Homework 3 - MATH 4322

Instructions

1. Due date: September 28, 2023
2. Scan or Type your answers and submit only one file. (If you submit several files only the recent one uploaded will be graded).
3. Preferably save your file as PDF before uploading. Submit in Canvas under Homework 3.
4. These questions are from *An Introduction to Statistical Learning with Applications in R* by James, et. al., chapter 4.

Problem 1

Suppose we collect data for a group of students in a statistics class with variables X_1 = hours studied, X_2 =undergrad GPA, and Y = receive an A. We fit a logistic regression and produce estimated coefficient, $\hat{\beta}_0 = -6$, $\hat{\beta}_1 = 0.05$, $\hat{\beta}_2 = 1$.

(a) Estimate the probability that a student who studies for 40 h and has an undergrad GPA of 3.5 gets an A in the class.

```r
b0 = -6
b1 = 0.05
b2 = 1
h = 40
GPA = 3.5
pHatX = exp(b0 + b1 * h + GPA) / (1 + exp(b0 + b1 * h + GPA))
pHatX
```

```
[1] 0.3775407
```

(b) How many hours would the student in part (a) need to study to have a 50% chance of getting an A in the class?

```r
pX = 0.5
newH = (log(pX / (1 - pX)) - (b0 + GPA)) / 0.05
newH
```

```
[1] 50
```

Problem 2

In this problem, you will develop a model to predict whether a given car gets high or low gas mileage based on the `Auto` data set in the `ISLR` package.

(a) Create a binary variable, `mpg01`, that contains a 1 if mpg contains a value above its median, and a 0 if mpg contains a value below its median. You can compute the median using the `median()` function. Note you may find it helpful to use the `data.frame()` function to create a single data set containing both `mpg01` and the other Auto variables.

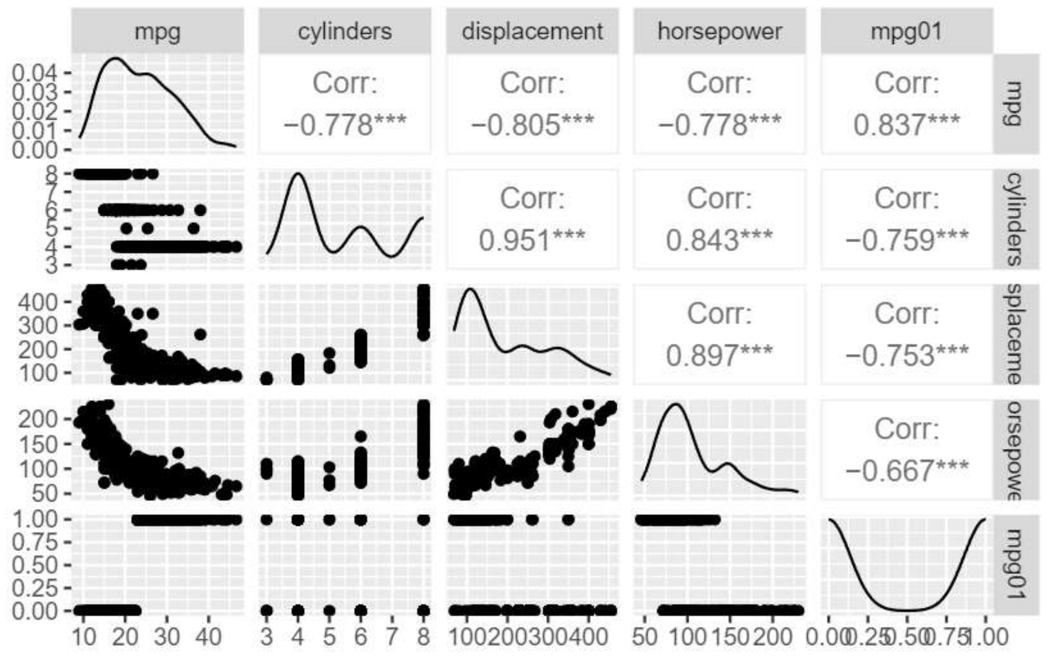
```r
library(ISLR)
Auto = Auto
medianMpg = median(Auto$mpg)
Auto$mpg01 = ifelse(Auto$mpg >= medianMpg, 1, 0)
```

(b) Explore the data graphically in order to investigate the association between `mpg01` and the other features. Which of the other features seem most likely to be useful in predicting `mpg01`? Scatterplots and boxplots may be useful tools to answer this question. Describe your findings.

