CIND 123 Summer 2019 - Assignment #3

Tasdeed Aziz

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

Use RStudio for this assignment. Edit the file `A3-S19-Q` and insert your R code where wherever you see the string "INSERT YOUR ANSWER HERE"

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document.

Sample Question and Solution

Use `seq()` to create the vector $(2, 4, 6, \ldots, 20)$.

```
#Insert your code here.
seq(2,20,by = 2)
```

```
##  [1]  2  4  6  8 10 12 14 16 18 20
```

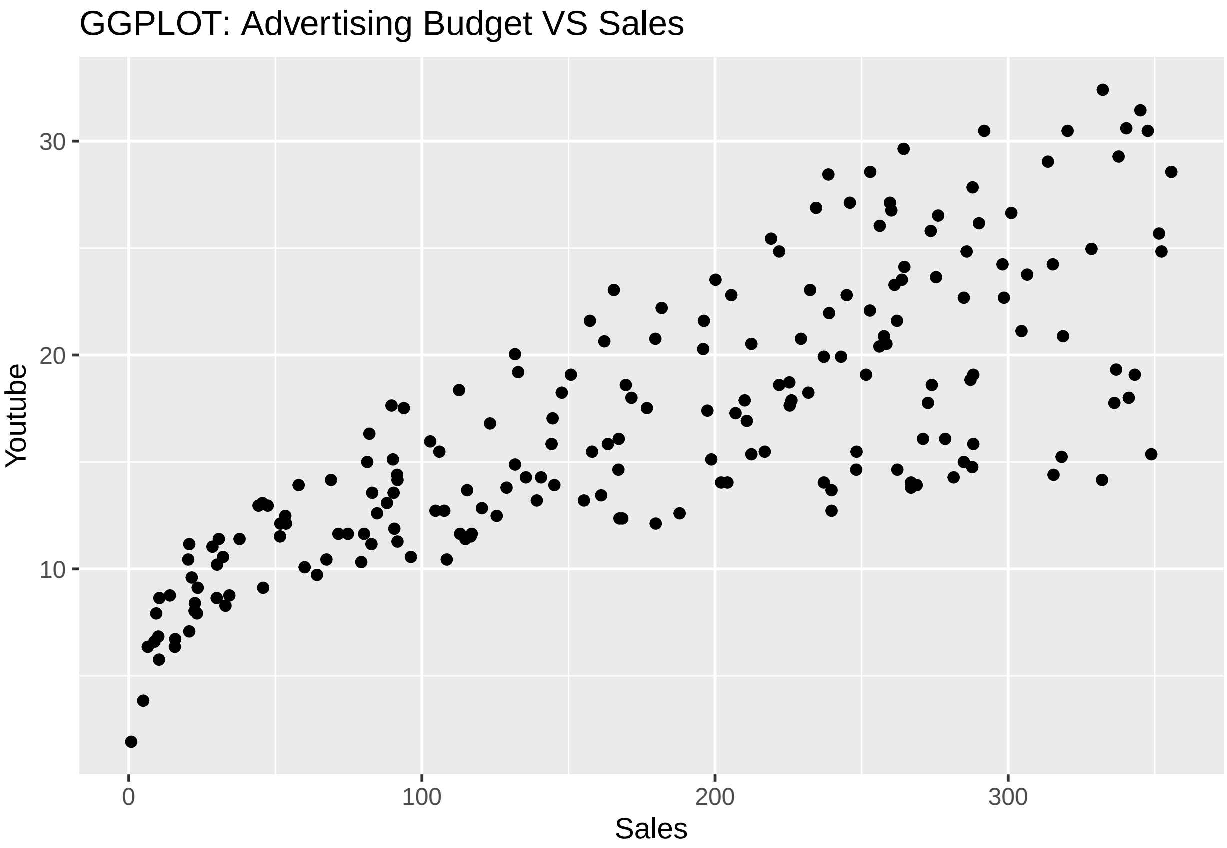
##Question 1

Install the `marketing` dataset on your computer using the command `install.packages("datarium")`. Then load the `datarium` package into your session using the following command. Understand the dataset by using `??marketing` command.

```
library(datarium)
attach(marketing)
```

a) Plot the advertising budget of `Youtube` against `Sales`. Comment on their relationship. Hint: You may use the `ggplot()` function from `ggplot2` package.

```
library(ggplot2)
ggplot(data = marketing,mapping = aes(x = youtube, y = sales)) + geom_point() + labs( x = "Sales", y =
```



b) Find the correlation between advertising budget of `Youtube` against `Sales`. Comment on the output. Does it match your intuition from part (a).

