BDA/IDAR Coursework 1 - Solutions

1. Statistical learning methods (10% | 20%)

Marking scheme: 2.5% | 5% each.

For each of parts (a) through (d), indicate whether we would generally expect the performance of a flexible statistical learning method to be better or worse than an inflexible method. Justify your answer.

(a) The sample size n is extremely large, and the number of predictors p is small.

Solution: Better. We expect that with a very large number of measurements n, a flexible learning method will fit the data closer and will be able to learn the signal (y) without as much fear of overfitting.

(b) The number of predictors p is extremely large, and the number of observations n is small.

Solution: Worse. If the number of predictors p is very large and n small then there is a greater possibility that a flexible learning method would overfit. Then we expect the inflexible method to be better in this case.

(c) The relationship between the predictors and response is highly non-linear.

Solution: Better. A higly non-linear relationship would most likely need a flexible statistical learning method to perform optimally.

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

Solution: Worse. With a very large error term variance σ^2 there is more worry about overfitting and thus an inflexible method would perform better.

2. Descriptive analysis (10% | 20%)

Marking scheme: 2.5% | 5% each.

In a higher educational institution the comprehensive applied mathematics exam is comprised of two parts. On the first day, 20 students took the exam, the results of which are presented below:

Oral exam results:	4, 1, 4, 5, 3, 2, 3, 4, 3, 5, 2, 2, 4, 3, 5, 5, 1, 1, 1, 2.
Written exam results:	2, 3, 1, 4, 2, 5, 3, 1, 2, 1, 2, 2, 1, 1, 2, 3, 1, 2, 3, 4.

(a) Use R to calculate the mean, the mode, the median, the variance and the standard deviation of the oral and written exams separately and added together as well.

Solution:

```
oral <-    c(4, 1, 4, 5, 3, 2, 3, 4, 3, 5, 2, 2, 4, 3, 5, 5, 1, 1, 1, 2)
written <- c(2, 3, 1, 4, 2, 5, 3, 1, 2, 1, 2, 2, 1, 1, 2, 3, 1, 2, 3, 4)
total <- oral + written

df <- data.frame(oral, written, total)
paste("The mean")   #paste, sprintf, print will all print some strings
```

```
## [1] "The mean"
```

```
apply(df,2,mean)  #2 means per column
```

```
##    oral written   total
##    3.00    2.25    5.25
```

```
sprintf("The median")
```

```
## [1] "The median"
```

```
apply(df,2,median)
```

```
##    oral written   total
##       3       2       5
```

```
print("The variance")
```

```
## [1] "The variance"
```

```
apply(df,2,var)
```

```
##     oral  written    total
## 2.105263 1.355263 2.828947
```

```
print("The standard deviation")
```

```
## [1] "The standard deviation"
```

```
apply(df,2,sd)
```

```
##     oral  written    total
## 1.450953 1.164158 1.681947
```

```
#define a new function
myMode <- function(x){
  names(sort(-table(x)))[1]
}

paste("mode")
```

```
## [1] "mode"
```

```
apply(df,2,myMode)
```

```
##    oral written   total
##     "1"     "2"     "4"
```

(b) Find the covariance and correlation between the oral and written exam scores.

Solution:

```
cov(df$oral,df$written)  #It's also fine to write cov(oral, written)
```

```
## [1] -0.3157895
```

```
cor(df$oral,df$written)
```

```
## [1] -0.1869531
```

```
cor(df$oral,df$total)
```

```
## [1] 0.7332629
```

```
cor(df$total,df$written)
```

```
## [1] 0.5308713
```

(c) Is there a positive or negative or no correlation between the two?

Solution: There is a very weak (low) negative correlation between the two. However there is quite a strong (high) postive correlation between the total and oral and a weak positive correlation between written and total.

(d) Is there causation between the two? Justify your answers.

Solution: Normally, when two variables are correlated, we start discussing about the causation. In this case, there is very low correlation between the two, and statistically it means that there is no linear relationship between the two. However, there might be non-linear relationship relationship and causation.

3. Descriptive analysis (10% | 0%)

Marking scheme: MSc students only. (b)(c) 1% each, the rest 2% each.

This exercise involves the `Auto` data set studied in the class.

(a) Which of the predictors are quantitative, and which are qualitative?

