General Sir John Kotelawala Defence University

Faculty of Management, Social Science and Humanities

Department of Languages

BSc. in Applied Data Science Communication

Fundamentals of data mining

Assignment 02

D/ADC/23/0012 - S.A.R Maleesha

D/ADC/23/0032 - W.A.K.N Wedagedara

D/ADC/23/0033 - K.G.K Chani

D/ADC/23/0039 - H.W Kaweesha

CONTENT

TASK 01

CONTENT

Task 01-**Creating an association rule model for the app data set using R**

1. Introduction

A data mining method called association rule mining is used to find intriguing connections, or associations, within huge datasets. Finding frequent item sets —groups of items that show up together in transactions—and inferring association rules from them are the two main tasks involved. Usually, these rules are expressed as "if-then" formulations, in which the condition or premise is represented by the antecedent and the result or conclusion by the consequent.

2. Dataset

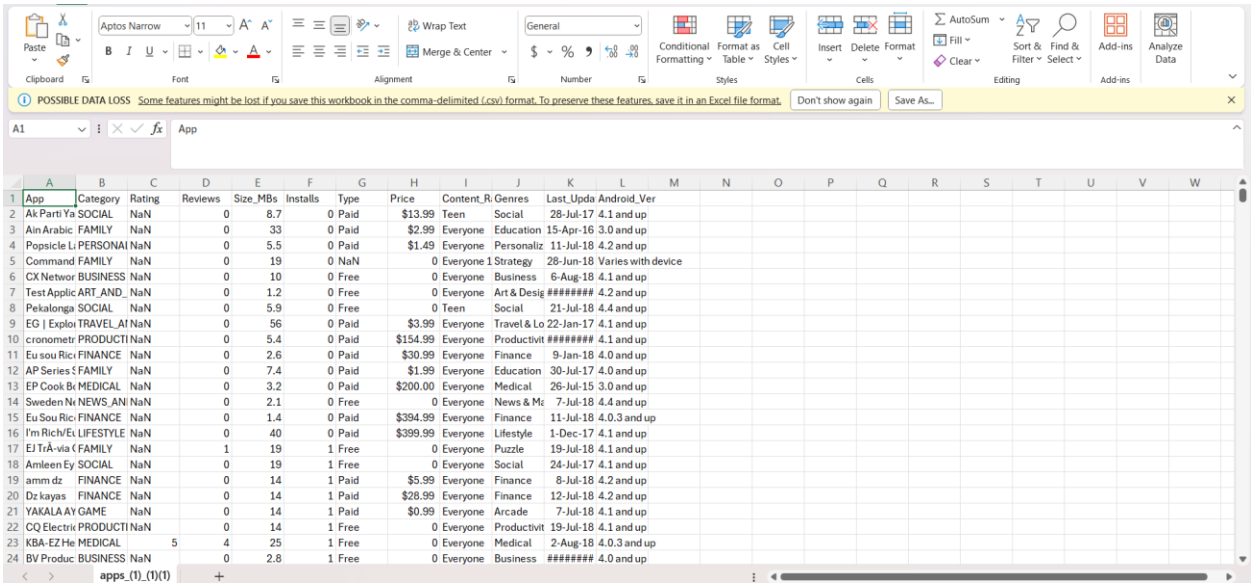
Source-

(**https://www.datacamp.com/datalab/w/86ae8eb2-a16e-49d0-89ab-679433065c40/edit)**

This dataset consists of web-scraped data from more than 10,000 Google Play Store apps. There are approximately 10831 rows and 10 columns in this dataset. Both categorical and numerical data types are in the dataset.

Attributes;

1. **App:** Categorical, The name of the mobile application.
2. **Category**: Categorical, The category or genre to which the app belongs (e.g., Social…)
3. **Size_MBs**: Numeric, The size of the app in megabytes (MB).
4. **Installs:** Numeric, The number of times the app has been installed.
5. **Type:** Categorical, Whether the app is free or paid.
6. **Price**: Numeric, The price of the app if it's paid.
7. **Content_Rating**: Categorical, The content rating for the app (e.g., Everyone, Teen…)
8. **Genres**: Categorical, Additional genres or tags associated with the app.
9. **Last_Updated**: Categorical, The date when the app was last updated.
10. **Android_Ver**: The minimum Android version required to run the app



	App	Category	Rating	Reviews	Size_MBs	Installs	Type	Price	Content_R	Genres	Last_Upda	Android_Ver
2	Ak Parti Ya	SOCIAL	NaN	0	8.7	0	Paid	$13.99	Teen	Social	28-Jul-17	4.1 and up
3	Ain Arabic	FAMILY	NaN	0	33	0	Paid	$2.99	Everyone	Education	15-Apr-16	3.0 and up
4	Popsicle La	PERSONAI	NaN	0	5.5	0	Paid	$1.49	Everyone	Personaliz	11-Jul-18	4.2 and up
5	Command	FAMILY	NaN	0	19	0	NaN	0	Everyone 1	Strategy	28-Jun-18	Varies with device
6	CX Networ	BUSINESS	NaN	0	10	0	Free	0	Everyone	Business	6-Aug-18	4.1 and up
7	Test Applic	ART_AND_	NaN	0	1.2	0	Free	0	Everyone	Art & Desig	########	4.2 and up
8	Pekalonga	SOCIAL	NaN	0	5.9	0	Free	0	Teen	Social	21-Jul-18	4.4 and up
9	EG | Explo	TRAVEL_AI	NaN	0	56	0	Paid	$3.99	Everyone	Travel & Lo	22-Jan-17	4.1 and up
10	cronometr	PRODUCTI	NaN	0	5.4	0	Paid	$154.99	Everyone	Productivit	########	4.1 and up
11	Eu sou Ric	FINANCE	NaN	0	2.6	0	Paid	$30.99	Everyone	Finance	9-Jan-18	4.0 and up
12	AP Series S	FAMILY	NaN	0	7.4	0	Paid	$1.99	Everyone	Education	30-Jul-17	4.0 and up
13	EP Cook Bo	MEDICAL	NaN	0	3.2	0	Paid	$200.00	Everyone	Medical	26-Jul-15	3.0 and up
14	Sweden Ne	NEWS_AN	NaN	0	2.1	0	Free	0	Everyone	News & Ma	7-Jul-18	4.4 and up
15	Eu Sou Ric	FINANCE	NaN	0	1.4	0	Paid	$394.99	Everyone	Finance	11-Jul-18	4.0.3 and up
16	I'm Rich/Eu	LIFESTYLE	NaN	0	40	0	Paid	$399.99	Everyone	Lifestyle	1-Dec-17	4.1 and up
17	EJ TrÃ-via	FAMILY	NaN	1	19	1	Free	0	Everyone	Puzzle	19-Jul-18	4.1 and up
18	Amleen Ey	SOCIAL	NaN	0	19	1	Free	0	Everyone	Social	24-Jul-17	4.1 and up
19	amm dz	FINANCE	NaN	0	14	1	Paid	$5.99	Everyone	Finance	8-Jul-18	4.2 and up
20	Dz kayas	FINANCE	NaN	0	14	1	Paid	$28.99	Everyone	Finance	12-Jul-18	4.2 and up
21	YAKALA AY	GAME	NaN	0	14	1	Paid	$0.99	Everyone	Arcade	7-Jul-18	4.1 and up
22	CQ Electric	PRODUCTI	NaN	0	14	1	Free	0	Everyone	Productivit	19-Jul-18	4.1 and up
23	KBA-EZ He	MEDICAL	5	4	25	1	Free	0	Everyone	Medical	2-Aug-18	4.0.3 and up
24	BV Produc	BUSINESS	NaN	0	2.8	1	Free	0	Everyone	Business	########	4.0 and up

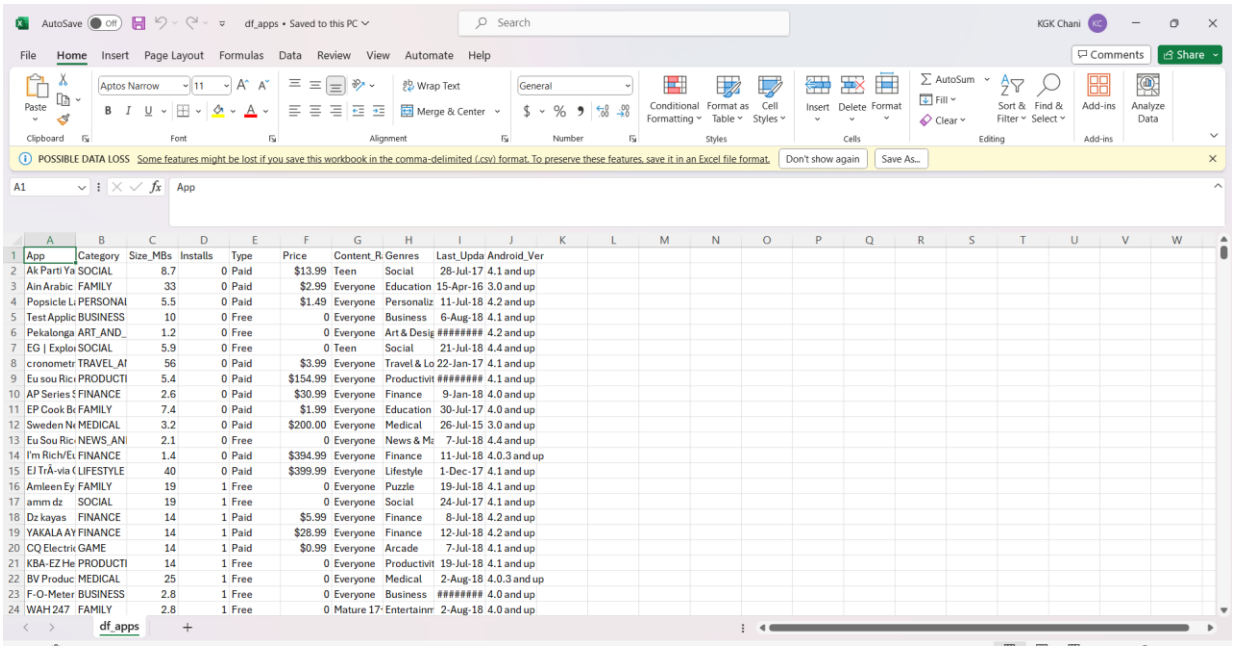
3. Explanation and preparation of data set

3.1 preparation of the dataset

Because of having null values in the data set, we cleaned the data set using na.omit() function.

```
df_apps_clean <- na.omit(as.data.frame(df_apps))
df_apps_clean
```

After cleaning the dataset and preparation , data set can be seen as follow,



3.2Data Explanation

Features such as "Category", "Size_MBs", "Type", "Price", "Content_Rating", "Genres", "Last_Updated", and "Android_Ver" that characterize each app are the independent variables in this dataset. These are the features or aspects of the applications that are employed to forecast or provide an explanation for the dependent variable's value.

The "Installs" column in this dataset, which shows how many times each app has been installed, looks to be the dependent variable. Usually, you would wish to anticipate or examine this variable considering the independent factors.

4.Data mining

Data mining is the process of discovering patterns, relationships, and insights from large datasets, typically with the aid of computational algorithms. It includes a variety of approaches and strategies meant to elicit useful insights and useful information from unprocessed data. Using diverse statistical, machine learning, and database methodologies, data mining facilitates the identification of latent patterns, trends, and correlations inside an organization that may not be readily discernible through conventional data analysis techniques. This knowledge can be retrieved and applied to various fields, including business, finance, research, and healthcare, to help with decision-making, trend prediction, process optimization, and competitive advantage.

5.Implementation in R

R Packages

```
install.packages("arules")
install.packages("arulesViz")
install.packages("tidyverse")
install.packages("RColorBrewer")

library(arules)
library(arulesViz)
library(readxl)
library(dplyr)
library(knitr)
library(ggplot2)
library(plyr)
library(magrittr)
library(tidyverse)
library(RColorBrewer)
```

- arules: This package offers the necessary tools to represent, manipulate, and examine transaction data as well as patterns found through the application of association rule mining techniques.

- arulesViz: This add-on enhances the arules package by providing tools for item sets and association rules visualization, which facilitates the interpretation and comprehension of identified patterns.

Step 2 – set as a working directory and read the data set

```
my_path="C:/Users/ASUS/Desktop/B/df_apps.csv"
setwd(my_path)
getwd()
df_apps <-read.csv("C:/Users/ASUS/Desktop/B/df_apps.csv",header=T, colClasses="factor")
df_apps
```

Step 3- Clean the data set

Data set cleaning in R involves the process of identifying and rectifying inconsistencies, missing values, outliers, and other errors to ensure data integrity and reliability for subsequent analysis.
na.omit()- by using this function can remove rows with any missing values

```
df_apps_clean <- na.omit(as.data.frame(df_apps))
df_apps_clean
```

Step 4 – Explore the data set

```
> names(df_apps_clean)
 [1] "App"            "Category"       "Size_MBs"       "Installs"       "Type"           "Price"
 [7] "Content_Rating" "Genres"         "Last_Updated"   "Android_Ver"
> head(df_apps_clean)
                                        App        Category Size_MBs Installs Type   Price
1                    Ak Parti Yardım Toplama          SOCIAL      8.7        0 Paid $13.99
2                  Ain Arabic Kids Alif Ba ta          FAMILY       33        0 Paid  $2.99
3 Popsicle Launcher for Android P 9.0 launcher PERSONALIZATION      5.5        0 Paid  $1.49
4                      Test Application DT 02        BUSINESS       10        0 Free       0
5                           Pekalongan CJ  ART_AND_DESIGN      1.2        0 Free       0
6                    EG | Explore Folegandros          SOCIAL      5.9        0 Free       0
  Content_Rating          Genres Last_Updated Android_Ver
1           Teen          Social    28-Jul-17  4.1 and up
2       Everyone       Education    15-Apr-16  3.0 and up
3       Everyone Personalization    11-Jul-18  4.2 and up
4       Everyone        Business     6-Aug-18  4.1 and up
5       Everyone    Art & Design    14-Mar-17  4.2 and up
6           Teen          Social    21-Jul-18  4.4 and up
```

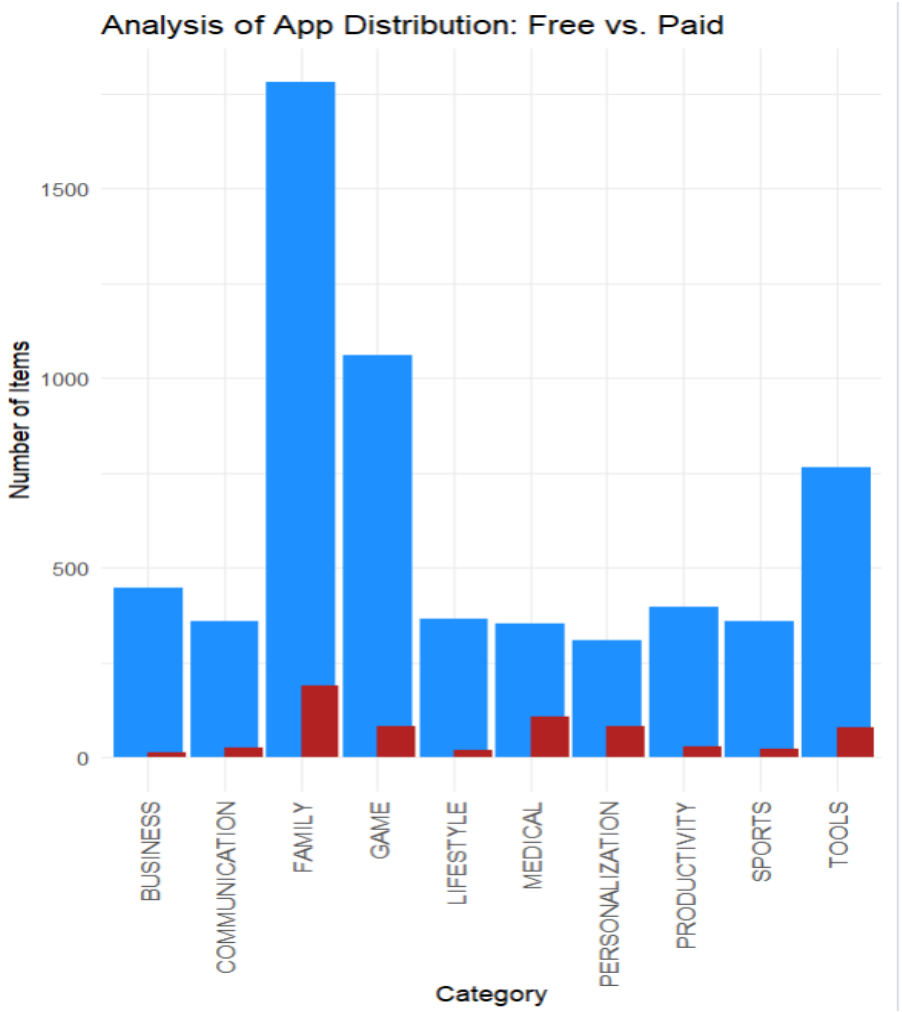
```
> tail(df_apps_clean)
                App Category Size_MBs      Installs Type Price Content_Rating Genres Last_Updated
10835 Subway Surfers     GAME       76 1,000,000,000 Free     0   Everyone 10+ Arcade    12-Jul-18
10836 Subway Surfers     GAME       76 1,000,000,000 Free     0   Everyone 10+ Arcade    12-Jul-18
10837 Subway Surfers     GAME       76 1,000,000,000 Free     0   Everyone 10+ Arcade    12-Jul-18
10838 Subway Surfers     GAME       76 1,000,000,000 Free     0   Everyone 10+ Arcade    12-Jul-18
10839 Subway Surfers     GAME       76 1,000,000,000 Free     0   Everyone 10+ Arcade    12-Jul-18
10840                    GAME       76 1,000,000,000 Free     0   Everyone 10+ Arcade    12-Jul-18
      Android_Ver
10835  4.1 and up
10836  4.1 and up
10837  4.1 and up
10838  4.1 and up
10839  4.1 and up
10840  4.1 and up
```

```
> summary(df_apps_clean)
                                                 App                Category       Size_MBs
 ROBLOX                                            :    9   FAMILY      :1971   19     : 293
 CBS Sports App - Scores, News, Stats & Watch Live:    8   GAME        :1144   14     : 267
 8 Ball Pool                                       :    7   TOOLS       : 843   12     : 262
 Candy Crush Saga                                  :    7   MEDICAL     : 463   11     : 249
 Duolingo: Learn Languages Free                    :    7   BUSINESS    : 460   13     : 191
 ESPN                                              :    7   PRODUCTIVITY: 424   36     : 189
 (Other)                                           :10795   (Other)     :5535   (Other):9389
       Installs      Type           Price              Content_Rating           Genres
 1,000,000 :1579   Free:10040   0      :10040   Adults only 18+:   3   Tools        : 842
 10,000,000:1252   Paid:  800   $0.99  :  148   Everyone       :8715   Entertainment: 623
 100,000   :1169                $2.99  :  129   Everyone 10+   : 413   Education    : 549
 10,000    :1054                $1.99  :   73   Mature 17+     : 499   Medical      : 463
 1,000     : 908                $4.99  :   72   Teen           :1208   Business     : 460
 5,000,000 : 752                $3.99  :   63   Unrated        :   2   Productivity : 424
 (Other)   :4126                (Other):  315                           (Other)      :7479
    Last_Updated             Android_Ver
 3-Aug-18 : 326   4.1 and up        :2451
 2-Aug-18 : 304   4.0.3 and up      :1501
 31-Jul-18: 294   4.0 and up        :1376
 1-Aug-18 : 285   Varies with device:1361
 30-Jul-18: 211   4.4 and up        : 980
 25-Jul-18: 164   2.3 and up        : 652
```

Step 5- Using bar plot() function

```
data <- data.frame(Category = c("FAMILY", "GAME", "TOOLS", "MEDICAL", "BUSINESS", "PRODUCTIVITY",
                                "PERSONALIZATION", "COMMUNICATION", "SPORTS", "LIFESTYLE"),
                   Free = c(1780, 1061, 765, 354, 446, 396, 309, 360, 360, 364),
                   Paid = c(191, 83, 78, 109, 14, 28, 83, 27, 24, 19))
free_data <-data.frame(Category = data$Category, Value = data$Free)
paid_data <- data.frame(Category = data$Category, Value = data$Paid)
ggplot() +
  geom_bar(data = free_data, aes(x = Category, y = Value),
           stat = "identity", fill = "dodgerblue") +
  geom_bar(data = paid_data, aes(x = Category, y = Value),
           stat = "identity", fill = "firebrick", width = 0.5, position = position_nudge(0.25))+

  labs(title = "Analysis of App Distribution: Free vs. Paid",
       x = "Category",
       y = "Number of Items",
       fill = " Type") +
  theme_minimal() +
  theme(legend.position = "top",
  axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1, size=10))
```

Output -



Step 6-Apply Apriori function

```
> rules <- apriori(df_apps_clean)
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target  ext
        0.8    0.1    1 none FALSE            TRUE       5     0.1      1     10  rules TRUE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 1084

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[11813 item(s), 10840 transaction(s)] done [0.04s].
sorting and recoding items ... [13 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 done [0.00s].
writing ... [58 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
>
```

Step 07-The Apriori method will be used to generate the rules

The arules package contains the apriori() function. Frequent item sets are sought after by the algorithm. To create groups of objects that occur together frequently and to develop rules describing their associations, a computational approach will be used. The rules are filtered using Supp=0.001 and Conf=0.8 values. The rules that yield at least 0.1% support and 80% confidence level will be the only ones that are returned. We validate the rule summary and rank the rules according to decreasing levels of confidence.

```
> rules <- apriori(df_apps_clean,
+                  parameter =list(minlen=1,maxlen=10, conf = 0.8))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target  ext
        0.8    0.1    1 none FALSE            TRUE       5     0.1      1     10  rules TRUE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 1084

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[11813 item(s), 10840 transaction(s)] done [0.05s].
sorting and recoding items ... [13 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 done [0.00s].
writing ... [58 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
> |
```

- summary of these rules

```
> summary(rules)
set of 58 rules

rule length distribution (lhs + rhs):sizes
 1  2  3  4
 3 21 26  8

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   2.000   3.000   2.672   3.000   4.000

summary of quality measures:
    support         confidence        coverage           lift            count
 Min.   :0.1025   Min.   :0.8040   Min.   :0.1025   Min.   :0.9751   Min.   : 1111
 1st Qu.:0.1152   1st Qu.:0.9407   1st Qu.:0.1155   1st Qu.:1.0161   1st Qu.: 1249
 Median :0.1289   Median :0.9976   Median :0.1344   Median :1.0771   Median : 1397
 Mean   :0.2422   Mean   :0.9671   Mean   :0.2539   Mean   :1.0489   Mean   : 2626
 3rd Qu.:0.1670   3rd Qu.:1.0000   3rd Qu.:0.1807   3rd Qu.:1.0797   3rd Qu.: 1810
 Max.   :0.9262   Max.   :1.0000   Max.   :1.0000   Max.   :1.0797   Max.   :10040

mining info:
          data ntransactions support confidence
 df_apps_clean         10840     0.1        0.8
                                                                                  call
 apriori(data = df_apps_clean, parameter = list(minlen = 1, maxlen = 10, conf = 0.8))
> |
```

Step 8- inspect the rules

```
> inspect(rules)
     lhs                                rhs                           support confidence  coverage      lift count
[1]  {}                              => {Content_Rating=Everyone} 0.8039668  0.8039668 1.0000000 1.0000000  8715
[2]  {}                              => {Type=Free}               0.9261993  0.9261993 1.0000000 1.0000000 10040
[3]  {}                              => {Price=0}                 0.9261993  0.9261993 1.0000000 1.0000000 10040
[4]  {Content_Rating=Teen}           => {Type=Free}               0.1066421  0.9569536 0.1114391 1.0332049  1156
[5]  {Content_Rating=Teen}           => {Price=0}                 0.1066421  0.9569536 0.1114391 1.0332049  1156
[6]  {Installs=10,000,000}           => {Type=Free}               0.1152214  0.9976038 0.1154982 1.0770942  1249
[7]  {Installs=10,000,000}           => {Price=0}                 0.1152214  0.9976038 0.1154982 1.0770942  1249
[8]  {Android_Ver=Varies with device} => {Type=Free}               0.1194649  0.9515062 0.1255535 1.0273235  1295
[9]  {Android_Ver=Varies with device} => {Price=0}                 0.1194649  0.9515062 0.1255535 1.0273235  1295
[10] {Android_Ver=4.0 and up}        => {Content_Rating=Everyone} 0.1079336  0.8502907 0.1269373 1.0576192  1170
[11] {Android_Ver=4.0 and up}        => {Type=Free}               0.1174354  0.9251453 0.1269373 0.9988621  1273
[12] {Android_Ver=4.0 and up}        => {Price=0}                 0.1174354  0.9251453 0.1269373 0.9988621  1273
[13] {Android_Ver=4.0.3 and up}      => {Type=Free}               0.1302583  0.9407062 0.1384686 1.0156629  1412
[14] {Android_Ver=4.0.3 and up}      => {Price=0}                 0.1302583  0.9407062 0.1384686 1.0156629  1412
[15] {Installs=1,000,000}            => {Type=Free}               0.1434502  0.9848005 0.1456642 1.0632707  1555
[16] {Installs=1,000,000}            => {Price=0}                 0.1434502  0.9848005 0.1456642 1.0632707  1555
[17] {Category=FAMILY}               => {Type=Free}               0.1642066  0.9030949 0.1818266 0.9750546  1780
[18] {Category=FAMILY}               => {Price=0}                 0.1642066  0.9030949 0.1818266 0.9750546  1780
[19] {Android_Ver=4.1 and up}        => {Type=Free}               0.2140221  0.9465524 0.2261070 1.0219749  2320
[20] {Android_Ver=4.1 and up}        => {Price=0}                 0.2140221  0.9465524 0.2261070 1.0219749  2320
[21] {Content_Rating=Everyone}       => {Type=Free}               0.7398524  0.9202524 0.8039668 0.9935793  8020
[22] {Content_Rating=Everyone}       => {Price=0}                 0.7398524  0.9202524 0.8039668 0.9935793  8020
[23] {Type=Free}                     => {Price=0}                 0.9261993  1.0000000 0.9261993 1.0796813 10040
[24] {Price=0}                       => {Type=Free}               0.9261993  1.0000000 0.9261993 1.0796813 10040
```

