جامعة الإسكندرية
ALEXANDRIA UNIVERSITY
كلية الحاسبات وعلوم البيانات

Data Science Project - 17

STUDENT'S NAME

Kareem Mohamed abdelghny elramady
2406247728

Amr Mohamed Shehata Mohamed Saleh
2406246457

Marwan Mohammed Shehata Ahmad
2406249854

Mazen amr Mahmoud fouad
2406248857

The problem

Businesses collect a large amount of data related to customers, including transaction histories, details about the customers. However, without proper analysis, it is challenging to extract meaningful insights from this data.

The role of each member

Mazen:

- Read data
- Clean data
- Visualisation

Marwan:

- K-means

Amr:

- Association

Kareem:

- GUI (Shiny)

What will the program do ?

The program takes a huge data set and turns it into numerous benefits which are:

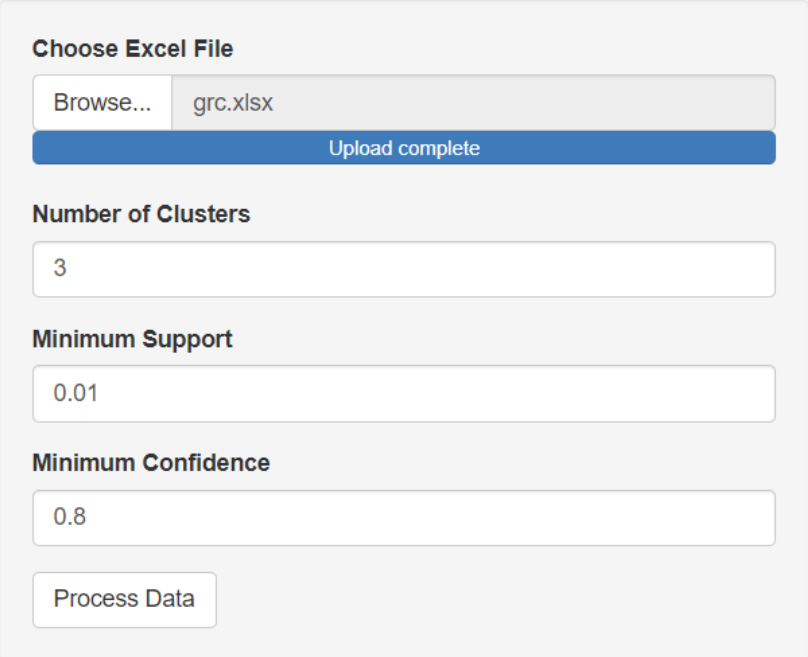
1. **Grouping Customers:** Divides customers by spending and age to create better marketing plans.

2. **Finding Patterns:** Identifies which products are often bought together to increase sales.

3. **Making Better Choices:** Helps businesses plan based on customer needs and spending habits.