Solution: Using the `summary` command on `Auto` dataset we easily see which predictors are quantitative: `mpg`, `displacement`, `horsepower`, `cylinders` and `acceleration`. While the variables `year` (values are 70 - 82), `origin` (values are 1, 2 or 3) and `name` are quantitative. Note that `year` is an ordinal variable as addition is meaningless. And an ordinal variable is qualitative.

```
library(ISLR)
```

```
## Warning: package 'ISLR' was built under R version 3.4.2
```

```
summary(Auto)
```

```
##       mpg          cylinders      displacement     horsepower
##  Min.   : 9.00   Min.   :3.000   Min.   : 68.0   Min.   : 46.0
##  1st Qu.:17.00   1st Qu.:4.000   1st Qu.:105.0   1st Qu.: 75.0
##  Median :22.75   Median :4.000   Median :151.0   Median : 93.5
##  Mean   :23.45   Mean   :5.472   Mean   :194.4   Mean   :104.5
##  3rd Qu.:29.00   3rd Qu.:8.000   3rd Qu.:275.8   3rd Qu.:126.0
##  Max.   :46.60   Max.   :8.000   Max.   :455.0   Max.   :230.0
##
##      weight      acceleration        year           origin
##  Min.   :1613   Min.   : 8.00   Min.   :70.00   Min.   :1.000
##  1st Qu.:2225   1st Qu.:13.78   1st Qu.:73.00   1st Qu.:1.000
##  Median :2804   Median :15.50   Median :76.00   Median :1.000
##  Mean   :2978   Mean   :15.54   Mean   :75.98   Mean   :1.577
##  3rd Qu.:3615   3rd Qu.:17.02   3rd Qu.:79.00   3rd Qu.:2.000
##  Max.   :5140   Max.   :24.80   Max.   :82.00   Max.   :3.000
##
##                  name
##  amc matador       :  5
##  ford pinto        :  5
##  toyota corolla    :  5
##  amc gremlin       :  4
##  amc hornet        :  4
##  chevrolet chevette:  4
##  (Other)           :365
```

```
table(Auto$year)
```

```
##
## 70 71 72 73 74 75 76 77 78 79 80 81 82
## 29 27 28 40 26 30 34 28 36 29 27 28 30
```

(b) What is the range of each quantitative predictor? You can answer this using the `range()` function.

Solution:

```
names(Auto)  # figure out which columns are quantitative 1:6
```

```
## [1] "mpg"          "cylinders"    "displacement" "horsepower"
## [5] "weight"       "acceleration" "year"         "origin"
## [9] "name"
```

```
sapply(Auto[,1:6], range) #sapply applies a function (here: range) to each column of the dataset.
```

```
##       mpg cylinders displacement horsepower weight acceleration
## [1,]  9.0         3           68         46   1613          8.0
## [2,] 46.6         8          455        230   5140         24.8
```

(c) What is the mean and standard deviation of each quantitative predictor?

Solution:

```
sapply(Auto[,1:6], mean)
```

```
##          mpg    cylinders displacement   horsepower       weight
##    23.445918     5.471939   194.411990   104.469388  2977.584184
## acceleration
##    15.541327
```

```
sapply(Auto[,1:6], sd)
```

```
##          mpg    cylinders displacement   horsepower       weight
##     7.805007     1.705783   104.644004    38.491160   849.402560
## acceleration
##     2.758864
```

(d) Now remove the 10th through 85th observations in the dataset. What is the range, mean, and standard deviation of each predictor in the subset of the data that remains?

Solution:

```
sapply(Auto[-(10:85),1:6], range)
```

```
##       mpg cylinders displacement horsepower weight acceleration
## [1,] 11.0         3           68         46   1649          8.5
## [2,] 46.6         8          455        230   4997         24.8
```

```
sapply(Auto[-(10:85),1:6], mean)
```

```
##          mpg    cylinders displacement   horsepower       weight
##    24.404430     5.373418   187.240506   100.721519  2935.971519
## acceleration
##    15.726899
```