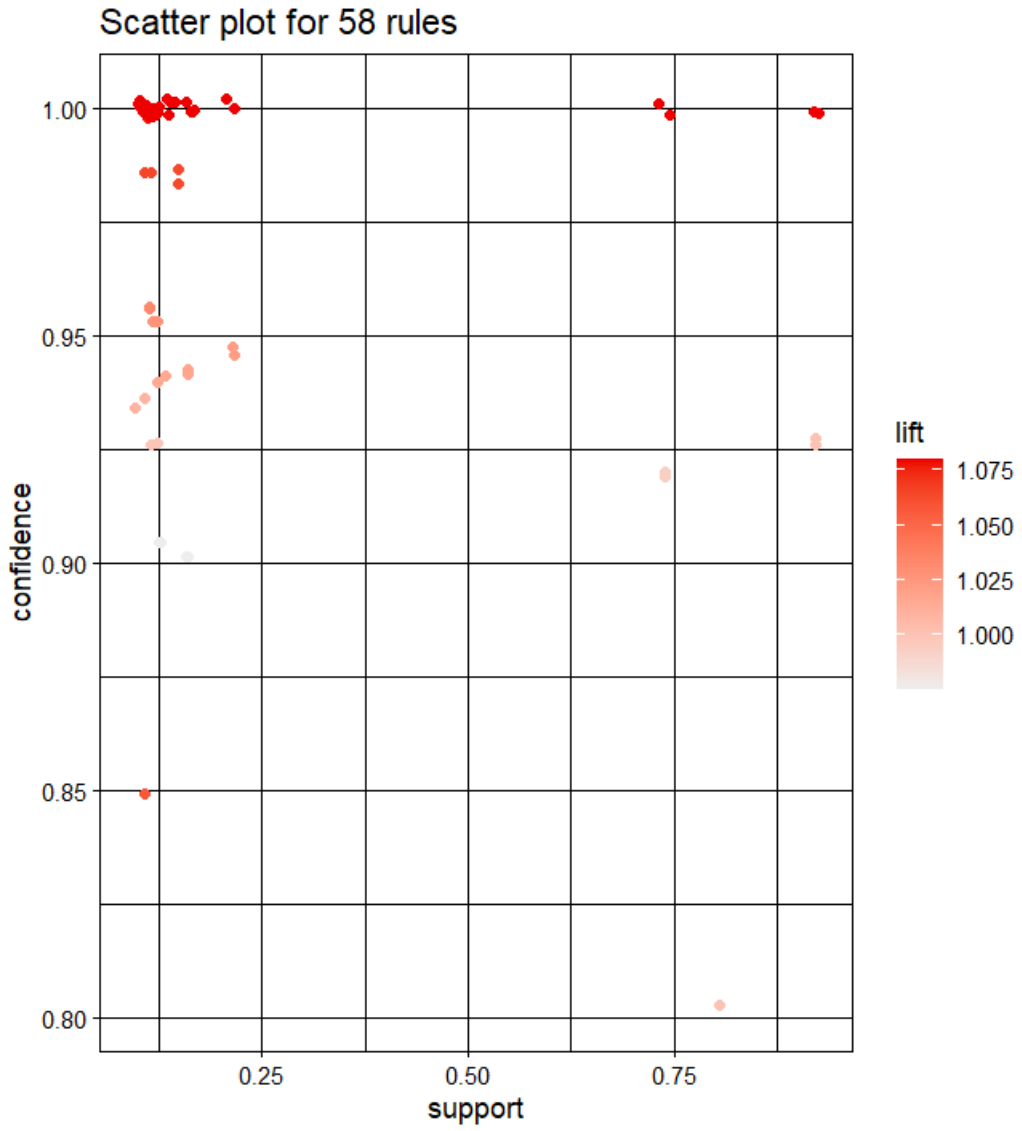
```
> summary(df_apps_clean)
                                          App                Category       Size_MBs          Installs
 ROBLOX                                     :    9   FAMILY      :1971   19     : 293   1,000,000 :1579
 CBS Sports App - Scores, News, Stats & Watch Live:    8   GAME        :1144   14     : 267   10,000,000:1252
 8 Ball Pool                                :    7   TOOLS       : 843   12     : 262   100,000   :1169
 Candy Crush Saga                           :    7   MEDICAL     : 463   11     : 249   10,000    :1054
 Duolingo: Learn Languages Free             :    7   BUSINESS    : 460   13     : 191   1,000     : 908
 ESPN                                       :    7   PRODUCTIVITY: 424   36     : 189   5,000,000 : 752
 (Other)                                    :10795   (Other)     :5535   (Other):9389   (Other)   :4126
   Type           Price              Content_Rating           Genres        Last_Updated
 Free:10040   0      :10040   Adults only 18+:   3   Tools        : 842   3-Aug-18 : 326
 Paid:  800   $0.99  :  148   Everyone       :8715   Entertainment: 623   2-Aug-18 : 304
              $2.99  :  129   Everyone 10+   : 413   Education    : 549   31-Jul-18: 294
              $1.99  :   73   Mature 17+     : 499   Medical      : 463   1-Aug-18 : 285
              $4.99  :   72   Teen           :1208   Business     : 460   30-Jul-18: 211
              $3.99  :   63   Unrated        :   2   Productivity : 424   25-Jul-18: 164
              (Other):  315                           (Other)      :7479   (Other)  :9256
            Android_Ver
 4.1 and up        :2451
 4.0.3 and up      :1501
 4.0 and up        :1376
 Varies with device:1361
 4.4 and up        : 980
 2.3 and up        : 652
 (Other)           :2519
>
```

Step 9-Visualization

Plot the rules

```
> library(arulesViz)
> plot(rules)
```

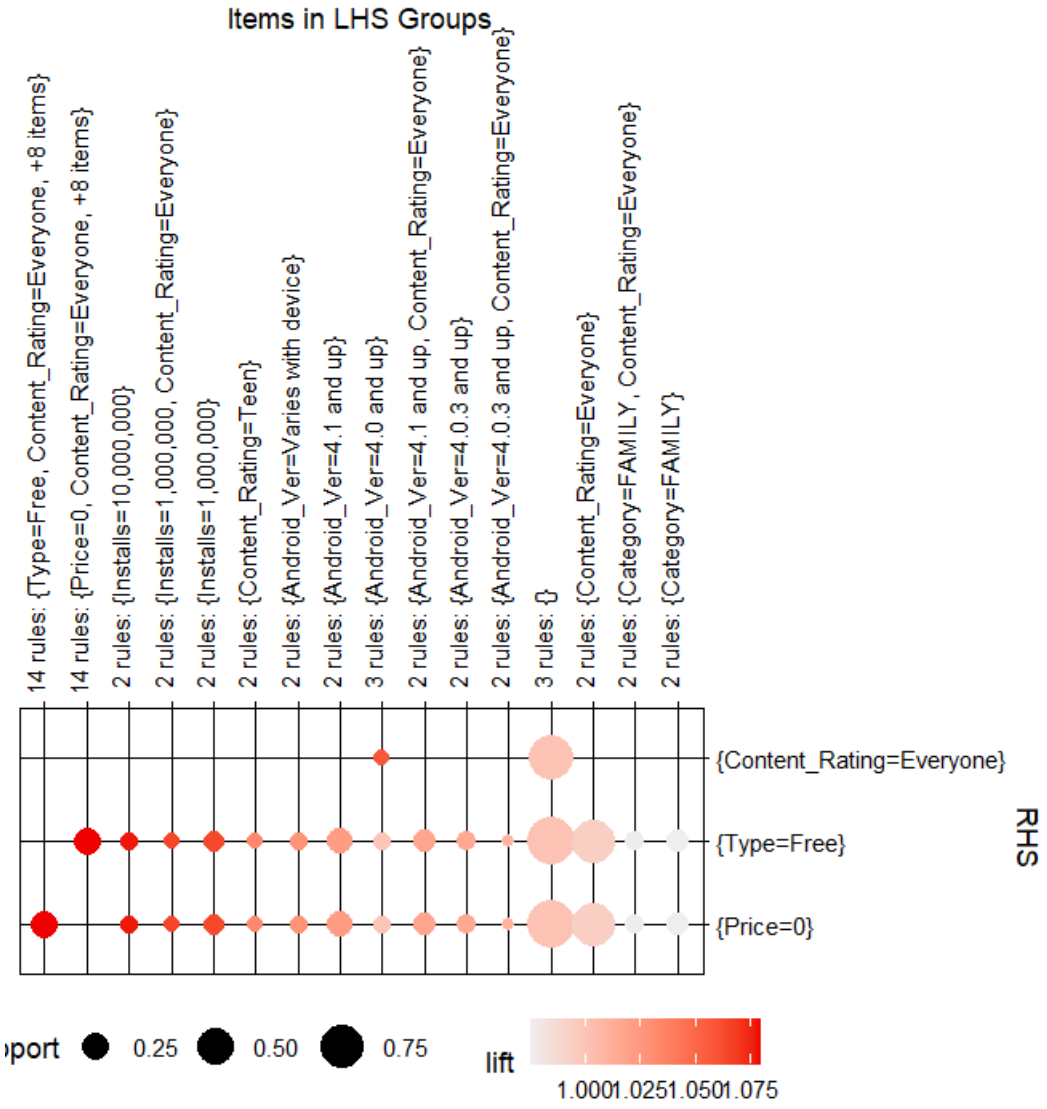
Output –



- Plot the rules in groups

```
> plot(rules, method="grouped")
>
```

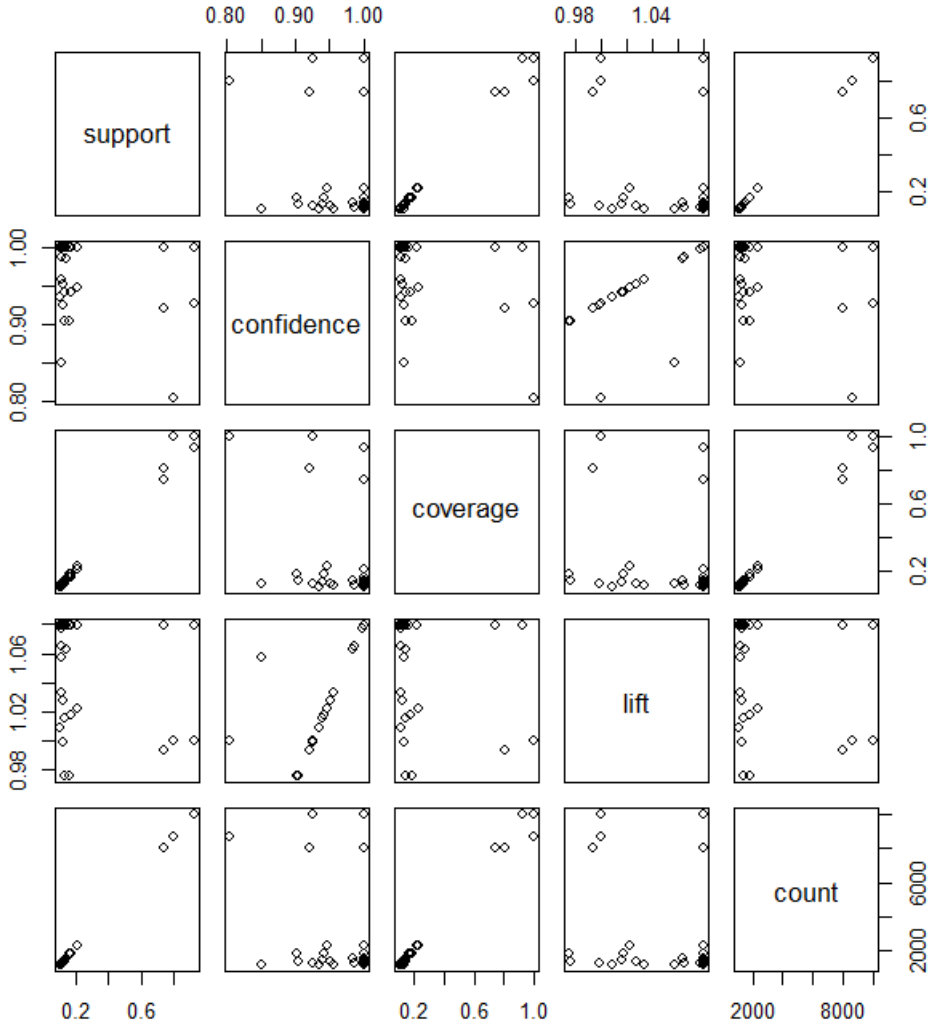
Output



The below code displays a scatterplot matrix to compare the support, confidence, and lift

```
> plot(rules@quality)
>
```

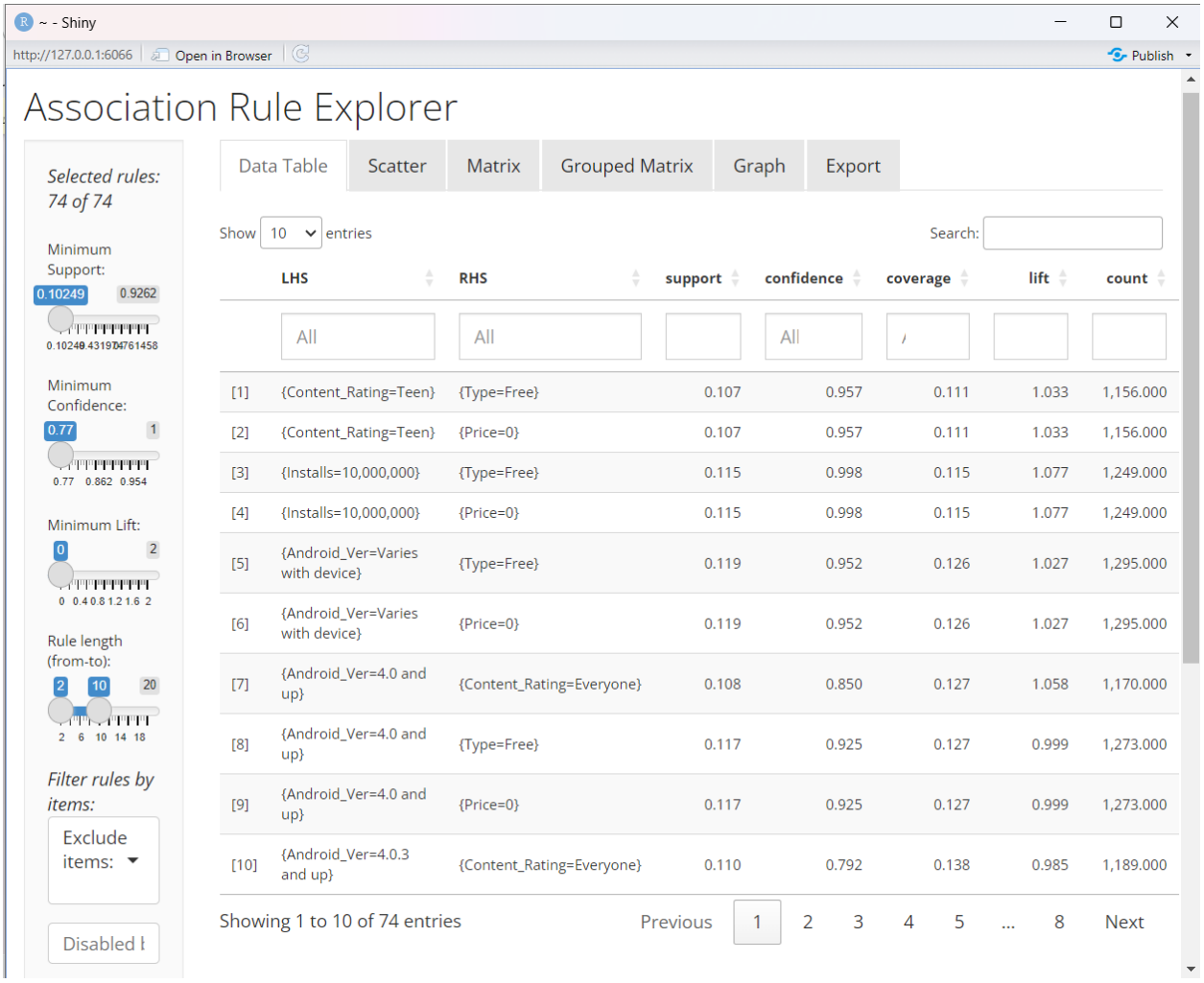
Output –



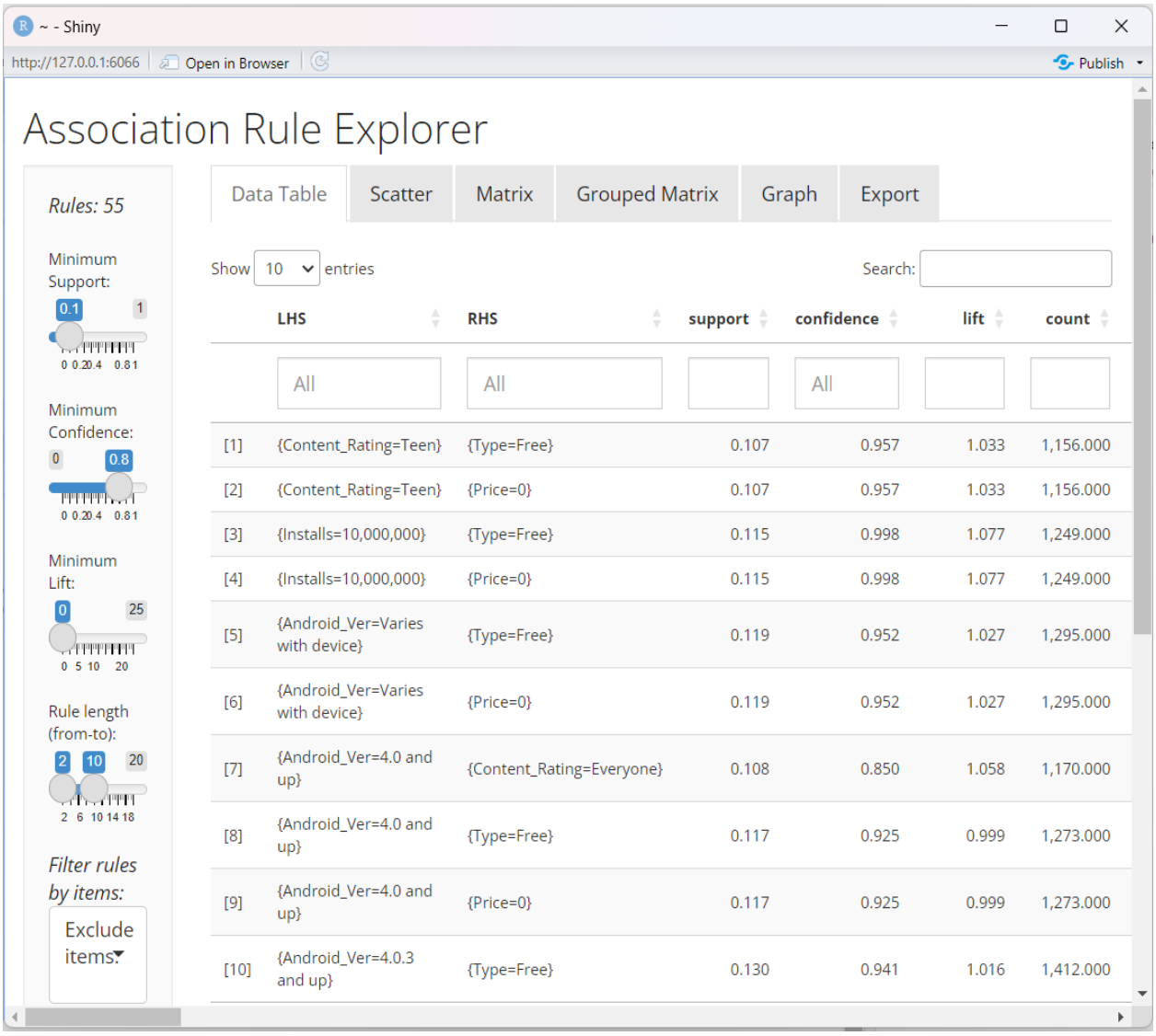
Step 10-Association Rules using ruleExplorer() function.

The ruleExplorer() function in R's arulesViz package provides an interactive graphical user interface (GUI) for exploring association rules generated by the arules package. The ruleExplorer() function in R's arulesViz package provides an interactive graphical user interface (GUI) for exploring association rules generated by the arules package

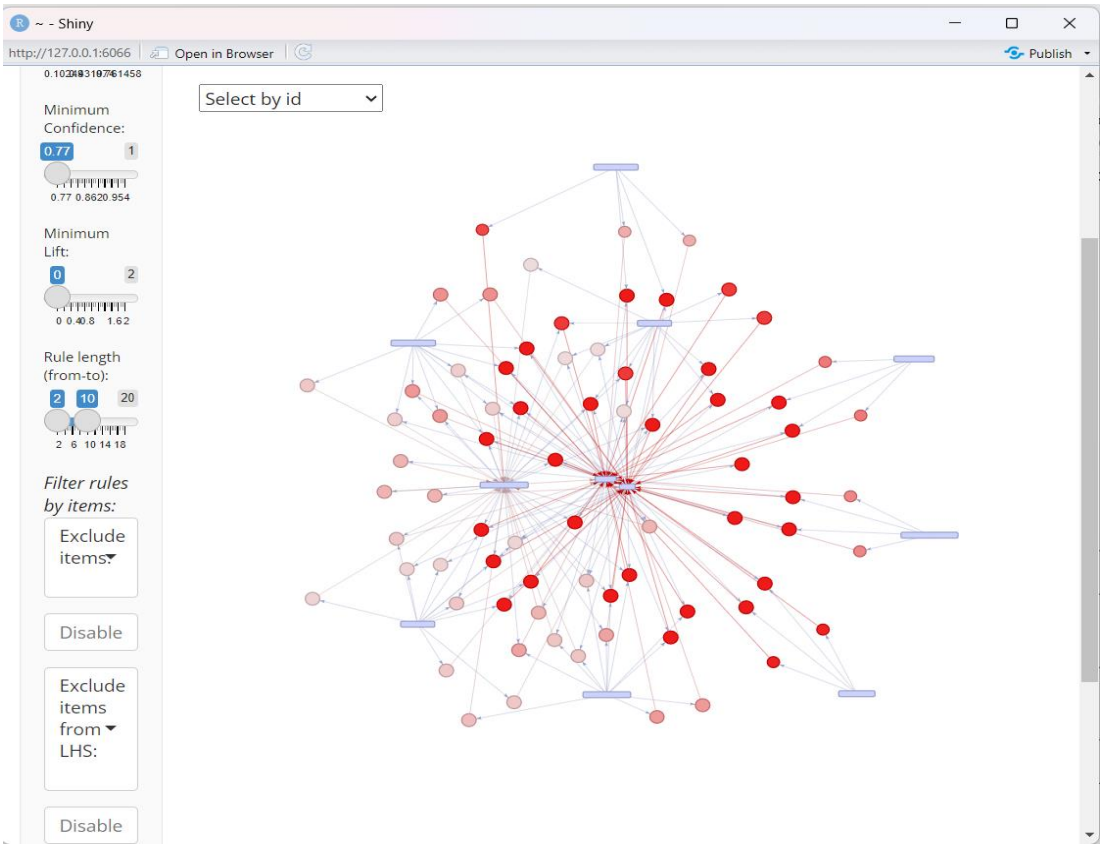
```
library(shiny)

rules_ex <-apriori(df_apps_clean,
                   parameter =list(minlen=2,maxlen=4,conf=0.75))
ruleExplorer(rules_ex)
```



Step 11-ruleExplorer() function use for the data set



Graph Tab



6. Results analysis and discussion

We must transform this dataset into a transactional format, where each row denotes a transaction (or a group of goods purchased together), to conduct association rule mining on it. In this instance, each distinct collection of Category, Type, Content_Rating, and Genres can be thought of as an item set within a transaction. Then, by using association rule mining tools, we can find intriguing connections between various items.

The scatter plot shows the relationship between lift and support for 58 rules at different confidence levels.

- Lift is a measure that indicates how much more accurate a particular rule is at forecasting a result than pure chance. If the lift is 1, then the rule is no better than chance, and if it is greater than 1, then the rule is superior.

- Support: This represents the portion of the population to which the regulation is applicable. A regulation that has a support of 0.25, for instance, indicates that it applies to 25% of the population.

The scatter plot suggests that there is a general tendency where lower support values are correlated with larger lift values. Accordingly, rules with a lower population share (lower support) typically have a higher lift or predictive power. This pattern is not unheard of, though, as certain rules have a high lift and strong support.

The points are dispersed around the plot at all confidence levels, suggesting that the confidence levels do not affect either lift or support.

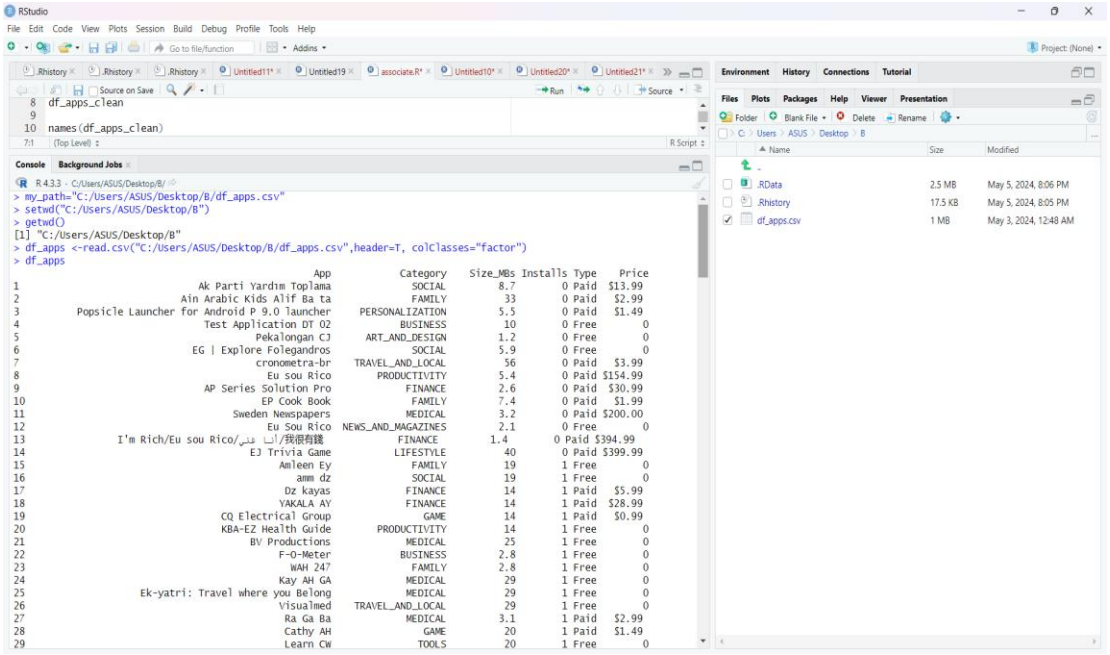
Conclusion

In conclusion, the dataset of over 10,000 Google Play Store apps was converted into a transactional format to facilitate association rule mining. Each different collection of Category, Type, Content_Rating, and Genres was treated as an item set within a transaction. The scatter plot depicts the link between lift and support for 58 rules with varying confidence levels.

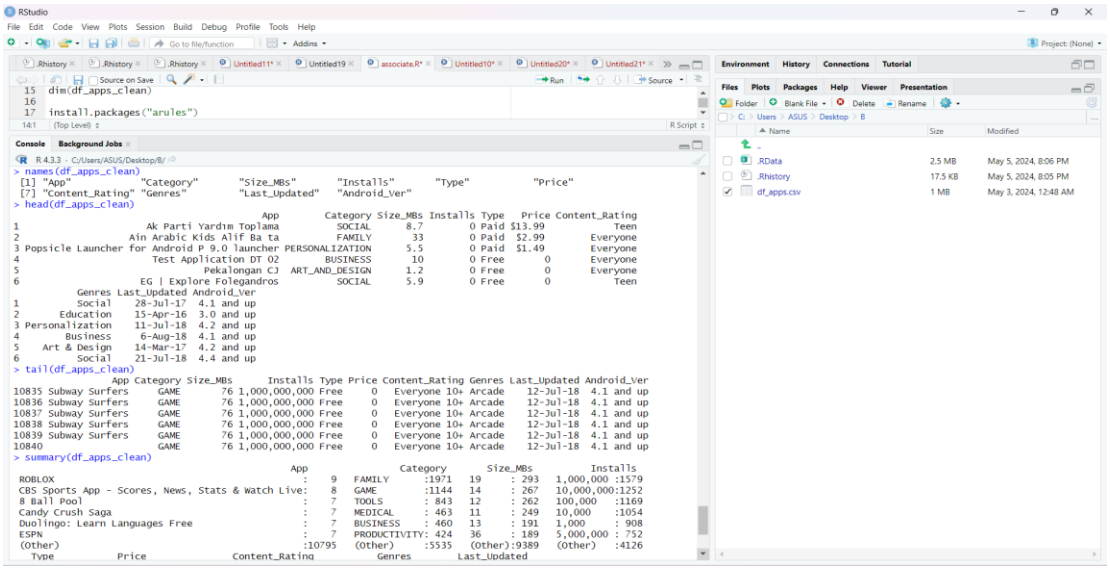
Lift is a measure of how well a rule predicts a result as compared to pure chance. The scatter plot indicates a general pattern in which lower support values are connected with higher lift values, implying that rules with a smaller population share have greater predictive potential. However, there are certain exceptions to this pattern, such as rules with high lift and strong support. Support represents the portion of the population to which the rule applies. Confidence levels do not appear to affect either lift or support, as the points are dispersed around the plot at all confidence levels.