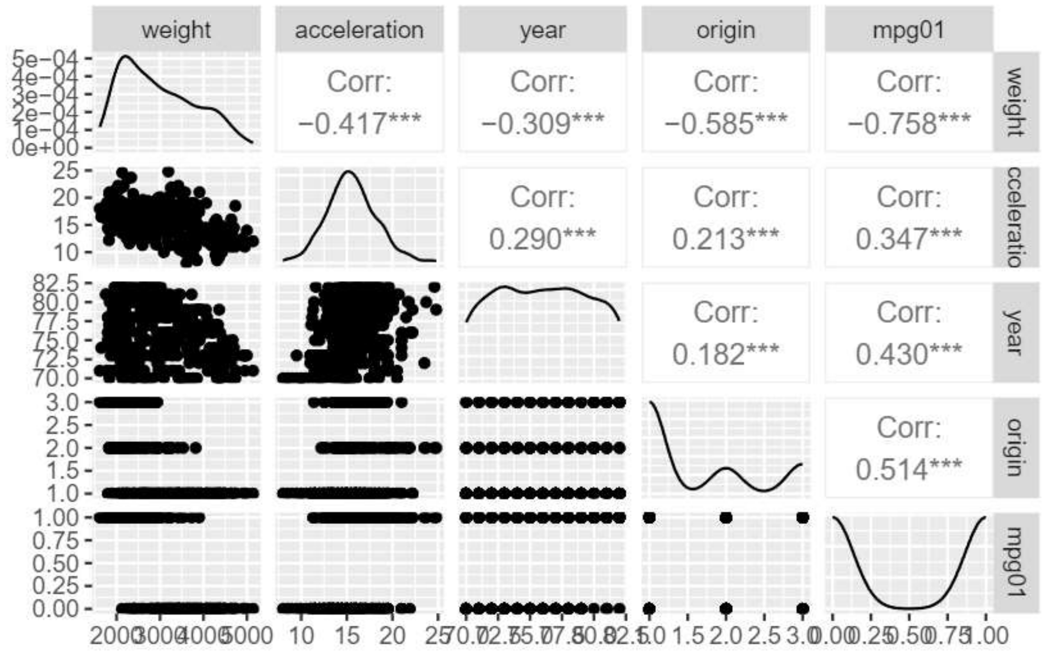
```r
library(ggplot2)
library(GGally)
```

```
Registered S3 method overwritten by 'GGally':
  method from
  +.gg   ggplot2
```

```r
ggpairs(Auto[,c(1:4, 10)])
```



```
ggpairs(Auto[,c(5:8, 10)])
```



Answer: Looking at the scatterplots and boxplots it looks like displacement, horsepower, and weight are associated with mpg01.

(c) Split the data into a training set and a test set.

```r
set.seed(101)
sample = sample.int(nrow(Auto), floor(.75*nrow(Auto)), FALSE)
train = Auto[sample,]
test = Auto[-sample,]
```

(d) Perform logistic regression on the training data in order to predict `mpg01` using the variables that seemed most associated with `mpg01` in (b). What is the test error of the model obtained? That is use the test data to predict and get the confusion matrix and determine the error rate.

```r
autoGLM = glm(mpg01 ~ (displacement + horsepower + weight), data = train, family = "b
summary(autoGLM)
```

```

Call:
glm(formula = mpg01 ~ (displacement + horsepower + weight), family = "binomial",
    data = train)

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)  11.4729989  1.7706588   6.480  9.2e-11 ***
displacement -0.0122141  0.0060840  -2.008 0.044689 *
horsepower   -0.0545029  0.0158444  -3.440 0.000582 ***
weight       -0.0014942  0.0007717  -1.936 0.052839 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 407.52  on 293  degrees of freedom
Residual deviance: 155.56  on 290  degrees of freedom
AIC: 163.56

Number of Fisher Scoring iterations: 7
```

```r
# confustion matrix
glmPrediction = predict(autoGLM, test, type = "response")
```

```
yHat = glmPrediction > 0.5
table(test$mpg01, yHat)
```

```
   yHat
    FALSE TRUE
  0    46    5
  1     5   42
```

```
errorRate = (5+5)/(5+5+46+42)
errorRate
```

```
[1] 0.1020408
```

Answer: Test error rate is 0.1020408

Problem 3

This problem involves writing functions.

(a) Write a function, `Power()`, that prints out the result of raising 2 to the 3rd power. In other words, your function should compute 2^3 and print out the results.

```
Power <- function(x) {print(x^3)}
Power(2)
```

```
[1] 8
```

Hint: Recall that x^a raises x to the power a. Use the `print()` function to output the result.