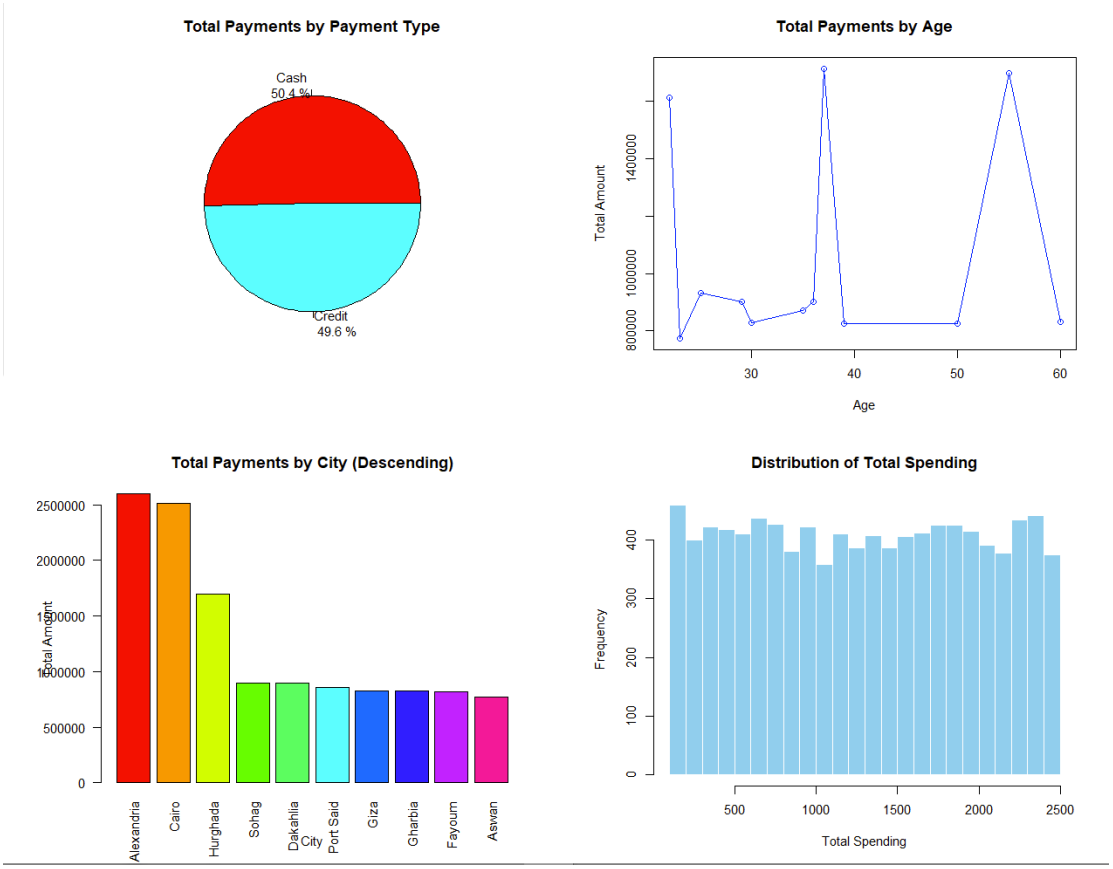
Input of the program

1. Excel file
2. Number of clusters
3. Minimum support
4. Minimum confidence



Output of the program

1. Graphs



2. Cluster results

Graphs | Cluster Results | Association Rules

Show 10 entries Search:

	customer	age	total	Cluster
1	Farida	22	794570	1
2	Hanan	22	818543	2
3	Eman	23	772871	1
4	Mohamed	25	930510	3
5	Walaa	29	900797	3
6	Ahmed	30	829587	2
7	Sameh	35	869668	3
8	Magdy	36	901010	3
9	Rania	37	893789	3
10	Sayed	37	820900	2

Showing 1 to 10 of 15 entries Previous 1 2 Next

3. Association Rules

Graphs | Cluster Results | Association Rules

Show 10 entries Search:

	rules	support	confidence	coverage	lift	count
1	{items=rolls/buns} => {count=[1,2)}	0.01108512152954337	1	0.01108512152954337	4.558646267964765	109
2	{items=bottled beer} => {count=[1,2)}	0.01220380351876335	1	0.01220380351876335	4.558646267964765	120
3	{items=whole milk} => {count=[1,2)}	0.01230550188141971	1	0.01230550188141971	4.558646267964765	121
4	{items=soda} => {count=[1,2)}	0.01576324621173599	1	0.01576324621173599	4.558646267964765	155
5	{items=canned beer} => {count=[1,2)}	0.02633987592799756	1	0.02633987592799756	4.558646267964765	259
6	{customer=Sayed} => {city=Cairo}	0.06284958812163124	1	0.06284958812163124	5.045151359671626	618
7	{customer=Sayed} => {rnd=[10,15]}	0.06284958812163124	1	0.06284958812163124	2.52906378600823	618
8	{customer=Sayed} => {age=[37,60]}	0.06284958812163124	1	0.06284958812163124	2.165382074432944	618
9	{customer=Ahmed} => {city=Giza}	0.06335807993491305	1	0.06335807993491305	15.78330658105939	623
10	{city=Giza} => {customer=Ahmed}	0.06335807993491305	1	0.06335807993491305	15.78330658105939	623

Showing 1 to 10 of 4,975 entries Previous 1 2 3 4 5 … 498 Next

The full description of the dataset

This dataset is an organized data which illustrates the customer's movements among the grocery.

In which in shows (items, total payment, customer names, age, city, payment type).

Projects steps

- Libraries used

```
1  ################## libraries
2  library(shiny)
3  library(readxl)
4  library(arules)
5  library(ggplot2)
6  library(DT)
7
```

- Build the main structure of shiny code,

And using “ title panel() ” to insert the title “ data science project ”.

```
10 ################## ui
11 ui <- fluidPage (
12   titlePanel("Data Science Project"),
13
```

- And write “ sidebarlayout() ” and use attributes (fileInput, numericInput) to allow users to insert data.

```
14  sidebarLayout(
15    sidebarPanel(
16      fileInput("file", "Choose Excel File", accept = ".xlsx"),
17      numericInput("n_clusters", "Number of Clusters", value = 3, min = 1),
18      numericInput("min_support", "Minimum Support", value = 0.01, min = 0, max = 1, step = 0.01),
19      numericInput("min_confidence", "Minimum Confidence", value = 0.8, min = 0, max = 1, step = 0.01),
20      actionButton("process_btn", "Process Data")
21    ),
22
```

