StartUp Insights

Rechel, Supraja, Kushal

2023-04-05

```
# Installing the required packages
# Installing plyr package from CRAN repository
install.packages('plyr', repos = "http://cran.us.r-project.org")
```

```
## Installing package into 'C:/Users/Rechel Sardar/AppData/Local/R/win-library/4.2'
## (as 'lib' is unspecified)
```

```
## package 'plyr' successfully unpacked and MD5 sums checked
```

```
## Warning: cannot remove prior installation of package 'plyr'
```

```
## Warning in file.copy(savedcopy, lib, recursive = TRUE): problem copying
## C:\Users\Rechel
## Sardar\AppData\Local\R\win-library\4.2\00LOCK\plyr\libs\x64\plyr.dll to
## C:\Users\Rechel Sardar\AppData\Local\R\win-library\4.2\plyr\libs\x64\plyr.dll:
## Permission denied
```

```
## Warning: restored 'plyr'
```

```
## 
## The downloaded binary packages are in
##  C:\Users\Rechel Sardar\AppData\Local\Temp\RtmpqwHQAU\downloaded_packages
```

```
# Setting the CRAN repository to RStudio
options(repos = list(CRAN="http://cran.rstudio.com/"))
install.packages("readxl")
```

```
## Installing package into 'C:/Users/Rechel Sardar/AppData/Local/R/win-library/4.2'
## (as 'lib' is unspecified)
```

```
## package 'readxl' successfully unpacked and MD5 sums checked
```

```
## Warning: cannot remove prior installation of package 'readxl'
## Warning in file.copy(savedcopy, lib, recursive = TRUE): problem copying
## C:\Users\Rechel
## Sardar\AppData\Local\R\win-library\4.2\00LOCK\readxl\libs\x64\readxl.dll to
## C:\Users\Rechel
## Sardar\AppData\Local\R\win-library\4.2\readxl\libs\x64\readxl.dll: Permission
## denied

## Warning: restored 'readxl'

##
## The downloaded binary packages are in
##  C:\Users\Rechel Sardar\AppData\Local\Temp\RtmpqwHQAU\downloaded_packages
```

```
install.packages("dplyr")
```

```
## Installing package into 'C:/Users/Rechel Sardar/AppData/Local/R/win-library/4.2'
## (as 'lib' is unspecified)

## package 'dplyr' successfully unpacked and MD5 sums checked

## Warning: cannot remove prior installation of package 'dplyr'

## Warning in file.copy(savedcopy, lib, recursive = TRUE): problem copying
## C:\Users\Rechel
## Sardar\AppData\Local\R\win-library\4.2\00LOCK\dplyr\libs\x64\dplyr.dll to
## C:\Users\Rechel
## Sardar\AppData\Local\R\win-library\4.2\dplyr\libs\x64\dplyr.dll: Permission
## denied

## Warning: restored 'dplyr'

##
## The downloaded binary packages are in
##  C:\Users\Rechel Sardar\AppData\Local\Temp\RtmpqwHQAU\downloaded_packages
```

```
install.packages('openxlsx')
```

```
## Installing package into 'C:/Users/Rechel Sardar/AppData/Local/R/win-library/4.2'
## (as 'lib' is unspecified)

## package 'openxlsx' successfully unpacked and MD5 sums checked
##
```

```
## The downloaded binary packages are in
##  C:\Users\Rechel Sardar\AppData\Local\Temp\RtmpqwHQAU\downloaded_packages
```

```
install.packages("ggplot2")
```

```
## Installing package into 'C:/Users/Rechel Sardar/AppData/Local/R/win-library/4.2'
## (as 'lib' is unspecified)
```

```
## package 'ggplot2' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\Rechel Sardar\AppData\Local\Temp\RtmpqwHQAU\downloaded_packages
```

```
install.packages("e1071")
```

```
## Installing package into 'C:/Users/Rechel Sardar/AppData/Local/R/win-library/4.2'
## (as 'lib' is unspecified)
```

```
## package 'e1071' successfully unpacked and MD5 sums checked
```

```
## Warning: cannot remove prior installation of package 'e1071'
```

```
## Warning in file.copy(savedcopy, lib, recursive = TRUE): problem copying
## C:\Users\Rechel
## Sardar\AppData\Local\R\win-library\4.2\00LOCK\e1071\libs\x64\e1071.dll to
## C:\Users\Rechel
## Sardar\AppData\Local\R\win-library\4.2\e1071\libs\x64\e1071.dll: Permission
## denied
```

```
## Warning: restored 'e1071'
```

```
##
## The downloaded binary packages are in
##  C:\Users\Rechel Sardar\AppData\Local\Temp\RtmpqwHQAU\downloaded_packages
```

```
install.packages("caret")
```

```
## Installing package into 'C:/Users/Rechel Sardar/AppData/Local/R/win-library/4.2'
## (as 'lib' is unspecified)
```