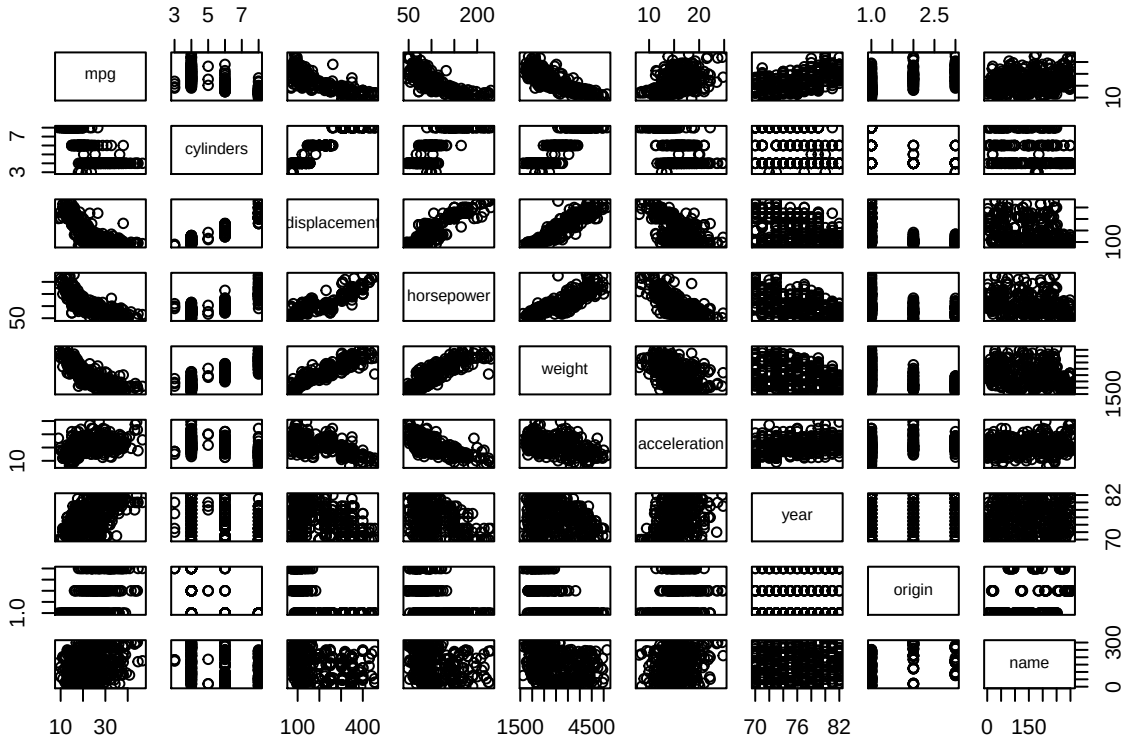
```
sapply(Auto[-(10:85),1:6], sd)
```

```
##          mpg    cylinders displacement   horsepower       weight
##     7.867283     1.654179    99.678367    35.708853   811.300208
## acceleration
##     2.693721
```

(e) Using the full data set, investigate the predictors graphically, using scatterplots or other tools of your choice. Create some plots highlighting the relationships among the predictors. Comment on your findings.

Solution:

```r
pairs(Auto)
```

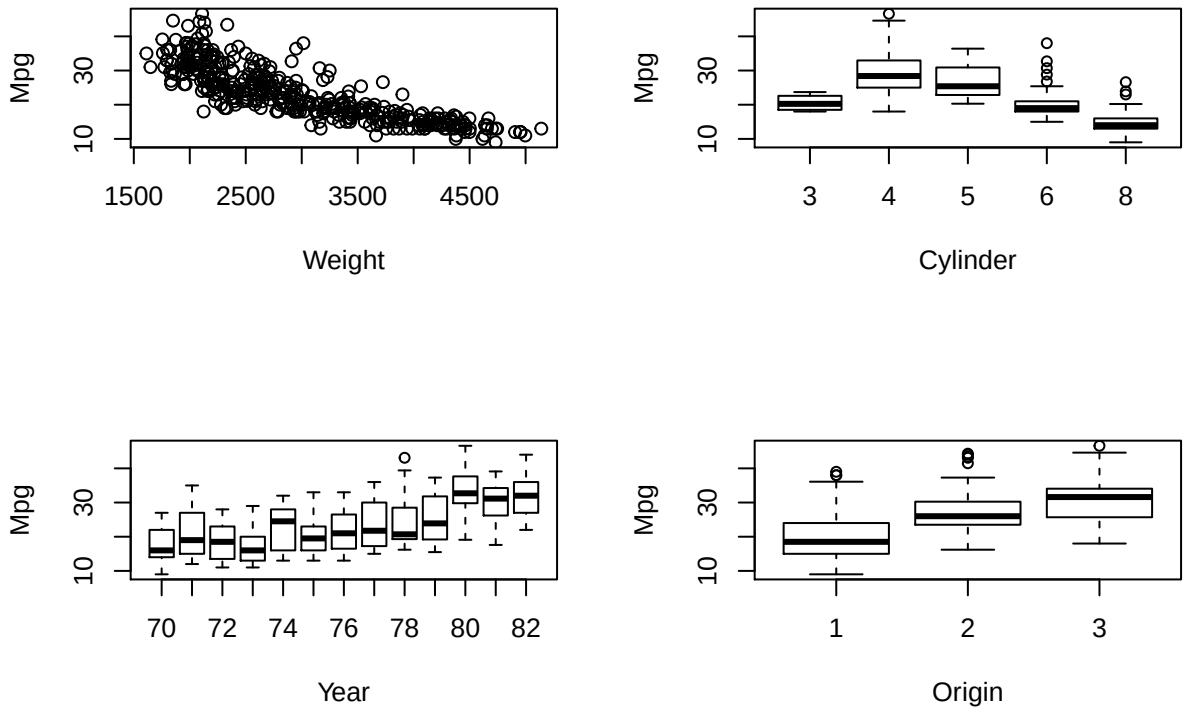



```r
par(mfrow=c(2,2))
# A scatter plot. Heavier cars lower mpg
plot(Auto$weight, Auto$mpg, xlab = "Weight", ylab = "Mpg")

# A boxplot. More cylinders, lower mpg.
# Certain cylinder sizes (eg.4) seems to give a larger mpg.
plot(as.factor(Auto$cylinders), Auto$mpg, xlab = "Cylinder", ylab = "Mpg")

# A boxplot. Year increased, mpg also increased.
plot(as.factor(Auto$year), Auto$mpg, xlab = "Year", ylab = "Mpg")

# A boxplot. We see very different values for mpg depending on origin.
# This indicates that this is perhaps a strong predictor of mpg.
plot(as.factor(Auto$origin), Auto$mpg, xlab = "Origin", ylab = "Mpg")
```



