 4$ pts)

a. If the number of predictors p is very large and the number of observations n is very small then an inflexible model is expected to perform better than a flexible model. Flexible models are not robust to small n and large p which can result in overfitting.

b. If the sample size n is very large and the number of predictors p is very small, then we expect a flexible method to perform better than an inflexible method. Since we have a large sample size with small p, the flexible model is unlikely to overfit the data and will have less bias than an inflexible model.

c. If the relationship between the predictors and response is highly non-linear then we expect that a flexible model to perform better than an inflexible model. For example, an inflexible model, such as linear regression with linear basis functions, cannot capture the non-linear behaviour of the response.

d. If the variance of the error terms $\sigma^2 = Var(\epsilon)$ is extremely high then an inflexible model is expected to have better performance than a flexible model. Flexible models will tend to overfit the data by modelling the error terms σ^2.

Problem 5 ($1 + 5 = 6$ pts)

One possible sketch is provided in Figure 1.

- (Squared) bias: The bias of the model decreases with increased flexibility.
- Variance: The variance will increase with the flexibility of the model as small changes in the training data (can) result in different fits for highly flexible models.
- Training error: The training error decreases as model flexibility increases.
- Test error: The test error is generally large for extremes in model flexibility: models with low flexibility tend to underfit whereas models with high flexibility tend to overfit. The lowest test error will be between the extremes.
- Irreducible error: This error is constant as it does not depend on the model flexibility. It is inherent in the data generation process and cannot be controlled through modeling.

