**Introduction to Data Science**

**ISCG 8026**

**Semester 1, 2020**

****

**Lab Exercise-Part B**

**Priya Payyadi Surendran**

**ID:1533656**

**Supervisor: Miss Natt Nuntalid**

**Question 1**

Develop a R Script file that can perform the following tasks on the data set given in “Data Set 1a.csv”:

1. Omitting all the transactions made before 1 Jan 2010

|  |
| --- |
| # clean the raw data  rm(list=ls())  cat("\14")  #install.packages('readr')  library(readr)  # Read the dataset  Sales <- read.csv("Data Set 1a.csv",na.strings = "")  #function to convert word-number to int  wordtonum <- function(word){  num <- list(zero=0, one=1, two=2, three=3, four=4, five=5,  six=6, seven=7, eight=8, nine=9) #print(word)  for(i in word) {  if(!is.null(num[[i]])) {  word[word==i]<-num[[i]]  } }    return(word) }  # change quantity column to numeric type  Sales$Quantity <- wordtonum(Sales$Quantity)  # set as numeric for quantity  Sales$Quantity = as.numeric(Sales$Quantity)  # change spelling for products  levels(Sales$Product) <- c(levels(Sales$Product),"Galaxy Edge S8")  Sales$Product[Sales$Product == "Galaxi Edge S8"] <-"Galaxy Edge S8"  # Correcting fonststyle of customer name  levels(Sales$Name) <- c(levels(Sales$Name), "Alice")  Sales$Name[Sales$Name == "ALice"] <- "Alice"  #omitting data with missing customer name  Sales<-na.omit(Sales)  #install.packages('lubridate')  library(lubridate) # for date  # date format clean up  Sales$Date = dmy(Sales$Date)  # Removing Anomalous Data  levels(Sales$Date) <- c(levels(Sales$Date),NA)  Sales$Date[Sales$Date == "1978-09-02"] <-NA  Sales<-na.omit(Sales)  # function for forex rate calculation  ForexRate<- function(arg1){  if (arg1=='NZD') {out<-1}  if (arg1=='AUD') {out<-1.2}  return(out)  }  #read csv for unit price for products  Products <- read.csv("Data Set 1b.csv")  # combine Sales table and Products by product  MergeTable = merge(Sales,Products,"Product")  # apply the forex rates.  MergeTable["Exchange.Rate"]<-sapply(MergeTable$Currency,ForexRate)  MergeTable$Unit.Price.NZD<-MergeTable$Unit.Price\*MergeTable$Exchange.Rate  MergeTable = na.omit(MergeTable)  # Step 1. Omitting all the Sales made before 1 Jan 2010  Final Sales = subset(MergeTable,MergeTable$Date >'2010-01-01') |

1. Calculating monthly total sale

|  |
| --- |
| #install.packages('zoo')  library(zoo)  #install.packages('plyr')  library(plyr)  # Step 2. Calculating monthly total sale  FinalSales$Month\_Year=as.yearmon(FinalSales$Date)  MonthYearColumns = c('Month\_Year')  QuantityPriceColumns = c('Quantity','Unit.Price.NZD')  MonthlySales=head(ddply(FinalSales,MonthYearColumns,function(x) colSums(x[QuantityPriceColumns]))) |

3. Visualizing the results (monthly total sale) by using appropriate charts/plots.

|  |
| --- |
| #install.packages('plotly')  library(plotly)  packageVersion('plotly')  # Step 3. Visualizing the results (monthly total sale) by using appropriate charts/plots.  VisualData=data.frame(MonthlySales)  barplot(VisualData$Quantity, main="Monthly Total Sales", xlab="Month & Year",names.arg=c(VisualData$Month\_Year), ylab = "Quantities Sold") |

A screenshot of a cell phone

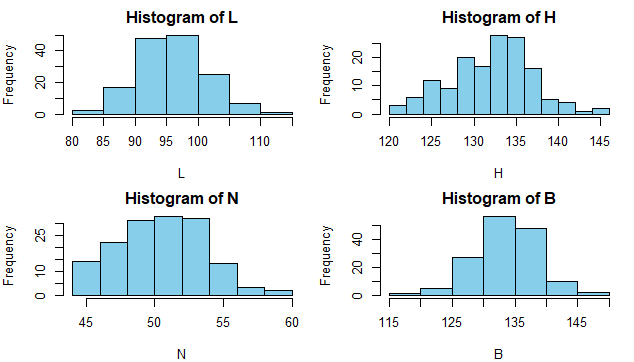
Description automatically generated

**Question 2**

Develop a R Script file that can perform the following tasks on the Skulls data set given in “Data Set 3.csv”.

