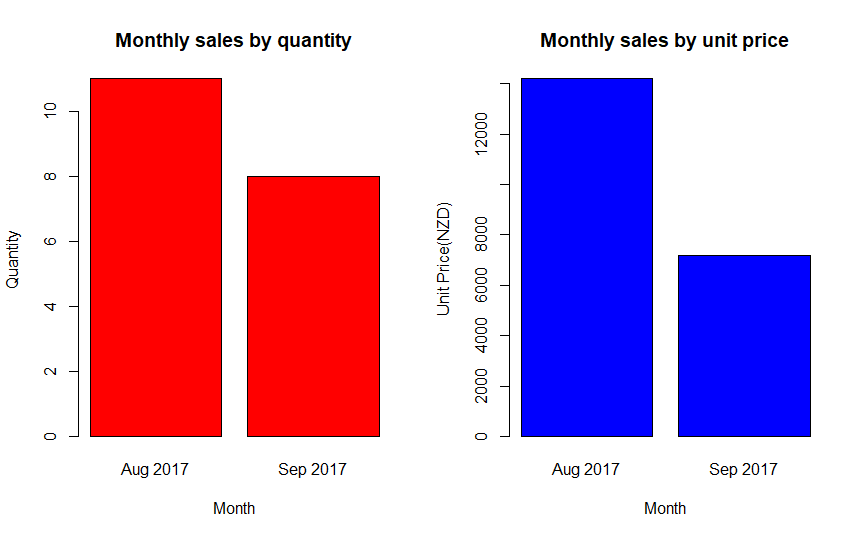
**Lab Exercises: B**

**Question: 1**

|  |
| --- |
| #Author: Dhaval Shah(Student Id: 1535740)  #########################  ## Loading the dataset ##  #########################  salesData1 <- read.csv("Data Set 1a.csv", na.strings = "");  salesData2 <- read.csv("Data Set 1b.csv", na.strings = "");  ##########################  ## Cleaning the dataset ##  ##########################  salesData1$Quantity = as.numeric(salesData1$Quantity)  levels(salesData1$Product) <- c(levels(salesData1$Product), "Galaxy Edge S8")  salesData1$Product[salesData1$Product == "Galaxi Edge S8"] <- "Galaxy Edge S8"  names(salesData1)[names(salesData1)=='Name']<-'Customer'  salesData1$Quantity[salesData1$Quantity=="one"]<-"1"  salesData1$Quantity[salesData1$Quantity=="two"]<-"2"  salesData1$Customer[salesData1$Customer == "ALice"] <- "Alice"  library(lubridate);  salesData1$Date<-dmy(salesData1$Date);  #Merging the dataset  mergedData = merge(salesData1,salesData2,"Product")  #Converting Unit  Calculate.Rate <- function(arg1){  if (arg1=='NZD') {out<-1}  if (arg1=='AUD') {out<-1.2}  return(out)  }  mergedData["Exchange.Rate"] <- sapply(mergedData$Currency,Calculate.Rate)  mergedData$Unit.Price.NZD <- mergedData$Unit.Price\*mergedData$Exchange.Rate  ###########################################################  # 1. Omitting all the transactions made before 1 Jan 2010 #  ###########################################################  filteredSalesData <- subset(mergedData, mergedData$Date > as.Date("2010-01-01"))  library(plyr)  library(zoo)  ###############################  # 2. Calculating monthly sale #  ###############################  filteredSalesData$Quantity = as.numeric(filteredSalesData$Quantity)  filteredSalesData$Month <- as.yearmon(filteredSalesData$Date)  GroupColumns <- c('Month')  DataColumns <- c('Quantity','Unit.Price.NZD')  monthWiseSale <- ddply(filteredSalesData,GroupColumns,function(x) colSums(x[DataColumns]))  monthWiseSale$Month = as.character(monthWiseSale$Month)  ###################################################  # 3. Visualizing the results (monthly total sale) #  ###################################################  par(mfrow=c(1,2))  barplot(monthWiseSale$Quantity, names.arg=monthWiseSale$Month, xlab = "Month", ylab = "Quantity", main="Monthly sales by quantity", col="red")  barplot(monthWiseSale$Unit.Price.NZD, names.arg=monthWiseSale$Month, xlab = "Month", ylab = "Unit Price(NZD)", main="Monthly sales by unit price", col="blue") |

