|  |  |  |
| --- | --- | --- |
| **S.no** | **Name of the Experiment** | **Signature** |
| **1** | Crash Course on Python – I & II |  |
| **2** | Implementation of Binary Search Algorithm in Python |  |
| **3** | Implementation of Bubble Sort Algorithm in Python |  |
| **4** | Implementation of Best First Search Algorithm |  |
| **5** | Implementation of A\* Algorithm |  |
| **6** | Building Semantic Network in Python |  |
| **7** | Design and Deployment of an Expert System |  |
| **8** | Building Bayesian Networks in Python |  |
| **9** | Building Markov Chain Model |  |
| **10** | Building a Hidden Markov Model in Python |  |
| **11** | Fundamentals of R Language |  |
| **12** | Web Scraping in R |  |
| **13** | Data Visualization in R |  |
| **14** | Build Data dashboard using Shinny Dashboard |  |

**CONTENTS**

Internal Examiner External Examiner

Name: Name:

**EXERCISE:10 Building a Hidden Markov Model in Python**

**Aim :**

To Build a Hidden Markov Model by the use of python library protopunica and pandas

**Requirements**:

1. Jupyter Notebook

**Coding:**

**1 Import Libraries**

from **protopunica** import **\***

import **pandas** as **pd**

# 2 Observation Model

sunny = DiscreteDistribution({"raincoat": 0.1,"Hat" : 0.9})

rainy = DiscreteDistribution({"raincoat": 0.7,"Hat" : 0.3})

# 3 Define States

states= [sunny,rainy]

# 4 Transition Model

transition\_model = numpy.array([[0.7,0.3],[0.4,0.6]])

# 5 Initial State

Initial\_state=numpy.array([0.3,0.7])

# 6 Build the Model

model=HiddenMarkovModel.from\_matrix(transition\_model,states,

# 7 Bake the Model

Initial\_state,state\_names=["sunny","rainy"])

model.bake()

# 8 Observation

observations=["Hat","Hat","raincoat","raincoat","Hat","Hat","raincoat","Hat",

"Hat"]

# 9 Predict the States for Given Observation

predictions = model.predict(observations)

predictions

# 10 Model

# Predicted states with respect to Observation

for prediction in predictions:

print(model.states[prediction].name)

predicted\_probabilities=model.predict\_proba(observations)

Most\_likely\_Weather=pd.DataFrame(predicted\_probabilities,columns=["Rainy","Sunny"])

Most\_likely\_Weather

**Result:**

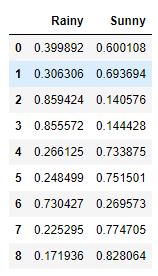
Thus the way we declare and execute Hidden Markov Model by the use of python library protopunica and pandas is Verified Successfully

**EXERCISE:10 Building a Hidden Markov Model in Python**

**Output:**

[1, 1, 0, 0, 1, 1, 0, 1, 1]

**Output:**



**EXERCISE:11 Fundamentals of R Language**

**Aim :**

To study and practice about the basic datatypes ,Conditional Statement, Loops ,Handling Data and Functions, of R using R Studio.

**Requirements**:

1. R Studio

**Coding:**

**Data Types**

***# Numeric***

num = 44.5

class(num)

***# Integer***

num1 = "100L"

class(num1)

***# Complex***

cmplx = 10i

class(cmplx)

***# Character***

Name = “VIKKI”

class(Name)

***# Logical***

num2 = TRUE

class(num2)

**Conditional Statements**

***# If Statement***

x <- 5

if(x > 0)

{

print("Positive number")

}

***# If Else Statement***

x <- -5

if(x > 0)

{

print("Positive number")

} else

{

print("Negative number")

}

***# Nested If-Else Statement***

x <- 0

if(x < 0)

{

print("Negative number")

} else if(x > 0)

{

print("Positive number")

} else

print("Zero")

**Loops**

***# For Loop***

x <- c(-10,5,10,44)

for (i in x)

{

print(i)

}

***# Nested for loop***

for (i in 1:3)

{

for (j in 1:i)

{

print(i \* j)

}

}

***# While Loop***

i = 0

count = 0

while (count <= 5)

{

print(i \* i)

i = i + 1

count = count + 1

}

**Functions**

***# Function for Squaring a Number***

number = function(x) {

return(x^2)

}

result = number(5)

print(result)

**Handling Data in R**

***# Creating data***

employee <- c('Leo', 'Andres', 'Messi')

salary <- c(20000, 23000, 28000)

ID\_num = c(30, 19, 10)

***# Creating data frame***

employee.data = data.frame(employee, salary, ID\_num)

***# Viewing data frame***

View(employee.data)

***# Basic info from data frame***

print(ncol(employee.data))

print(nrow(employee.data))