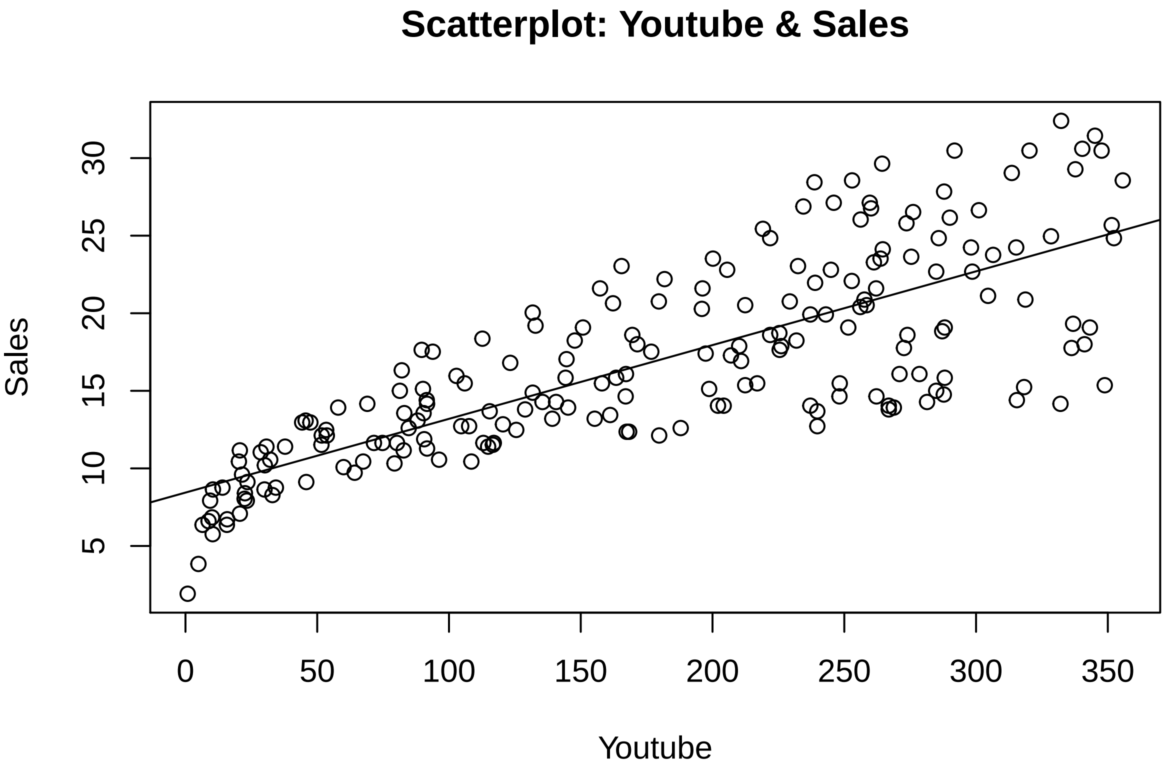
```
c = cor( x = marketing$sales, y = marketing$youtube)
print(c)
```

```
## [1] 0.7822244
```

```
#The correlation is close to positive 1 which means the variables have high positive correlation.
```

c) Plot the `Sales` as a function of `Youtube` variable using a scatterplot, and graph the least-square line on the same plot.

```
plot( x = youtube, y= sales, xlab = 'Youtube',ylab = 'Sales', main = 'Scatterplot: Youtube & Sales')
abline(lm(sales~youtube))
```