OUTPUT:

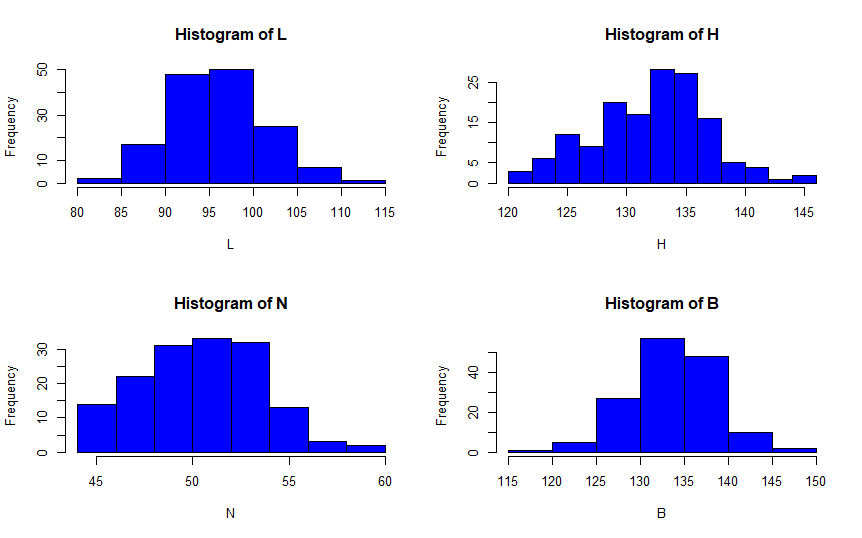


**Question:2**

1. Histogram:

|  |
| --- |
| skullData <- read.csv("Data Set 3.csv", na.strings = "")  par(mfrow=c(2,2))  hist(skullData$L, col = "blue", main="Histogram of L", xlab="L")  hist(skullData$H, col = "blue", main="Histogram of H", xlab="H")  hist(skullData$N, col = "blue", main="Histogram of N", xlab="N")  hist(skullData$B, col = "blue", main="Histogram of B", xlab="B") |

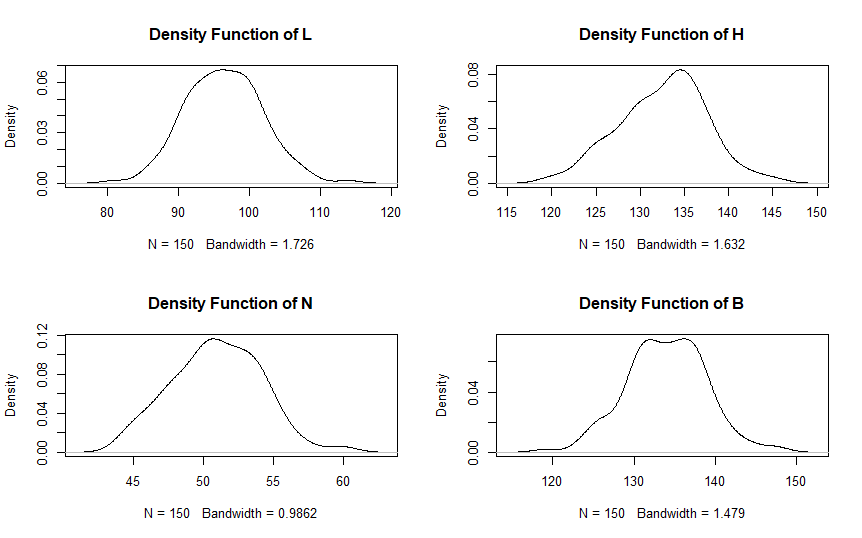
OUTPUT:



1. Density functions plot:

|  |
| --- |
| skullData <- read.csv("Data Set 3.csv", na.strings = "")  par(mfrow=c(2,2))  plot(density(skullData$L), main="Density Function of L")  plot(density(skullData$H), main="Density Function of H")  plot(density(skullData$N), main="Density Function of N")  plot(density(skullData$B), main="Density Function of B") |

OUTPUT:



