CIA LAB TEST-I

MANOJ KUMAR - 2048015

12/02/2021

# 1-a. Create a character vector of 5 elements and assign a unique value to each element in the vector.

# To create a character vector of 5 elements named Students\_details.  
character\_details <- make.unique(c("character","character","character","character","character"), sep = "-")  
  
# To check the type of the object.  
typeof(character\_details)

## [1] "character"

# Listing the character elements which created.  
character\_details

## [1] "character" "character-1" "character-2" "character-3" "character-4"

*Insights*

-We have created Character vector of 5 elements using make.unique() and c() combain function, namely \*character\_details\*.  
-Type of Vector is identified using typeof() predefined function. Here our character\_details is of \*Character\* type.  
-Listing the unique elements by directly calling \*character\_details\*.

# 1-b.Create the matrix below

[,1][,2][,3]  
[1,] 5 -5 5  
[2,] 5 -5 5  
[3,] 5 -5 5 and also display the transpose of the matrix

# Create a matrix.  
E = c(5,-5,5)  
M <- matrix(rep(E,3), nrow = 3, byrow = TRUE)  
print(M)

## [,1] [,2] [,3]  
## [1,] 5 -5 5  
## [2,] 5 -5 5  
## [3,] 5 -5 5

# Element list  
matrix\_element = c(5, -5, 5, 5, -5, 5, 5, -5, 5)  
  
# Creating Matrix using matrix() predefined function.   
main\_matrix = matrix( matrix\_element,   
 nrow = 3, # No. of row  
 ncol = 3, # No. of Column  
 byrow = TRUE # Order type by Row  
 )  
  
# To display our main matrix   
main\_matrix

## [,1] [,2] [,3]  
## [1,] 5 -5 5  
## [2,] 5 -5 5  
## [3,] 5 -5 5

*Insights*

-There are different types of methods to define or create the Matrix, Here I have listed all the provided elements \*matrix\_element\* and used \*matrix()\* to buuild our required matrix.   
-nrow arg used to mention no. of row we needed  
-ncol arg used to mention no. of column we needed  
-byrow arg used to specify the filling type.

# To display the transpose of the matrix  
  
matrix\_transpose = t(main\_matrix)  
matrix\_transpose

## [,1] [,2] [,3]  
## [1,] 5 5 5  
## [2,] -5 -5 -5  
## [3,] 5 5 5

*Insights*

-t() In-built function is used to transpose of the matrix

# 

# 2.Study and interpret the results for the given data and fit the appropriate regression model

#1. Import the income data and perform the exploratory data analysis

# Importing the Income Data.  
  
Income\_dataframe <- read.csv("~/Downloads/Other/Christ University/SEM 2/R/Lab/R CIA 1/income.csv")

str(Income\_dataframe)

## 'data.frame': 30 obs. of 2 variables:  
## $ Experience: num 1.1 1.3 1.5 2 2.2 2.9 3 3.2 3.2 3.7 ...  
## $ Salary : int 39343 46205 37731 43525 39891 56642 60150 54445 64445 57189 ...

# Exploring data  
  
Income\_dataframe

## Experience Salary  
## 1 1.1 39343  
## 2 1.3 46205  
## 3 1.5 37731  
## 4 2.0 43525  
## 5 2.2 39891  
## 6 2.9 56642  
## 7 3.0 60150  
## 8 3.2 54445  
## 9 3.2 64445  
## 10 3.7 57189  
## 11 3.9 63218  
## 12 4.0 55794  
## 13 4.0 56957  
## 14 4.1 57081  
## 15 4.5 61111  
## 16 4.9 67938  
## 17 5.1 66029  
## 18 5.3 83088  
## 19 5.9 81363  
## 20 6.0 93940  
## 21 6.8 91738  
## 22 7.1 98273  
## 23 7.9 101302  
## 24 8.2 113812  
## 25 8.7 109431  
## 26 9.0 105582  
## 27 9.5 116969  
## 28 9.6 112635  
## 29 10.3 122391  
## 30 10.5 121872

*Insights*

-Here shared \*income.csv\* dataset having two numerical columns in it, namely Experience and Salary of an Employee we assume.  
-Experience is haven't having rounded values by years. Insteard of its having Actucal Decimal value, which may include months perios as well.  
-Salary is populated in normal form which actually makes sense to work.

is.na(Income\_dataframe)