1. Plot the histograms of L, H, N, B in one page.

|  |
| --- |
| rm(list=ls())  cat("\14")  MainData<- read.csv("Data Set 3.csv")  #1 Plot the histograms of L, H, N, B in one page.  par(mfrow = c(2,2), mar=c(4, 4, 2,1))  hist( MainData$L,col = "Sky Blue", xlab = "L", main = "Histogram of L")  hist( MainData$H,col = "Sky Blue", xlab = "H", main = "Histogram of H")  hist( MainData$N,col = "Sky Blue", xlab = "N", main = "Histogram of N")  hist( MainData$B,col = "Sky Blue", xlab = "B", main = "Histogram of B") |



2. Plot the density functions of L, H, N, B in one page

|  |
| --- |
| #2 Plot the density functions of L, H, N, B in one page  par(mfrow = c(2,2), mar=c(4, 4, 2,1))  plot(density(MainData$L), xlab = "N= 150 Bandwidth= 1.726" , main= "Density function of L")  plot(density(MainData$H), xlab = "N= 150 Bandwidth= 1.632" , main= "Density function of H")  plot(density(MainData$N), xlab = "N= 150 Bandwidth= 0.9862" , main="Density function of N")  plot(density(MainData$B), xlab = "N= 150 Bandwidth= 1.479" , main= "Density function of B") |

A close up of a map

Description automatically generated

3. Compare the density functions against a normal density function. Comments on the symmetry and sharpness of the density functions.

|  |
| --- |
| #3 Compare the density functions against a normal density function. Comments on the symmetry and sharpness of the density functions. [0.4 Marks]  install.packages("e1071")  library(e1071)  skewness(MainData$L)  skewness(MainData$H)  skewness(MainData$N)  skewness(MainData$B)  kurtosis(MainData$L)  kurtosis(MainData$H)  kurtosis(MainData$N)  kurtosis(MainData$B) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | L | H | N | B |
| Skewness | 0.1380888 | -0.1742758 | 0.08172942 | -0.02808971 |
| Kurtosis | 0.1432677 | -0.1485636 | -0.2068898 | 0.2162337 |

4. Create the boxplots of L, H, N and B using a similar scale.

|  |
| --- |
| #4 Create the boxplots of L, H, N and B using a similar scale.  par(mfrow = c(1,4), mar=c(4, 4, 2,1))  boxplot(MainData$L,main = "L")  boxplot(MainData$H,main = "H")  boxplot(MainData$N,main = "N")  boxplot(MainData$B,main = "B") |

A screenshot of a cell phone

Description automatically generated

5. Calculate the mean, variance and standard deviation of L, H, N and B and complete the following table.

|  |
| --- |
| #5 Calculate the mean, variance and standard deviation of L, H, N and B  mean(MainData$L)  mean(MainData$H)  mean(MainData$N)  mean(MainData$B)  var(MainData$L)  var(MainData$H)  var(MainData$N)  var(MainData$B)  sd(MainData$L)  sd(MainData$H)  sd(MainData$N)  sd(MainData$B) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | L | H | N | B |
| Mean | 96.46 | 132.5467 | 50.93333 | 133.9733 |
| Variance | 28.92121 | 24.39714 | 10.29083 | 23.91875 |
| Standard Deviation | 5.377844 | 4.939346 | 3.207932 | 4.89068 |

**Question 3**

Develop a R Script file that can perform the following tasks on the Iris data set given in “Data Set 4.csv”.