1. Comparison of the density functions against a normal density function

|  |
| --- |
| install.packages(“e1071”)  library(e1071)  skewness(skullData$L)  kurtosis(skullD0ata$L)  skewness(skullData$H)  kurtosis(skullData$H)  skewness(skullData$N)  kurtosis(skullData$N)  skewness(skullData$B)  kurtosis(skullData$B) |

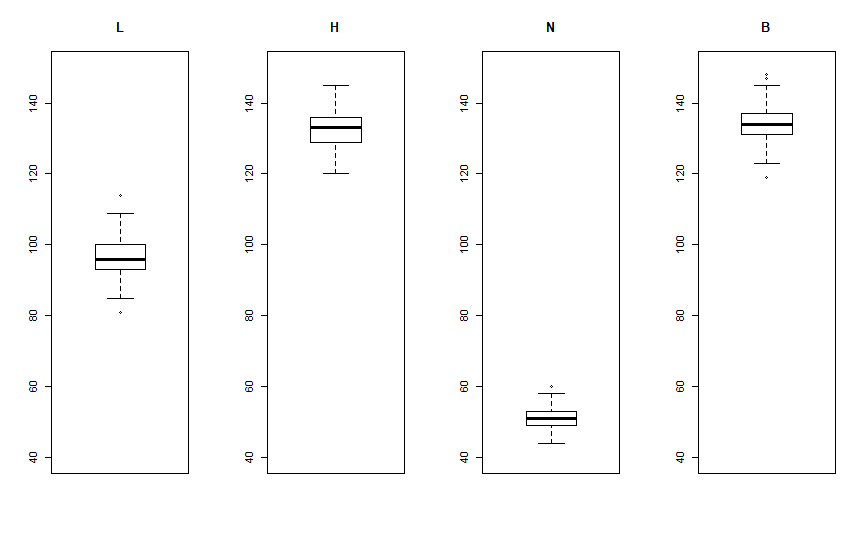
OUTPUT:

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

1. Boxplots:

|  |
| --- |
| par(mfrow=c(1,4))  boxplot(skullData$L, main="L", ylim = c(40,150))  boxplot(skullData$H, main="H", ylim = c(40,150))  boxplot(skullData$N, main="N", ylim = c(40,150))  boxplot(skullData$B, main="B", ylim = c(40,150)) |

OUTPUT:



1. Calculate the mean, variance and standard deviation

|  |
| --- |
| mean(skullData$L)  var(skullData$L)  sd(skullData$L)  mean(skullData$H)  var(skullData$H)  sd(skullData$H)  mean(skullData$N)  var(skullData$N)  sd(skullData$N)  mean(skullData$B)  var(skullData$B)  sd(skullData$B) |

OUTPUT:

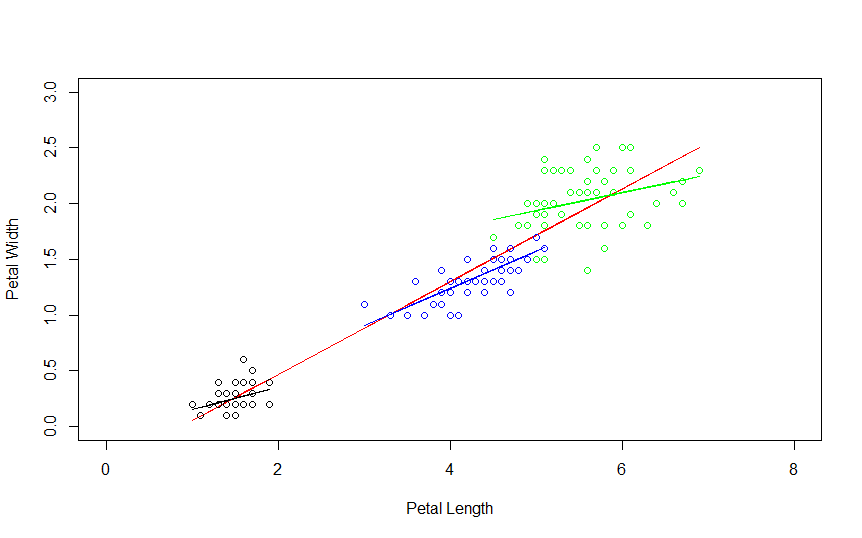
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | L | H | N | B |
| Mean | 96.46 | 132.5467 | 50.93333 | 133.9733 |
| Variance | 28.92121 | 24.39714 | 10.29083 | 23.91875 |
| Standard  deviation | 5.377844 | 4.939346 | 3.207932 | 4.89068 |