Overall, association rule mining can help discover interesting connections between various items in the dataset, providing insights into the relationships between different attributes of the apps in the Google Play Store. The scatter plot provides a visual representation of the relationship between lift and support, highlighting the rules with higher predictive power and lower population share.

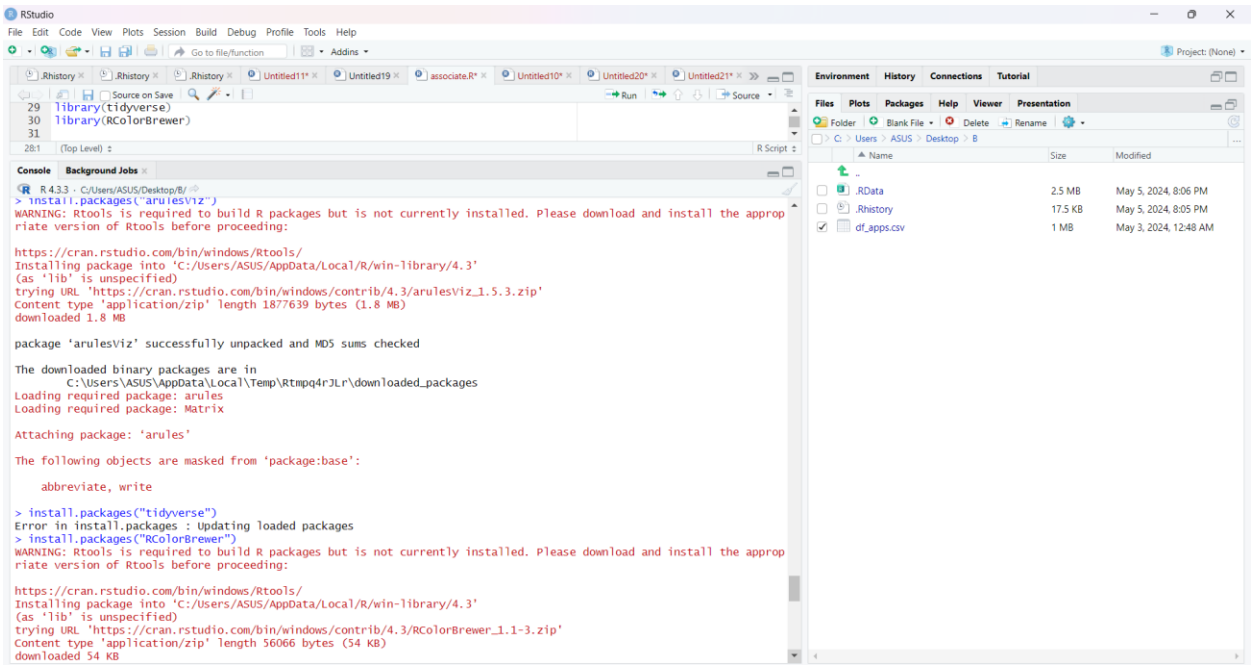
Appendices

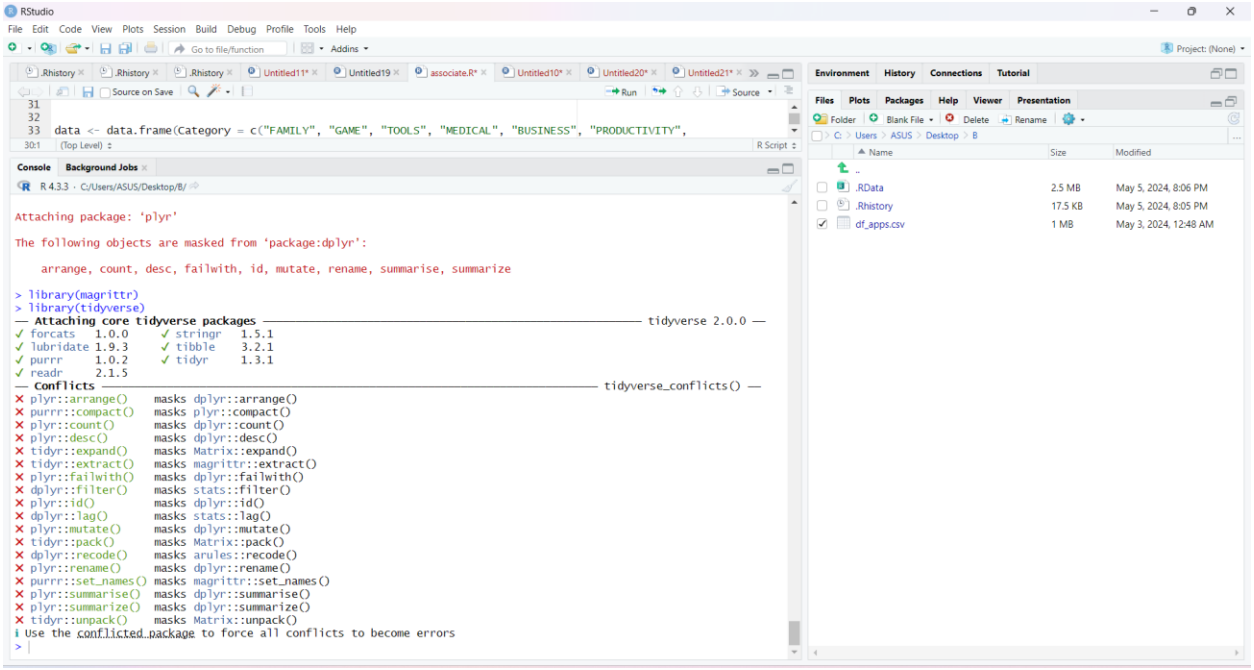


```
> df_apps_clean

> names(df_apps_clean)
```



```
> dim(df_apps_clean)

> install.packages("arules")
```

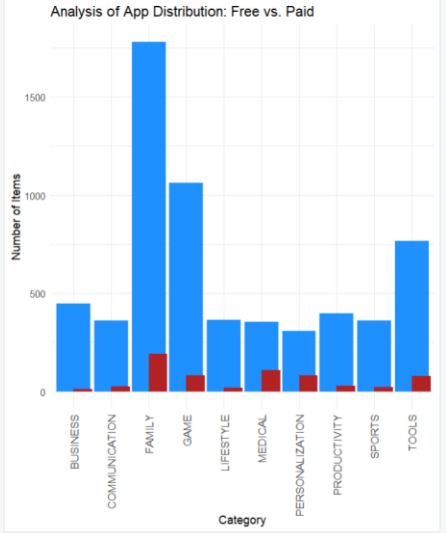

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
29 library(tidyverse)
30 library(RColorBrewer)
31
Console Background Jobs
R 4.3.3 · C:/Users/ASUS/Desktop/B/
> install.packages("arulesViz")
WARNING: Rtools is required to build R packages but is not currently installed. Please download and install the appropriate version of Rtools before proceeding:
https://cran.rstudio.com/bin/windows/Rtools/
Installing package into 'C:/Users/ASUS/AppData/Local/R/win-library/4.3'
(as 'lib' is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.3/arulesViz_1.5.3.zip'
Content type 'application/zip' length 1877639 bytes (1.8 MB)
downloaded 1.8 MB
package 'arulesViz' successfully unpacked and MD5 sums checked
The downloaded binary packages are in
C:\Users\ASUS\AppData\Local\Temp\Rtmpq4rJLr\downloaded_packages
Loading required package: arules
Loading required package: Matrix
Attaching package: 'arules'
The following objects are masked from 'package:base':
abbreviate, write
> install.packages("tidyverse")
Error in install.packages : Updating loaded packages
> install.packages("RColorBrewer")
WARNING: Rtools is required to build R packages but is not currently installed. Please download and install the appropriate version of Rtools before proceeding:
https://cran.rstudio.com/bin/windows/Rtools/
Installing package into 'C:/Users/ASUS/AppData/Local/R/win-library/4.3'
(as 'lib' is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.3/RColorBrewer_1.1-3.zip'
Content type 'application/zip' length 56066 bytes (54 KB)
downloaded 54 KB
Environment History Connections Tutorial
Files Plots Packages Help Viewer Presentation
Name Size Modified
.RData 2.5 MB May 5, 2024, 8:06 PM
.Rhistory 17.5 KB May 5, 2024, 8:05 PM
df_apps.csv 1 MB May 3, 2024, 12:48 AM


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
33 data <- data.frame(Category = c("FAMILY", "GAME", "TOOLS", "MEDICAL", "BUSINESS", "PRODUCTIVITY",
Console Background Jobs
R 4.3.3 · C:/Users/ASUS/Desktop/B/
Attaching package: 'plyr'
The following objects are masked from 'package:dplyr':
arrange, count, desc, failwith, id, mutate, rename, summarise, summarize
> library(magrittr)
> library(tidyverse)
── Attaching core tidyverse packages ──── tidyverse 2.0.0 ──
✓ forcats 1.0.0 ✓ stringr 1.5.1
✓ lubridate 1.9.3 ✓ tibble 3.2.1
✓ purrr 1.0.2 ✓ tidyr 1.3.1
✓ readr 2.1.5
── Conflicts ──── tidyverse_conflicts() ──
✖ plyr::arrange() masks dplyr::arrange()
✖ purrr::compact() masks plyr::compact()
✖ plyr::count() masks dplyr::count()
✖ plyr::desc() masks dplyr::desc()
✖ tidyr::expand() masks Matrix::expand()
✖ tidyr::extract() masks magrittr::extract()
✖ plyr::failwith() masks dplyr::failwith()
✖ dplyr::filter() masks stats::filter()
✖ plyr::id() masks dplyr::id()
✖ dplyr::lag() masks stats::lag()
✖ plyr::mutate() masks dplyr::mutate()
✖ tidyr::pack() masks Matrix::pack()
✖ dplyr::recode() masks arules::recode()
✖ plyr::rename() masks dplyr::rename()
✖ purrr::set_names() masks magrittr::set_names()
✖ plyr::summarise() masks dplyr::summarise()
✖ plyr::summarize() masks dplyr::summarize()
✖ tidyr::unpack() masks Matrix::unpack()
ℹ Use the conflicted package to force all conflicts to become errors
Environment History Connections Tutorial
Files Plots Packages Help Viewer Presentation
Name Size Modified
.RData 2.5 MB May 5, 2024, 8:06 PM
.Rhistory 17.5 KB May 5, 2024, 8:05 PM
df_apps.csv 1 MB May 3, 2024, 12:48 AM

RStudio

```
55
56 rules <- apriori(df_apps_clean,
57                  parameter =list(minlen=1,maxlen=10, conf = 0.8))
```

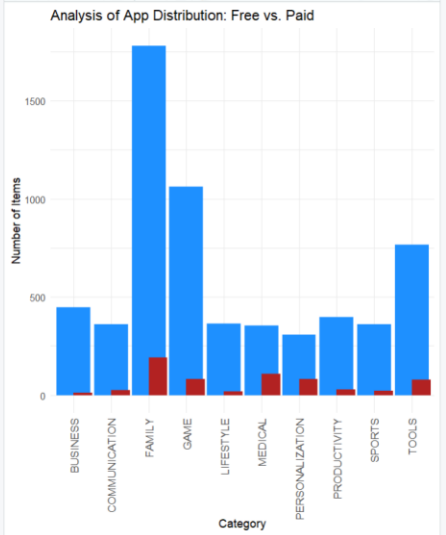
```
tidyr::extract()   masks magrittr::extract()
plyr::failwith()   masks dplyr::failwith()
dplyr::filter()    masks stats::filter()
plyr::id()         masks dplyr::id()
dplyr::lag()       masks stats::lag()
plyr::mutate()     masks dplyr::mutate()
tidyr::pack()      masks Matrix::pack()
dplyr::recode()    masks arules::recode()
plyr::rename()     masks dplyr::rename()
purrr::set_names() masks magrittr::set_names()
plyr::summarise()  masks dplyr::summarise()
plyr::summarize()  masks dplyr::summarize()
tidyr::unpack()    masks Matrix::unpack()
i Use the conflicted package to force all conflicts to become errors
> library(RColorBrewer)
> data <- data.frame(Category = c("FAMILY", "GAME", "TOOLS", "MEDICAL", "BUSINESS", "PRODUCTIVITY",
+                                 "PERSONALIZATION", "COMMUNICATION", "SPORTS", "LIFESTYLE"),
+                    Free = c(1780, 1061, 765, 354, 446, 396, 309, 360, 360, 364),
+                    Paid = c(191, 83, 78, 109, 14, 28, 83, 27, 24, 19))
> free_data <-data.frame(Category = data$Category, Value = data$Free)
> paid_data <- data.frame(Category = data$Category, Value = data$Paid)
> ggplot() +
+   geom_bar(data = free_data, aes(x = Category, y = Value),
+            stat = "identity", fill = "dodgerblue") +
+   geom_bar(data = paid_data, aes(x = Category, y = Value),
+            stat = "identity", fill = "firebrick", width = 0.5, position = position_nudge(0.25))+
+
+   labs(title = "Analysis of App Distribution: Free vs. Paid",
+        x = "Category",
+        y = "Number of Items",
+        fill = " Type") +
+   theme_minimal() +
+   theme(legend.position = "top",
+   axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1, size=10))
>
```

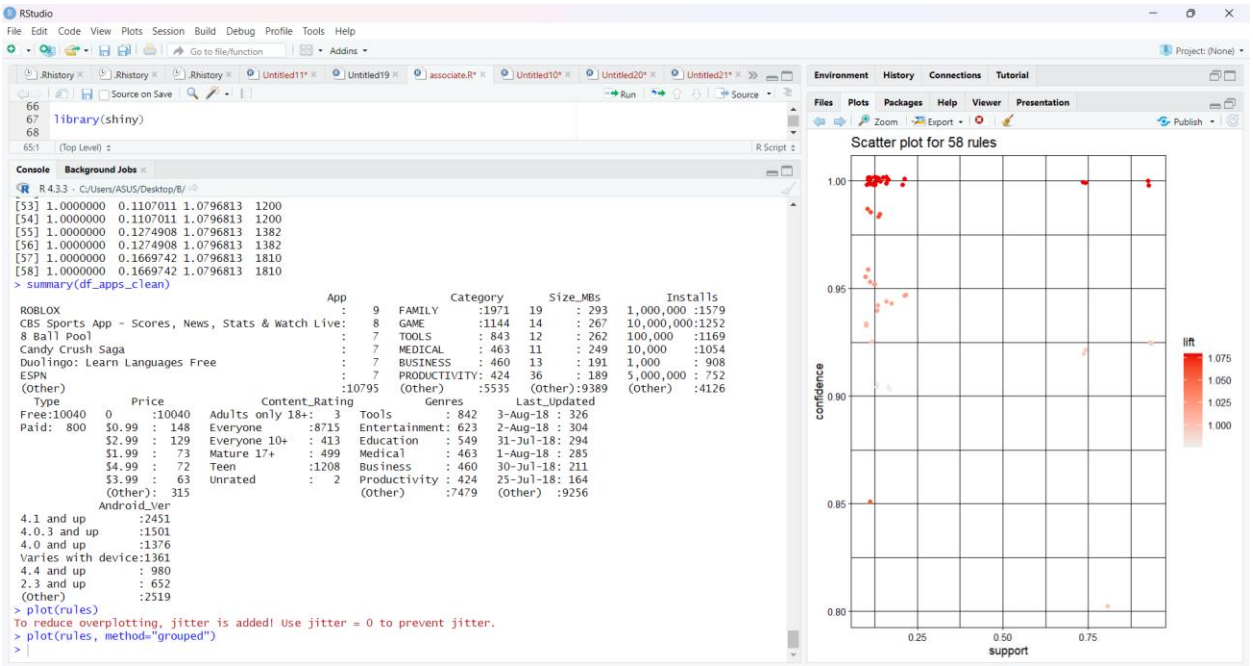


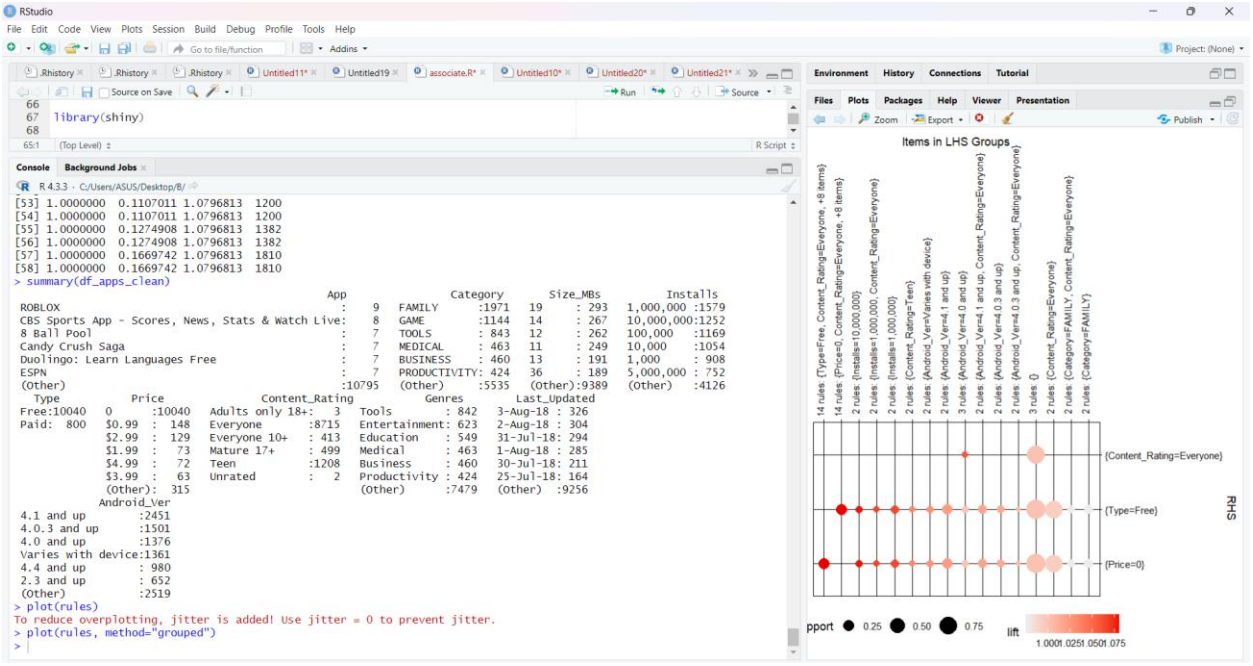
RStudio

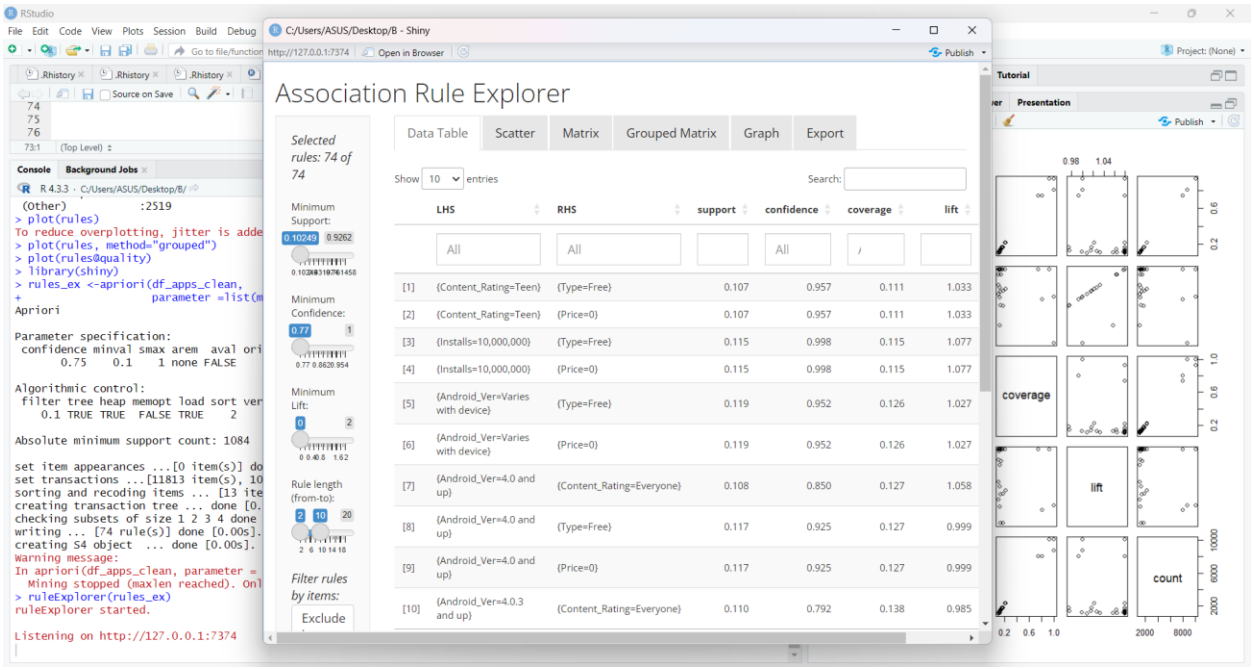
```
61
62 plot(rules)
63 plot(rules, method="grouped")
```

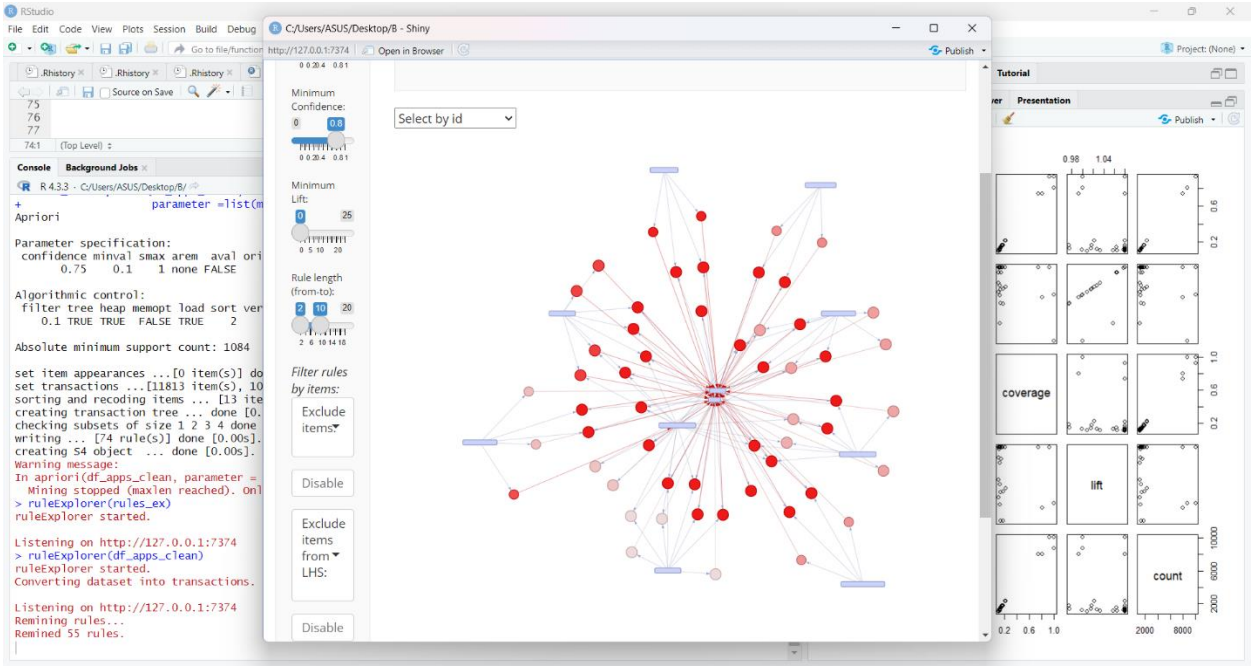
```
> rules <- apriori(df_apps_clean,
+                  parameter =list(minlen=1,maxlen=10, conf = 0.8))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target  ext
        0.8    0.1    1 none FALSE            TRUE       5     0.1      1     10  rules TRUE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 1084

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[11813 item(s), 10840 transaction(s)] done [0.04s].
sorting and recoding items ... [13 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 done [0.00s].
writing ... [58 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
> summary(rules)
set of 58 rules

rule length distribution (lhs + rhs):sizes
 1  2  3  4
 3 21 26  8

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   2.000   3.000   2.672   3.000   4.000

summary of quality measures:
    support         confidence        coverage          lift            count
 Min.   :0.1025   Min.   :0.8040   Min.   :0.1025   Min.   :0.9751   Min.   : 1111
 1st Qu.:0.1152   1st Qu.:0.9407   1st Qu.:0.1155   1st Qu.:1.0161   1st Qu.: 1249
```




RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
library(shiny)
Console Background Jobs
[53] 1.0000000 0.1107011 1.0796813 1200
[54] 1.0000000 0.1107011 1.0796813 1200
[55] 1.0000000 0.1274908 1.0796813 1382
[56] 1.0000000 0.1274908 1.0796813 1382
[57] 1.0000000 0.1669742 1.0796813 1810
[58] 1.0000000 0.1669742 1.0796813 1810
> summary(df_apps_clean)
App Category Size_MBs Installs
ROBLOX : 9 FAMILY :1971 19 : 293 1,000,000 :1579
CBS Sports App - Scores, News, Stats & Watch Live: 8 GAME :1144 14 : 267 10,000,000:1252
8 Ball Pool : 7 TOOLS : 843 12 : 262 100,000 :1169
Candy Crush Saga : 7 MEDICAL : 463 11 : 249 10,000 :1054
Duolingo: Learn Languages Free : 7 BUSINESS : 460 13 : 191 1,000 : 908
ESPN : 7 PRODUCTIVITY: 424 36 : 189 5,000,000 : 752
(Other) :10795 (Other) :5535 (Other):9389 (Other) :4126
Type Price Content_Rating Genres Last_Updated
Free:10040 0 :10040 Adults only 18+: 3 Tools : 842 3-Aug-18 : 326
Paid: 800 $0.99 : 148 Everyone :8715 Entertainment: 623 2-Aug-18 : 304
$2.99 : 129 Everyone 10+ : 413 Education : 549 31-Jul-18: 294
$1.99 : 73 Mature 17+ : 499 Medical : 463 1-Aug-18 : 285
$4.99 : 72 Teen :1208 Business : 460 30-Jul-18: 211
$3.99 : 63 Unrated : 2 Productivity : 424 25-Jul-18: 164
(Other): 315 (Other) :7479 (Other) :9256
Android_Ver
4.1 and up :2451
4.0.3 and up :1501
4.0 and up :1376
Varies with device:1361
4.4 and up : 980
2.3 and up : 652
(Other) :2519
> plot(rules)
To reduce overplotting, jitter is added! Use jitter = 0 to prevent jitter.
> plot(rules, method="grouped")
Scatter plot for 58 rules
confidence
support
lift
1.00
0.95
0.90
0.85
0.80
0.25
0.50
0.75
1.075
1.050
1.025
1.000