1. Create a scatter plot for petal length and width variables.

|  |
| --- |
| # 1. Create a scatter plot for petal length and width variables.  rm(list=ls())  cat("\14")  IrisData <- read.csv("Data Set 4.csv")  plot(IrisData$Petal.Length, IrisData$Petal.Width, xlab="Petal length", ylab="Petal width") |

2. Calculate a liner model between petal length and width and show it in the scatter plot.

|  |
| --- |
| #2 Calculate a liner flower between petal length and width and show it in the scatter plot.  flower<-lm(Petal.Width ~ Petal.Length, data=IrisData)  lines(IrisData$Petal.Length,flower$fitted.values) |

A close up of a map

Description automatically generated

3. Based on the Species data, subdivide the iris dataset into three separate subsets (for each species).

|  |
| --- |
| #3. Based on the Species data, subdivide the iris dataset into three separate subsets (for each species).  setosa <- subset(IrisData, Species == "setosa")  versicolor <- subset(IrisData, Species == "versicolor")  virginica <- subset(IrisData, Species == "virginica") |

4. Repeat steps 1 and 2 for each subset.

|  |
| --- |
| #4. Repeat steps 1 and 2 for each subset  ##scatter plot for species setosa  plot(setosa$Petal.Length, setosa$Petal.Width, xlab="Petal length of setosa", ylab="Petal width of setosa")  setosaflower<-lm(Petal.Width ~ Petal.Length, data=setosa)  lines(setosa$Petal.Length,setosaflower$fitted.values, col= "black")  ##scatter plot for species versicolor  plot(versicolor$Petal.Length, versicolor$Petal.Width, xlab="Petal length of versicolor", ylab="Petal width of versicolor")  versiflower<-lm(Petal.Width ~ Petal.Length, data=versicolor)  lines(versicolor$Petal.Length,versiflower$fitted.values,col= "blue")  ##scatter plot for species virginica  plot(virginica$Petal.Length, virginica$Petal.Width, xlab="Petal length of virginica", ylab="Petal width of virginica")  virginicaflower<-lm(Petal.Width ~ Petal.Length, data=virginica)  lines(virginica$Petal.Length,virginicaflower$fitted.values, col= "green") |

A close up of a map

Description automatically generated

5. Plot all the results including scatter plots and linear models in one plot.

|  |
| --- |
| #5. Plot all the results including scatter plots and linear flowers in one plot.  plot(IrisData$Petal.Length, IrisData$Petal.Width, col=c("black","blue","green")[unclass(IrisData$Species)],xlab="Petal length", ylab="Petal width")  flower<-lm(Petal.Width ~ Petal.Length, data=IrisData)  lines(IrisData$Petal.Length,flower$fitted.values, col= "red")  setosaflower<-lm(Petal.Width ~ Petal.Length, data=setosa)  lines(setosa$Petal.Length,setosaflower$fitted.values, col= "black")  versiflower<-lm(Petal.Width ~ Petal.Length, data=versicolor)  lines(versicolor$Petal.Length,versiflower$fitted.values, col= "blue")  virginicaflower<-lm(Petal.Width ~ Petal.Length, data=virginica)  lines(virginica$Petal.Length,virginicaflower$fitted.values, col= "green") |

A close up of a map

Description automatically generated

6. Show that the linear models obtained for the three subsets can describe the relationship between petal length and width more accurately.

|  |
| --- |
| #6. Show that the linear flowers obtained for the three subsets can describe the relationship between petal length and width more accurately.  #Calculating Error  flower<-lm(Petal.Width ~ Petal.Length, data=IrisData)  error = flower$fitted.values - IrisData$Petal.Width  mean(error^2)  setosaflower<-lm(Petal.Width ~ Petal.Length, data=setosa)  error1 = setosaflower$fitted.values - setosa$Petal.Width  mean(error1^2)  versiflower<-lm(Petal.Width ~ Petal.Length, data=versicolor)  error2 = versiflower$fitted.values - versicolor$Petal.Width  mean(error2^2)  virginicaflower<-lm(Petal.Width ~ Petal.Length, data=virginica)  error3 = virginicaflower$fitted.values - virginica$Petal.Width  mean(error3^2) |