**Question: 3**

1. Scatter Plot for petal length and width variables

|  |
| --- |
| #Author: Dhaval Shah(Student Id: 1535740)  #########################  ## Loading the dataset ##  #########################  speciesData <- read.csv("Data Set 4.csv", na.strings = "");  par(mfrow=c(1,1))  with(speciesData, plot(Petal.Length, Petal.Width, xlim = c(0,8), ylim=c(0.0,3.0),xlab="Petal Length",ylab="Petal Width"))  model<-lm(speciesData$Petal.Width~speciesData$Petal.Length)  lines(speciesData$Petal.Length,model$fitted.values, col="red")  summary(speciesData$Species)  with(subset(speciesData, Species=="setosa"), points(Petal.Length, Petal.Width, col = "black"))  with(subset(speciesData, Species=="versicolor"), points(Petal.Length, Petal.Width, col = "blue"))  with(subset(speciesData, Species=="virginica"), points(Petal.Length, Petal.Width, col = "green"))  setosaData <- subset(speciesData, Species=="setosa")  setosaModel<-lm(setosaData$Petal.Width~setosaData$Petal.Length)  lines(setosaData$Petal.Length,setosaModel$fitted.values, col="black")  versicolorData <- subset(speciesData, Species=="versicolor")  versicolorModel<-lm(versicolorData$Petal.Width~versicolorData$Petal.Length)  lines(versicolorData$Petal.Length,versicolorModel$fitted.values, col="blue")  virginicaData <- subset(speciesData, Species=="virginica")  virginicaModel<-lm(virginicaData$Petal.Width~virginicaData$Petal.Length)  lines(virginicaData$Petal.Length,virginicaModel$fitted.values, col="green")  setosaError <- setosaModel$fitted.values - setosaData$Petal.Width  setosaMSE <- mean(setosaError^2)  print(setosaMSE)  versicolorError <- versicolorModel$fitted.values - versicolorData$Petal.Width  versicolorMSE <- mean(versicolorError^2)  print(versicolorMSE)  virginicaError <- virginicaModel$fitted.values - virginicaData$Petal.Width  virginicaMSE <- mean(virginicaError^2)  print(virginicaMSE) |

OUTPUT:



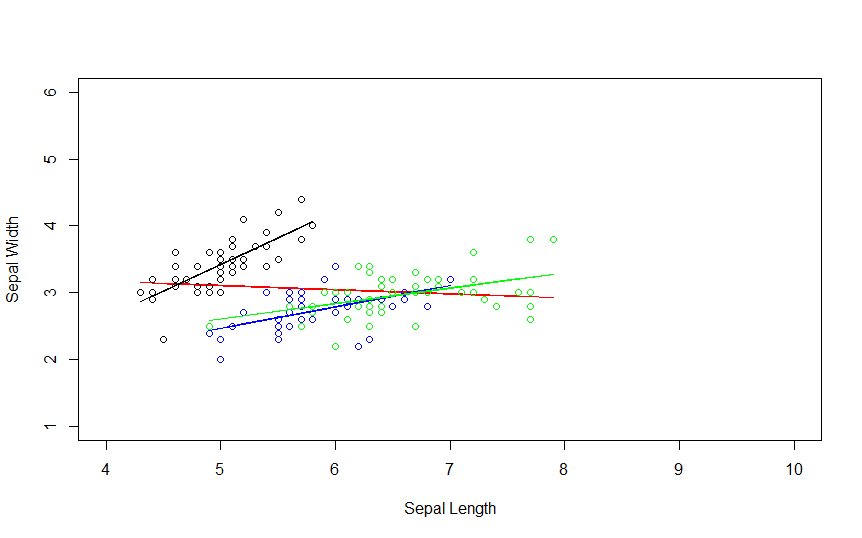
Mean Square Error for the linear models obtained for the three subsets which describe the relationship between petal length and width

|  |  |
| --- | --- |
| **Species** | **MSE** |
| setosa | 0.009686994 |
| versicolor | 0.01460732 |
| virginica | 0.06625411 |