## Experience Salary  
## [1,] FALSE FALSE  
## [2,] FALSE FALSE  
## [3,] FALSE FALSE  
## [4,] FALSE FALSE  
## [5,] FALSE FALSE  
## [6,] FALSE FALSE  
## [7,] FALSE FALSE  
## [8,] FALSE FALSE  
## [9,] FALSE FALSE  
## [10,] FALSE FALSE  
## [11,] FALSE FALSE  
## [12,] FALSE FALSE  
## [13,] FALSE FALSE  
## [14,] FALSE FALSE  
## [15,] FALSE FALSE  
## [16,] FALSE FALSE  
## [17,] FALSE FALSE  
## [18,] FALSE FALSE  
## [19,] FALSE FALSE  
## [20,] FALSE FALSE  
## [21,] FALSE FALSE  
## [22,] FALSE FALSE  
## [23,] FALSE FALSE  
## [24,] FALSE FALSE  
## [25,] FALSE FALSE  
## [26,] FALSE FALSE  
## [27,] FALSE FALSE  
## [28,] FALSE FALSE  
## [29,] FALSE FALSE  
## [30,] FALSE FALSE

# Summary-Exploratory Data Analysis  
summary(Income\_dataframe)

## Experience Salary   
## Min. : 1.100 Min. : 37731   
## 1st Qu.: 3.200 1st Qu.: 56721   
## Median : 4.700 Median : 65237   
## Mean : 5.313 Mean : 76003   
## 3rd Qu.: 7.700 3rd Qu.:100545   
## Max. :10.500 Max. :122391

*Insights*

-Minimum years of Experience is 1.1 in year   
-Minimum Amount of Salary is 37731 (preferably currency is not mentioned)   
-Maximum years of Experience is 10.5, which is 10 and half years.   
-Maximum Amount of Salary provided is 122391 (preferably currency is not mentioned)   
-Using \*summary()\* we come to all other basic stats features.

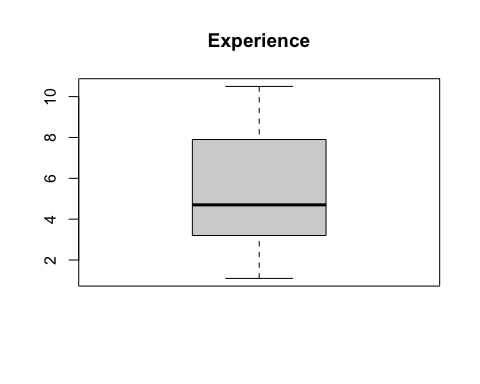
# 2. Use appropriate plot to visualize the relationship between the variables and outliers

# Using scatter plot  
  
x = Income\_dataframe$Experience  
y = Income\_dataframe$Salary  
  
scatter.smooth(x,   
 y,   
 xlab = "Experience", ylab = "Salary", # X,Y-Lab  
 main = "Relationship between the Experience & Salary", # Title  
 col.main = "Black", # Title lab color  
 cex.lab = "1" , # X,Y-axis Lab color  
 pch = 19  
)

 *Insights*

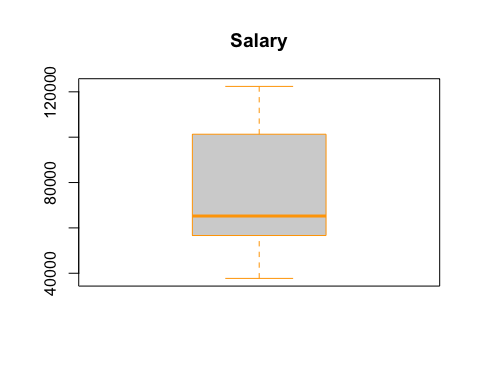
-Scatter plot is used to visualize the relationship between the variables  
-The position of each dot on the horizontal and vertical axis indicates values for an individual data point.  
-Relationships between variables can be described and found \*positive, strong, and linear\*.

boxplot(x, main = "Experience", border = "black" )

 *Insights*

-boxplots are useful to detect potential outliers.  
-Clear no outliers to seem to be present in Experience, as our boxplot came out well.

boxplot.default(y, main = "Salary", border = "orange")



# no outliers

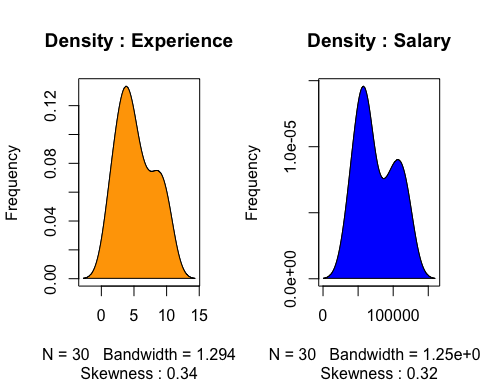
*Insights*

-boxplots are useful to detect potential outliers.  
-Clear no outliers to seem to be present in Salary, as our boxplot came out well and good.  
-Both variables are right skewed and have no outliers.