|  |
| --- |
| > flower<-lm(Petal.Width ~ Petal.Length, data=IrisData)  > error = flower$fitted.values - IrisData$Petal.Width  > mean(error^2)  [1] 0.04206731  >  > setosaflower<-lm(Petal.Width ~ Petal.Length, data=setosa)  > error1 = setosaflower$fitted.values - setosa$Petal.Width  > mean(error1^2)  [1] 0.009686994  >  > versiflower<-lm(Petal.Width ~ Petal.Length, data=versicolor)  > error2 = versiflower$fitted.values - versicolor$Petal.Width  > mean(error2^2)  [1] 0.01460732  >  > virginicaflower<-lm(Petal.Width ~ Petal.Length, data=virginica)  > error3 = virginicaflower$fitted.values - virginica$Petal.Width  > mean(error3^2)  [1] 0.06625411 |

7. Repeat the above steps for sepal length and sepal width data.

|  |
| --- |
| #7. Repeat the above steps for sepal length and sepal width data.  ##sepal length and width  ##scatter plot for width and length of sepal  plot(IrisData$Sepal.Length, IrisData$Sepal.Width, xlab="Sepal length", ylab="Sepal width")  flower<-lm(Sepal.Width ~ Sepal.Length, data=IrisData)  lines(IrisData$Sepal.Length,flower$fitted.values)  ##scatter plot for species setosa  setosa <- subset(IrisData, Species == "setosa")  plot(setosa$Sepal.Length, setosa$Sepal.Width, xlab="Sepal length of setosa", ylab="Sepal width of setosa")  setosaflower<-lm(Sepal.Width ~ Sepal.Length, data=setosa)  lines(setosa$Sepal.Length,setosaflower$fitted.values, col= "black")  ##scatter plot for species versicolor  versicolor <- subset(IrisData, Species == "versicolor")  plot(versicolor$Sepal.Length, versicolor$Sepal.Width, xlab="Sepal length of versicolor", ylab="Sepal width of versicolor")  versiflower<-lm(Sepal.Width ~ Sepal.Length, data=versicolor)  lines(versicolor$Sepal.Length,versiflower$fitted.values,col= "blue")  ##scatter plot for species virginica  virginica <- subset(IrisData, Species == "virginica")  plot(virginica$Sepal.Length, virginica$Sepal.Width, xlab="Sepal length of virginica ", ylab="Sepal width of virginica")  virginicaflower<-lm(Sepal.Width ~ Sepal.Length, data=virginica)  lines(virginica$Sepal.Length,virginicaflower$fitted.values, col= "green")  ##scatter plot for all the species  plot(IrisData$Sepal.Length, IrisData$Sepal.Width, col=c("black","blue","green")[unclass(IrisData$Species)],xlab="Sepal length", ylab="Sepal width")  flower<-lm(Sepal.Width ~ Sepal.Length, data=IrisData)  lines(IrisData$Sepal.Length,flower$fitted.values, col= "red")  setosaflower<-lm(Sepal.Width ~ Sepal.Length, data=setosa)  lines(setosa$Sepal.Length,setosaflower$fitted.values, col= "black")  versiflower<-lm(Sepal.Width ~ Sepal.Length, data=versicolor)  lines(versicolor$Sepal.Length,versiflower$fitted.values, col= "blue")  virginicaflower<-lm(Sepal.Width ~ Sepal.Length, data=virginica)  lines(virginica$Sepal.Length,virginicaflower$fitted.values, col= "green")  flower<-lm(Sepal.Width ~ Sepal.Length, data=IrisData)  error4 = flower$fitted.values - IrisData$Sepal.Width  mean(error4^2)  setosaflower<-lm(Sepal.Width ~ Sepal.Length, data=setosa)  error5 = setosaflower$fitted.values - setosa$Sepal.Width  mean(error5^2)  versiflower<-lm(Sepal.Width ~ Sepal.Length, data=versicolor)  error6 = versiflower$fitted.values - versicolor$Sepal.Width  mean(error6^2)  virginicaflower<-lm(Sepal.Width ~ Sepal.Length, data=virginica)  error7 = virginicaflower$fitted.values - virginica$Sepal.Width  mean(error7^2) |