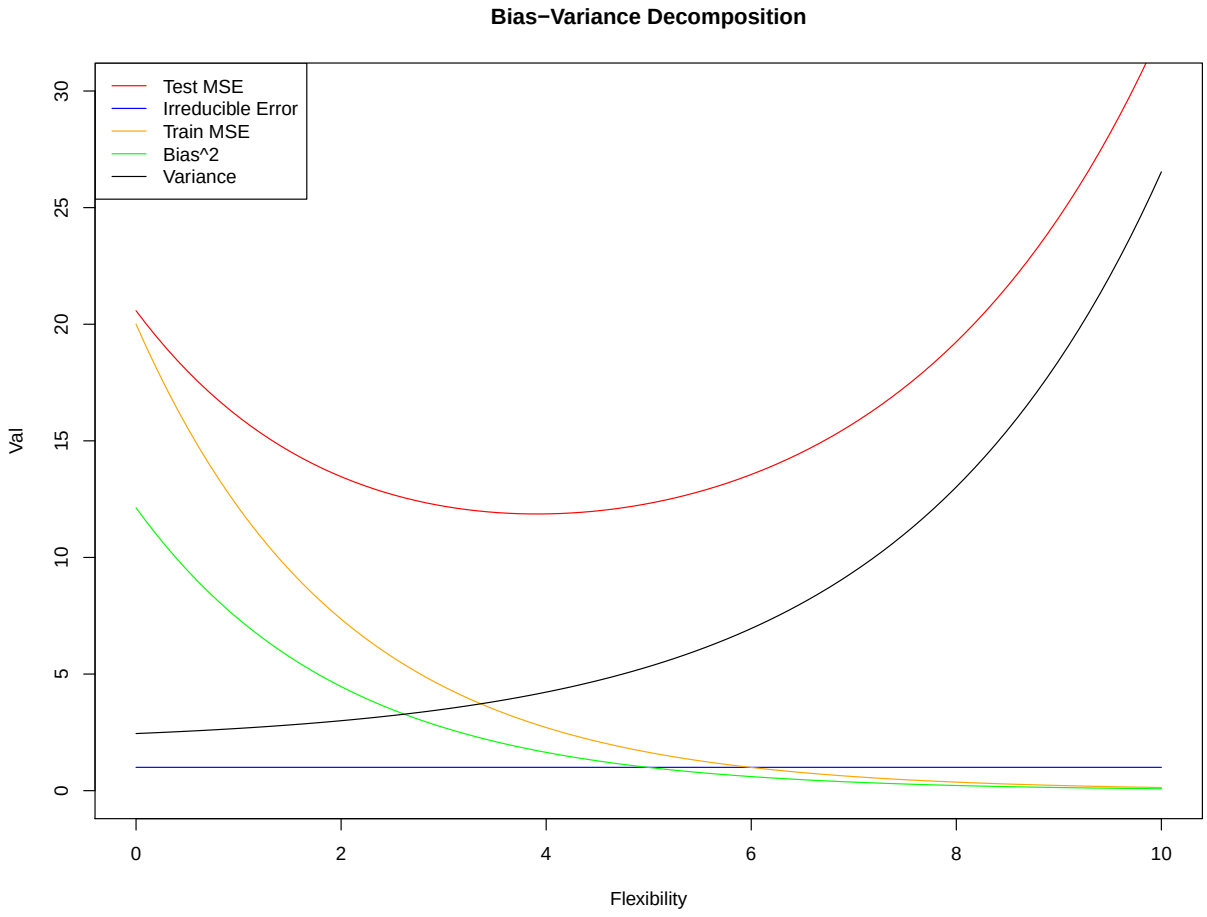


Figure 1: Illustration of Bias-Variance Trade-off

Problem 6 ($1+2+1+2+1+1+2+2=12$ pts)

a. There are 506 rows and 14 columns in the data. Each row represent one observation of a neighborhood, and each column represent one feature of the neighborhoods. The details of each column can be found by running `?Boston` and is omitted in this key.

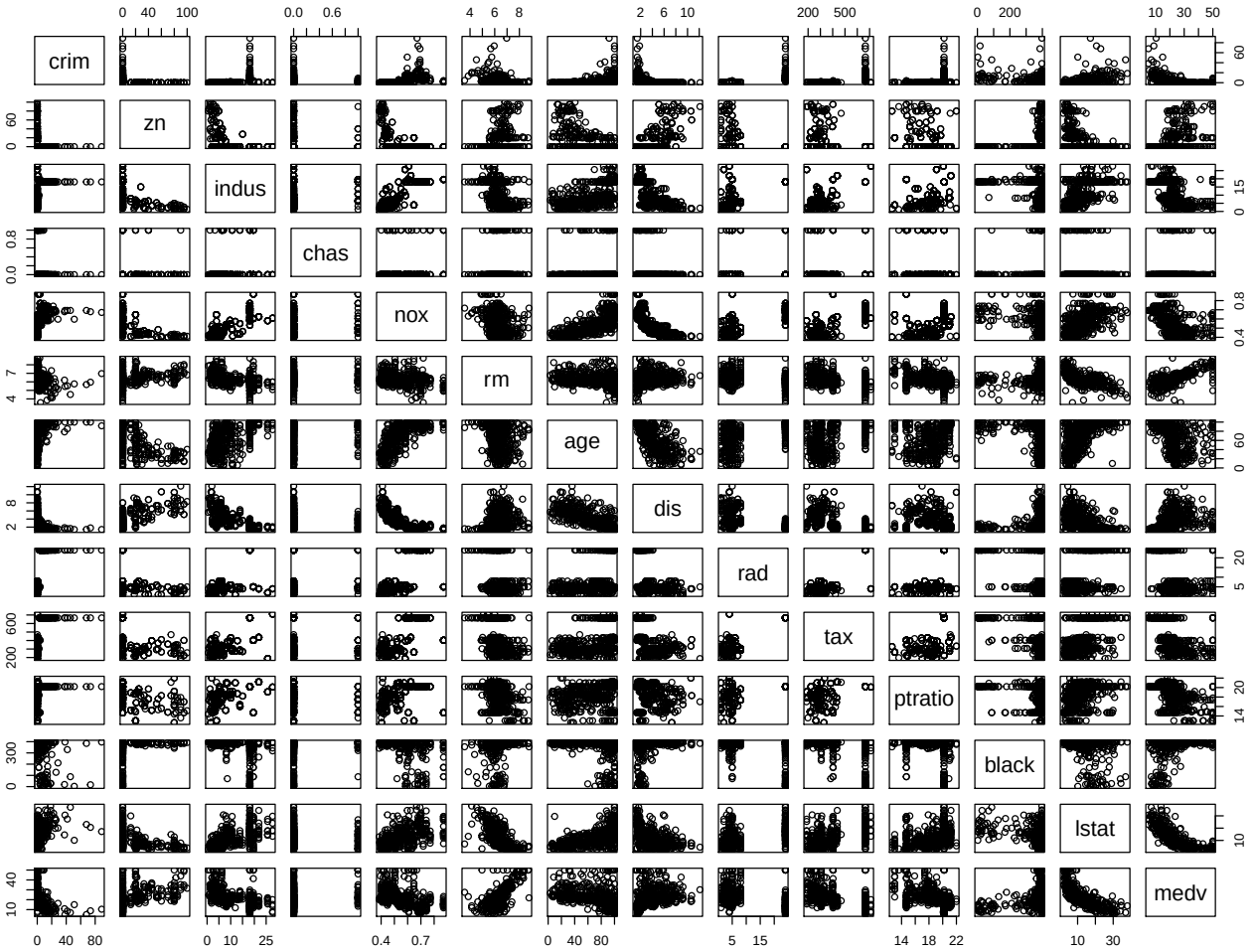
```
library(MASS)
data(Boston)
dim(Boston)
```

```
## [1] 506  14
```

```
?Boston
```

b. To examine pairwise relationships between the variables a standard pairwise plot is presented below.

```
pairs(Boston)
```