```
## package 'caret' successfully unpacked and MD5 sums checked
## Warning: cannot remove prior installation of package 'caret'
```

```
## Warning in file.copy(savedcopy, lib, recursive = TRUE): problem copying
## C:\Users\Rechel
## Sardar\AppData\Local\R\win-library\4.2\00LOCK\caret\libs\x64\caret.dll to
## C:\Users\Rechel
## Sardar\AppData\Local\R\win-library\4.2\caret\libs\x64\caret.dll: Permission
## denied
```

```
## Warning: restored 'caret'
```

```
##
## The downloaded binary packages are in
##  C:\Users\Rechel Sardar\AppData\Local\Temp\RtmpqwHQAU\downloaded_packages
```

```
install.packages("caTools")
```

```
## Installing package into 'C:/Users/Rechel Sardar/AppData/Local/R/win-library/4.2'
## (as 'lib' is unspecified)
```

```
## package 'caTools' successfully unpacked and MD5 sums checked
```

```
## Warning: cannot remove prior installation of package 'caTools'
```

```
## Warning in file.copy(savedcopy, lib, recursive = TRUE): problem copying
## C:\Users\Rechel
## Sardar\AppData\Local\R\win-library\4.2\00LOCK\caTools\libs\x64\caTools.dll to
## C:\Users\Rechel
## Sardar\AppData\Local\R\win-library\4.2\caTools\libs\x64\caTools.dll: Permission
## denied
```

```
## Warning: restored 'caTools'
```

```
##
## The downloaded binary packages are in
##  C:\Users\Rechel Sardar\AppData\Local\Temp\RtmpqwHQAU\downloaded_packages
```

```
install.packages("rcompanion")
```

```
## Installing package into 'C:/Users/Rechel Sardar/AppData/Local/R/win-library/4.2'
## (as 'lib' is unspecified)
## package 'rcompanion' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\Rechel Sardar\AppData\Local\Temp\RtmpqwHQAU\downloaded_packages
```

```
install.packages("rpart.plot")
```

```
## Installing package into 'C:/Users/Rechel Sardar/AppData/Local/R/win-library/4.2'
## (as 'lib' is unspecified)
```

```
## package 'rpart.plot' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\Rechel Sardar\AppData\Local\Temp\RtmpqwHQAU\downloaded_packages
```

```
# Merging two datasets
# Reading the existing datasets from Excel file
library(readxl)
existing_data <- read_excel("D:\\dsr.xlsx")
# Printing the class of each column
sapply(existing_data,class)
```

```
##      Company Name    Employee Count            IPO         Founded
##       "character"       "character"    "character"       "numeric"
##      Headquarters            Sector    Description        Founders
##       "character"       "character"    "character"     "character"
##         Investors Amount(in dollars)         Stage           Month
##       "character"       "character"    "character"       "numeric"
##              Type
##       "character"
```

```
# Reading the new dataset to be added from Excel file
new_data <- read_excel("D:\\dsr2.xlsx")
# Printing the class of each column
sapply(new_data,class)
```

```
##      Company Name    Employee Count            IPO         Founded
##       "character"       "character"    "character"       "numeric"
##      Headquarters            Sector    Description        Founders
##       "character"       "character"    "character"     "character"
##         Investors Amount(in dollars)         Stage           Month
##       "character"       "character"       "character"         "logical" 
##              Type 
##       "character" 
```

```
# Manipulating the class to match them across the two datasets
# Converting Month column to character type
new_data$Month=as.character(new_data$Month)
# Loading the dplyr package for data manipulation
library(dplyr)
```

```
## 
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
## 
##     filter, lag
```

```
## The following objects are masked from 'package:base':
## 
##     intersect, setdiff, setequal, union
```

```
# Replacing empty strings with NA in Month column and converting it to numeric type
new_data$Month <- na_if(new_data$Month, '')
new_data$Month=as.numeric(new_data$Month)

# Adding new rows to the existing dataset
data <- rbind(existing_data, new_data)