A screenshot of a cell phone

Description automatically generated

A close up of a map

Description automatically generated

A close up of a map

Description automatically generated

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  | | --- | | > flower<-lm(Sepal.Width ~ Sepal.Length, data=IrisData)  > error4 = flower$fitted.values - IrisData$Sepal.Width  > mean(error4^2)  [1] 0.1861044  >  >  > setosaflower<-lm(Sepal.Width ~ Sepal.Length, data=setosa)  > error5 = setosaflower$fitted.values - setosa$Sepal.Width  > mean(error5^2)  [1] 0.0631735  >  > versiflower<-lm(Sepal.Width ~ Sepal.Length, data=versicolor)  > error6 = versiflower$fitted.values - versicolor$Sepal.Width  > mean(error6^2)  [1] 0.06980983  >  > virginicaflower<-lm(Sepal.Width ~ Sepal.Length, data=virginica)  > error7 = virginicaflower$fitted.values - virginica$Sepal.Width  > mean(error7^2)  [1] 0.08061605 | |  | | |  | | --- | | > | | |

**Question 4**

Develop a R Script file that can perform the following tasks on the Power Consumption data set given in “Data Set 5.csv”. (The data are the electricity consumptions in kilowatt-hours per month from different houses and the areas in square meter of those houses)

1. Create a scatter plot for the two variables.

|  |
| --- |
| rm(list=ls())  cat("\14")  Power<- read.csv("Data Set 5.csv")  par(mfrow = c(1,3), mar=c(4, 4, 2,1))  #1. Create a scatter plot for the two variables.  #Plotting the scatter plot for Size,KW variables  plot(Power$Size, Power$KW, xlab ="x" ,ylab = "y" , main="Observed Data") |

A screenshot of a cell phone

Description automatically generated

2. Calculate a linear regression model.

|  |
| --- |
| #plotting the linear,polynomial regression models  data1<- lm(Power$KW~poly(Power$Size,1))  lines(Power$Size, data1$fitted.values, col="Red") |

3. Calculate polynomial regression models of order 2 and 3.

|  |
| --- |
| plot(Power$Size, Power$KW, xlab ="x" ,ylab = "y" , main="Observed Data")  data2<- lm(Power$KW~poly(Power$Size,2))  lines(Power$Size, data2$fitted.values, col="Red")  plot(Power$Size, Power$KW, xlab ="x" ,ylab = "y" , main="Observed Data")  data3<- lm(Power$KW~poly(Power$Size,3))  lines(Power$Size, data3$fitted.values, col="Red") |

4. Plot the regression models

A close up of a map

Description automatically generated

5. Compare the accuracy of the three models

|  |
| --- |
| #Compare the accuracy of the three models  error1 = data1$fitted.values - Power$KW  mean(error1^2)  error2 = data2$fitted.values - Power$KW  mean(error2^2)  error3 = data3$fitted.values - Power$KW  mean(error3^2) |

|  |
| --- |
| > error1 = data1$fitted.values - Power$KW  > mean(error1^2)  [1] 14320.65  >  > error2 = data2$fitted.values - Power$KW  > mean(error2^2)  [1] 1541.255  >  > error3 = data3$fitted.values - Power$KW  > mean(error3^2)  [1] 1529.475 |

**Question 5**

Two machines are used for filling plastic bottles with a net volume of 16.0 ounces. The filling process can be assumed to be normal. The quality engineering department suspects that both machines fill to the same net volume. An experiment is performed by taking a random sample from the output of each machine. Would you reject or accept the quality engineering department hypothesis? Perform your calculation by using R. (alpha = 0.05).

|  |
| --- |
| m1 = c(16.03,16.04,16.05,16.05,16.02,16.01,15.96,15.98,16.02,15.99)  m2 = c(16.02,15.97,15.96,16.01,15.99,16.03,16.04,16.02,16.01,16.00)  ##Calculating Mean  mean1 <- mean(m1)  mean2 <- mean(m2)  n1=10  n2=10  sd1<-sd(m1)  sd2<-sd(m2)  ##Calculating Variance  variance1 <- var(m1)  variance2 <- var(m2)  ## Z Factor  z1 <- (m1 - mean1)/sd(m1)  z2 <- (m2 - mean2)/sd(m2)  sp= sqrt(((n1 - 1)\*sd1^2 + (n2-1)\*sd2^2)/(n1 + n2 - 2))  t0 = (mean1 -mean2)/(sp\*sqrt(1/n1 + 1/n2))  print(t0)  t1 = t.test(m1,m2) #hypothesis t-test > print(t1)  print(t1)  alpha = 0.05  t0.025 <- 2.101 ## Value from the z distribution table [t 0.025,18 = 2.101]  if (t0 <= t0.025) {  print("Accept")  } else {  print("Reject")  } |