(f) Suppose that we wish to predict gas mileage (`mpg`) on the basis of the other variables. Do your plots suggest that any of the other variables might be useful in predicting `mpg`? Justify your answer.

Solution: The plots in (e) indicate that there are several predictors that could be predictive of `mpg`, e.g., `year`, `weight`, `cylinders` and `origin`.

4. Linear regression (20% | 20%)

Marking scheme: (a) i - iv 2.5% each, (b)(c) 5% each.

This question involves the use of simple linear regression on the `Auto` data set.

(a) Use the `lm()` function to perform a simple linear regression with `mpg` as the response and `horsepower` as the predictor. Use the `summary()` function to print the results. Comment on the output. For example:

i. Is there a relationship between the predictor and the response?

Solution:

```
library(ISLR)
lm.fit <- lm(mpg ~ horsepower, data = Auto)
summary(lm.fit)
```

```
##
## Call:
## lm(formula = mpg ~ horsepower, data = Auto)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
```

```
## -13.5710  -3.2592  -0.3435   2.7630  16.9240
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 39.935861   0.717499   55.66   <2e-16 ***
## horsepower  -0.157845   0.006446  -24.49   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.906 on 390 degrees of freedom
## Multiple R-squared:  0.6059, Adjusted R-squared:  0.6049
## F-statistic: 599.7 on 1 and 390 DF,  p-value: < 2.2e-16
```

The p-value for the model is quite small. This suggests that there is a relationship between `mpg` and `horsepower`.

ii. How strong is the relationship between the predictor and the response?

Solution:

The value of R^2 measures the strength of a relationship. In this case, adjusted $R^2 = 0.6049$, which says that 60% of the variance has been explained using the predictor `horsepower`.

iii. Is the relationship between the predictor and the response positive or negative?

Solution:

The estimate of the coefficient for `horsepower` is -0.157845. This suggests that the relationship is negative, i.e., as `horsepower` increases `mpg` will decrease.

iv. What is the predicted `mpg` associated with a `horsepower` of 98? What are the associated 95% confidence and prediction intervals?

Solution:

```
predict(lm.fit, data.frame(horsepower = 98), interval = "confidence")
```

```
##        fit      lwr      upr
## 1 24.46708 23.97308 24.96108
```

```
predict(lm.fit, data.frame(horsepower = 98), interval = "prediction")
```

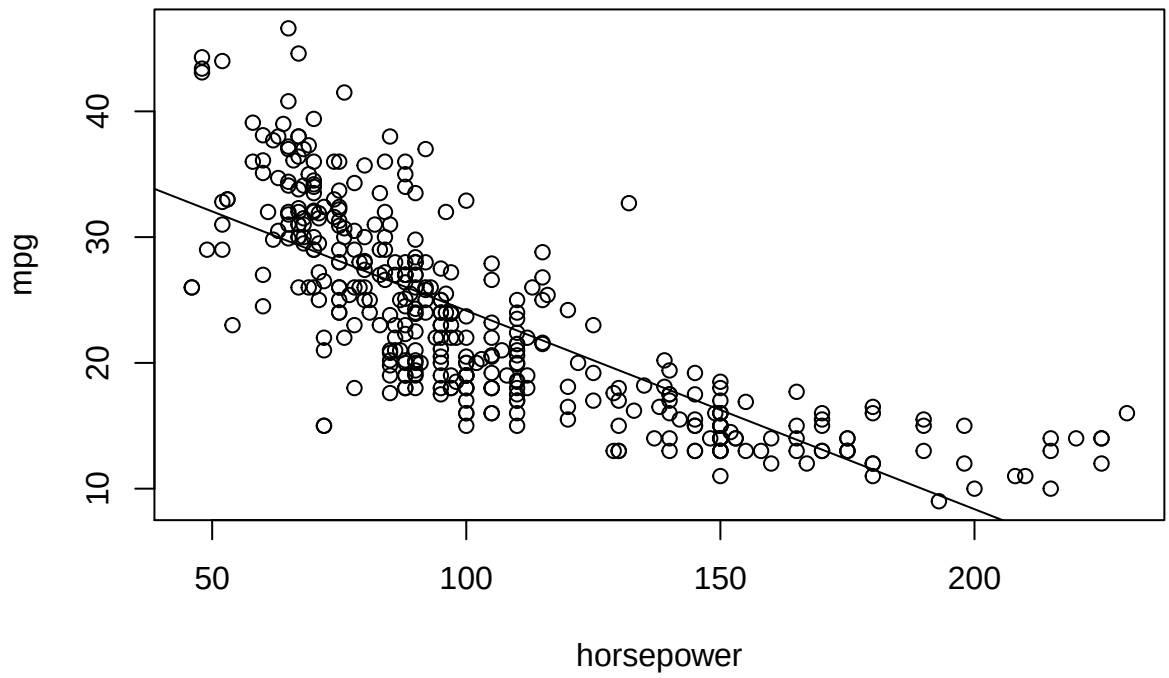
```
##        fit     lwr      upr
## 1 24.46708 14.8094 34.12476
```