# Printing the class of each column in merged dataset
sapply(data,class)
```

```
##      Company Name    Employee Count               IPO           Founded 
##       "character"       "character"       "character"         "numeric" 
##      Headquarters            Sector       Description          Founders 
##       "character"       "character"       "character"       "character" 
##         Investors Amount(in dollars)             Stage             Month 
##       "character"       "character"       "character"         "numeric" 
##              Type 
##       "character" 
```

```
# Preprocessing of Employee Count
# Printing unique values in Employee Count column
unique(data$`Employee Count`)
```

```
##  [1] "(201-500)"    "(501-1000)"   "(51-200)"     "(11-50)"      "(2-10)"
##  [6] "(1001-5000)"  NA             "(5001-10000)" "(10001+)"     "44936.0"
## [11] "45250.0"      "1.0"          "(101-250)"    "(51-100)"     "(251-500)"
## [16] "(10,001+)"    "(1-10)"
```

```
# Counting the number of occurrences of each value in Employee Count column
table(data$`Employee Count`)
```

```
##
##       (1-10)    (10,001+)     (10001+)  (1001-5000)    (101-250)      (11-50)
##            1            3            5           93            2          493
##       (2-10)    (201-500)    (251-500) (5001-10000)   (501-1000)     (51-100)
##          141          173            1           12          111            2
##     (51-200)          1.0      44936.0      45250.0
##          414            1            5            1
```

```
# Converting Employee Count column to a factor type with ordered levels
data$`Employee Count`<- factor(data$`Employee Count`,
                         levels =
c("(2-10)","(11-50)","(51-100)","(51-200)","(101-250)","(201-500)","(251-500)","(501-1000)","(1001-5000)","(5001-10000)","(10001+)"),
                         labels = c("A", "B", "C","D","E","F","G","H","I","J","K"))
# Counting the number of occurrences of each value in Employee Count column
table(data$`Employee Count`)
```

```
##
##   A   B   C   D   E   F   G   H   I   J   K
## 141 493   2 414   2 173   1 111  93  12   5
```

```
# Preprocessing of IPO

# Replacing "YES" with "Yes"
data$IPO=gsub("YES","Yes",data$IPO)
# Replacing "No"/"Yes" with 0/1 in IPO column
data$IPO=ifelse(data$IPO=="No",0,1)
# Counting the number of occurrences of each value in IPO column
table(data$IPO)
```

```
## 
##    0    1 
## 1479   11
```

```
# Preprocessing of Headquarters
```

```
# Renaming the "Headquarters" column to "cities"
colnames(data)[colnames(data) == "Headquarters"] = "cities"
# Printing the class of each column in merged dataset
sapply(data,class)
```

```
##       Company Name     Employee Count                IPO            Founded 
##        "character"           "factor"          "numeric"          "numeric" 
##             cities             Sector        Description           Founders 
##        "character"        "character"        "character"        "character" 
##          Investors Amount(in dollars)              Stage              Month 
##        "character"        "character"        "character"          "numeric" 
##               Type 
##        "character"
```

```
# Printing unique values in cities column
unique(data$cities)
```

```
##  [1] "Bangalore"             "Gurugram"            
##  [3] "Pune"                  "Mumbai"              
##  [5] "New Delhi"             "Noida"               
##  [7] "Chandigarh"            "Chennai"             
##  [9] "Bhubaneswar"           "Haryana"             
## [11] "Powai"                 "Bhilwara"            
## [13] "Satara"                "Hyderabad"           
## [15] "Goa"                   "Coimbatore"          
## [17] "Telengana"             "Jaipur"              
## [19] "Ghaziabad"             "Ahmedabad"           
## [21] "Gujarat"               "Rajsamand"           
## [23] "Samsitpur"             "Kochi"               
## [25] "Indore"                "Ambernath"           