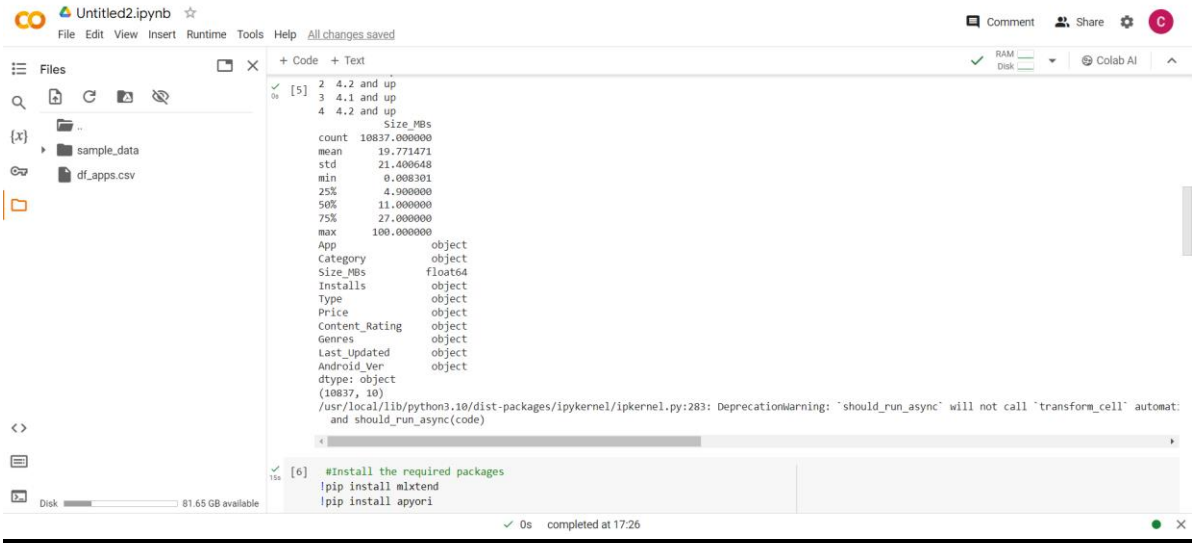

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
library(shiny)
Console Background Jobs
[53] 1.0000000 0.1107011 1.0796813 1200
[54] 1.0000000 0.1107011 1.0796813 1200
[55] 1.0000000 0.1274908 1.0796813 1382
[56] 1.0000000 0.1274908 1.0796813 1382
[57] 1.0000000 0.1669742 1.0796813 1810
[58] 1.0000000 0.1669742 1.0796813 1810
> summary(df_apps_clean)
App Category Size_MBs Installs
ROBLOX : 9 FAMILY :1971 19 : 293 1,000,000 :1579
CBS Sports App - Scores, News, Stats & Watch Live: 8 GAME :1144 14 : 267 10,000,000:1252
8 Ball Pool : 7 TOOLS : 843 12 : 262 100,000 :1169
Candy Crush Saga : 7 MEDICAL : 463 11 : 249 10,000 :1054
Duolingo: Learn Languages Free : 7 BUSINESS : 460 13 : 191 1,000 : 908
ESPN : 7 PRODUCTIVITY: 424 36 : 189 5,000,000 : 752
(Other) :10795 (Other) :5535 (Other):9389 (Other) :4126
Type Price Content_Rating Genres Last_Updated
Free:10040 0 :10040 Adults only 18+: 3 Tools : 842 3-Aug-18 : 326
Paid: 800 $0.99 : 148 Everyone :8715 Entertainment: 623 2-Aug-18 : 304
$2.99 : 129 Everyone 10+ : 413 Education : 549 31-Jul-18: 294
$1.99 : 73 Mature 17+ : 499 Medical : 463 1-Aug-18 : 285
$4.99 : 72 Teen :1208 Business : 460 30-Jul-18: 211
$3.99 : 63 Unrated : 2 Productivity : 424 25-Jul-18: 164
(Other): 315 (Other) :7479 (Other) :9256
Android_Ver
4.1 and up :2451
4.0.3 and up :1501
4.0 and up :1376
Varies with device:1361
4.4 and up : 980
2.3 and up : 652
(Other) :2519
> plot(rules)
To reduce overplotting, jitter is added! Use jitter = 0 to prevent jitter.
> plot(rules, method="grouped")
Items in LHS Groups
{Content_Rating=Everyone}
{Type=Free}
{Price=0}
RHS
pport 0.25 0.50 0.75
lift
1.000 1.025 1.050 1.075


C:/Users/ASUS/Desktop/B - Shiny
Association Rule Explorer
Selected rules: 74 of 74
Minimum Support:
Minimum Confidence:
Minimum Lift:
Rule length (from-to):
Filter rules by items:
Exclude
Data Table
Scatter
Matrix
Grouped Matrix
Graph
Export
Show 10 entries
Search:
LHS
RHS
support
confidence
coverage
lift
[1] {Content_Rating=Teen} {Type=Free} 0.107 0.957 0.111 1.033
[2] {Content_Rating=Teen} {Price=0} 0.107 0.957 0.111 1.033
[3] {Installs=10,000,000} {Type=Free} 0.115 0.998 0.115 1.077
[4] {Installs=10,000,000} {Price=0} 0.115 0.998 0.115 1.077
[5] {Android_Ver=Varies with device} {Type=Free} 0.119 0.952 0.126 1.027
[6] {Android_Ver=Varies with device} {Price=0} 0.119 0.952 0.126 1.027
[7] {Android_Ver=4.0 and up} {Content_Rating=Everyone} 0.108 0.850 0.127 1.058
[8] {Android_Ver=4.0 and up} {Type=Free} 0.117 0.925 0.127 0.999
[9] {Android_Ver=4.0 and up} {Price=0} 0.117 0.925 0.127 0.999
[10] {Android_Ver=4.0.3 and up} {Content_Rating=Everyone} 0.110 0.792 0.138 0.985
coverage
lift
count


C:/Users/ASUS/Desktop/B - Shiny
Minimum Confidence:
Minimum Lift:
Rule length (from-to):
Filter rules by items:
Select by id
Exclude items
Disable
Exclude items from LHS:
Disable
coverage
lift
count

Python Code


Untitled2.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
Files
+ Code + Text
Colab AI
sample_data
df_apps.csv
Disk 81.65 GB available
[1] import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules
[3] # Load the dataset
my_path = "/content/df_apps.csv"
df_apps = pd.read_csv(my_path, header=0)
[4] # Clean the dataset
df_apps_clean = df_apps.dropna()
# Print the first few rows of the cleaned dataset
print(df_apps_clean.head())
# Print the summary statistics of the cleaned dataset
print(df_apps_clean.describe())
# Print the structure of the cleaned dataset
print(df_apps_clean.dtypes)
0s completed at 17:26


Untitled2.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
Files
+ Code + Text
Colab AI
sample_data
df_apps.csv
Disk 81.65 GB available
[5] 2 4.2 and up
3 4.1 and up
4 4.2 and up
Size_MBs
count 10837.000000
mean 19.771471
std 21.400648
min 0.008301
25% 4.900000
50% 11.000000
75% 27.000000
max 100.000000
App object
Category object
Size_MBs float64
Installs object
Type object
Price object
Content_Rating object
Genres object
Last_Updated object
Android_Ver object
dtype: object
(10837, 10)
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `transform_cell` automat
and should_run_async(code)
[6] #Install the required packages
!pip install mlxtend
!pip install apyori
0s completed at 17:26

Untitled2.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share C

Files

+ Code + Text

RAM Disk Colab AI

sample_data

df_apps.csv

```
[6]  #Install the required packages
     !pip install mlxtend
     !pip install apyori

     import plotly.express as px
     apps_per_category = df_apps_clean['Category'].value_counts()
     apps_per_category

     fig = px.bar(apps_per_category, x=apps_per_category.index, y=apps_per_category.values)
     fig.show()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `transform_cell` autom
  and should_run_async(code)
Requirement already satisfied: mlxtend in /usr/local/lib/python3.10/dist-packages (0.22.0)
Requirement already satisfied: scipy>=1.2.1 in /usr/local/lib/python3.10/dist-packages (from mlxtend) (1.11.4)
Requirement already satisfied: numpy>=1.16.2 in /usr/local/lib/python3.10/dist-packages (from mlxtend) (1.25.2)
Requirement already satisfied: pandas>=0.24.2 in /usr/local/lib/python3.10/dist-packages (from mlxtend) (2.0.3)
Requirement already satisfied: scikit-learn>=1.0.2 in /usr/local/lib/python3.10/dist-packages (from mlxtend) (1.2.2)
Requirement already satisfied: matplotlib>=3.0.0 in /usr/local/lib/python3.10/dist-packages (from mlxtend) (3.7.1)
Requirement already satisfied: joblib>=0.13.2 in /usr/local/lib/python3.10/dist-packages (from mlxtend) (1.4.0)
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from mlxtend) (67.7.2)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib>=3.0.0->mlxtend) (1.2.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib>=3.0.0->mlxtend) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib>=3.0.0->mlxtend) (4.51.0)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib>=3.0.0->mlxtend) (1.4.5)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib>=3.0.0->mlxtend) (24.0)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib>=3.0.0->mlxtend) (9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib>=3.0.0->mlxtend) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib>=3.0.0->mlxtend) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=0.24.2->mlxtend) (2023.4)
```

Disk 81.65 GB available

0s completed at 17:26

Untitled2.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share C

Files

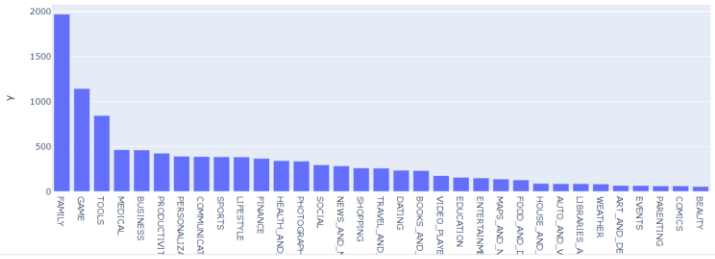
+ Code + Text

RAM Disk Colab AI

sample_data

df_apps.csv

```
[6]  Downloading apyori-1.1.2.tar.gz (8.6 kB)
     Preparing metadata (setup.py) ... done
Building wheels for collected packages: apyori
  Building wheel for apyori (setup.py) ... done
  Created wheel for apyori: filename=apyori-1.1.2-py3-none-any.whl size=5955 sha256=442ad9b408cea10f1d74b4d0ba1f0e4977f8c47f6ed4bf6fdb9de346e1
  Stored in directory: /root/.cache/pip/wheels/c4/1a/79/20f55c470a50bb3702a8cb7c94d8ada15573538c7f4baebe2d
Successfully built apyori
Installing collected packages: apyori
Successfully installed apyori-1.1.2
```



Disk 81.65 GB available

0s completed at 17:26

Untitled2.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share C

Files

+ Code + Text

RAM Disk Colab AI

sample_data

df_apps.csv

```
# Import the required libraries
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules

df_apps_clean = df_apps_clean[df_apps_clean['Category'] != "Ak Parti Yardım Toplama"]
frequent_itemsets = apriori(df_apps_clean, min_support=0.01, use_colnames=True)
rules = association_rules(frequent_itemsets, metric="lift", min_threshold=1)

# Print the summary statistics of the rules
print(rules.describe())

# Print the first few rules
print(rules.head())

# Plot the rules
plt.figure(figsize=(10,6))
sns.scatterplot(x='support', y='lift', data=rules, hue='antecedents', palette='viridis')
plt.title('Association Rules: Support vs. Lift')
plt.xlabel('Support')
plt.ylabel('Lift')
plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)
plt.show()
```

Disk 81.65 GB available

0s completed at 17:26

Task 02

CONTENT

Task 2: Apply Logistic Regression on a selected dataset using R or Python

1.Introduction

This report aims to analyze employee attrition using logistic regression techniques. Employee attrition can significantly impact a company, leading to increased recruitment costs and loss of skilled labor. By identifying factors that contribute to employee attrition, companies can implement targeted interventions to improve retention. The specific research question addressed in this report is: "What factors significantly contribute to employee attrition, and how can these insights be used to predict future attrition?".

2.Dataset

The dataset is sourced from the IBM HR Analytics department and is publicly available on github(**https://github.com/Niranjankumar-c/HRAnalyticsEmployeeAttrition**). It is designed to provide insights into employee attrition and factors that might influence it within a fictional version of IBM.

Contents: The dataset consists of approximately 1,470 employee records. Each record includes details about the employee's professional and personal life. It features 35 attributes, including both numerical and categorical data types.

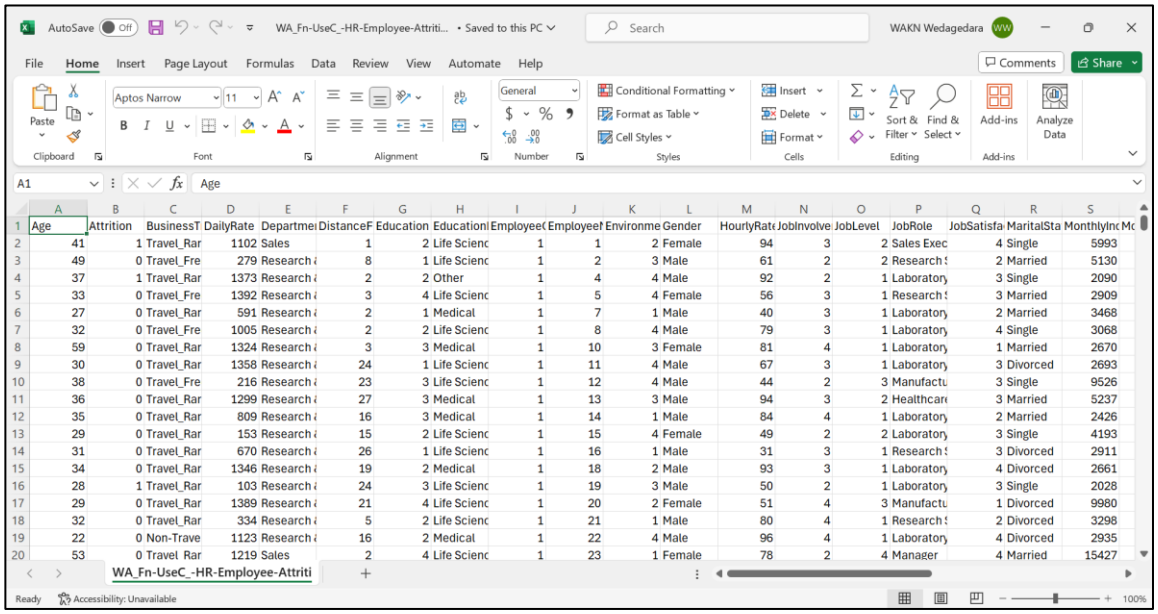
Attributes:

1. **Age:** Numeric, representing the age of the employee.
2. **Attrition:** ('1' or '0), indicating if the employee left the company.
3. **BusinessTravel:** Categorical, representing travel requirements (e.g., 'Travel_Rarely', 'Travel_Frequently', 'Non-Travel').
4. **DailyRate:** Numeric, representing the daily rate of the employee.
5. **Department:** Categorical, indicating department (e.g., 'Research & Development', 'Sales', 'Human Resources').

6. **DistanceFromHome:** Numeric, indicating the distance from work to home (in miles).
7. **Education:** Categorical, representing the level of education (1 to 5 scale).
8. **EducationField:** Categorical, such as 'Life Sciences', 'Medical', 'Technical', etc.
9. **EmployeeCount:** Numeric, always equal to 1 (likely a placeholder).
10. **EmployeeNumber:** Numeric, a unique identifier for each employee.
11. **EnvironmentSatisfaction:** Categorical, indicating satisfaction with the work environment (1 to 4 scale).
12. **Gender:** Categorical, 'Male' or 'Female'.
13. **HourlyRate:** Numeric, indicating the hourly pay rate.
14. **JobInvolvement:** Categorical, indicating the level of job involvement (1 to 4 scale).
15. **JobLevel:** Numeric, representing the level of job position.
16. **JobRole:** Categorical, indicating job title.
17. **JobSatisfaction:** Categorical, indicating job satisfaction (1 to 4 scale).
18. **Marital Status:** Categorical, such as 'Single', 'Married', or 'Divorced'.
19. **Monthly Income:** Numeric, representing monthly salary.
20. **Monthly Rate:** Numeric, representing monthly rate (not further specified).
21. **NumCompaniesWorked:** Numeric, indicating the number of companies the employee has worked for.
22. **Over18:** Categorical, indicating if the employee is over 18 years of age.
23. **Overtime:** Binary ('Yes' or 'No'), indicating if the employee works overtime.
24. **PercentSalaryHike:** Numeric, representing the percentage increase in salary last year.
25. **PerformanceRating:** Categorical, representing the performance rating (1 to 4 scale).

26. **RelationshipSatisfaction:** Categorical, indicating satisfaction with relationships at work (1 to 4 scale).
27. **StandardHours:** Numeric, likely a standard value for all employees.
28. **StockOptionLevel:** Numeric, representing the level of stock options.
29. **TotalWorkingYears:** Numeric, representing the total number of years the employee has worked.
30. **TrainingTimesLastYear:** Numeric, representing the number of trainings attended last year.
31. **WorkLifeBalance:** Categorical, indicating work-life balance satisfaction (1 to 4 scale).
32. **YearsAtCompany:** Numeric, representing the number of years spent at the company.
33. **YearsInCurrentRole:** Numeric, indicating the number of years in the current job role.
34. **YearsSinceLastPromotion:** Numeric, indicating the number of years since the last promotion.
35. **YearsWithCurrManager:** Numeric, indicating the number of years with the current manager.

Usage: To better understand the factors that contribute to employee attrition, predictive modelling makes extensive use of this dataset. Scholars and data scientists utilize diverse statistical and machine learning techniques, including logistic regression, to forecast employee attrition and detect noteworthy predictors that influence employee turnover. This analysis helps to better understand workforce dynamics and develop retention strategies.



3. **Explanation and Preparation of Datasets**

Brief Description of the Dataset

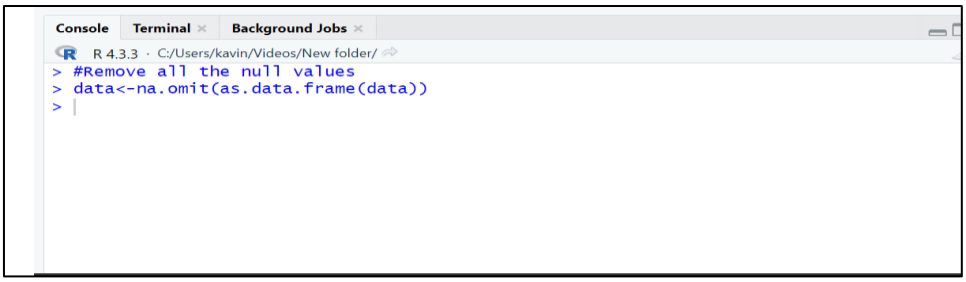
The IBM HR Analytics Employee Attrition & Performance dataset comprises comprehensive employee data from a fictional version of IBM. The dataset includes 1,470 records, each representing an individual employee, and features 35 attributes that cover various aspects of an employee's professional and personal life. These attributes include age, department, job satisfaction, monthly income, and many others.

3.1 Independent and Dependent Variables

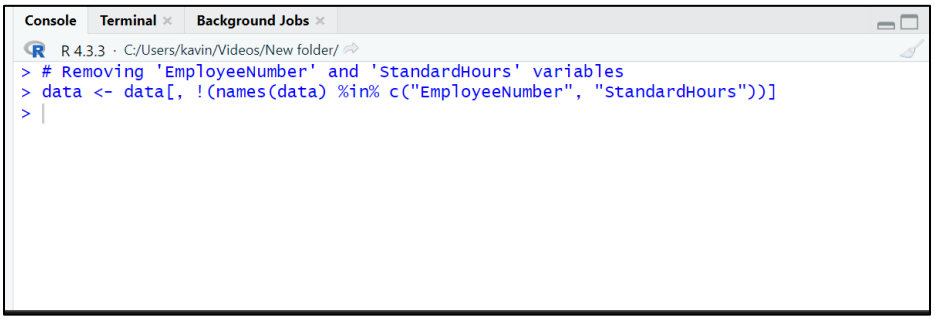
- **Dependent Variable:**
    - **Attrition:** This binary variable is the outcome of interest, indicating whether an employee has left the company ('1') or not ('0'). It is the primary variable of interest for the logistic regression analysis.
- **Independent Variables:**
    - **Age, MonthlyIncome, YearsAtCompany, JobSatisfaction,** and other similar variables serve as predictors. These variables are expected to influence the likelihood of an employee leaving the company.

3.2 Data Cleaning

1.Handling Missing Values**:** Missing data were addressed by removing rows with any missing values using the **na.omit()** function.

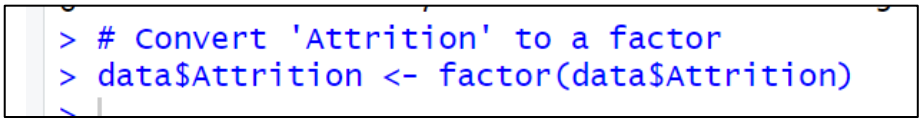


2.Removing Irrelevant Variables**:** Variables that do not influence attrition, such as **EmployeeNumber** and **StandardHours**, were removed to focus the analysis on relevant predictors.



3.3 **Data Transformation**

1. Converting Categorical Variables to Factors: Categorical variables, crucial for the analysis, were converted into factors to be properly interpreted in the logistic regression model.



4. Data Mining Techniques

4.1. Implementation in R

R Packages Used

1. **party:** Used for creating complex visualizations and implementing conditional inference trees, although it wasn't specifically mentioned in the model building, it's useful for classification problems.
2. **epitools:** Provides tools for epidemiological analysis including rates and proportions, which can enhance the understanding of attribute relationships.
3. **ggplot2:** A powerful visualization package for creating complex and aesthetically pleasing graphics in R. It was used to plot the logistic regression results and various data distributions.
4. **GGally:** Extends ggplot2 by adding several functions to reduce the complexity of combining multiple ggplot2 plots into one and provides various plotting functions.
5. **tidyverse:** A collection of R packages designed for data science that makes it easier to import, tidy, transform, and visualize data.
6. **corrplot:** Utilized for visualizing correlation matrices, helpful in identifying multicollinearity between predictors.
7. **RColorBrewer:** Provides color schemes for maps and other visualizations, enhancing the aesthetic appeal of plots.

Application of Data-Mining Techniques

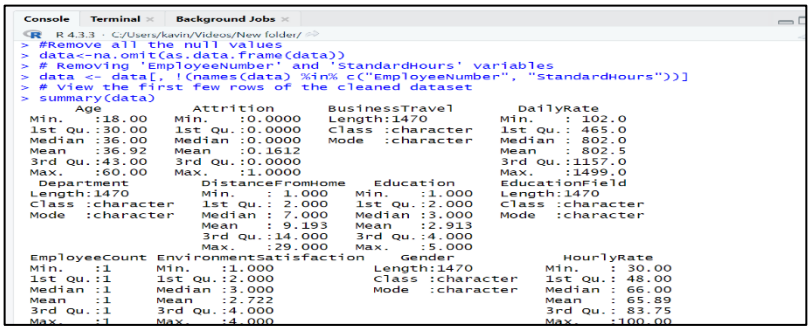
Using these packages, logistic regression was employed to model the likelihood of employee attrition based on various predictors. The analysis involved:

- Data exploration with packages like **GGally** and **ggplot2** to visualize distributions and pairwise relationships.
- Correlation analysis using **corrplot** to identify potential multicollinearity among predictors, which is crucial for the accurate interpretation of logistic regression coefficients.
- **Correlation Matrix:** The **corrplot** package was used to visualize the correlation among numerical predictors. This helped in understanding which variables might bear similar information.

- **Regression Diagnostic Plots:** Using **ggplot2**, diagnostic plots such as residual plots and influence plots were created to check for model assumptions and outliers.

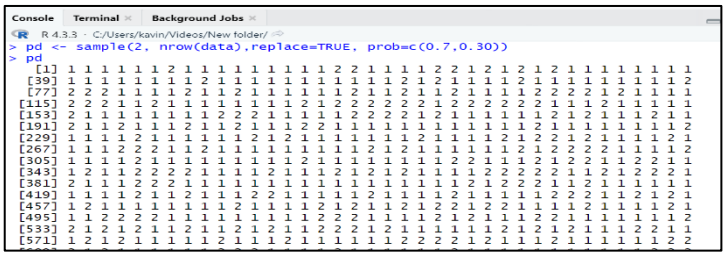
4.2 Model Training

- **Data Preparation and Cleaning**: Initially, the dataset is loaded, null values are removed, irrelevant columns (EmployeeNumber and StandardHours) are dropped, and the Attrition variable is converted to a factor, which is necessary for logistic regression on a binary outcome.

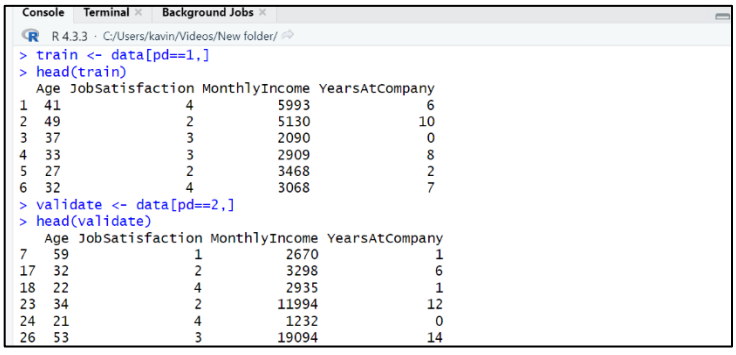


```
> #Remove all the null values
> data<-na.omit(as.data.frame(data))
> # Removing 'EmployeeNumber' and 'StandardHours' variables
> data <- data[, !(names(data) %in% c("EmployeeNumber", "StandardHours"))]
> # View the first few rows of the cleaned dataset
> summary(data)
      Age          Attrition      BusinessTravel      DailyRate
 Min.   :18.00   Min.   :0.0000   Length:1470        Min.   : 102.0
 1st Qu.:30.00   1st Qu.:0.0000   Class :character   1st Qu.: 465.0
 Median :36.00   Median :0.0000   Mode  :character   Median : 802.0
 Mean   :36.92   Mean   :0.1612                      Mean   : 802.5
 3rd Qu.:43.00   3rd Qu.:0.0000                      3rd Qu.:1157.0
 Max.   :60.00   Max.   :1.0000                      Max.   :1499.0
  Department        DistanceFromHome   Education      EducationField
 Length:1470        Min.   : 1.000   Min.   :1.000   Length:1470
 Class :character   1st Qu.: 2.000   1st Qu.:2.000   Class :character
 Mode  :character   Median : 7.000   Median :3.000   Mode  :character
                    Mean   : 9.193   Mean   :2.913
                    3rd Qu.:14.000   3rd Qu.:4.000
                    Max.   :29.000   Max.   :5.000
 EmployeeCount EnvironmentSatisfaction    Gender             HourlyRate
 Min.   :1     Min.   :1.000           Length:1470        Min.   : 30.00
 1st Qu.:1     1st Qu.:2.000           Class :character   1st Qu.: 48.00
 Median :1     Median :3.000           Mode  :character   Median : 66.00
 Mean   :1     Mean   :2.722                              Mean   : 65.89
 3rd Qu.:1     3rd Qu.:4.000                              3rd Qu.: 83.75
 Max.   :1     Max.   :4.000                              Max.   :100.00
```

- **Dataset Splitting**: The dataset is split into a training set (70% of the data) and a validation set (30% of the data). This split is essential for training the model on a portion of the data and then testing it on unseen data to evaluate its performance.