The prediction associated with `horsepower = 98` is 24.46708. The confidence interval is [23.97308, 24.96108], and the prediction interval is [14.8094, 34.12476].

(b) Plot the response and the predictor. Use the `abline()` function to display the least squares regression line.

Solution:

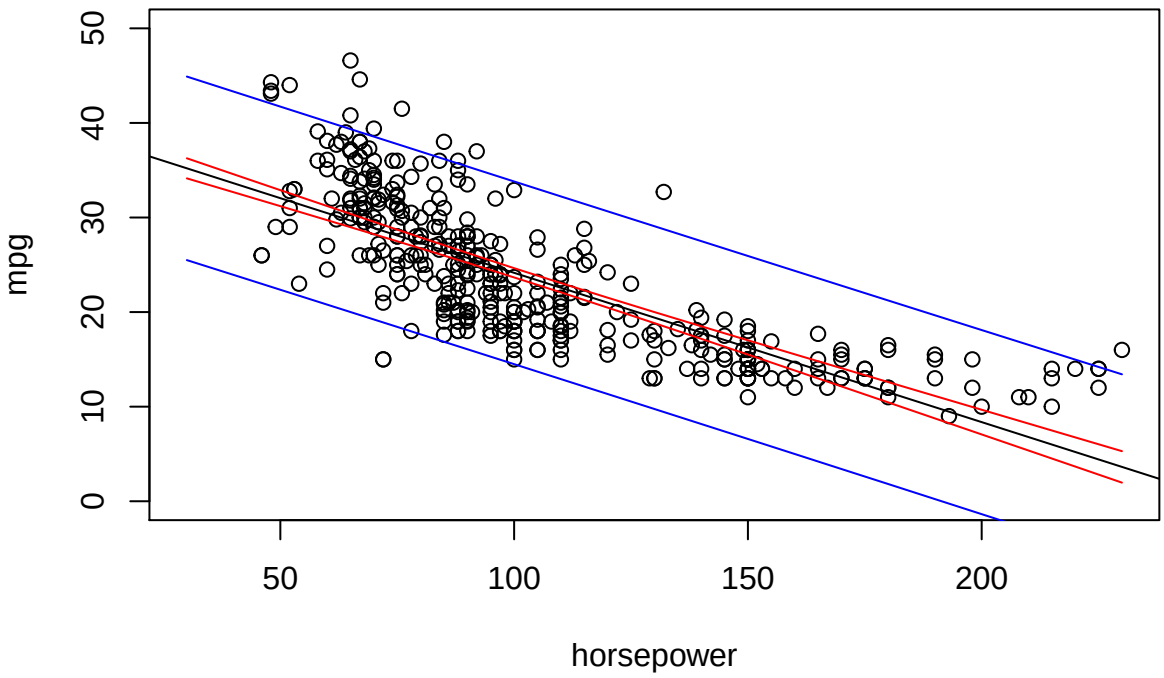
```
plot(Auto$horsepower, Auto$mpg, xlab = "horsepower", ylab = "mpg")
abline(lm.fit)
```



(c) Plot the 95% confidence interval and prediction interval in the same plot as (b) using different colours and legends.

Solution:

```
plot(Auto$horsepower, Auto$mpg,
     xlab = "horsepower", ylab = "mpg",
     ylim=c(0, 50), xlim=c(30, 230))
abline(lm.fit)
nd <- data.frame(horsepower=seq(30, 230, by=10))
p_conf <- predict(lm.fit,interval="confidence",newdata=nd)
p_pred <- predict(lm.fit,interval="prediction",newdata=nd)
lines(nd$horsepower, p_conf[,"lwr"], col="red", type="l", pch="+")
lines(nd$horsepower, p_conf[,"upr"], col="red", type="l", pch="+")
lines(nd$horsepower, p_pred[,"upr"], col="blue", type="l", pch="*")
lines(nd$horsepower, p_pred[,"lwr"], col="blue", type="l", pch="*")
```



5. Logistic regression (10% | 20%)

Marking scheme: (1) create crim01 and add it to data set (2) split dataset into train and testing sets (3) build model (4) calculate test error rate (5) try several models 2% | 4% each.