# 

# 3. Use Density plot to check the normality and also check the correlation analysis.

# A density plot shows the distribution of a numeric variable.  
  
library(e1071) # For skewness function  
par(mfrow = c(1,2)) # Dividing graph area in 2 columns  
  
# Density plot for 'Experience'  
plot(density(x),   
 main = "Density : Experience",  
 ylab = "Frequency",  
 sub=paste("Skewness :",  
 round(e1071::skewness(x),  
 2  
 )  
 )  
 )   
polygon(density(x), col = "orange")  
  
# Density plot for 'Salary'  
plot(density(y),   
 main = "Density : Salary",  
 ylab = "Frequency",  
 sub=paste("Skewness :",  
 round(e1071::skewness(y),  
 2  
 )  
 )  
 )   
  
polygon(density(y), col = "blue")



cor(x, y)

## [1] 0.9782416

*Insights*

-A density plot is a representation of the distribution of a numeric variable.  
-Numeric variables namely \*Experience\* and \*Salary\*.  
-We estimated density to show the probability density function of the \*Experience\* and \*Salary\*  
-We can clearly see the hump in between the distribution of salary.  
  
-Correlation analysis studies the strength of relationship between two continuous variables.  
-Correlation values are colser to 1 so, \*strong positive correlation\*.

# 4. Calculate the model summary, model coefficients, estimate and error for the predictor variable.

# Simple Linear Regression Model  
LinearRegressionModel <- lm(formula = Salary ~ Experience,   
 data = Income\_dataframe)  
  
# Coefficients  
LinearRegressionModel

##   
## Call:  
## lm(formula = Salary ~ Experience, data = Income\_dataframe)  
##   
## Coefficients:  
## (Intercept) Experience   
## 25792 9450

*Insights*

-The sign of each coefficient indicates the direction of the relationship between a predictor variable and the response variable.   
-A positive sign indicates that as the predictor variable increases, the response variable also increases.

# Summary   
modelsummary <- summary(LinearRegressionModel)  
modelCoeffs <- modelsummary$coefficients  
modelCoeffs

## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) 25792.200 2273.0534 11.34694 5.511950e-12  
## Experience 9449.962 378.7546 24.95009 1.143068e-20

estimate <-modelCoeffs["Experience","Estimate"]  
stdError <-modelCoeffs["Experience", "Std. Error"]  
  
tValue <- estimate/stdError  
tValue

## [1] 24.95009

*Insights*

-When p Value is less than significance level p Value < 0.05 , we can \*Reject Null Hypothesis\* that the co-efficient β of the predictor is zero.  
-Hence, there is an significant difference.

# 5. Fit the appropriate Regression model.

# Create the training and test data (70:30)  
set.seed(100)  
rows = sample(nrow(Income\_dataframe))  
  
# Randomly order data:  
data = Income\_dataframe[rows, ]  
  
# Identify row to split on: split  
split = round(nrow(data) \* .70)  
  
# Create train  
train = data[1:split,]  
train

## Experience Salary  
## 10 3.7 57189  
## 23 7.9 101302  
## 6 2.9 56642  
## 16 4.9 67938  
## 19 5.9 81363  
## 25 8.7 109431  
## 14 4.1 57081  
## 12 4.0 55794  
## 22 7.1 98273  
## 28 9.6 112635  
## 4 2.0 43525  
## 21 6.8 91738  
## 2 1.3 46205  
## 7 3.0 60150  
## 17 5.1 66029  
## 27 9.5 116969  
## 11 3.9 63218  
## 8 3.2 54445  
## 18 5.3 83088  
## 3 1.5 37731  
## 24 8.2 113812

# Create test  
test = data[(split + 1): nrow(data),]  
test

## Experience Salary  
## 13 4.0 56957  
## 29 10.3 122391  
## 1 1.1 39343  
## 9 3.2 64445  
## 30 10.5 121872  
## 20 6.0 93940  
## 26 9.0 105582  
## 5 2.2 39891  
## 15 4.5 61111