From this plot we see that there are a number of pairs of variables with a moderate absolute correlation ($\approx 70\%$). We list a few pairs here and the sign of the correlation: $(\texttt{age}, \texttt{dis}, -)$, $(\texttt{age}, \texttt{nox}, +)$, $(\texttt{indus}, \texttt{dis}, -)$, $(\texttt{indus}, \texttt{tax}, +)$, $(\texttt{nox}, \texttt{dist}, -)$, $(\texttt{rm}, \texttt{medv}, +)$, $(\texttt{lstat}, \texttt{medv}, -)$. These seem to be fairly reasonable; for example, higher number of rooms per dwelling is associated with higher median home value.

c. The predictors associated with per capita crime rate can be determined through calculating correlation between *crim* and each of the other variables, or visually examined from the pairs plot. The two largest absolute correlation between crime rate and another covariate are 0.63 (rad), and 0.58 (tax). All other correlations are fairly weak. There are also mild associations between *zn, chas, nox, dist, medv*. This

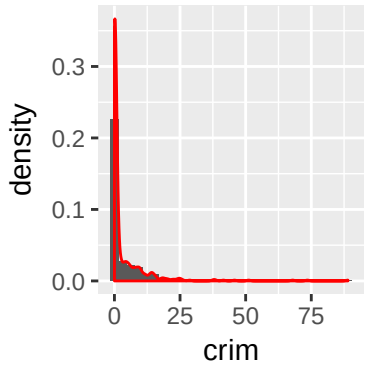
can be confirmed through an exploratory linear model of crime rate regressed on each of the other covariates.

```
round(cor(Boston)["crim", ], 2)
```

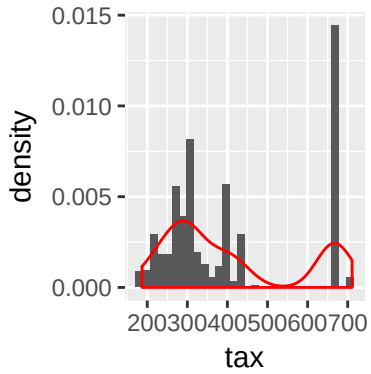
```
##    crim      zn   indus    chas     nox      rm     age     dis     rad
##    1.00   -0.20    0.41   -0.06    0.42   -0.22    0.35   -0.38    0.63
##     tax ptratio   black   lstat    medv
##    0.58    0.29   -0.39    0.46   -0.39
```

d. Boston suburbs with high rates of crime, tax rates, and pupil-teacher ratios can be explored via the density plots within the pairs plot. Most neighborhoods have low crime, but the right tail of the distribution shows that there are neighborhoods with high crime; it appears that there are two types of tax regimes in Boston towns: high and low (with more people in lower tax rates); and there are a few lucky schools with low pupil-teacher ratios, but most towns have high pupil-teacher ratios. The range of each predictor are:
   - Crime rate: $[0.00632, 88.97620]$;
   - Tax rate: $[187, 711]$;
   - Pupil-teacher ratio: $[12.6, 22.0]$.

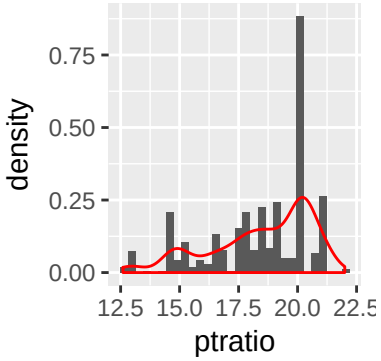
```
library(ggplot2)
ggplot(Boston, aes(x=crim))+ geom_histogram(aes(y = ..density..)) + geom_density(color = "red")
```




```
ggplot(Boston, aes(x=tax))+ geom_histogram(aes(y = ..density..)) + geom_density(color = "red")
```




```
ggplot(Boston, aes(x=ptratio))+ geom_histogram(aes(y = ..density..)) + geom_density(color = "red")
```



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

```
sum(Boston$chas==1)
```

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

f. The mean pupil-teacher ratio among the towns in this dataset is 18.45, and the standard deviation is 2.16.