Using the Boston data set, fit classification models in order to predict whether a given suburb has a crime rate above or below the median. Explore logistic regression models using various subsets of the predictors. Describe your findings.

Solution:

```
library(MASS)
data("Boston")
n <- dim(Boston)[1]

# Introduce a variable whether or not the crime rate is above=1 / below=0 the median
Boston$crim01 <- rep(0, n) #add a column in Boston in this way
#or you can use Boston <- data.frame(Boston, crim01)
Boston$crim01[Boston$crim >= median(Boston$crim)] <- 1
Boston$crim <- NULL #remove the column of crim in this way
```

```r
# Look to see what features are most strongly correlated with crim01:
#
Boston.cor = cor(Boston)
print(sort(Boston.cor[,'crim01']))
```

```
##         dis          zn       black        medv          rm        chas
## -0.61634164 -0.43615103 -0.35121093 -0.26301673 -0.15637178  0.07009677
##     ptratio       lstat       indus         tax         age         rad
##  0.25356836  0.45326273  0.60326017  0.60874128  0.61393992  0.61978625
##         nox      crim01
##  0.72323480  1.00000000
```

```r
# Split the data set into testing and training parts:
set.seed(10)
train <- sample(1:n,0.7*n)  #70% data for training, 30% for testing.
#You may also choose half/half.

Boston.train <- Boston[train,]
Boston.test <- Boston[-train,]

# Fit several models to the training data
#
# Define a function to calculate and print the error rate given a glm model
printErrorRate <- function(glm.model){
  crim01_pred <- predict(glm.model,
                         newdata=Boston.test, type="response")
  yhat <- rep(0, length(crim01_pred))
  yhat[crim01_pred > 0.5] <- 1
  CM <- table(predicted=yhat, truth=Boston.test$crim01)
  #CM is the confusion matrix
  print(CM)
  print(paste("The error rate is", (CM[1,2] + CM[2,1])/sum(CM)))
}

#model 1
glm_model_1 <- glm(crim01 ~ .,
                   data=Boston.train, family=binomial)
printErrorRate(glm_model_1)
```

```
##          truth
## predicted  0  1
##         0 75  8
##         1  8 61
## [1] "The error rate is 0.105263157894737"
```

```r
#model 2
glm_model_2 <- glm(crim01 ~ nox + rad + dis,
                   data=Boston.train, family=binomial)
printErrorRate(glm_model_2)
```

```
##          truth
## predicted  0  1
##         0 75 16
##         1  8 53
## [1] "The error rate is 0.157894736842105"
```

```r
#model 3
glm_model_3 <- glm(crim01 ~ nox + rad + dis + lstat,
                   data=Boston.train, family=binomial)
printErrorRate(glm_model_3)
```

```
##           truth
## predicted  0  1
##         0 75 16
##         1  8 53
## [1] "The error rate is 0.157894736842105"
```

```r
#model 4
glm_model_4 <- glm(crim01 ~ nox,
                   data=Boston.train, family=binomial)
printErrorRate(glm_model_4)
```

```
##           truth
## predicted  0  1
##         0 68  6
##         1 15 63
## [1] "The error rate is 0.138157894736842"
```

6. Resampling methods (20% | 0%)

Marking scheme: (MSc students only) Point which method to use (Either method will do). (5%) Describe how the method works in this problem. (10%) Describe how to derive the SD (It's not enough to end at SE or variance.)(5%)

Suppose that we use some statistical learning method to make a prediction for the response Y for a particular value of the predictor X. Carefully describe how we might estimate the standard deviation of our prediction. (You don't need to give a concrete example.)

Solution:

We could apply a LOOCV type technique (less computationally intensive methods could be devised) so that we would have $n-1$ estimates of Y. The variance of these Y could be used to estimate the variance of our prediction. The standard deviation is the square root of the variance.

Or alternatively, we may estimate the standard deviation of our prediction by using the bootstrap method. In this case, rather than obtaining new independant data sets from the population and fitting our model on those data sets, we instead obtain repeated random samples from the original data set. In this case, we perform sampling with replacement B times and then find the corresponding estimates and the standard error of those B estimates by using equation (5.8) in the book. The standard deviation (SD) can be calculated from the standard error (SE) by the following formula:

$$SD = SE \times \sqrt{N}$$

7. Resampling methods (20% | 20%)

Marking scheme: (c)(d) 4% each, the rest 3% each.

We will now perform cross-validation on a simulated data set.