1. Scatter Plot for sepal length and width variables

|  |
| --- |
| #Author: Dhaval Shah(Student Id: 1535740)  #########################  ## Loading the dataset ##  #########################  speciesData <- read.csv("Data Set 4.csv", na.strings = "");  ##Question: 3(7)  par(mfrow=c(1,1))  with(speciesData, plot(Sepal.Length, Sepal.Width, xlim = c(4,10), ylim=c(1.0,6.0),xlab="Sepal Length",ylab="Sepal Width"))  model<-lm(speciesData$Sepal.Width~speciesData$Sepal.Length)  lines(speciesData$Sepal.Length,model$fitted.values, col="red")  summary(speciesData$Species)  with(subset(speciesData, Species=="setosa"), points(Sepal.Length, Sepal.Width, col = "black"))  with(subset(speciesData, Species=="versicolor"), points(Sepal.Length, Sepal.Width, col = "blue"))  with(subset(speciesData, Species=="virginica"), points(Sepal.Length, Sepal.Width, col = "green"))  setosaData <- subset(speciesData, Species=="setosa")  setosaModel<-lm(setosaData$Sepal.Width~setosaData$Sepal.Length)  lines(setosaData$Sepal.Length,setosaModel$fitted.values, col="black")  versicolorData <- subset(speciesData, Species=="versicolor")  versicolorModel<-lm(versicolorData$Sepal.Width~versicolorData$Sepal.Length)  lines(versicolorData$Sepal.Length,versicolorModel$fitted.values, col="blue")  virginicaData <- subset(speciesData, Species=="virginica")  virginicaModel<-lm(virginicaData$Sepal.Width~virginicaData$Sepal.Length)  lines(virginicaData$Sepal.Length,virginicaModel$fitted.values, col="green")  setosaError <- setosaModel$fitted.values - setosaData$Sepal.Width  setosaMSE <- mean(setosaError^2)  print(setosaMSE)  versicolorError <- versicolorModel$fitted.values - versicolorData$Sepal.Width  versicolorMSE <- mean(versicolorError^2)  print(versicolorMSE)  virginicaError <- virginicaModel$fitted.values - virginicaData$Sepal.Width  virginicaMSE <- mean(virginicaError^2)  print(virginicaMSE) |

OUTPUT:



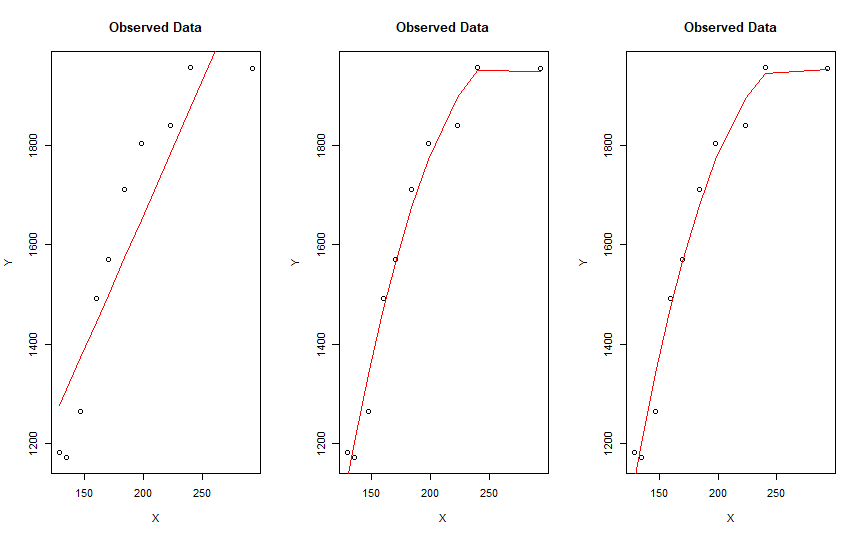
Mean Square Error for the linear models obtained for the three subsets which describe the relationship between sepal length and width

|  |  |
| --- | --- |
| **Species** | **MSE** |
| setosa | 0.0631735 |
| versicolor | 0.06980983 |
| virginica | 0.08061605 |