d) Use the regression line to predict the `Sales` amount when `Youtube` budget is `$69K`.

```
lm(sales~youtube)
```

```
##
## Call:
## lm(formula = sales ~ youtube)
##
## Coefficients:
## (Intercept)      youtube
##     8.43911      0.04754
```

```
x = 69
predict = 8.43911 + 0.04754*x
cat(predict,'K')
```

```
## 11.71937 K
```

e) Use `youtube` and `facebook` variables to build a linear regression model to predict `sales` Display a summary of your model indicating Residuals, Coefficients, . . . , etc. What conclusion can you draw from this summary?

```
regression = lm(sales~youtube+facebook)
regression
```

```
##
```

```
## Call:
## lm(formula = sales ~ youtube + facebook)
##
## Coefficients:
## (Intercept)      youtube     facebook
##     3.50532      0.04575      0.18799
```

```
summary(regression)
```

```
##
## Call:
## lm(formula = sales ~ youtube + facebook)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.5572  -1.0502   0.2906   1.4049   3.3994
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.50532    0.35339   9.919   <2e-16 ***
## youtube      0.04575    0.00139  32.909   <2e-16 ***
## facebook     0.18799    0.00804  23.382   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.018 on 197 degrees of freedom
## Multiple R-squared:  0.8972, Adjusted R-squared:  0.8962
## F-statistic: 859.6 on 2 and 197 DF,  p-value: < 2.2e-16
```

```
#Residuals tells the distribution is symmetric.
# y = 3.50532 + 0.04575(youtube) + 0.18799(facebook)
#R-square : 0.8972 (distance between the variables)
```

f) Use the regression line to predict the `Sales` amount when `youtube` budget is `$69K` and `facebook` is `$39.36K`.

```
a = 3.50532
b = 0.04575
c = 0.18799
x = 69
y = 39.36

Sales_amount = a + b*x + c*y
cat(Sales_amount,'K')
```

```
## 14.06136 K
```

g) What is the difference between the output in (f) and the output in (d)

```
#Output (d) gives prediction of sales only for youtube whereas output(f) predict the sales for two vari
#facebook and youtube.
```

h) Display the correlation matrix of the variables: `youtube`, `facebook`, `newspaper` and `sales`. What conclusion can you draw?

```
correlation = cor(marketing, method = c("pearson", "kendall", "spearman"))
print(correlation)
```

```
##              youtube   facebook  newspaper     sales
## youtube   1.00000000 0.05480866 0.05664787 0.7822244
## facebook  0.05480866 1.00000000 0.35410375 0.5762226
## newspaper 0.05664787 0.35410375 1.00000000 0.2282990
## sales     0.78222442 0.57622257 0.22829903 1.0000000
```

```
#The corelation matrix is symmetric, variables are positively corelated.
```

i) In your opinion, which statistical test should be used to discuss the relationship between `youtube` and `sales`? Hint: Review the differnce between Pearson and Spearman tests.

```
#Pearson should be used as the variables are continous.
```