(a) Generate a simulated data set as follows:

```
set.seed(500)
y = rnorm(500)
x = 4 - rnorm(500)
y = x - 2*x^2 + 3*x^4 + rnorm(500)
```

In this data set, what is n and what is p? Write out the model used to generate the data in equation form.

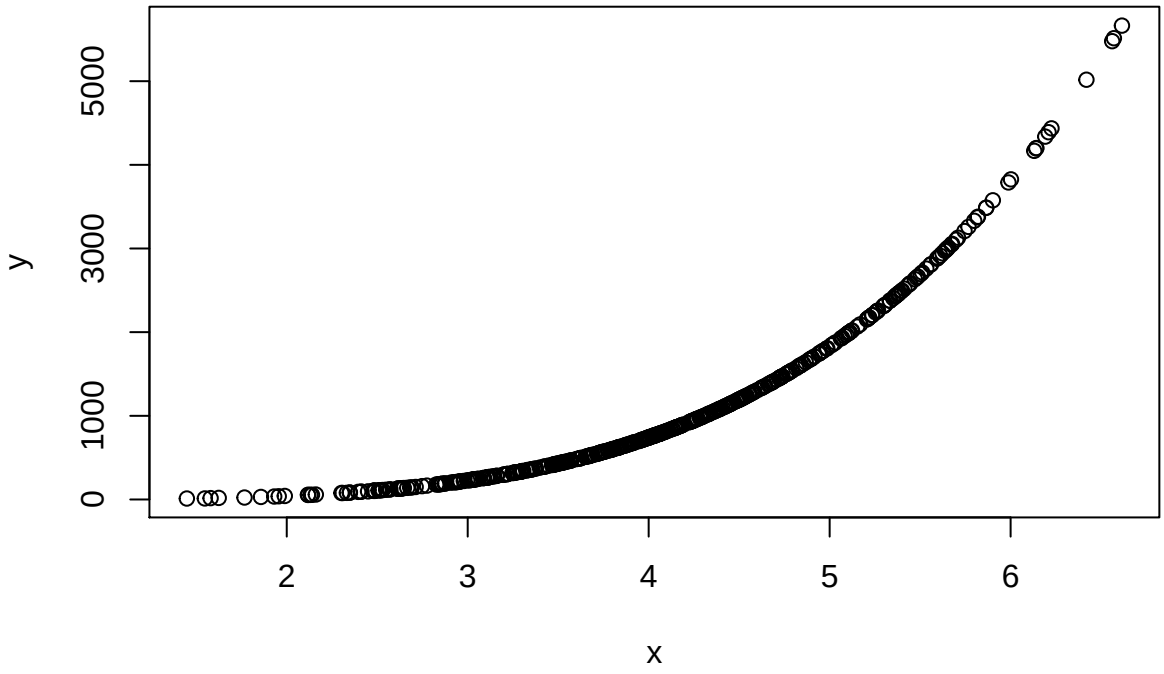
Solution: $n = 500$ and $p = 3$, assuming the predictors to be x, x^2 and x^4. The model is

$$Y = X - 2X^2 + 3X^4 + \varepsilon$$

(b) Create a scatterplot of X against Y. Comment on what you find.

Solution:

```
set.seed(500)
y = rnorm(500)
x = 4 - rnorm(500)
y = x - 2*x^2 + 3*x^4 + rnorm(500)
plot(x,y)
```



(c) Set the seed to be 23, and then compute the LOOCV and 10-fold CV errors that result from fitting the following four models using least squares:

i. $Y = \beta_0 + \beta_1 X + \varepsilon$
ii. $Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \varepsilon$
iii. $Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \varepsilon$
iv. $Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \beta_4 X^4 + \varepsilon$.

Note you may find it helpful to use the `data.frame()` function to create a single data set containing both X and Y.