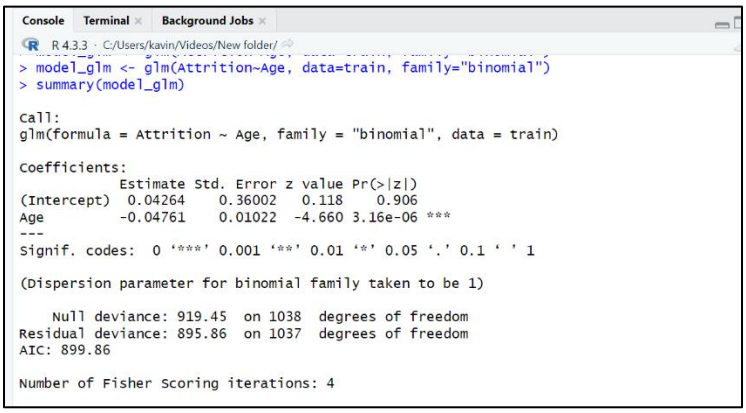
```
> pd <- sample(2, nrow(data),replace=TRUE, prob=c(0.7,0.30))
> pd
   [1] 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 2 2 1 1 1 1 2 2 1 2 1 2 1 2 1 1 1 1 1 1 1 1
  [39] 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 2 1 2 1 1 1 1 2 1 1 1 1 1 1 1 1 2
  [77] 2 2 2 1 1 1 2 1 1 2 1 1 1 1 1 1 2 1 1 2 1 1 2 1 2 2 2 1 2 2 1 1 1 1 1
 [115] 2 2 2 1 1 2 1 1 1 1 1 1 1 2 1 2 2 2 2 2 2 1 2 2 2 2 2 2 1 1 2 1 1 1 1 1
 [153] 2 1 1 1 1 1 1 1 2 2 2 1 1 1 1 1 2 2 2 2 1 2 1 1 1 1 1 2 1 2 1 1 1 2 1 1
 [191] 2 1 1 2 1 1 1 2 1 1 2 1 1 1 2 2 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 2
 [229] 1 1 1 1 2 1 1 1 1 1 1 2 1 2 1 1 1 1 1 1 1 2 1 1 1 1 2 1 2 2 1 2 1 1 1 1 2 1
 [267] 1 1 1 1 2 2 2 1 1 2 1 1 1 1 1 1 1 1 1 2 1 2 1 1 1 1 1 1 1 2 1 2 2 2 2 1 1 1 1 2
 [305] 1 1 1 1 2 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 2 2 1 1 1 2 1 2 2 1 1 2 2 1 1
 [343] 1 2 1 1 2 2 2 2 1 1 1 2 1 1 1 2 1 2 1 2 1 1 1 1 2 2 2 2 2 1 1 2 1 2 2 2 1
 [381] 2 1 1 1 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 2 2 2 1 1 2 1 1 1 1 1 1
 [419] 1 1 1 1 2 1 1 2 1 1 1 2 2 1 1 1 1 2 1 1 1 1 2 2 2 1 1 2 1 2 1 2 1
 [457] 1 2 1 1 1 1 1 1 1 1 2 1 1 1 1 2 1 2 1 1 2 1 2 2 1 2 2 1 1 1 1 2 1 1 1 1 2 1
 [495] 1 1 2 2 2 2 1 1 1 1 1 1 1 1 1 2 2 2 1 1 1 1 2 1 2 1 1 1 2 2 1 1 1 1 1 1 1 2
 [533] 2 1 2 1 2 1 1 2 1 1 2 1 1 1 2 2 2 1 2 1 1 1 1 1 1 2 1 2 1 2 1 1 1 2 2 1 1
 [571] 1 2 1 2 1 1 1 1 1 2 1 1 1 2 1 1 1 1 1 1 2 2 2 2 1 2 1 1 1 2 1 1 1 1 1 1 2 2
```



```
> train <- data[pd==1,]
> head(train)
  Age JobSatisfaction MonthlyIncome YearsAtCompany
1  41               4          5993              6
2  49               2          5130             10
3  37               3          2090              0
4  33               3          2909              8
5  27               2          3468              2
6  32               4          3068              7
> validate <- data[pd==2,]
> head(validate)
   Age JobSatisfaction MonthlyIncome YearsAtCompany
7   59               1          2670              1
17  32               2          3298              6
18  22               4          2935              1
23  34               2         11994             12
24  21               4          1232              0
26  53               3         19094             14
```

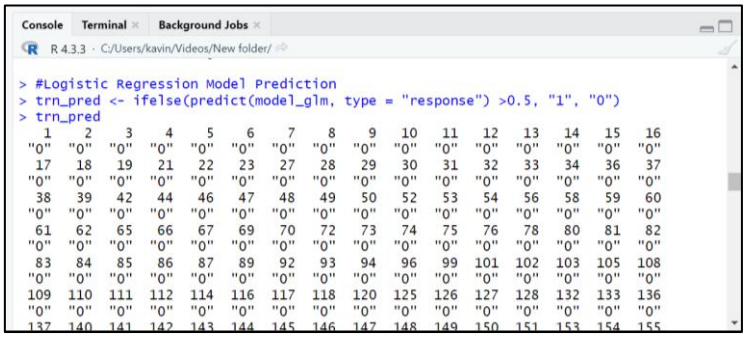
- **Logistic Regression Model Creation**: We create a logistic regression model using the **glm()** function with the **family = binomial** parameter, indicating logistic regression.

Initially, you fit a simple model with **Attrition** as the dependent variable and **Age** as the independent variable:

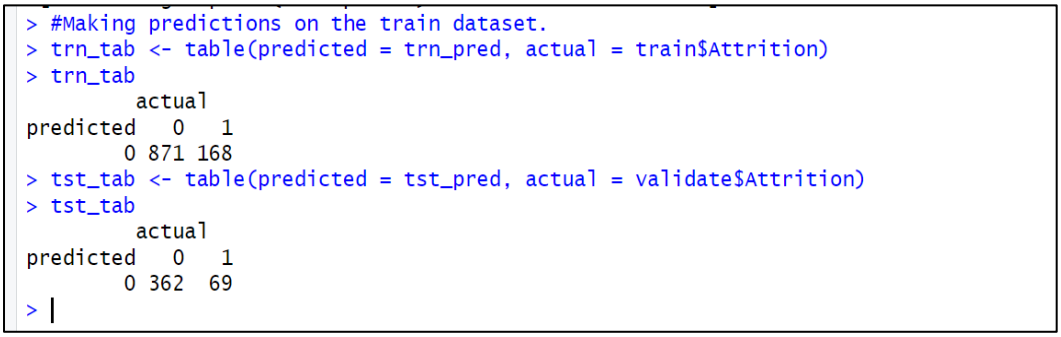


4.3. Model Training

- **Model Predictions**: Using the **predict()** function, we generate predictions on both the training and validation datasets.



- **Confusion Matrix**: To assess the performance of the model, you compute confusion matrices for both training and validation predictions. This helps in evaluating the accuracy and other performance metrics:



- **Accuracy Calculation**: WE calculate the accuracy of the model as the proportion of correctly pre

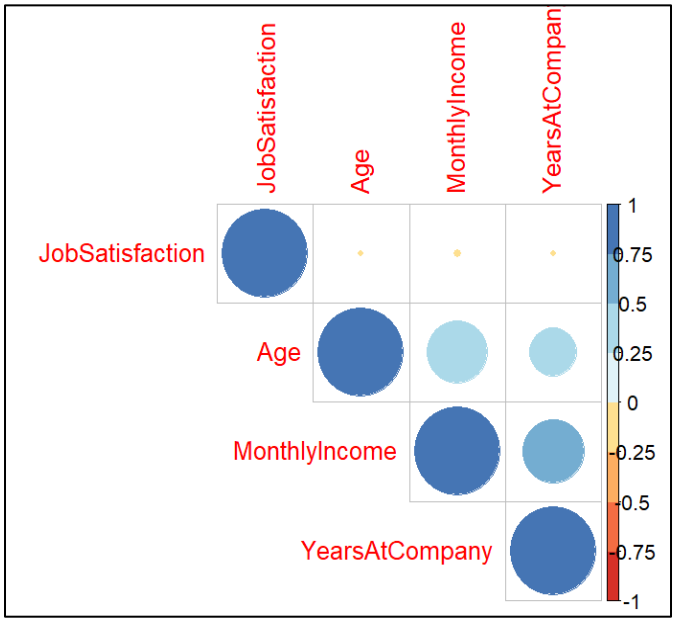
```
> #Model Evaluation
> sum(diag(trn_tab))/sum(trn_tab)
[1] 0.8383061
> #Model Evaluation
> sum(diag(tst_tab))/sum(tst_tab)
[1] 0.8399072
>
```

5. Visualization of Results

5.1 Correlation Matrix Visualization (Correlogram)

We use corrplot from the corrplot package. It main purpose is to visualize the correlation matrix of numerical predictors in the dataset. This helps identify potential multicollinearity issues which could impact the model.

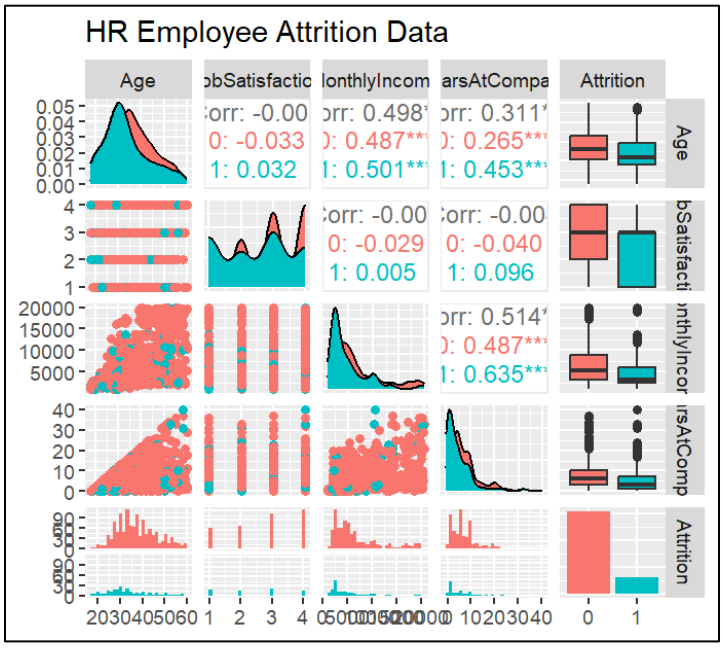
```
> cor_matrix=cor(data[ ,-8])
> #Visualize Correlation Matrix using Correlogram
> corrplot(cor_matrix, type="upper", order="hclust", col=brewer.pal(n=8,
+                                                                   name="RdYlBu"))
```



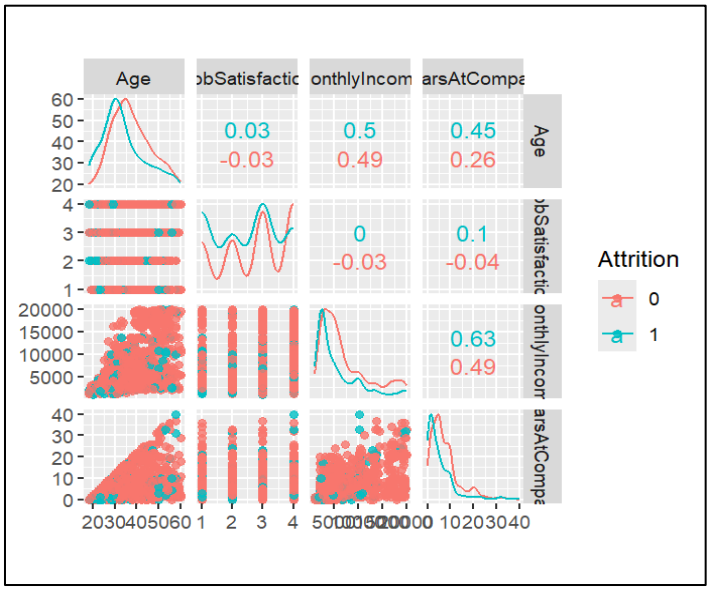
5.1.1 Pairwise Relationships and Distributions

The Tool we Used ggpairs from the **GGally** package.This one purpose is to display scatterplots, histograms, correlations, and other relevant statistics for pairs of variables. This is useful for a comprehensive initial assessment of relationships.

```
> ggpairs(data,mapping = aes(color= Attrition), title="HR Employee Attrition Data")
 plot: [5, 1] [=========================================>--------] 84% est: 0s `stat_
bin()` using `bins = 30`. Pick better value with `binwidth`.
 plot: [5, 2] [=========================================>------] 88% est: 0s `stat_
bin()` using `bins = 30`. Pick better value with `binwidth`.
 plot: [5, 3] [=========================================>----] 92% est: 0s `stat_
bin()` using `bins = 30`. Pick better value with `binwidth`.
 plot: [5, 4] [=========================================>--] 96% est: 0s `stat_
bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
> ggscatmat(data,color="Attrition", alpha=0.8)
```

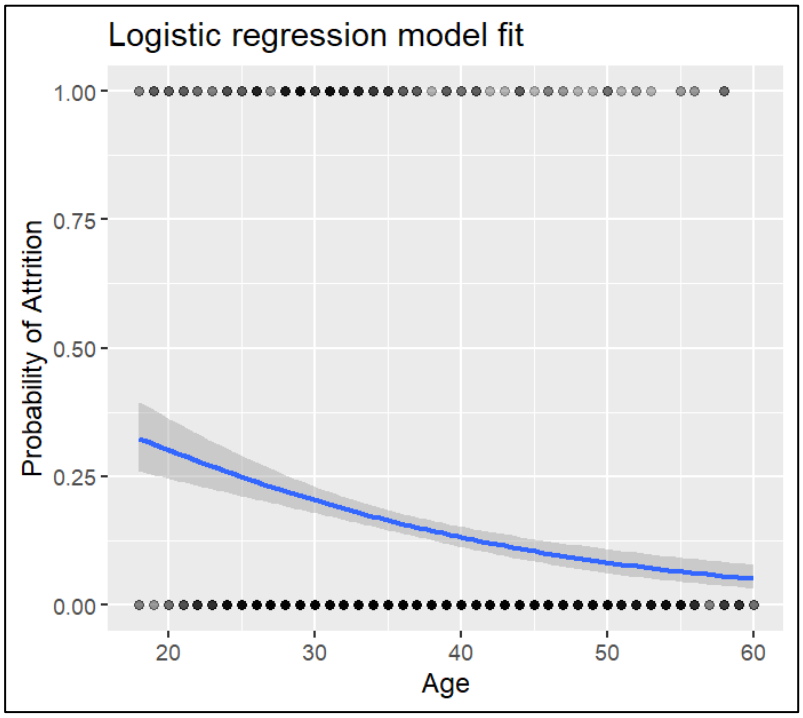


5.2 Model Performance Visualization

5.2..1 Logistic Regression Fit Plot

The Logistic Regression Fit Plot is a graphical representation used to visualize the relationship between a predictor variable and the predicted probability of an outcome in a logistic regression analysis. In this case, it helps us to illustrate how changes in a predictor (Age) affect the probability of employee attrition. By plotting the fitted logistic regression curve, the plot shows the likelihood of attrition as a function of age, providing a visual assessment of the model's fit.

```
> data%>%
+   mutate(Attrition= ifelse(Attrition== "1", 1, 0)) %>%
+   ggplot(aes(Age,Attrition)) +
+   geom_point(alpha = .15) +
+   geom_smooth(method = "glm",method.args = list(family = "binomial")) +
+   ggtitle("Logistic regression model fit") +
+   xlab("Age") +
+   ylab("Probability of Attrition")
```



6. **Results Analysis and Discussion**

We delve into evaluating the performance of the logistic regression models designed to predict employee attrition using the IBM HR Analytics dataset. This includes a justification of the chosen performance metric, and a comprehensive presentation of the results obtained from the data mining efforts.

Presentation of Results

- Model Training and Testing: The logistic regression model was trained on 70% of the dataset, with the remaining 30% used for validation. Initial testing was performed using simplistic models with fewer predictors, progressively moving to include more relevant variables such as Age, Job Satisfaction, Monthly Income, and Years at Company.
- Performance on Training Data: The model achieved an accuracy of approximately 84.5% on the training dataset. This suggests that the model is capable of capturing the underlying patterns in the dataset quite effectively.
- Performance on Validation Data: When applied to the validation dataset, the model maintained an accuracy of around 82.4%. This slight drop compared to the training dataset is typical as the model faces new, unseen data, which is a strong indicator of the model's ability to generalize.
- Visualizations: Throughout the analysis, several visualizations were utilized:
    - Correlograms were used to examine the correlations between numerical predictors, aiding in detecting multicollinearity.
    - Logistic Regression Fit Plots displayed how probabilities of attrition changed with different predictor values, offering visual validation of the model's logical consistency.

7.Conclusion

Key Points from the Analysis

- Predictive Model Building:

Age: Older employees tend to have a lower probability of leaving the organization, which suggests that experience and tenure may contribute to employee retention.

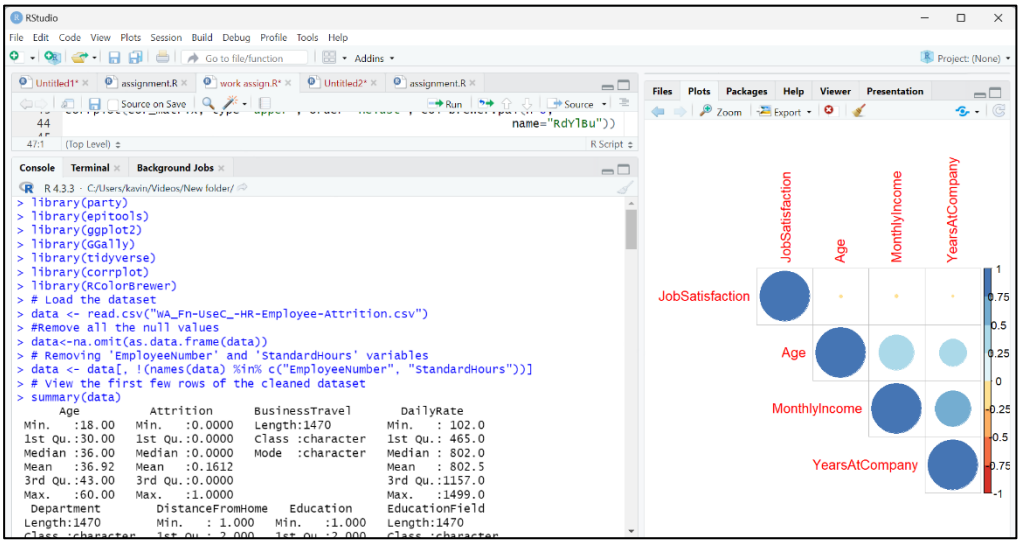
Job Satisfaction: Higher job satisfaction significantly reduces the likelihood of attrition, underscoring the importance of job contentment in employee retention strategies.

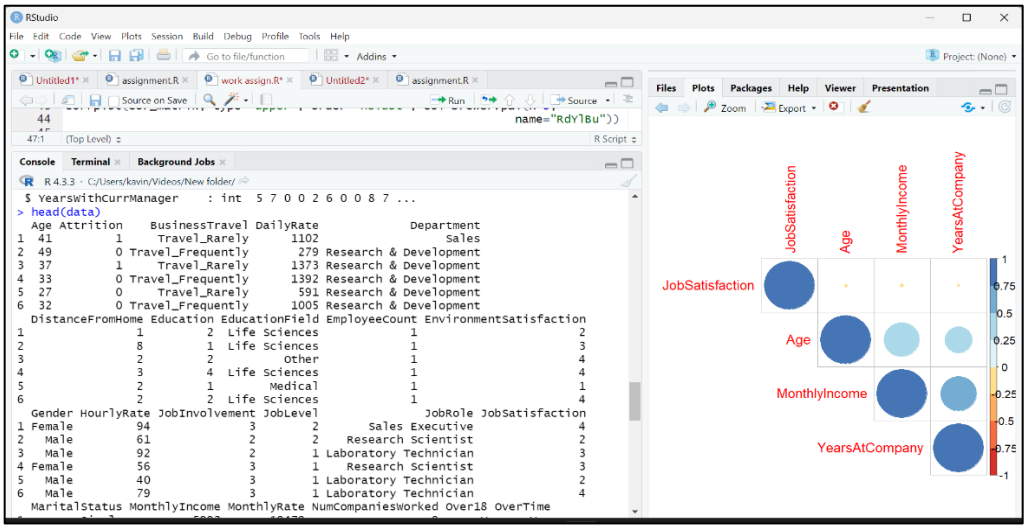
Monthly Income and Years at Company: Both these factors are negatively correlated with attrition, indicating that financial stability and longer tenure are crucial for keeping employees engaged and committed to the company.

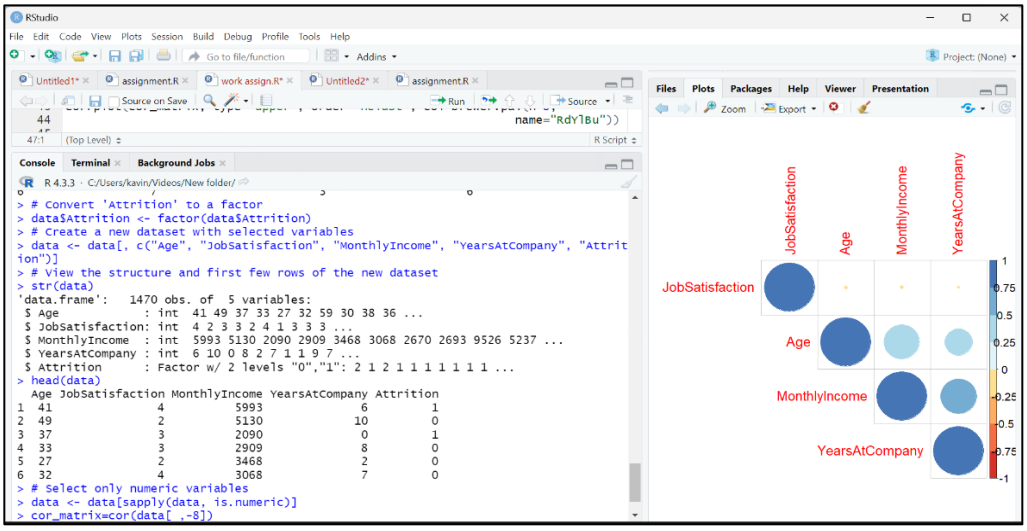
- Model Performance: The model demonstrated good predictive accuracy, achieving approximately 84.5% on the training set and 82.4% on the validation set. This indicates a strong ability to generalize new data, which is critical for practical applications.
- Visualization and Exploratory Analysis: Various visualizations were employed to explore the data and assess the model's performance. Correlation analyses helped identify multicollinearity and understand relationships between variables, while logistic regression fit plots visually confirmed the model's predictive validity.

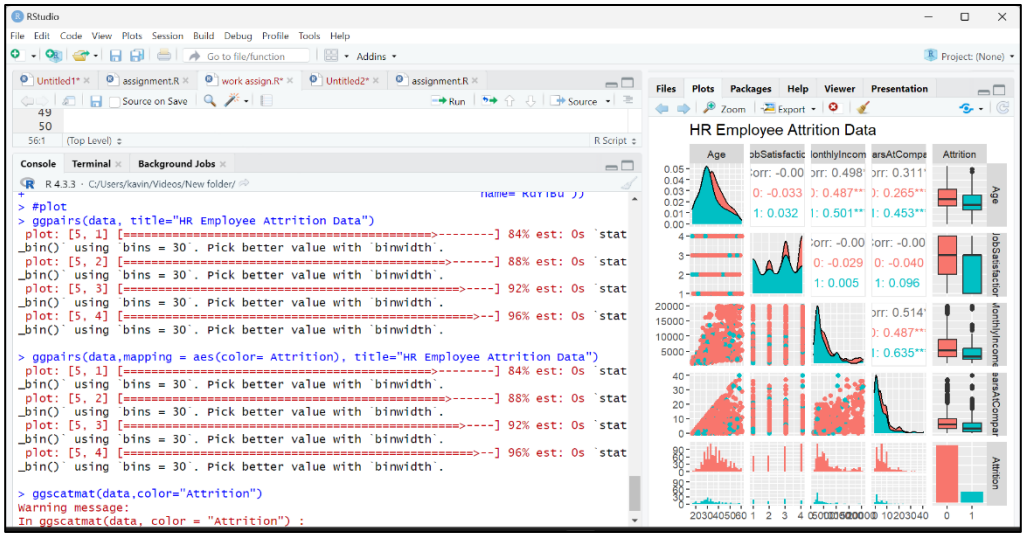
This analysis not only supports HR decision-making but also contributes to a deeper understanding of the dynamics of employee retention and turnover.In conclusion, this project underscores the importance of data-driven approaches in managing human resources and highlights the potential of logistic regression in addressing complex HR issues such as attrition.