|  |
| --- |
| > t0 = (mean1 -mean2)/(sp\*sqrt(1/n1 + 1/n2))  > print(t0)  [1] 0.7989355  >  > t1 = t.test(m1,m2) #hypothesis t-test > print(t1)  > print(t1)  Welch Two Sample t-test  data: m1 and m2  t = 0.79894, df = 17.493, p-value = 0.435  alternative hypothesis: true difference in means is not equal to 0  95 percent confidence interval:  -0.01635123 0.03635123  sample estimates:  mean of x mean of y  16.015 16.005  >  > alpha = 0.05  >  > t0.025 <- 2.101 ## Value from the z distribution table [t 0.025,18 = 2.101]  >  > if (t0 <= t0.025) {  + print("Accept")  + } else {  + print("Reject")  + }  [1] "Accept"  >  >  > ## since t0<=t0.025 and t0 is 0.79835 which does not fall under critical region,we accept the null hypothesis  >  >  > ## t0 is 0.79835 which does not fall under critical region,  > ##we accept the null hypothesis |

**Result**: Since t0 is 0.79835 which is not being in the critical region, we accept the Null

Hypothesis (H0: μ1 = μ2 ).

**Question 6**

In the population, the average IQ is 100 with a standard deviation of 10. A team of scientists wants to test a new medication to see if it has either a positive or negative effect on intelligence, or no effect at all. A sample of 30 participants who have taken the medication has a mean of 110. Did the medication affect intelligence, using alpha = 0.01? Find the confidence interval?

|  |
| --- |
| # H0;  IQ = 100 #average IQ  # H1;IQ != 100  # A sample of 30 participants who have taken the medication has a mean of 110  sample = 30  mean = 110  sd = 10 # average IQ 100 with a standard deviation of 10  ## Z is calculated as +/- 2.58  z = (mean-IQ)/(sd/sqrt(sample))  print(z)  alpha = 0.01  # 1-0.5a = 0.995  # In table, only find nearest value 0.9951  ##Result: Reject the Null Hypothesis.  ##Conclusion: Medication significantly affect the intelligence Z = 5.477, p < 0.01.  ## Z is calculated as +/- 2.58  z0 = 2.5+0.08  if (z <= z0\*(-1) | z > z0 ) {  print("Reject")  } else {  print("Accept")  }  #checking Hypothesis  L = 110 - z0 \* sd/sqrt(sample)  U = 110 + z0 \* sd/sqrt(sample)  ##110 +/- 4.7104 is the confidence interval  print(L)  print(U) |

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | | ## Z is calculated as +/- 2.58  > z = (mean-IQ)/(sd/sqrt(sample))  > print(z)  [1] 5.477226  > alpha = 0.01  > # 1-0.5a = 0.995  > # In table, only find nearest value 0.9951  >  > ##Result: Reject the Null Hypothesis.  > ##Conclusion: Medication significantly affect the intelligence Z = 5.477, p < 0.01.  > ## Z is calculated as +/- 2.58  >  > z0 = 2.5+0.08  >  > if (z <= z0\*(-1) | z > z0 ) {  + print("Reject")  + } else {  + print("Accept")  + }  [1] "Reject"  > #checking Hypothesis  > L = 110 - z0 \* sd/sqrt(sample)  > U = 110 + z0 \* sd/sqrt(sample)  > ##110 +/- 4.7104 is the confidence interval  > print(L)  [1] 105.2896  > print(U)  [1] 114.7104 | |  | |  | |

**Result**: Reject the Null Hypothesis.

**Conclusion**: Medication significantly affect the intelligence **Z** = 5.477, **p** < 0.01.

**Confidence Interval**: 110 +/- 4.7104