Solution:

```
library(boot)
DF <- data.frame(x,y)
set.seed(23)

# Do LOOCV on each model:
#
m_1 <- glm(y ~ x, data=DF)
cv.err <- cv.glm(DF, m_1)
print(paste("Model (1): The LOOCV output is ", cv.err$delta[1]))

m_2 <- glm(y ~ poly(x,2), data=DF)
cv.err <- cv.glm(DF, m_2)
print(paste("Model (2): The LOOCV output is ", cv.err$delta[1]))

m_3 <- glm(y ~ poly(x,3), data=DF)
cv.err <- cv.glm(DF, m_3)
print(paste("Model (3): The LOOCV output is ", cv.err$delta[1]))

m_4 <- glm(y ~ poly(x,4), data=DF)
cv.err <- cv.glm(DF, m_4)
print(paste("Model (4): The LOOCV output is ", cv.err$delta[1]))

# Do 10 fold CV on each model:
#
cv.err <- cv.glm(DF, m_1, K=10)
print(paste("Model (1): The 10 fold CV output is ", cv.err$delta[1]))

cv.err <- cv.glm(DF, m_2, K=10)
print(paste("Model (2): The 10 fold CV output is ", cv.err$delta[1]))

cv.err <- cv.glm(DF, m_3, K=10)
print(paste("Model (3): The 10 fold CV output is ", cv.err$delta[1]))

cv.err <- cv.glm(DF, m_4, K=10)
print(paste("Model (4): The 10 fold CV output is ", cv.err$delta[1]))
```

```
## Warning: package 'boot' was built under R version 3.4.2
## [1] "Model (1): The LOOCV output is  155977.093496369"
## [1] "Model (2): The LOOCV output is  7400.42772222348"
## [1] "Model (3): The LOOCV output is  64.0080549629829"
## [1] "Model (4): The LOOCV output is  0.926181170235929"
## [1] "Model (1): The 10 fold CV output is  155677.162068933"
## [1] "Model (2): The 10 fold CV output is  7382.781237403"
## [1] "Model (3): The 10 fold CV output is  63.7063401237125"
## [1] "Model (4): The 10 fold CV output is  0.939151110146169"
```

(d) Repeat (c) using random seed 46, and report your results. Are your results the same as what you got in (c)? Why?

Solution:

```
library(boot)
DF <- data.frame(x,y)
set.seed(46)
# Do LOOCV on each model:
#
m_1 <- glm(y ~ x, data=DF)
cv.err <- cv.glm(DF, m_1)
print(paste("Model (1): The LOOCV output is ", cv.err$delta[1]))

m_2 <- glm(y ~ poly(x,2), data=DF)
cv.err <- cv.glm(DF, m_2)
print(paste("Model (2): The LOOCV output is ", cv.err$delta[1]))

m_3 <- glm(y ~ poly(x,3), data=DF)
cv.err <- cv.glm(DF, m_3)
print(paste("Model (3): The LOOCV output is ", cv.err$delta[1]))

m_4 <- glm(y ~ poly(x,4), data=DF)
cv.err <- cv.glm(DF, m_4)
print(paste("Model (4): The LOOCV output is ", cv.err$delta[1]))

# Do 10 fold CV on each model:
#
cv.err <- cv.glm(DF, m_1, K=10)
print(paste("Model (1): The 10 fold CV output is ", cv.err$delta[1]))

cv.err <- cv.glm(DF, m_2, K=10)
print(paste("Model (2): The 10 fold CV output is ", cv.err$delta[1]))

cv.err <- cv.glm(DF, m_3, K=10)
print(paste("Model (3): The 10 fold CV output is ", cv.err$delta[1]))

cv.err <- cv.glm(DF, m_4, K=10)
print(paste("Model (4): The 10 fold CV output is ", cv.err$delta[1]))
```

```
## [1] "Model (1): The LOOCV output is  155977.093496369"

## [1] "Model (2): The LOOCV output is  7400.42772222348"

## [1] "Model (3): The LOOCV output is  64.0080549629829"

## [1] "Model (4): The LOOCV output is  0.92618117023593"

## [1] "Model (1): The 10 fold CV output is  154019.656200593"

## [1] "Model (2): The 10 fold CV output is  7417.09827634986"

## [1] "Model (3): The 10 fold CV output is  62.3149478794098"

## [1] "Model (4): The 10 fold CV output is  0.923674066388046"
```

Under two different seeds, the respective LOOCV errors are the same, but the 10-fold CV errors are different.

(e) Which of the models in (c) had the smallest LOOCV and 10-fold CV error? Is this what you expected? Explain your answer.

Solution:

The quartic (degree 4) model has the smallest CV error, which is expected as the data was generated from a quartic model.

(f) Comment on the statistical significance of the coefficient estimates that results from fitting each of the models in (c) using least squares. Do these results agree with the conclusions drawn based on the cross-validation results?

Solution:

```
summary(m_4)
```

```
##
## Call:
## glm(formula = y ~ poly(x, 4), data = DF)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.88395  -0.61614   0.02642   0.64776   2.70284
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.071e+03  4.267e-02 25086.5   <2e-16 ***
## poly(x, 4)1 2.003e+04  9.542e-01 20990.7   <2e-16 ***
## poly(x, 4)2 8.547e+03  9.542e-01  8956.9   <2e-16 ***
## poly(x, 4)3 1.857e+03  9.542e-01  1945.8   <2e-16 ***
## poly(x, 4)4 1.664e+02  9.542e-01   174.3   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.9105089)
##
##     Null deviance: 4.777e+08  on 499  degrees of freedom
## Residual deviance: 4.507e+02  on 495  degrees of freedom
## AIC: 1379
##
## Number of Fisher Scoring iterations: 2
```

The p values of the quartic model coefficients are all very small. All the four degree terms are statistically significant. This strongly agrees with the CV results that it is a quartic model.