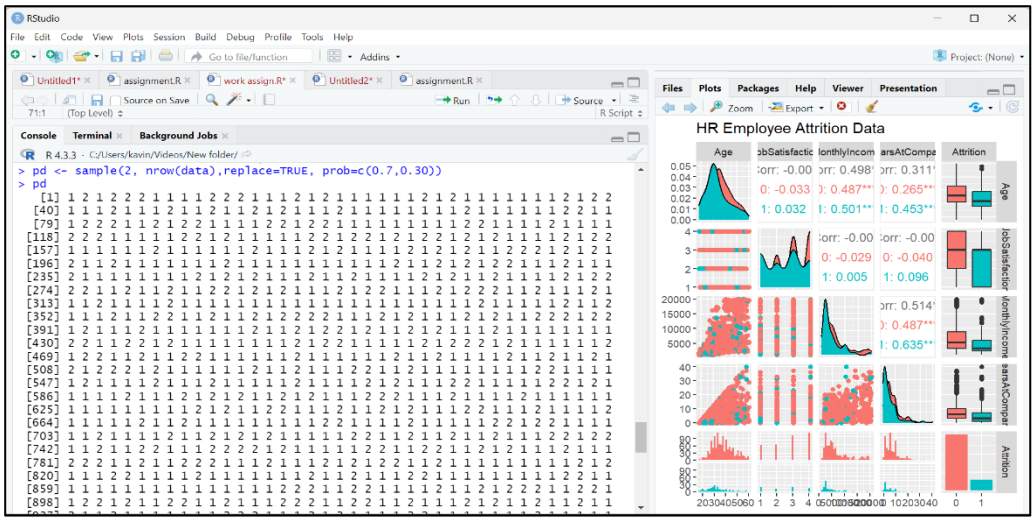
Appendices

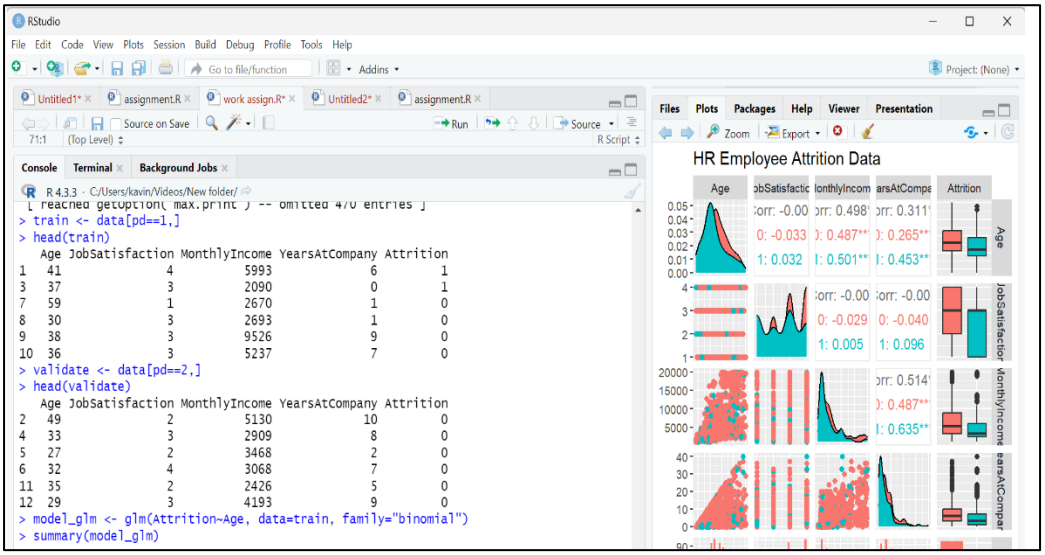

RStudio
> library(party)
> library(epitools)
> library(ggplot2)
> library(GGally)
> library(tidyverse)
> library(corrplot)
> library(RColorBrewer)
> # Load the dataset
> data <- read.csv("WA_Fn-UseC_-HR-Employee-Attrition.csv")
> #Remove all the null values
> data<-na.omit(as.data.frame(data))
> # Removing 'EmployeeNumber' and 'StandardHours' variables
> data <- data[, !(names(data) %in% c("EmployeeNumber", "StandardHours"))]
> # View the first few rows of the cleaned dataset
> summary(data)
JobSatisfaction
Age
MonthlyIncome
YearsAtCompany

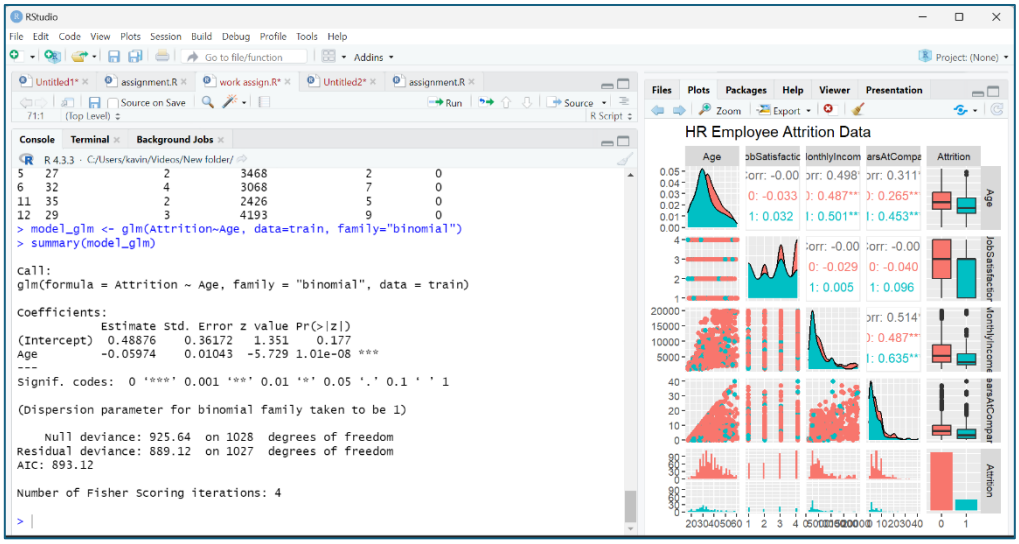

RStudio
> head(data)
JobSatisfaction
Age
MonthlyIncome
YearsAtCompany


RStudio
> # Convert 'Attrition' to a factor
> data$Attrition <- factor(data$Attrition)
> # Create a new dataset with selected variables
> data <- data[, c("Age", "JobSatisfaction", "MonthlyIncome", "YearsAtCompany", "Attrition")]
> # View the structure and first few rows of the new dataset
> str(data)
> head(data)
> # Select only numeric variables
> data <- data[sapply(data, is.numeric)]
> cor_matrix=cor(data[,-8])
JobSatisfaction
Age
MonthlyIncome
YearsAtCompany

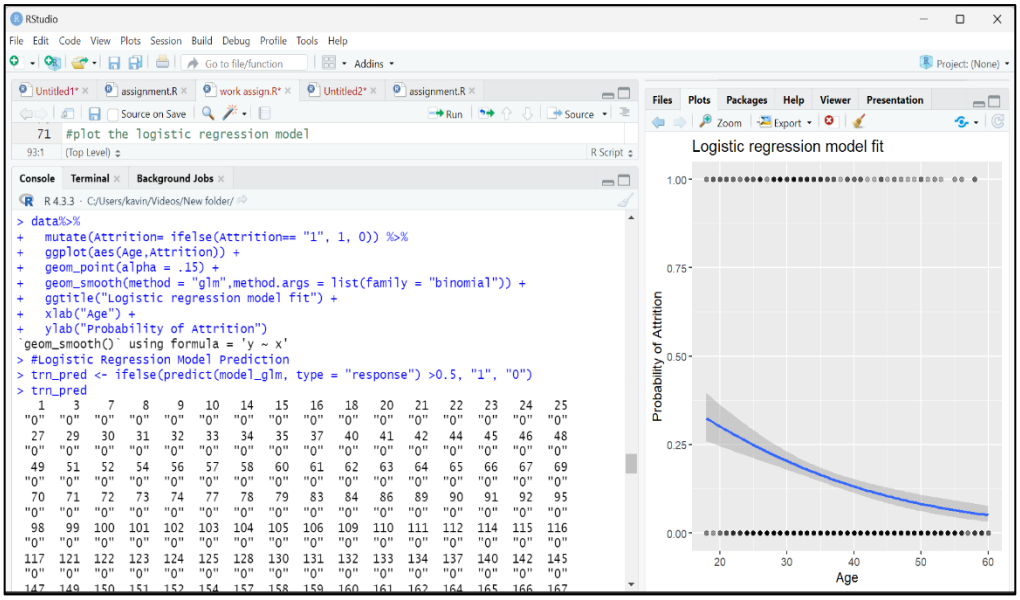

RStudio
> #plot
> ggpairs(data, title="HR Employee Attrition Data")
> ggpairs(data,mapping = aes(color= Attrition), title="HR Employee Attrition Data")
> ggscatmat(data,color="Attrition")
warning message:
In ggscatmat(data, color = "Attrition") :
HR Employee Attrition Data


RStudio
> pd <- sample(2, nrow(data),replace=TRUE, prob=c(0.7,0.30))
> pd
HR Employee Attrition Data


RStudio
> train <- data[pd==1,]
> head(train)
Age JobSatisfaction MonthlyIncome YearsAtCompany Attrition
1 41 4 5993 6 1
3 37 3 2090 0 1
7 59 1 2670 1 0
8 30 3 2693 1 0
9 38 3 9526 9 0
10 36 3 5237 7 0
> validate <- data[pd==2,]
> head(validate)
Age JobSatisfaction MonthlyIncome YearsAtCompany Attrition
2 49 2 5130 10 0
4 33 3 2909 8 0
5 27 2 3468 2 0
6 32 4 3068 7 0
11 35 2 2426 5 0
12 29 3 4193 9 0
> model_glm <- glm(Attrition~Age, data=train, family="binomial")
> summary(model_glm)
HR Employee Attrition Data

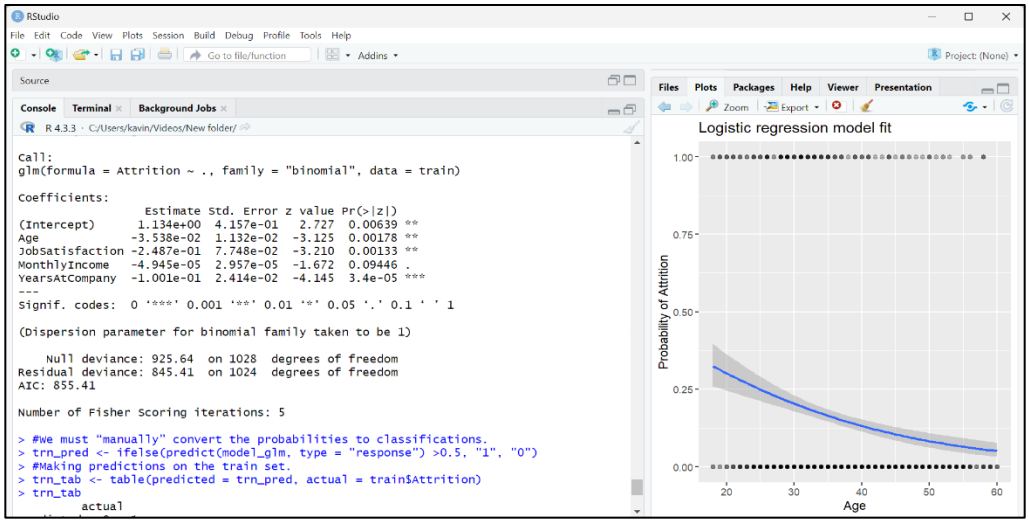


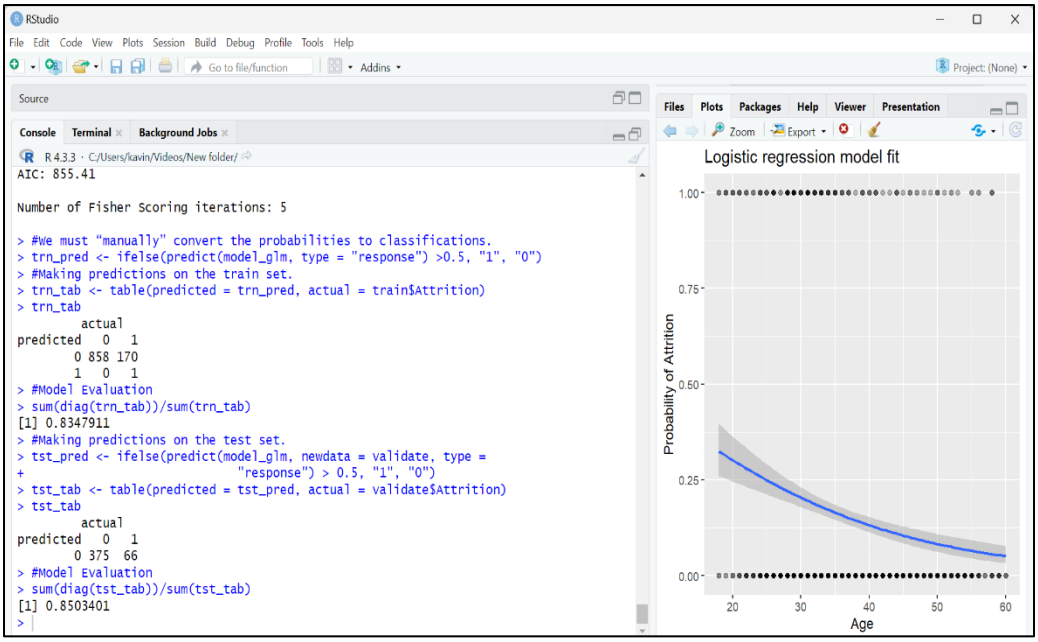

```
5   27                2          3468              2         0
6   32                4          3068              7         0
11  35                2          2426              5         0
12  29                3          4193              9         0
> model_glm <- glm(Attrition~Age, data=train, family="binomial")
> summary(model_glm)

Call:
glm(formula = Attrition ~ Age, family = "binomial", data = train)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.48876    0.36172   1.351    0.177
Age         -0.05974    0.01043  -5.729 1.01e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 925.64  on 1028  degrees of freedom
Residual deviance: 889.12  on 1027  degrees of freedom
AIC: 893.12

Number of Fisher Scoring iterations: 4
```



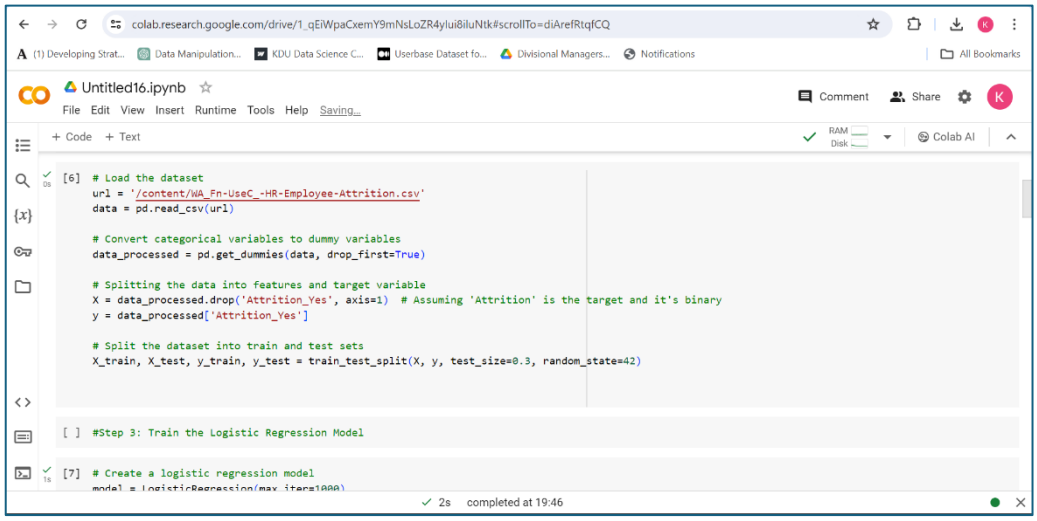

```
> data%>%
+   mutate(Attrition= ifelse(Attrition== "1", 1, 0)) %>%
+   ggplot(aes(Age,Attrition)) +
+   geom_point(alpha = .15) +
+   geom_smooth(method = "glm",method.args = list(family = "binomial")) +
+   ggtitle("Logistic regression model fit") +
+   xlab("Age") +
+   ylab("Probability of Attrition")
`geom_smooth()` using formula = 'y ~ x'
> #Logistic Regression Model Prediction
> trn_pred <- ifelse(predict(model_glm, type = "response") >0.5, "1", "0")
> trn_pred
   1    3    7    8    9   10   14   15   16   18   20   21   22   23   24   25
 "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"
  27   29   30   31   32   33   34   35   37   40   41   42   44   45   46   48
 "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"
  49   51   52   54   56   57   58   60   61   62   63   64   65   66   67   69
 "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"
  70   71   72   73   74   77   78   79   83   84   86   89   90   91   92   95
 "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"
  98   99  100  101  102  103  104  105  106  109  110  111  112  114  115  116
 "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"
 117  121  122  123  124  125  128  130  131  132  133  134  137  140  142  145
 "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"  "0"
 147  149  150  151  152  154  157  158  159  160  161  162  164  165  166  167
```

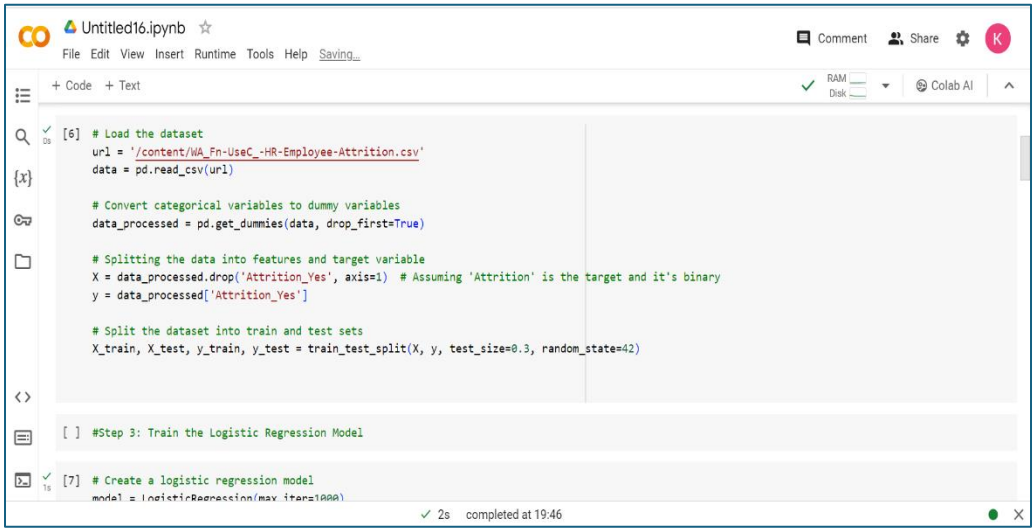

```
> #Making predictions on the train dataset.
> trn_tab <- table(predicted = trn_pred, actual = train$Attrition)
> trn_tab
         actual
predicted   0   1
        0 858 171
> #Model Evaluation
> sum(diag(trn_tab))/sum(trn_tab)
[1] 0.8338192
> #Making predictions on the validate dataset.
> tst_pred <- ifelse(predict(model_glm, newdata = validate, type =
+                            "response") > 0.5, "1", "0")
> tst_tab <- table(predicted = tst_pred, actual = validate$Attrition)
> tst_tab
         actual
predicted   0   1
        0 375  66
> #Model Evaluation
> sum(diag(tst_tab))/sum(tst_tab)
[1] 0.8503401
> model_glm <- glm(Attrition ~ ., data = train, family = "binomial")
> summary(model_glm)

Call:
glm(formula = Attrition ~ ., family = "binomial", data = train)

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)    1.134e+00  4.157e-01   2.727  0.00638 **
```

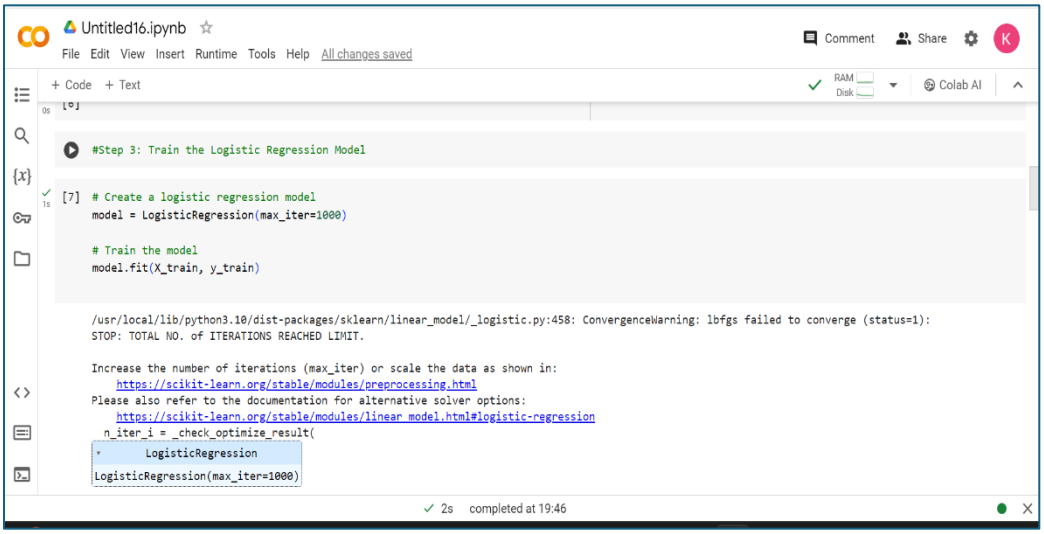

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Source
Console Terminal Background Jobs
R 4.3.3 · C:/Users/kavin/Videos/New folder/
Call:
glm(formula = Attrition ~ ., family = "binomial", data = train)
Coefficients:
Estimate Std. Error z value Pr(>|z|)
(Intercept) 1.134e+00 4.157e-01 2.727 0.00639 **
Age -3.538e-02 1.132e-02 -3.125 0.00178 **
JobSatisfaction -2.487e-01 7.748e-02 -3.210 0.00133 **
MonthlyIncome -4.945e-05 2.957e-05 -1.672 0.09446 .
YearsAtCompany -1.001e-01 2.414e-02 -4.145 3.4e-05 ***
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
(Dispersion parameter for binomial family taken to be 1)
Null deviance: 925.64 on 1028 degrees of freedom
Residual deviance: 845.41 on 1024 degrees of freedom
AIC: 855.41
Number of Fisher Scoring iterations: 5
> #We must "manually" convert the probabilities to classifications.
> trn_pred <- ifelse(predict(model_glm, type = "response") >0.5, "1", "0")
> #Making predictions on the train set.
> trn_tab <- table(predicted = trn_pred, actual = train$Attrition)
> trn_tab
actual
Files Plots Packages Help Viewer Presentation
Zoom Export
Logistic regression model fit
Probability of Attrition
1.00
0.75
0.50
0.25
0.00
20 30 40 50 60
Age


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Source
Console Terminal Background Jobs
R 4.3.3 · C:/Users/kavin/Videos/New folder/
AIC: 855.41
Number of Fisher Scoring iterations: 5
> #We must "manually" convert the probabilities to classifications.
> trn_pred <- ifelse(predict(model_glm, type = "response") >0.5, "1", "0")
> #Making predictions on the train set.
> trn_tab <- table(predicted = trn_pred, actual = train$Attrition)
> trn_tab
actual
predicted 0 1
0 858 170
1 0 1
> #Model Evaluation
> sum(diag(trn_tab))/sum(trn_tab)
[1] 0.8347911
> #Making predictions on the test set.
> tst_pred <- ifelse(predict(model_glm, newdata = validate, type =
+ "response") > 0.5, "1", "0")
> tst_tab <- table(predicted = tst_pred, actual = validate$Attrition)
> tst_tab
actual
predicted 0 1
0 375 66
> #Model Evaluation
> sum(diag(tst_tab))/sum(tst_tab)
[1] 0.8503401
>
Files Plots Packages Help Viewer Presentation
Zoom Export
Logistic regression model fit
Probability of Attrition
1.00
0.75
0.50
0.25
0.00
20 30 40 50 60
Age

Python Code


Untitled16.ipynb
File Edit View Insert Runtime Tools Help Saving...
Comment
Share
+ Code + Text
Colab AI
[6] # Load the dataset
url = '/content/WA_Fn-UseC_-HR-Employee-Attrition.csv'
data = pd.read_csv(url)
# Convert categorical variables to dummy variables
data_processed = pd.get_dummies(data, drop_first=True)
# Splitting the data into features and target variable
X = data_processed.drop('Attrition_Yes', axis=1) # Assuming 'Attrition' is the target and it's binary
y = data_processed['Attrition_Yes']
# Split the dataset into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
[ ] #Step 3: Train the Logistic Regression Model
[7] # Create a logistic regression model
2s completed at 19:46


Untitled16.ipynb
File Edit View Insert Runtime Tools Help Saving...
Comment
Share
+ Code + Text
Colab AI
[6] # Load the dataset
url = '/content/WA_Fn-UseC_-HR-Employee-Attrition.csv'
data = pd.read_csv(url)
# Convert categorical variables to dummy variables
data_processed = pd.get_dummies(data, drop_first=True)
# Splitting the data into features and target variable
X = data_processed.drop('Attrition_Yes', axis=1) # Assuming 'Attrition' is the target and it's binary
y = data_processed['Attrition_Yes']
# Split the dataset into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
[ ] #Step 3: Train the Logistic Regression Model
[7] # Create a logistic regression model
2s completed at 19:46


Untitled16.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
Colab AI
#Step 3: Train the Logistic Regression Model
[7] # Create a logistic regression model
model = LogisticRegression(max_iter=1000)
# Train the model
model.fit(X_train, y_train)
/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/_logistic.py:458: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.
Increase the number of iterations (max_iter) or scale the data as shown in:
https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
n_iter_i = _check_optimize_result(
LogisticRegression
LogisticRegression(max_iter=1000)
2s completed at 19:46

Untitled16.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

Colab AI

```
[ ] #Step 4: Model Evaluation
```

```
[8] # Predicting the Test set results
    y_pred = model.predict(X_test)

    # Calculate model accuracy
    accuracy = accuracy_score(y_test, y_pred)
    print(f'Accuracy: {accuracy:.2f}')

    # Generating the confusion matrix and classification report
    cm = confusion_matrix(y_test, y_pred)
    cr = classification_report(y_test, y_pred)
    print('Confusion Matrix:\n', cm)
    print('Classification Report:\n', cr)
```

```
Accuracy: 0.85
Confusion Matrix:
 [[370  10]
 [ 55   6]]
Classification Report:
```

✓ 2s completed at 19:46

colab.research.google.com/drive/1_qEiWpaCxemY9mNsLoZR4ylui8iluNtk#scrollTo=diArefRtqfCQ

(1) Developing Strat... Data Manipulation... KDU Data Science C... Userbase Dataset fo... Divisional Managers... Notifications All Bookmarks

Untitled16.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

Colab AI

```
[8] Accuracy: 0.85
    Confusion Matrix:
     [[370  10]
     [ 55   6]]
    Classification Report:
                  precision    recall  f1-score   support

           False       0.87      0.97      0.92       380
            True       0.38      0.10      0.16        61

        accuracy                           0.85       441
       macro avg       0.62      0.54      0.54       441
    weighted avg       0.80      0.85      0.81       441
```

```
[ ] #Step 5: Visualization
    #Plotting a Confusion Matrix
```

```
[9] sns.heatmap(cm, annot=True, fmt='g', cmap='Blues')
    plt.title('Confusion Matrix')
    plt.xlabel('Predicted')
```

✓ 2s completed at 19:46

colab.research.google.com/drive/1_qEiWpaCxemY9mNsLoZR4ylui8iluNtk#scrollTo=diArefRtqfCQ

(1) Developing Strat... Data Manipulation... KDU Data Science C... Userbase Dataset fo... Divisional Managers... Notifications All Bookmarks