**Question: 4**

|  |
| --- |
| #Author: Dhaval Shah(Student Id: 1535740)  #########################  ## Loading the dataset ##  #########################  electricityConsumptionData <- read.csv("Data Set 5.csv", na.strings = "");  par(mfrow=c(1,3))  with(electricityConsumptionData, plot(Size, KW, xlab="X", ylab="Y", main="Observed Data"))  model<-lm(electricityConsumptionData$KW ~ electricityConsumptionData$Size)  lines(electricityConsumptionData$Size, model$fitted.values, col="red")  linearRegError <- model$fitted.values - electricityConsumptionData$KW  linearRegMSE = mean(linearRegError^2)  print(linearRegMSE)  with(electricityConsumptionData, plot(Size, KW, xlab="X", ylab="Y", main="Observed Data"))  model <-lm(electricityConsumptionData$KW ~ poly(electricityConsumptionData$Size,2))  lines(electricityConsumptionData$Size, model$fitted.values, col="red")  PolyRegOrd2Error <- model$fitted.values - electricityConsumptionData$KW  PolyRegOrd2MSE = mean(PolyRegOrd2Error^2)  print(PolyRegOrd2MSE)  with(electricityConsumptionData, plot(Size, KW, xlab="X", ylab="Y", main="Observed Data"))  model <-lm(electricityConsumptionData$KW ~ poly(electricityConsumptionData$Size,3))  lines(electricityConsumptionData$Size, model$fitted.values, col="red")  PolyRegOrd3Error <- model$fitted.values - electricityConsumptionData$KW  PolyRegOrd3MSE = mean(PolyRegOrd3Error^2)  print(PolyRegOrd3MSE) |

OUTPUT:



Comparison of accuracy of the models:

|  |  |
| --- | --- |
| **Model** | **MSE** |
| Linear Regression Model | 14320.65 |
| Polynomial Regression Model with order 2 | 1541.255 |
| Polynomial Regression Model with order 3 | 1529.475 |

**Question: 5**

|  |
| --- |
| #Author: Dhaval Shah(Student Id: 1535740)  machine1 = c(16.03,16.04,16.05,16.05,16.02,16.01,15.96,15.98,16.02,15.99)  machine2 = c(16.02,15.97,15.96,16.01,15.99,16.03,16.04,16.02,16.01,16.00)  meanM1 = mean(machine1)  meanM2 = mean(machine2)  sdM1 = sd(machine1)  sdM2 = sd(machine2)  n1 = 10  n2 = 10  calculateSp <- function(n1,n2,sd1,sd2)  {  return(sqrt(((n1 - 1)\*sd1^2 + (n2-1)\*sd2^2)/(n1 + n2 - 2)));  }  sp = calculateSp(n1,n2,sdM1,sdM2);  t0 = (meanM1 - meanM2)/(sp\*sqrt(1/n1 + 1/n2))  print(t0)  t1 = t.test(machine1,machine2)  print(t1)  alpha = 0.05  # +- t 0.025,18 = +1 2.101  t = 2.101  if (t0 <= t) {  print("Hypothesis accepted")  } else {  print("Hypothesis rejected")  } |

OUTPUT:

|  |
| --- |
| [1] 0.7989355  Welch Two Sample t-test  data: machine1 and machine2  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  [1] "Hypothesis accepted" |

**Question: 6**

|  |
| --- |
| #Author: Dhaval Shah(Student Id: 1535740)  # H0;  u = 100  # H1;u != 100  x = 110  Q = 10  n = 30  #Calculation of test statistics  z = (x-u)/(Q/sqrt(n))  print(paste("Z =",z))  alpha = 0.01  # 1-0.5alpha = 0.995  # In NORMAL DISTRIBUTION TABLE, only find nearest value 0.9951  z0 = 2.5+0.08  if (z <= z0\*(-1) | z > z0 ) {  print("Medication didn't affect intelligence")  } else {  print("Medication significantly affect intelligence")  }  L = 110 - z0 \* Q/sqrt(n)  U = 110 + z0 \* Q/sqrt(n)  print("Confidence Interval:")  print(paste("Lower:", L))  print(paste("Upper:", U)) |

OUTPUT:

|  |
| --- |
| [1] "Z = 5.47722557505166"  [1] "Medication didn't affect intelligence"  [1] "Confidence Interval:"  [1] "Lower: 105.289586005456"  [1] "Upper: 114.710413994544" |