Question 2

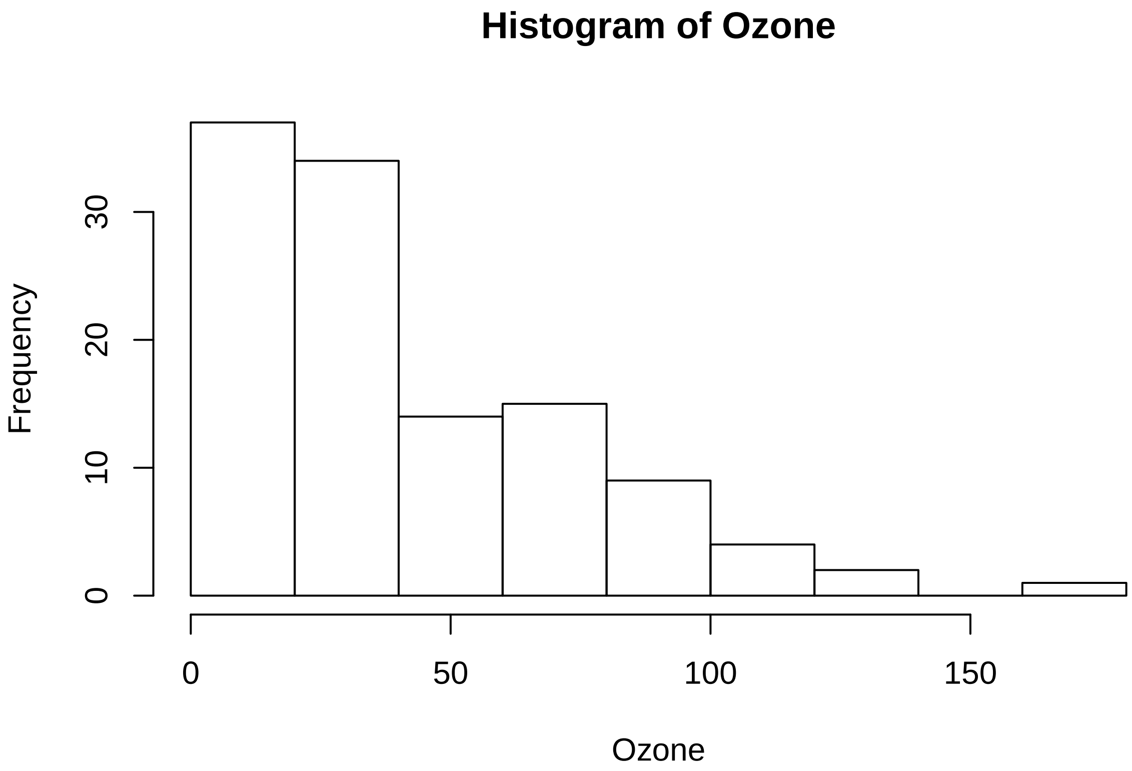
This question makes use of package "ISwR". Please load `airquality` dataset as following:

```
#install.packages("ISwR")
library(ISwR)
data(airquality)
str(airquality)
```

```
## 'data.frame':    153 obs. of  6 variables:
##  $ Ozone  : int  41 36 12 18 NA 28 23 19 8 NA ...
##  $ Solar.R: int  190 118 149 313 NA NA 299 99 19 194 ...
##  $ Wind   : num  7.4 8 12.6 11.5 14.3 14.9 8.6 13.8 20.1 8.6 ...
##  $ Temp   : int  67 72 74 62 56 66 65 59 61 69 ...
##  $ Month  : int  5 5 5 5 5 5 5 5 5 5 ...
##  $ Day    : int  1 2 3 4 5 6 7 8 9 10 ...
```

a) Use a histogram to assess the normality of the `Ozone` variable, then explain why it does not appear normally distributed.