# Fit the model on training data and predict Salary on test data.  
  
Model = lm(formula = Salary ~ Experience, data = train )  
Model

##   
## Call:  
## lm(formula = Salary ~ Experience, data = train)  
##   
## Coefficients:  
## (Intercept) Experience   
## 25016 9661

Predict = predict(Model, test)

*Insights*

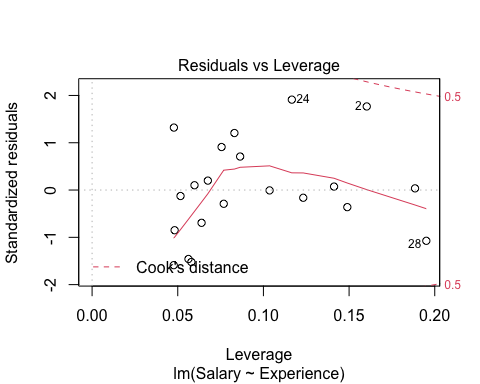
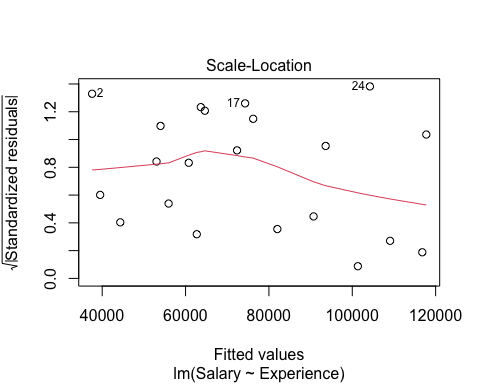
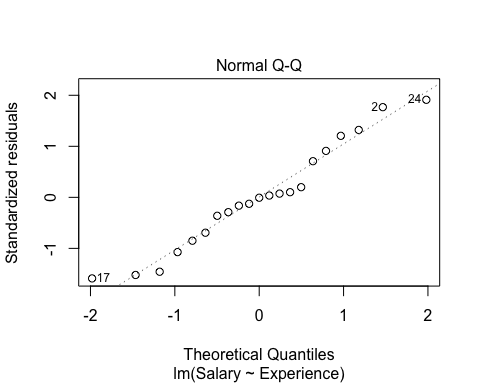
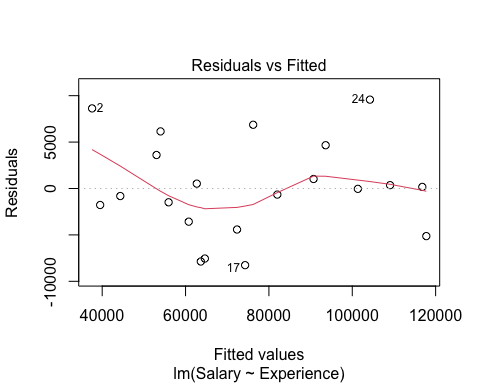
-Create the training and test data (70:30) and Randomly order data:  
-Fit the model on training data and predict Salary on test data.

# 6. Capture the summary of the model and review the diagnostic measures.

summary(Model)

##   
## Call:  
## lm(formula = Salary ~ Experience, data = train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -8259.9 -3573.9 -38.8 3608.3 9572.7   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 25016 2657 9.414 1.38e-08 \*\*\*  
## Experience 9661 462 20.911 1.42e-14 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 5328 on 19 degrees of freedom  
## Multiple R-squared: 0.9584, Adjusted R-squared: 0.9562   
## F-statistic: 437.3 on 1 and 19 DF, p-value: 1.415e-14

plot(Model)



*Insights*

-The fitted model equation is given by 'Y = 26690 X \* 9448'. It has a Coefficient of Determination value 0.9504 and an adjusted R2 value of 0.9478 which shows that the variability of Salaries of employees through their Experience is explained less than the full model.

-The p value is 1.2 \* 10^-8, which is still considerably much lower than required, and we can state we can reject H0 and claim that there is a significant relationship between the Experience and Salary of an Employee.

-The residual plot in the second model is slithly more linear which indicates more varibility. The Q-Q plot shos that there is a higher digression of residuals from the complete model which shows the residuals are less normal.

-The Scale Location Graph shows its values concentrated even more towards the positive side which indicates the residuals are not balanced. The Residuals v/s Leverage graph shows that though there are no visible outliers through Cook's Distance method, nonetheless there are more further apart in the fitted model.