- And use “ mainPanel ” that shows us “The graphs, Cluster results, Association rules”.

```
    mainPanel(
      tabsetPanel(
        tabPanel("Graphs",
                 fluidRow(
                   column(6, plotOutput("payment_plot")),
                   column(6, plotOutput("age_plot"))
                 ),
                 fluidRow(
                   column(6, plotOutput("city_plot")),
                   column(6, plotOutput("hist_plot"))
                 )
        ),
        tabPanel("Cluster Results", DTOutput("cluster_table")),
        tabPanel("Association Rules", DTOutput("assoc_rules"))
      )
    )
  )
)

```

- Declare variable “data, processed_data, assoc_rules”, And make sure that data not empty.

```
45  ################## server logic
46  server <- function(input, output, session) {
47
48    # Temporary data recording
49    data <- reactiveVal(NULL)
50    processed_data <- reactiveVal(NULL)
51    assoc_rules <- reactiveVal(NULL)
52
53
54    observeEvent(input$file, {
55      req(input$file)
56      data(read_xlsx(input$file$datapath))
57    })
58
```

- When pressing the button “process_btn”, the app check if data is empty or not, and clean the data.

```
60    observeEvent(input$process_btn, {
61      req(data())
62      cleaned_data <- data()
63      cleaned_data <- unique(cleaned_data)
64      cleaned_data <- na.omit(cleaned_data)
65
```

- Implement K-means algorithm to compare between total payment with customer and age.
  And use cluster number taken from user.

```
    ##### K-means clustering
    aggregated_data <- aggregate(total ~ customer + age, data = cleaned_data, sum)
    kmeans_result <- kmeans(aggregated_data[, c("age", "total")], centers = input$n_clusters)
    aggregated_data$Cluster <- kmeans_result$cluster
    processed_data(aggregated_data)

    # view K-means clustring
    output$cluster_table <- renderDT({ req(processed_data()); datatable(processed_data()) })
```

- Implement association rule and take minimum support and minimum confidence from user
  And convert the data into transactions type and apply association algorithm .

```
    ###### Association Rule Mining
    transactions <- as(cleaned_data, "transactions")
    rules <- apriori(transactions, parameter = list(supp = input$min_support, conf = input$min_confidence))
    assoc_rules(rules)

    # view association rules
    output$assoc_rules <- renderDT({ req(assoc_rules()); datatable(as(assoc_rules(), "data.frame")) })
  })
```

- This graph shows a pie chart for a comparison between total of cash and credit .

```
  ##### graphs
  # pie chart ==> total ~ payment type
  output$payment_plot <- renderPlot({
    req(data())
    payment_totals <- aggregate(total ~ paymentType, data = data(), sum)
    pie(payment_totals$total, labels = paste(payment_totals$paymentType, "\n",
                          round(100 * payment_totals$total / sum(payment_totals$total), 1), "%"),
                          col = rainbow(length(payment_totals$paymentType)), main = "Total Payments by Payment Type")
  })
```

- This graph shows a line chart for a comparison between total and age .

```
# Line Charts ==> total ~ age
output$age_plot <- renderPlot({
  req(data())
  age_totals <- aggregate(total ~ age, data = data(), sum)
  plot(age_totals$age, age_totals$total, type = "o", col = "blue", main = "Total Payments by Age", xlab = "Age", ylab = "Total Amount")
})
```

- This graph shows a bar plot for a comparison between total and city .

```
# bar plot ==> total ~ city
output$city_plot <- renderPlot({
  req(data())
  city_totals <- aggregate(total ~ city, data = data(), sum)
  city_totals <- city_totals[order(-city_totals$total), ]
  barplot(city_totals$total, names.arg = city_totals$city, col = rainbow(length(city_totals$city)),
          main = "Total Payments by City (Descending)", las = 2, xlab = "City", ylab = "Total Amount")
})
```

- This graph shows a hist plot for distribution of total spending .

```
# hist plot ==> Distribution of Total Spending
output$hist_plot <- renderPlot({
  req(data())
  hist(data()$total, breaks = 20, col = "skyblue", main = "Distribution of Total Spending",
       xlab = "Total Spending", ylab = "Frequency", border = "white")
})
}
```

- Open app using the function shiny app.

```
# open the app
shinyApp(ui = ui, server = server)
```