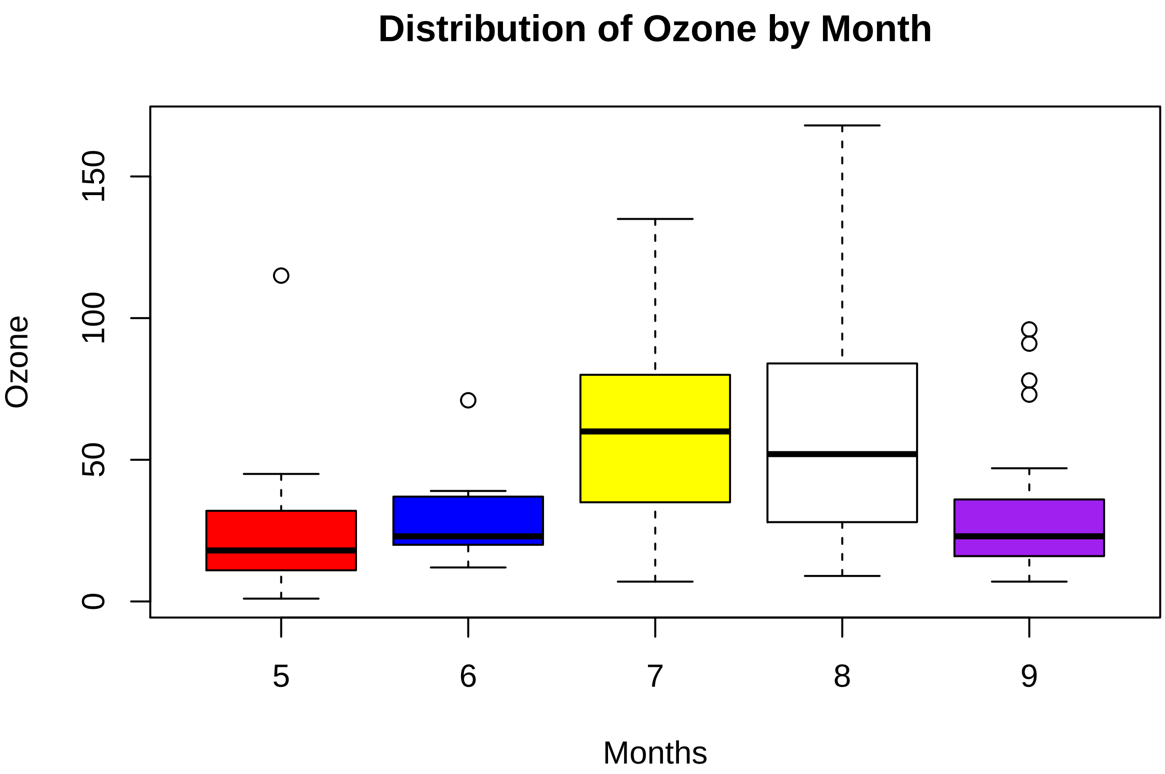
```
hist(airquality$Ozone, xlab = 'Ozone', main = 'Histogram of Ozone' )
```



```
#Due to the presense of outliers the histogram does not look symmetrical, thus not normally distributed
```

b) Create a boxplot that shows the distribution of `Ozone` in each month. Use different colors for each month.

```
boxplot(airquality$Ozone~airquality$Month, col = c('Red','Blue','Yellow','White','Purple'),
        ylab = 'Ozone',xlab = 'Months', main = 'Distribution of Ozone by Month')
```



Question 3

π appears in the formula for the standard normal distribution, the most important probability distribution in statistics. Why not give it a try to calculate π using statistics! In fact, you'll use a simulation technique called the *Monte Carlo Method*.

Recall that the area of a circle of radius r is $A = \pi r^2$. Therefore the area of a circle of radius 1, aka a *unit circle*, is π. You'll compute an approximation to the area of this circle using the Monte Carlo Method.

a) The Monte Carlo Method uses random numbers to simulate some process. Here the process is throwing darts at a square. Assume the darts are uniformly distributed over the square. Imagine a unit circle enclosed by a square whose sides are of length 2. Set an R variable `area.square` to be the area of a square whose sides are of length 2.

```
area.square = 2^2
```

b) The points of the square can be given x-y coordinates. Let both x and y range from -1 to +1 so that the square is centred on the origin of the coordinate system. Throw some darts at the square by generating random numeric vectors x and y, each of length `N = 10,000`. Set R variables `x` and `y` each to be uniformly distributed random numbers in the range -1 to +1. (hint: runif() generates random number for the uniform distribution)

```
x = runif(n=10000, min = -1, max = 1)
y = runif(n=10000, min = -1, max = 1)
```

c) Now count how many darts landed inside the unit circle. Recall that a point is inside the unit circle when $x^2 + y^2 < 1$. Save the result of successful hits in a variable named hit. (hint: a for loop over the length of x and y is one option to reach hit)

```
hit = 0

for (i in 1:length(x)) if (x[i]^2 + y[i]^2 < 1) hit = hit + 1
print(hit)
```

```
## [1] 7859
```

d) The probability that a dart hits inside the circle is proportional to the ratio of the area of the circle to the area of the square. Use this fact to calculate an approximation to π and print the result

```
proportional = hit/length(x)
print(proportional)
```

```
## [1] 0.7859
```

```
approximation = proportional * area.square
print(approximation)
```

```
## [1] 3.1436
```

You got the first estimate for `pi` π, congratulations you have completed the first run of the Monte Carlo simulation. If there is further interest put all the above logic in a function, and call it 50 times store the results in a vector called pi then take the mean of pi vector.