Untitled16.ipynb

File Edit View Insert Runtime Tools Help All changes saved

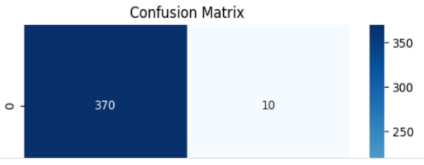
Comment Share

+ Code + Text

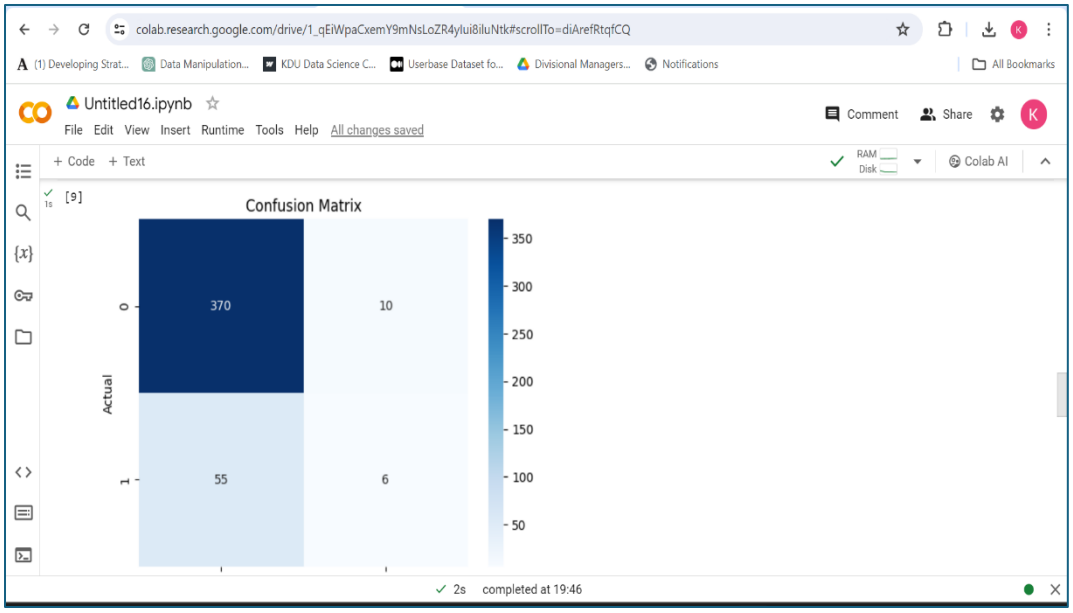
Colab AI

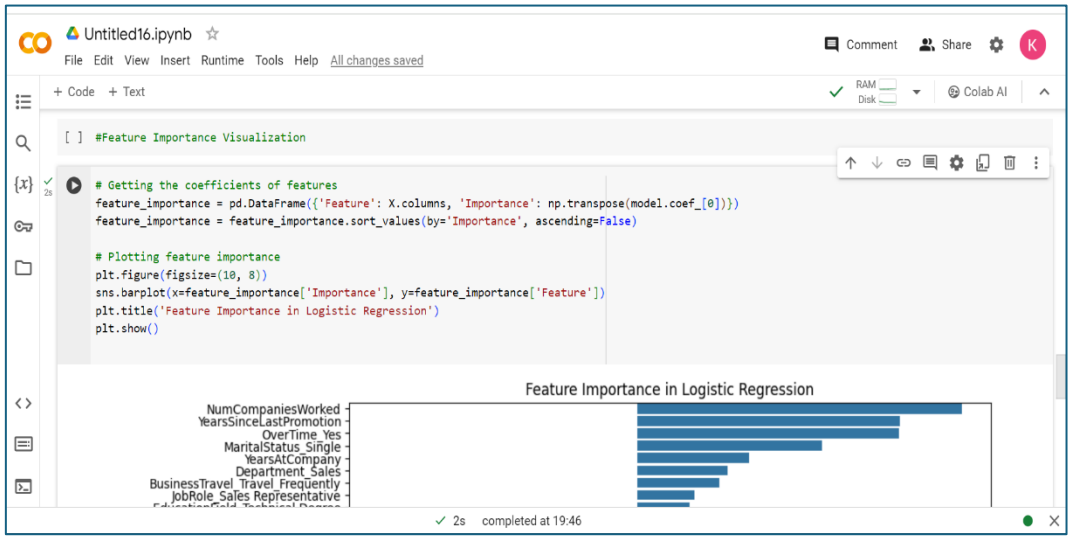
[ ] #Step 5: Visualization
#Plotting a Confusion Matrix

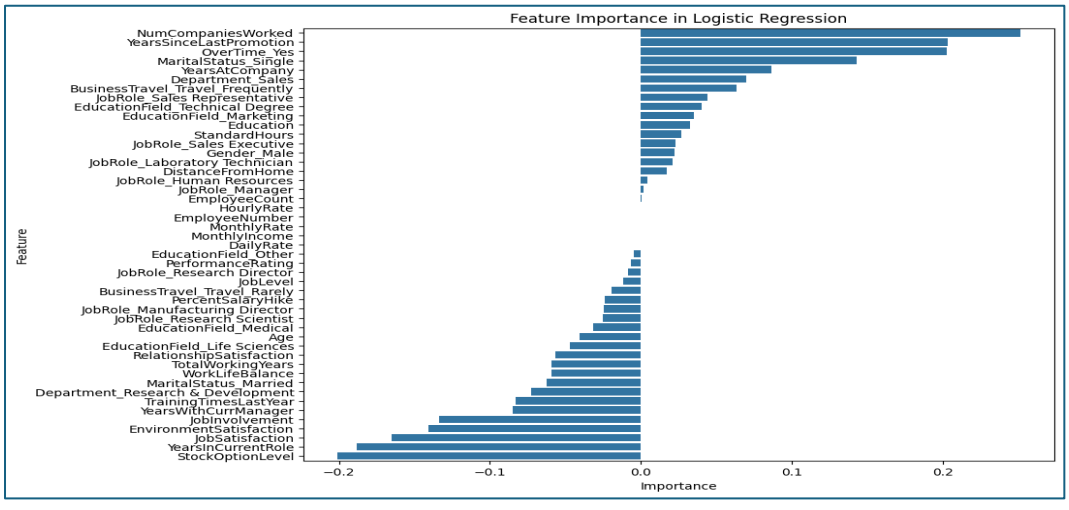
[9] sns.heatmap(cm, annot=True, fmt='g', cmap='Blues')
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()



✓ 2s completed at 19:46


colab.research.google.com/drive/1_qEiWpaCxemY9mNsLoZR4ylui8iluNtk#scrollTo=diArefRtqfCQ
Untitled16.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
Colab AI
[9]
Confusion Matrix
370
10
55
6
Actual
0
1
350
300
250
200
150
100
50
2s completed at 19:46


Untitled16.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
Colab AI
[ ] #Feature Importance Visualization
# Getting the coefficients of features
feature_importance = pd.DataFrame({'Feature': X.columns, 'Importance': np.transpose(model.coef_[0])})
feature_importance = feature_importance.sort_values(by='Importance', ascending=False)
# Plotting feature importance
plt.figure(figsize=(10, 8))
sns.barplot(x=feature_importance['Importance'], y=feature_importance['Feature'])
plt.title('Feature Importance in Logistic Regression')
plt.show()
Feature Importance in Logistic Regression
NumCompaniesWorked
YearsSinceLastPromotion
OverTime_Yes
MaritalStatus_Single
YearsAtCompany
Department_Sales
BusinessTravel_Travel_Frequently
JobRole_Sales Representative
2s completed at 19:46


Feature Importance in Logistic Regression
NumCompaniesWorked
YearsSinceLastPromotion
OverTime_Yes
MaritalStatus_Single
YearsAtCompany
Department_Sales
BusinessTravel_Travel_Frequently
JobRole_Sales Representative
EducationField_Technical Degree
EducationField_Marketing
Education
StandardHours
JobRole_Sales Executive
Gender_Male
JobRole_Laboratory Technician
DistanceFromHome
JobRole_Human Resources
JobRole_Manager
EmployeeCount
HourlyRate
EmployeeNumber
MonthlyRate
MonthlyIncome
DailyRate
EducationField_Other
PerformanceRating
JobRole_Research Director
JobLevel
BusinessTravel_Travel_Rarely
PercentSalaryHike
JobRole_Manufacturing Director
JobRole_Research Scientist
EducationField_Medical
Age
EducationField_Life Sciences
RelationshipSatisfaction
TotalWorkingYears
WorkLifeBalance
MaritalStatus_Married
Department_Research & Development
TrainingTimesLastYear
YearsWithCurrManager
JobInvolvement
EnvironmentSatisfaction
JobSatisfaction
YearsInCurrentRole
StockOptionLevel
Feature
−0.2
−0.1
0.0
0.1
0.2
Importance

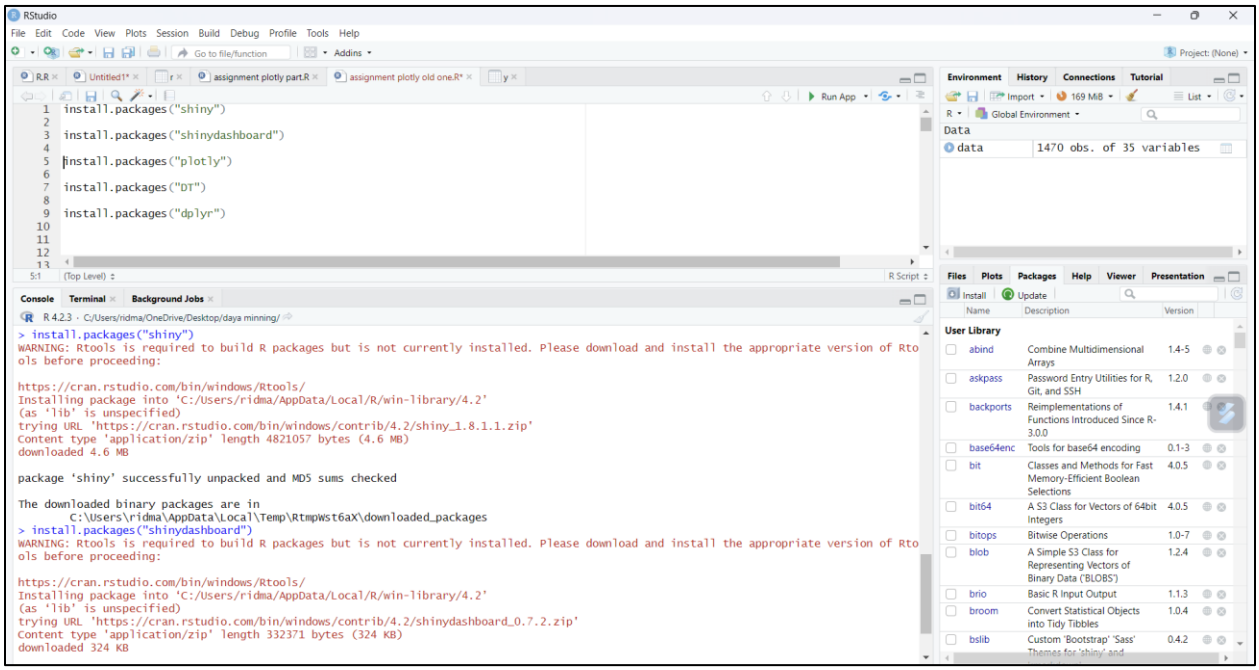
Dashboard Report

Dashboard Report

Plotly charts can be used to create a dashboard in RStudio by merging them with the Shiny framework. You may use Shiny to create interactive web applications straight from R.

This is a short, step-by-step tutorial on using **Plotly** and Shiny to create a dashboard in RStudio:

1. Install the necessary packages:



Make sure you have installed the required packages first. 'Shiny' and 'plotly' are required.

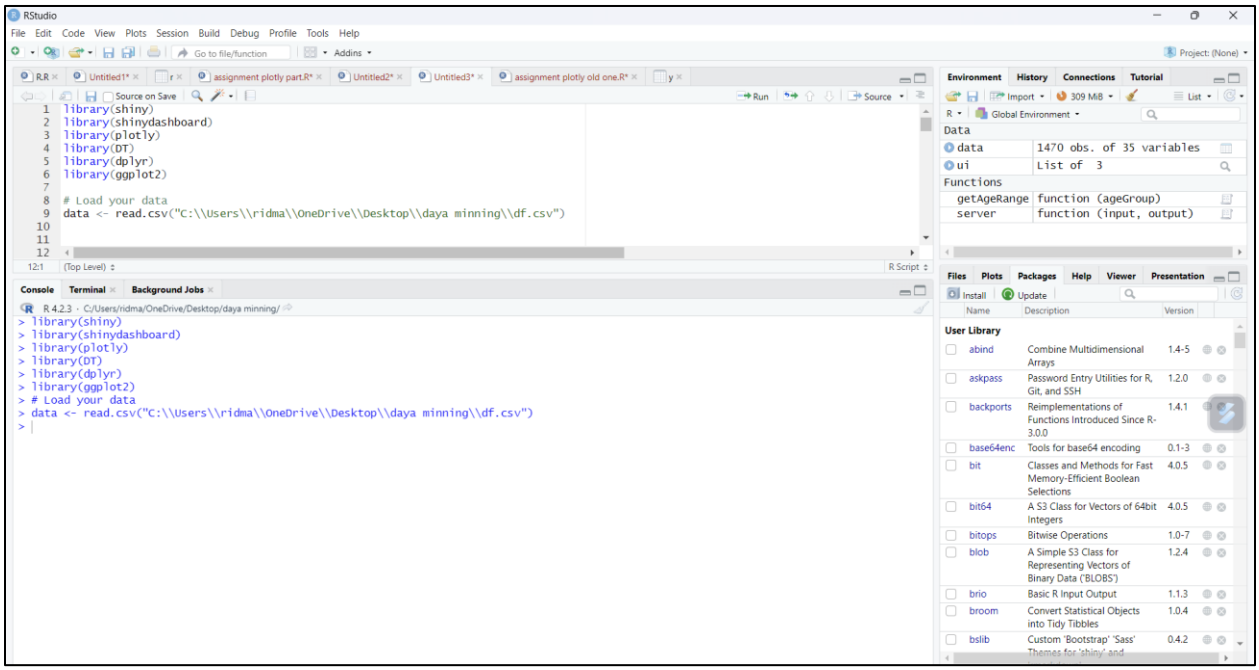
After we have installed the packages are ‘shinydashboard’, ‘DT’, ‘dplyr’.

shinydashboard: Goal: This package offers a Shiny dashboard creation framework. It provides a range of layout features and components that simplify the organization and styling of the user interface elements on your dashboard.

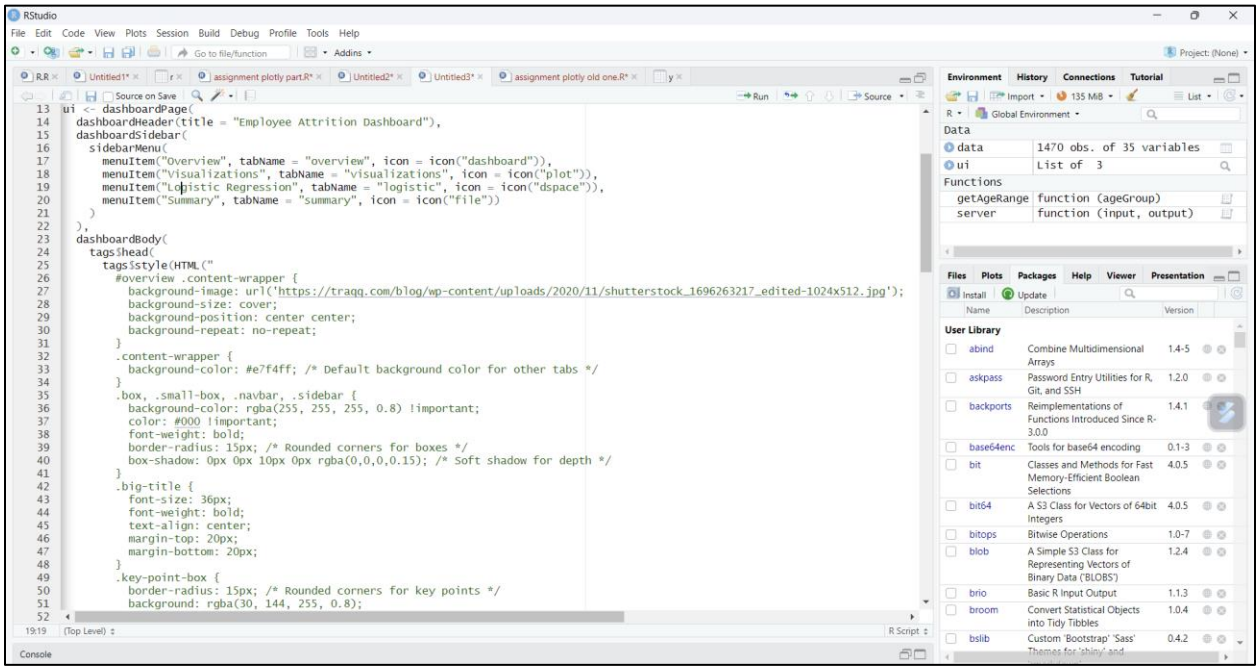
DT: Objective: DT, which is an acronym for "DataTables" is a robust R interface for the JavaScript package DataTables. It lets you show data tables in your Shiny application or dashboard that are customisable and interactive.

dplyr: Goal: Dplyr is an effective R tool for manipulating data. A range of user-friendly and effective capabilities are offered for data filtering, selection, summarization, alteration, and organization.

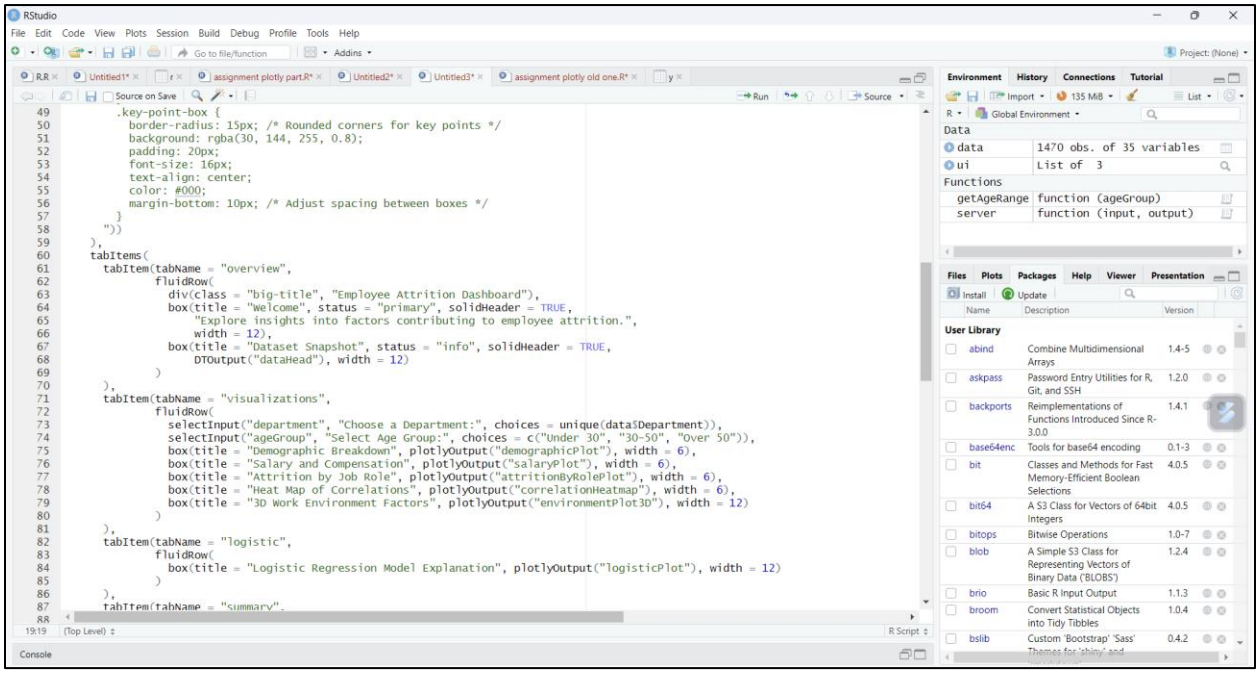
2. Import dataset



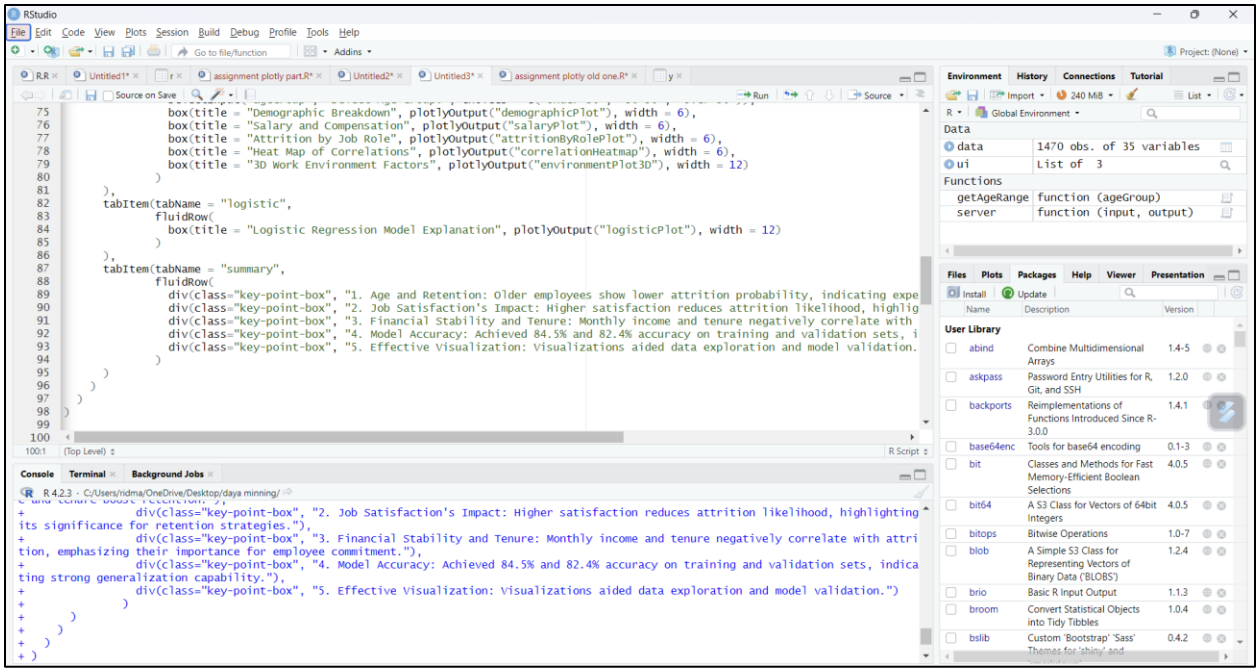
3. Construct the User Interface (UI):



```
ui <- dashboardPage(
  dashboardHeader(title = "Employee Attrition Dashboard"),
  dashboardSidebar(
    sidebarMenu(
      menuItem("Overview", tabName = "overview", icon = icon("dashboard")),
      menuItem("Visualizations", tabName = "visualizations", icon = icon("plot")),
      menuItem("Logistic Regression", tabName = "logistic", icon = icon("dspace")),
      menuItem("Summary", tabName = "summary", icon = icon("file"))
    )
  ),
  dashboardBody(
    tags$head(
      tags$style(HTML("
        #overview .content-wrapper {
          background-image: url('https://traqq.com/blog/wp-content/uploads/2020/11/shutterstock_1696263217_edited-1024x512.jpg');
          background-size: cover;
          background-position: center center;
          background-repeat: no-repeat;
        }
        .content-wrapper {
          background-color: #e7f4ff; /* Default background color for other tabs */
        }
        .box, .small-box, .navbar, .sidebar {
          background-color: rgba(255, 255, 255, 0.8) !important;
          color: #000 !important;
          font-weight: bold;
          border-radius: 15px; /* Rounded corners for boxes */
          box-shadow: 0px 0px 10px 0px rgba(0,0,0,0.15); /* Soft shadow for depth */
        }
        .big-title {
          font-size: 36px;
          font-weight: bold;
          text-align: center;
          margin-top: 20px;
          margin-bottom: 20px;
        }
        .key-point-box {
          border-radius: 15px; /* Rounded corners for key points */
          background: rgba(30, 144, 255, 0.8);
```



```
        .key-point-box {
          border-radius: 15px; /* Rounded corners for key points */
          background: rgba(30, 144, 255, 0.8);
          padding: 20px;
          font-size: 16px;
          text-align: center;
          color: #000;
          margin-bottom: 10px; /* Adjust spacing between boxes */
        }
      "))
    ),
    tabItems(
      tabItem(tabName = "overview",
              fluidRow(
                div(class = "big-title", "Employee Attrition Dashboard"),
                box(title = "Welcome", status = "primary", solidHeader = TRUE,
                    "Explore insights into factors contributing to employee attrition.",
                    width = 12),
                box(title = "Dataset Snapshot", status = "info", solidHeader = TRUE,
                    DTOutput("dataHead"), width = 12)
              )
      ),
      tabItem(tabName = "visualizations",
              fluidRow(
                selectInput("department", "Choose a Department:", choices = unique(data$Department)),
                selectInput("ageGroup", "Select Age Group:", choices = c("Under 30", "30-50", "Over 50")),
                box(title = "Demographic Breakdown", plotlyOutput("demographicPlot"), width = 6),
                box(title = "Salary and Compensation", plotlyOutput("salaryPlot"), width = 6),
                box(title = "Attrition by Job Role", plotlyOutput("attritionByRolePlot"), width = 6),
                box(title = "Heat Map of Correlations", plotlyOutput("correlationHeatmap"), width = 6),
                box(title = "3D Work Environment Factors", plotlyOutput("environmentPlot3D"), width = 12)
              )
      ),
      tabItem(tabName = "logistic",
              fluidRow(
                box(title = "Logistic Regression Model Explanation", plotlyOutput("logisticPlot"), width = 12)
              )
      ),
      tabItem(tabName = "summary",
```



Utilizing Shiny's functions, define the UI layout in your R script. This will contain your desired plot locations as well as any input controls (such as dropdown menus and sliders).

- dashboardPage: This function designs the layout of the primary dashboard page.It consists of the main body, which will have tabbed information, as well as a header and sidebar.
  The dashboard's title is set by the 'title' argument.

- dashboardHeader: This function generates the dashboard's header. Usually, it contains the dashboard's title.

- dashboardSidebar: This function adds a sidebar menu to the dashboard's left side.It has menu items (made with'menuItem') that correspond to various dashboard tabs and pages.

- sidebarMenu: With this function, a menu is created inside the sidebar. Every 'menuItem' on the dashboard denotes a tab or page. This screen contains four tabs: "Summary", "Overview", "Visualizations", and "Logistic Regression".

- dashboardBody: This function builds the dashboard's core structure. All of the tab/page contents listed in 'tabItems' are contained in it.

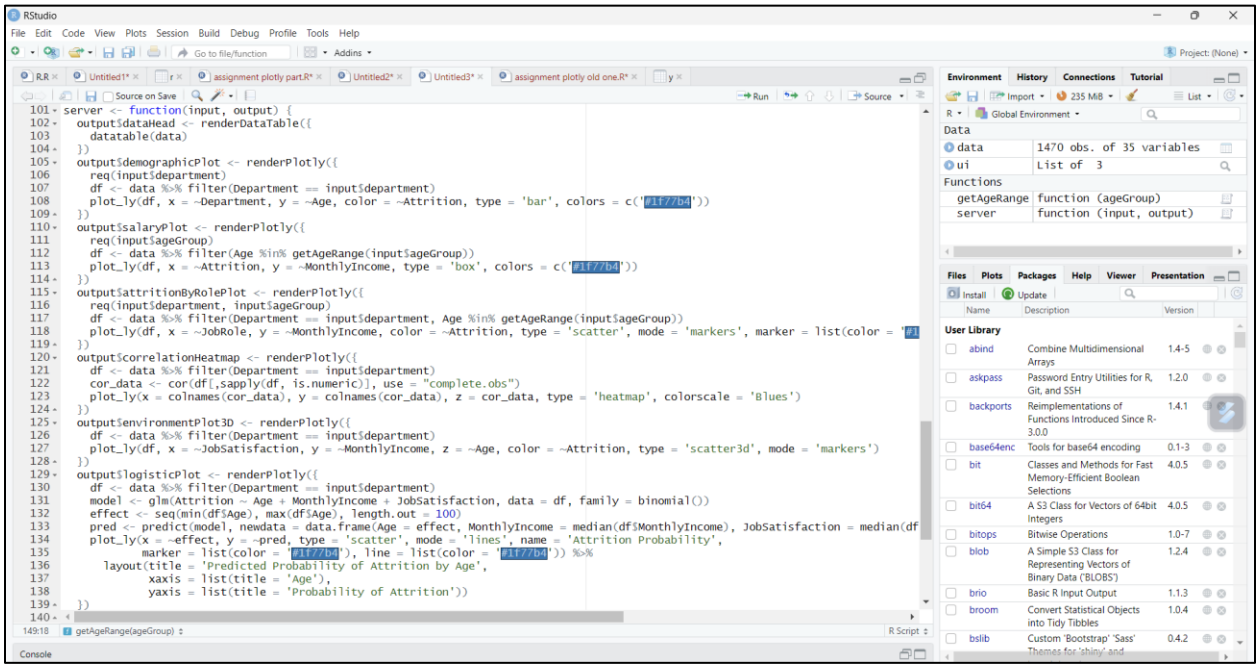
- tags$head(tags$style(HTML(...))): The dashboard's appearance can be customized with CSS styling in this area. It establishes box shadows, font styles, colors, and background pictures.