(b) Create a new function, `Power2()`, that allows you to pass any two numbers, x and a, and prints out the value of x^a. You can do this by beginning your function with the line

`Power2 <- function(x, a) {`

You should be able to call your function by entering, for instance,

`Power2(3, 8)`

on the command line. This should output the value of 3^8, namely, 6,561.

```
Power2 <- function(x, a) {print(x^a)}
Power2(3, 8)
```

```
[1] 6561
```

(c) Using the `Power2()` function that you just wrote, compute 10^3, 8^{17}, and 131^3.

```r
Power2(10, 3)
```

```
[1] 1000
```

```r
Power2(8, 17)
```

```
[1] 2.2518e+15
```

```r
Power2(131, 3)
```

```
[1] 2248091
```

(d) Now create a new function, `Power3()`, that actually returns the result x^a as an `R` object, rather than simply printing it to the screen. That is, if you store the value x^a in an object called `result` within your function, then you can simply `return()` this result, using the following line:

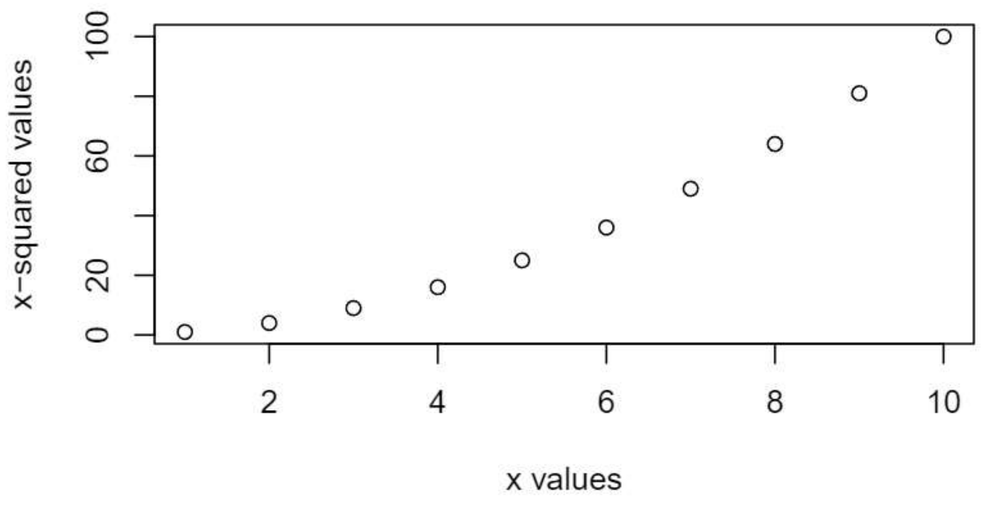
```
return(result)
```

The line above should be the last line in your function, before the } symbol.

```r
Power3 <- function(x, a) {
  result = x^a
  return(result)
}
```

(e) Now using the `Power3()` function, create a plot of $f(x) = x^2$. The x-axis should display a range of integers from 1 to 10, and the y-axis should display x^2. Label the axes appropriately, and use an appropriate title for the figure.

```r
plot(1:10, Power3(1:10, 2), xlab = "x values", ylab = "x-squared values")
```

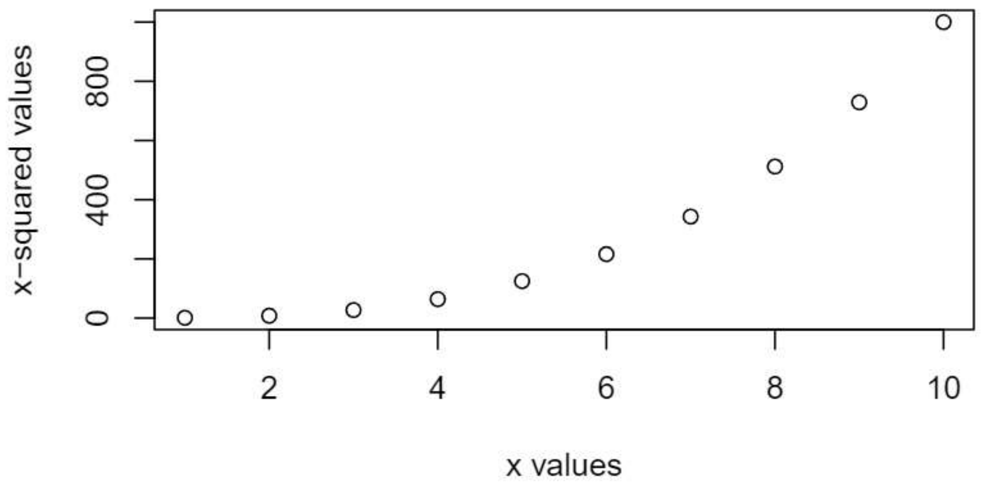


(f) Create a function, `PlotPower()`, that allows you to create a plot of x against x^a for a fixed a and for a range of values of x. For instance, if you call

```
PlotPower(1:10, 3)
```

then a plot should be created with an x-axis taking on values $1, 2, \dots, 10$, and a y-axis taking on values $1^3, 2^3, \dots, 10^3$.

```
PlotPower <- function(x, a) {
  plot(x, Power3(x,a), xlab = "x values", ylab = "x-squared values")
}
PlotPower(1:10, 3)
```


x−squared values
0
400
800
2
4
6
8
10
x values