***# Slicing Column***

print(employee.data$employee)

***# Slicing Rows***

print(employee.data[2:3, ])

***# Modifying data***

employee.data[2, "ID\_num"] <- 15

employee.data

***# Saving data frame***

write.csv(employee.data, "Employee Details.csv")

***# Reading data frame***

read.csv("Employee Details.csv")

**Result:**

Thus the way we declare and execute basic datatypes ,Conditional Statement, Loops , Data Handling and Functions of R using R Studio is verified Successfully

**EXERCISE:11 Fundamentals of R Language**

**OUTPUT:**

***# Numeric***

"numeric"

***# Integer***

"integer"

***# Complex***

"complex"

***# Character***

"character"

***# Logical***

"logical"

**OUTPUT:**

***# If Statement***

"Positive number"

***# If Else Statement***

"Negative number"

***# Nested If-Else Statement***

“Zero”

**OUTPUT:**

***# For Loop***

-10

5

10

44

***# Nested for loop***

1

2

4

3

6

9

***# While Loop***

0

1

4

9

16

25

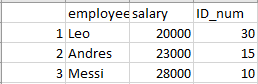
**OUTPUT:**

***# Function for Squaring a Number***

25

**OUTPUT:**

**Handling Data in R**



|  |
| --- |
|  |

**EXERCISE:12 Web Scraping in R**

**Aim :**

To Scrape and Specific types of information from website using R by the use of R library

**Requirements**:

1. R Studio

**Coding:**

***# IMPORTING LIBRARIES***

library(rvest)

library(dplyr)

library(robotstxt)

library(stringr)

***# SCRAPPING WEBSITE***

link <- "https://editorial.rottentomatoes.com/guide/10-best-reviewed-pro-football-movies/"

***# Allowability***

path = paths\_allowed(link)

***# HTML ELEMENTS FROM WEBSITE***

web <- read\_html(link)

View(web)

name <- web %>% html\_nodes(".article\_movie\_title a") %>% html\_text()

View(name)

year <- web %>% html\_nodes(".start-year") %>% html\_text()

View(year)

rating <- web %>% html\_nodes(".tMeterScore") %>% html\_text()

View(rating)

rank <- web %>% html\_nodes(".countdown-index") %>% html\_text()

View(rank)

Director <- web %>% html\_nodes(".director .descriptor+ a") %>% html\_text()

View(Director)

***# CREATING DATAFRAME***

movies.ratings <- data.frame(name, year, rating, rank, Director)

***# VIEW DATAFRAME***

View(movies.ratings)

***# SAVING DATA***

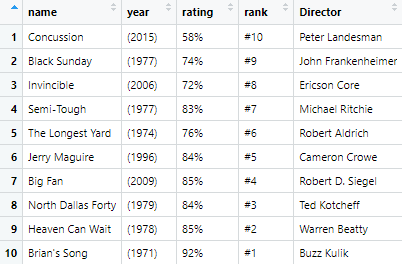
write.csv (movies.ratings, "My movie data.csv")

**Result:**

Thus the way we scrape and execute the web scraping by the use of R Studio is verified Successfully

**EXERCISE:12 Web Scraping in R**

**Output:**

****

**EXERCISE:13 Data Visualization in R**

**Aim :**

To Create and Manipulate various Data Visualization like Bar Plot, Line Plot, Scatter Plot, Histogram for Data set using R Studio.

**Requirements**:

1. R Studio
2. Data Set

**Coading:**

library(ggplot2)

library(dplyr)

***# Importing Dataset***

expenses <- read.csv("dailyexpenses.csv")

***# Viewing Dataset***

View(expenses)

str(expenses)

***# Date type Conversion***

expenses$Date <- as.Date(expenses$Date, "%d/%m/%y")

View(expenses)

str(expenses)

***# Data frame for Comparing Expenses***

tot <- expenses[, 2:9]

total <- colSums(tot)

detail <- c("food", "groceries", "medical", "fuel", "toll",

"phone", "electricity", "miscellaneous")

tot\_each <- data.frame(detail, total)

View(tot\_each)

***# Bar plot of expenses***

ggplot(data = tot\_each, aes(x = detail, y = total))+ geom\_bar(stat = "identity", fill = "green") +

labs(x = "Items", y = "Expenses", title = "Comparing Expenses")

***# Dataframe for Daily Total Expenses***

cm <- expenses[, 2:9]

dt <- expenses[, 1]

rs <- rowSums(cm)

daily\_tot <- data.frame(dt, rs)

View(daily\_tot)

***# Line Plot for daily total expenses***

ggplot(data = daily\_tot, aes(x = dt, y = rs)) +

geom\_line(color = "red", size = 2) +

labs(x = "Date", y = "Expenses", title = "Total Expenses")