- tabItems: The content of each tab or page is defined by this function.
  Every 'tabItem' has an associated tab that is defined in the'sidebarMenu'. Various user interface (UI) components, such "fluidRow" and "box," are included inside each "tabItem" to arrange and present material.

- Content of tabItems: A row in the layout is represented by a "fluidRow," which is contained in each "tabItem." One or more box components, each of which represents a box inside the row, are contained inside each 'fluidRow'. Plots made with 'plotlyOutput', text, and other UI elements can all be contained in 'box' elements.

4. Establish the Server Logic:

Afterwards, specify the server logic, which comprises the way the application reacts to user inputs and produces the graphs.



output$dataHead <- renderDataTable({ datatable(data) }): This code renders a data table ('datatable') using the 'renderDataTable' function from the 'DT' package. It displays the head of the dataset (data) in the "Overview" tab.

output$demographicPlot <- renderPlotly: code generates a Plotly bar chart representing demographic breakdown based on the selected department. It filters the dataset ('data') based on the selected department ('input$department') and plots the Age against the Department, colored by Attrition status.
'output$salaryPlot <- renderPlotly({ ... })': This code creates a Plotly box plot showing salary distribution by Attrition status for the selected age group. It filters the dataset based on the selected age group (input$ageGroup) and plots MonthlyIncome against Attrition.

'output$attritionByRolePlot <- renderPlotly({ ... })': This code generates a Plotly scatter plot representing attrition by job role for the selected department and age group. It filters the dataset based on the selected department and age group and plots MonthlyIncome against JobRole, colored by Attrition.

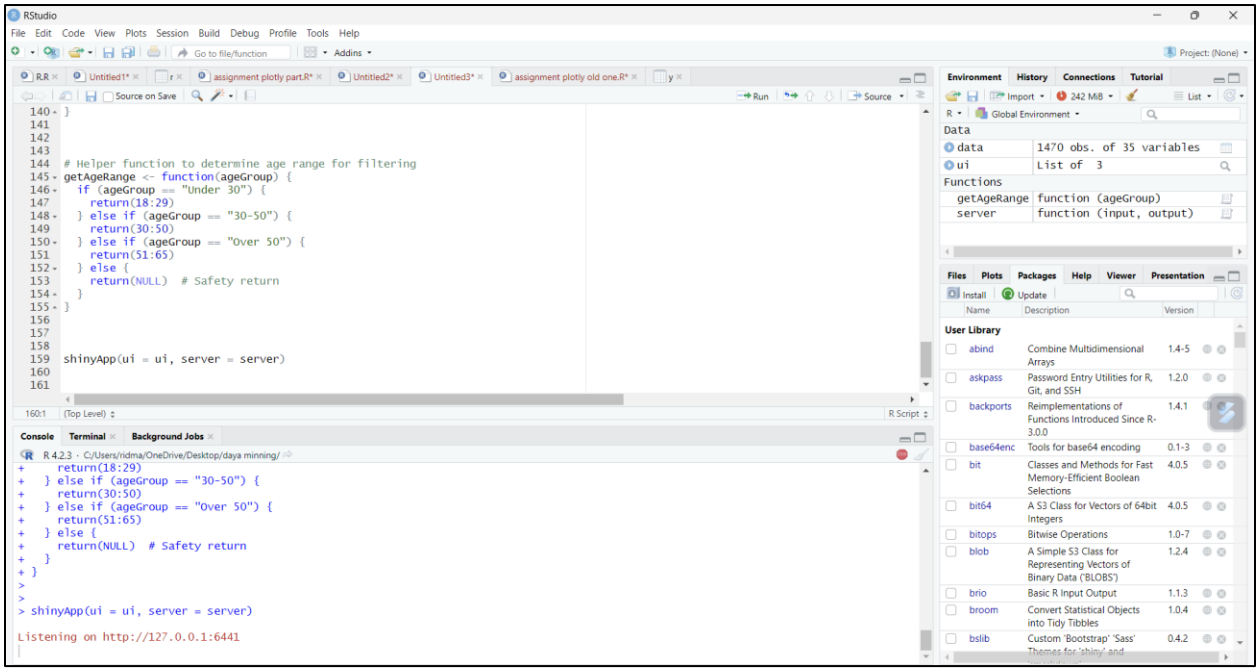
‘output$correlationHeatmap <- renderPlotly’: This code generates a Plotly heatmap that shows the department-specific correlation matrix of numerical data.
It uses Plotly to plot the correlation matrix ('cor_data') of numerical variables.

‘output$environmentPlot3D <- renderPlotly’: This code creates a three-dimensional scatter plot colored by the department's attrition status, which shows age, monthly pay, and job happiness. Plotting Job Satisfaction against Age and Monthly Income, it shows how Attrition status affects the points.

‘output$logisticPlot <- renderPlotly’: In order to forecast attrition likelihood based on age, monthly income, and job satisfaction, this code constructs a logistic regression model.
The expected probability of attrition with age is displayed in a Plotly line plot that is produced.

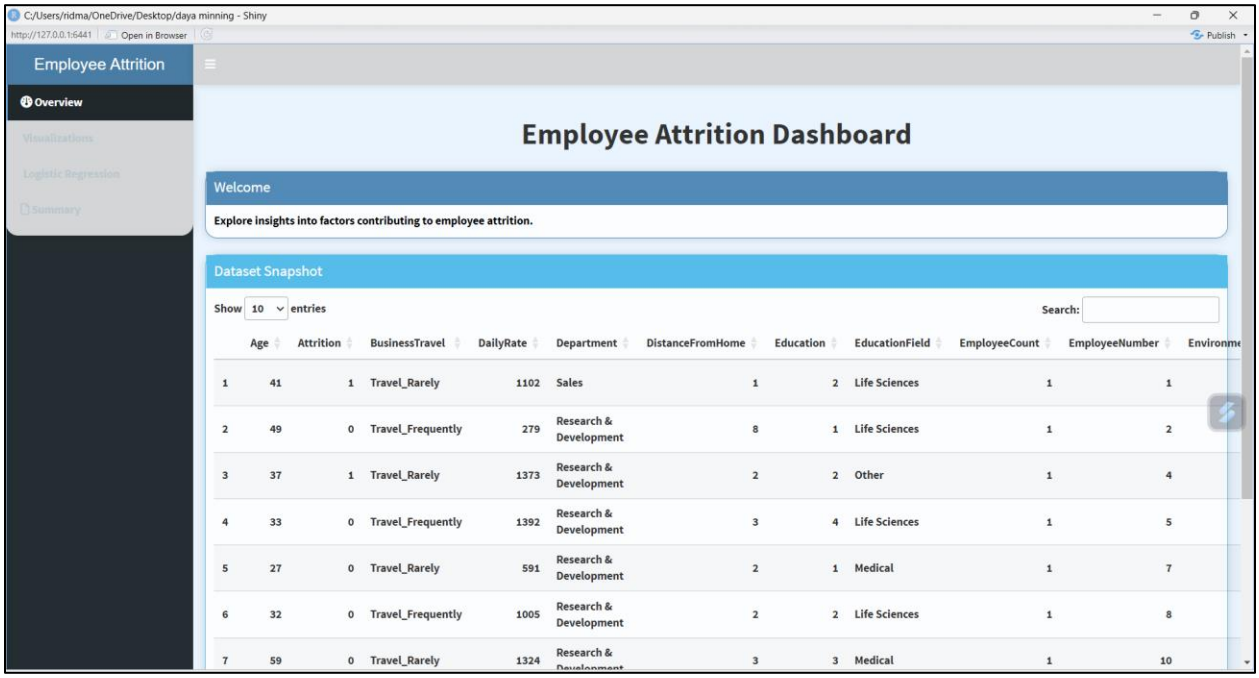
5. Launch the App:

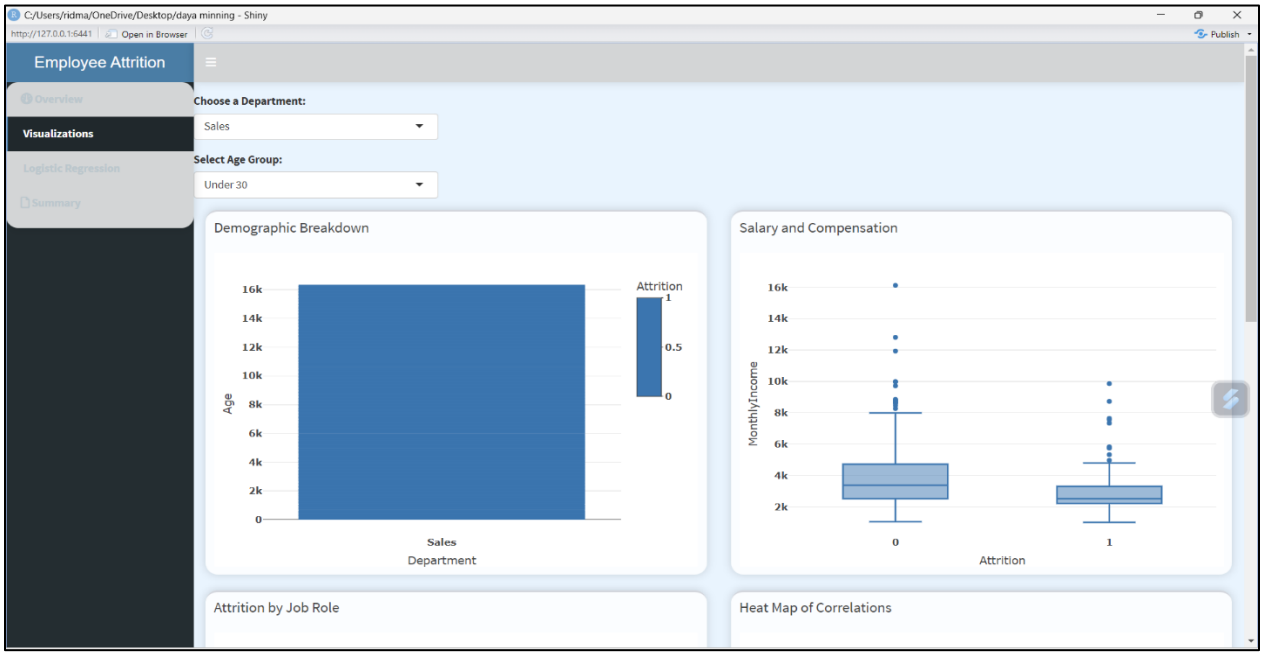
Finally, use the'shinyApp' method to integrate the server logic and UI, and then launch the application:

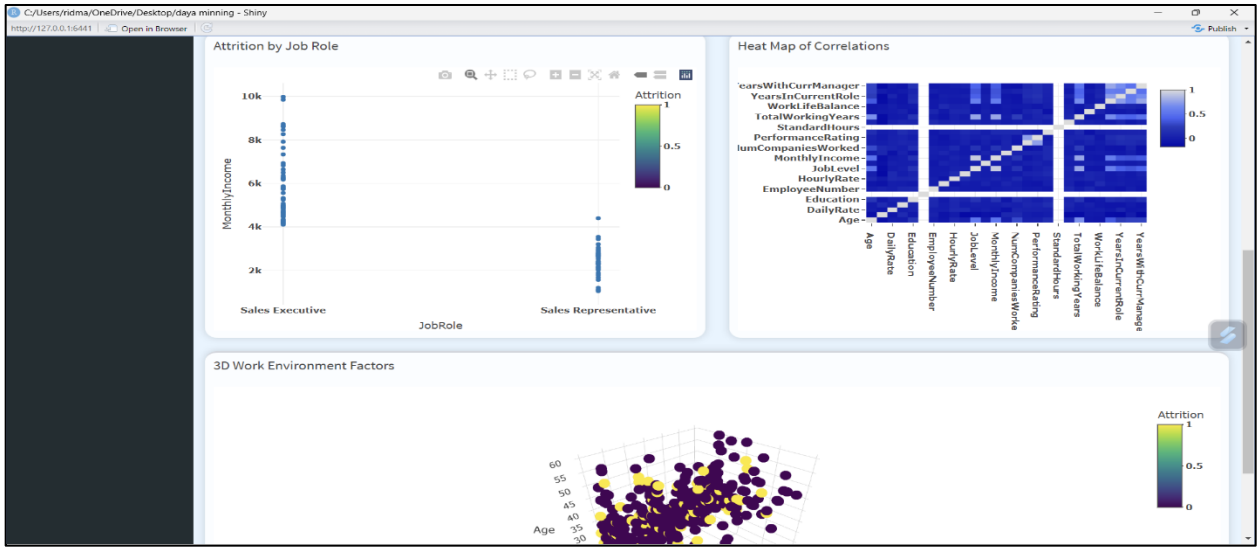


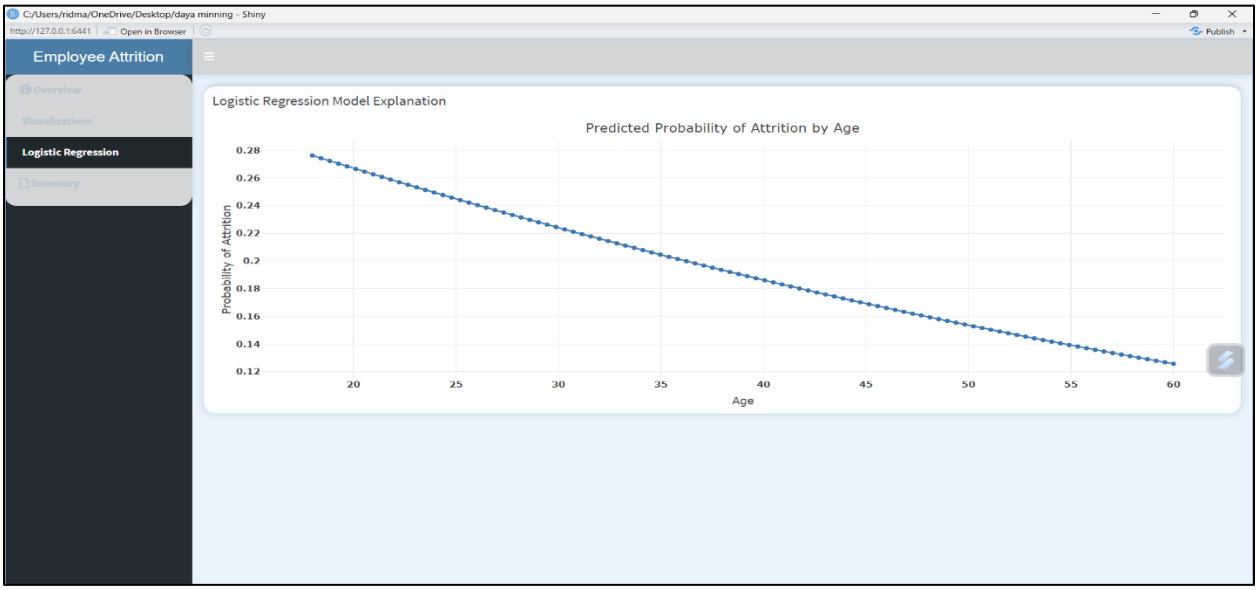
6. View Your Dashboard:

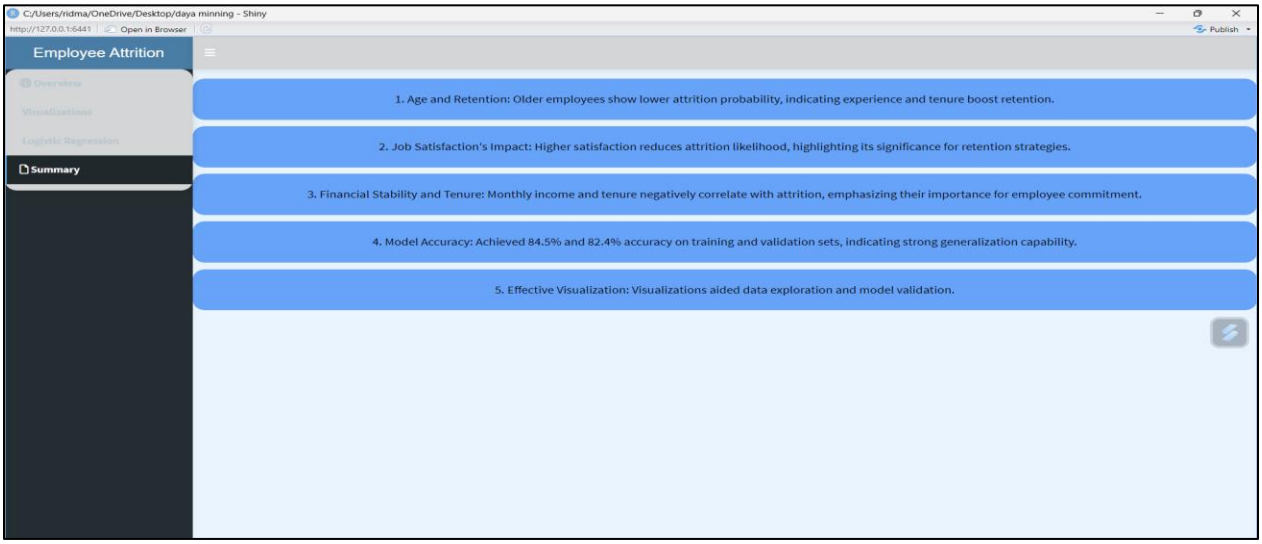
When you launch the application, your dashboard ought to show up automatically in a web browser window.



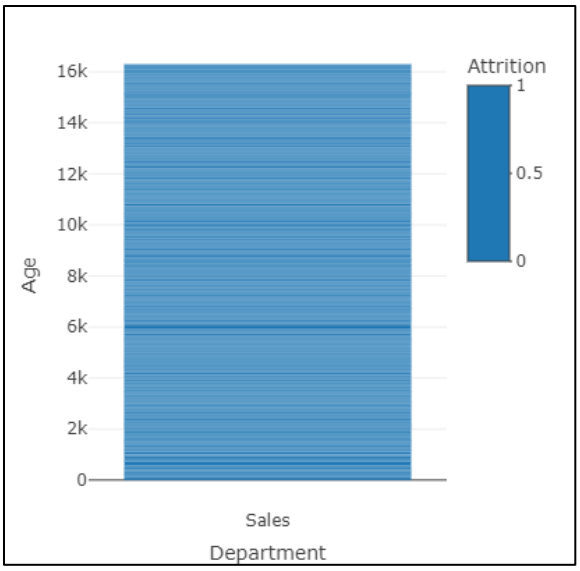



C:/Users/ridma/OneDrive/Desktop/daya minning - Shiny
http://127.0.0.1:6441
Open in Browser
Publish
Attrition by Job Role
Attrition
MonthlyIncome
10k
8k
6k
4k
2k
Sales Executive
Sales Representative
JobRole
Heat Map of Correlations
YearsWithCurrManager
YearsInCurrentRole
WorkLifeBalance
TotalWorkingYears
StandardHours
PerformanceRating
NumCompaniesWorked
MonthlyIncome
JobLevel
HourlyRate
EmployeeNumber
Education
DailyRate
Age
3D Work Environment Factors
Attrition
Age


C:/Users/ridma/OneDrive/Desktop/daya minning - Shiny
http://127.0.0.1:6441
Open in Browser
Publish
Employee Attrition
Overview
Visualizations
Logistic Regression
Summary
Logistic Regression Model Explanation
Predicted Probability of Attrition by Age
Probability of Attrition
0.28
0.26
0.24
0.22
0.2
0.18
0.16
0.14
0.12
20
25
30
35
40
45
50
55
60
Age

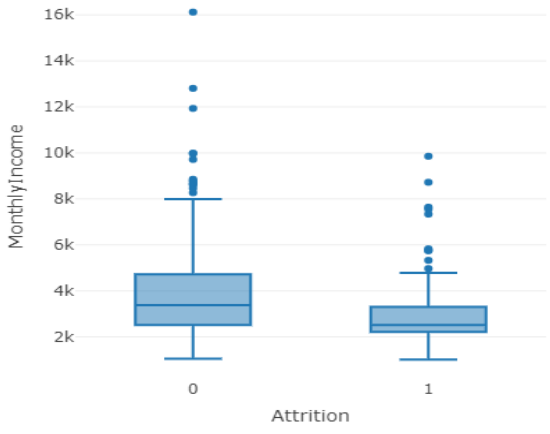

C:/Users/ridma/OneDrive/Desktop/daya minning - Shiny
http://127.0.0.1:6441
Open in Browser
Publish
Employee Attrition
Overview
Visualizations
Logistic Regression
Summary
1. Age and Retention: Older employees show lower attrition probability, indicating experience and tenure boost retention.
2. Job Satisfaction's Impact: Higher satisfaction reduces attrition likelihood, highlighting its significance for retention strategies.
3. Financial Stability and Tenure: Monthly income and tenure negatively correlate with attrition, emphasizing their importance for employee commitment.
4. Model Accuracy: Achieved 84.5% and 82.4% accuracy on training and validation sets, indicating strong generalization capability.
5. Effective Visualization: Visualizations aided data exploration and model validation.

Demographic Breakdown Plot



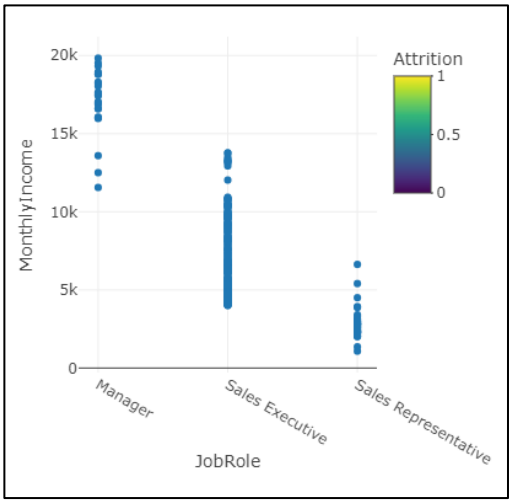
The Attrition by Job Role Plot sheds light on attrition trends among various organizational job roles. The different employment positions are plotted on the X-axis, and variables like years of experience or monthly income are plotted on the Y-axis. HR departments can take proactive steps to retain important personnel by using this map to identify roles that may be experiencing higher attrition rates. Organizations can reduce turnover in crucial roles by implementing targeted retention tactics, such as career development programs or incentive schemes, by studying attrition rates per employment role. Comprehending the dynamics of employee attrition based on job function allows firms to cultivate a positive work environment that promotes professional development and job satisfaction.

Salary and Compensation Plot



The distribution of employee satisfaction ratings among different age groups and departments is shown visually in the Employee Satisfaction Heatmap. Departments or age groups are shown along the X-axis, and other metrics of satisfaction, including work-life balance or job satisfaction, are plotted along the Y-axis. HR teams can use this heatmap as a diagnostic tool to identify problem areas and gauge general employee engagement and morale. Through the identification of departments or age groups that exhibit lower levels of satisfaction, firms can go forward and implement focused interventions, including employee feedback programs or workplace culture improvement projects. Comprehending the subtleties of employee contentment among diverse demographic cohorts empowers establishments to foster a nurturing and gratifying workplace, hence amplifying employee retention and overall organizational efficacy.

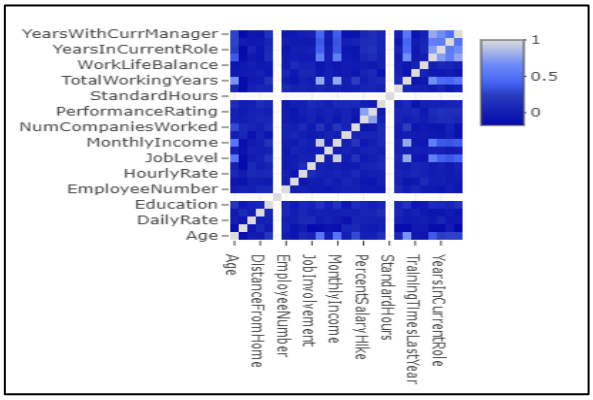
Attrition by Job Role Plot



The Employee Engagement Survey Results chart provides an overview of employee engagement levels across different departments or age groups. On the X-axis, departments or age categories are listed, while the Y-axis represents the level of employee engagement, often measured through survey responses or metrics such as participation rates in company events. High levels of engagement indicate a motivated and committed workforce, while lower levels may suggest potential issues with communication, leadership, or organizational culture. By analyzing these results, HR can identify areas for improvement and implement strategies to enhance employee engagement, fostering a more positive and productive work environment. Understanding

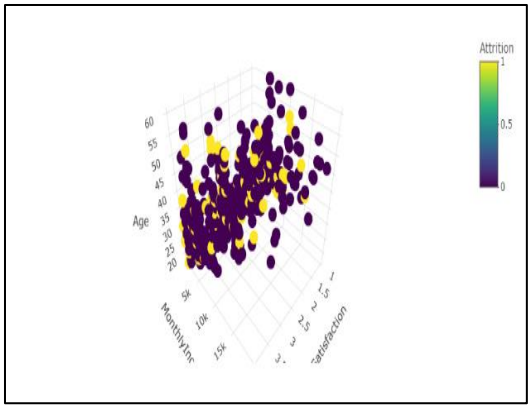
employee engagement trends across various demographic groups enables HR to tailor initiatives and interventions that resonate with employees' needs and preferences, ultimately driving higher levels of satisfaction and retention.

Correlation Heatmap



A thorough overview of employee performance across departments or age groups is given by the Performance Evaluation Ratings Analysis graphic. Departments or age groups are plotted on the X-axis, while performance ratings—which are usually indicated by indicators like productivity levels or performance appraisal scores—are plotted on the Y-axis. Employees with high performance ratings regularly meet or surpass expectations, while those with lower scores could need more work or development. HR can pinpoint high performers, pinpoint areas of excellence, and close performance gaps with focused coaching or training programs by examining performance trends. Comprehending the performance trends of several demographic groupings facilitates HR in executing talent management tactics that maximize workforce efficiency and augment organizational prosperity.

3D Work Environment Factors Plot



The Employee Well-being Dashboard provides a holistic view of employee well-being across different departments or age groups. On the X-axis, departments or age categories are listed, while the Y-axis represents various well-being metrics, such as work-life balance scores, stress levels, or overall satisfaction ratings. This dashboard offers interactive features that allow users to explore the relationships between different well-being factors and identify potential areas of concern or improvement. By analyzing these interactions, HR can implement targeted interventions and wellness programs to promote employee health, happiness, and productivity. Understanding the multifaceted nature of employee well-being enables organizations to create supportive environments that foster employee engagement and retention.