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

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

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

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

g. There are 16 suburbs that have the highest median value of homes. The characteristics of these homes are summarized below. It appears to consist of two groups of suburbs: one group with high proportion of non-retail business acres per town (*indus*) and no residential land zoned for lots over 25,000 sq.ft (*zn*); and another group of opposite characteristics. The crime rates are all low, especially in the second group, which may explain why those suburbs have high median value of owner-occupied homes. (*caveat: this is an open question so all reasonable comments are acceptable*).

```
subset(Boston, medv == max(Boston$medv))
```

```
##        crim zn indus chas    nox    rm   age    dis rad tax ptratio  black
## 162 1.46336  0 19.58    0 0.6050 7.489  90.8 1.9709   5 403    14.7 374.43
## 163 1.83377  0 19.58    1 0.6050 7.802  98.2 2.0407   5 403    14.7 389.61
## 164 1.51902  0 19.58    1 0.6050 8.375  93.9 2.1620   5 403    14.7 388.45
## 167 2.01019  0 19.58    0 0.6050 7.929  96.2 2.0459   5 403    14.7 369.30
## 187 0.05602  0  2.46    0 0.4880 7.831  53.6 3.1992   3 193    17.8 392.63
## 196 0.01381 80  0.46    0 0.4220 7.875  32.0 5.6484   4 255    14.4 394.23
## 205 0.02009 95  2.68    0 0.4161 8.034  31.9 5.1180   4 224    14.7 390.55
## 226 0.52693  0  6.20    0 0.5040 8.725  83.0 2.8944   8 307    17.4 382.00
## 258 0.61154 20  3.97    0 0.6470 8.704  86.9 1.8010   5 264    13.0 389.70
## 268 0.57834 20  3.97    0 0.5750 8.297  67.0 2.4216   5 264    13.0 384.54
## 284 0.01501 90  1.21    1 0.4010 7.923  24.8 5.8850   1 198    13.6 395.52
## 369 4.89822  0 18.10    0 0.6310 4.970 100.0 1.3325  24 666    20.2 375.52
## 370 5.66998  0 18.10    1 0.6310 6.683  96.8 1.3567  24 666    20.2 375.33
## 371 6.53876  0 18.10    1 0.6310 7.016  97.5 1.2024  24 666    20.2 392.05
## 372 9.23230  0 18.10    0 0.6310 6.216 100.0 1.1691  24 666    20.2 366.15
## 373 8.26725  0 18.10    1 0.6680 5.875  89.6 1.1296  24 666    20.2 347.88
##     lstat medv
## 162  1.73   50
## 163  1.92   50
```

```
## 164  3.32   50
## 167  3.70   50
## 187  4.45   50
## 196  2.97   50
## 205  2.88   50
## 226  4.63   50
## 258  5.12   50
## 268  7.44   50
## 284  3.16   50
## 369  3.26   50
## 370  3.73   50
## 371  2.96   50
## 372  9.53   50
## 373  8.88   50
```

h. There are 133 suburbs average more than six rooms per dwelling and 13 suburbs average more than eight rooms per dwelling. Towns that on average have more than 8 rooms per dwelling have lower crime rates (mean 0.71 versus 3.69), have lower tax rates (325.08 versus 410.43), have lower pupil to teacher ratios (16.36 versus 18.51), and higher median home values (44.20 versus 21.96).

```
sum(Boston$rm > 6)
```

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

```
sum(Boston$rm > 8)
```

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

```
apply(subset(Boston, rm < 8), 2, mean)
```

```
##         crim           zn        indus         chas          nox
##   3.68985513  11.30425963  11.24379310   0.06693712   0.55510264
##           rm          age          dis          rad          tax
##   6.23021095  68.49675456   3.80466349   9.60446247 410.43002028
##      ptratio        black        lstat         medv
##  18.51075051 355.92154158  12.87306288  21.96146045
```

```
apply(subset(Boston, rm > 8), 2, mean)
```

```
##        crim          zn       indus        chas         nox          rm
##   0.7187954  13.6153846   7.0784615   0.1538462   0.5392385   8.3485385
##         age         dis         rad         tax     ptratio       black
##  71.5384615   3.4301923   7.4615385 325.0769231  16.3615385 385.2107692
##       lstat        medv
##   4.3100000  44.2000000
```