***# Dataframe of Daily Medical Expenses***

md <- expenses[, 3]

de <- expenses[, 1]

daily\_med <- data.frame(de, md)

View(daily\_med)

***# Scatter Plot of Daily Medical Expense***

ggplot(data = daily\_med, aes(x = de, y = md)) + geom\_point() +

labs(x = "Expenses", y = "Medical", title = "Medical Expenses")

***# Histogram of Medical Expenses***

ggplot(expenses, aes(x = Medical)) +

geom\_histogram(binwidth = 5, color = "black", fill = "blue")

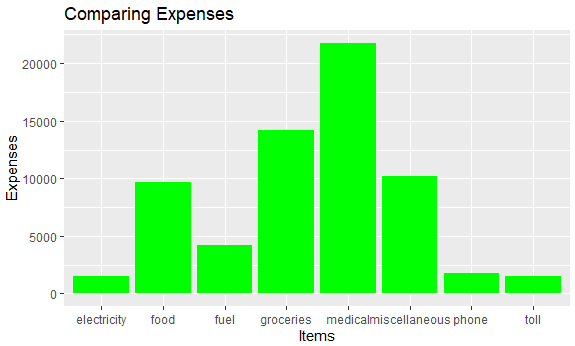
**Result:**

Thus the way we Create and Manipulate various Data Visualization like Bar Plot, Line Plot, Scatter Plot, Histogram for Data set using R Studio is verified Successfully

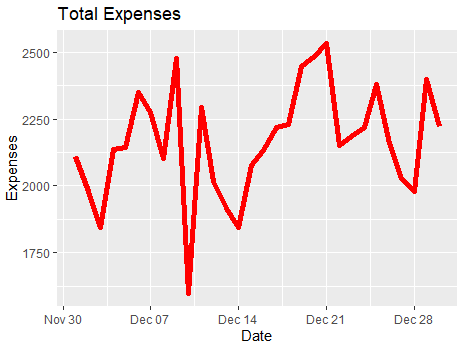
**EXERCISE:13 Data Visualization in R**

**OUTPUT:**

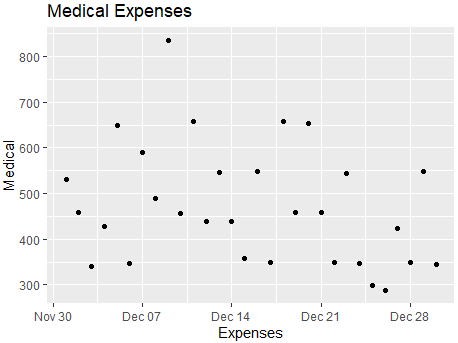
***# Bar plot of expenses***



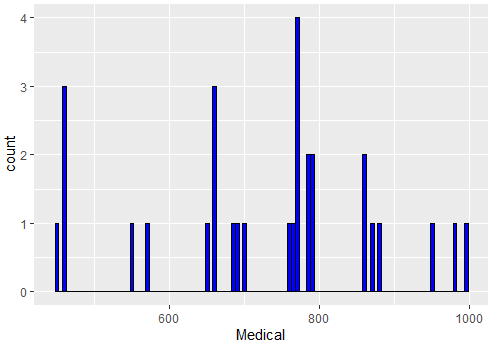
***# Line Plot for daily total expenses***



***# Scatter Plot of Daily Medical Expense***



***# Histogram of Medical Expenses***



**EXERCISE:14 Build Data dashboard using Shinny Dashboard**

**Aim :**

To Build Data dashboard using Shinny Dashboard using R Studio

**Requirements**:

1. R Studio

**Coading:**

library(shinydashboard)

library(shiny)

ui<-dashboardPage(

dashboardHeader(title="Basic Dashboard"),

dashboardSidebar(

sidebarMenu(

menuItem("Dashboard", tabName = "dashboard", icon = icon("dashboard")),

menuItem("Widgets", tabName = "widgets", icon = icon("th"))

)

),

dashboardBody(

tabItems(

tabItem(tabName = "dashboard",

fluidRow(

box(plotOutput("plot1",height=400)),

box(title="Controls",

sliderInput("slider","Number of Observations",1,1000,500)

)

)),

tabItem(tabName = "widgets",

h2("Widgets tab page under construction"))

)

)

)

server <- function(input, output) {

set.seed(122)

histdata <- rnorm(1000)

output$plot1 <- renderPlot({

data <- histdata[seq\_len(input$slider)]

hist(data)

})

}

shinyApp(ui,server)

**Result:**

Thus the way we create and execute Data dashboard using Shinny Dashboard in R Studio is verified Successfully

**EXERCISE:14 Build Data dashboard using Shinny Dashboard**

**OUTPUT:**