## [27] "Thiruvananthapuram"    "Roorkee"             
## [29] "Gandhinagar"           "Panchkula"           
## [31] "Trivandrum"            "Thane"               
## [33] "Lucknow"               "Jharkhand"           
## [35] "Kolkata"               "Faridabad"
## [37] "West Bengal"                "Orissa"
## [39] "Mangalore"                  "London"
## [41] "Andheri"                    "Guwahati"
## [43] "Kanpur"                     "The Nilgiris"
## [45] "Ranchi"                     "Jodhpur"
## [47] "Small Towns, Andhra Pradesh" "Surat"
## [49] "Nagpur"                     "Kottayam"
## [51] "Vadodara"                   "Delhi"
## [53] "Cochin"                     "Ahemdabad"
## [55] "Gurgaon"                    "Bhubhneshwar"
## [57] "Taramani"                   "Tirivanthapuram"
## [59] "Mohali"                     "Tiruchirappalli"
## [61] "Dehradun"                   "Karnataka"
## [63] "Mysore"                     "Lonavala"
## [65] "kanpur"                     "Nasik"
## [67] "Visakhapatnam"              "Raipur"
## [69] "Karachi"                    "gurgaon"
## [71] "Bhopal"                     "Maharashtra"
```

```
# Preprocessing of Sector
```

```
# Counting the number of occurrences of each value in Sector column
head(table(data$Sector),50)
```

```
##
##          3D AI company           Advertisement
##                      1                       1
##             Aeorspace  Aerospace, Manufacturing
##                      1                       2
##            Agriculture                 AgriTech
##                      3                      17
##                 Agtech               AI Chatbot
##                      1                       1
##             AI company               AI startup
##                      2                       8
##              Analytics        Apparel & Fashion
##                      1                      12
##             AR startup  Artificial Intelligence
##                      1                       3
##          Arts & Crafts        Augmented reality
##                      2                       1
##                      Automation                  Automobile 
##                               3                           1 
##        Automobile Manufacturing                  Automotive 
##                               1                          36 
##              Autonomous Vehicle                    Aviation 
##                               1                           1 
##           Aviation & Aerospace                         B2B 
##                               2                           2 
##                  B2B E-commerce             B2B marketplace 
##                               3                           1 
##                 B2B Marketplace                 B2B service 
##                               2                           2 
##                     B2B startup                  B2B Travel 
##                               1                           1 
##                         Banking                      Beauty 
##                               2                           2 
##                 Beauty products                   Beverages 
##                               1                           1 
##                     Bike Rental               Biotechnology 
##                               1                          12 
##                   BioTechnology                  Blockchain 
##                               5                           3 
##              Blockchain startup             Broadcast Media 
##                               1                           1 
##                    Broadcasting          Building Materials 
##                               1                           2 
## Business Supplies & Equipment            Cannabis startup 
##                               1                           1 
##                 Capital Markets        Celebrity Engagement 
##                               1                           1 
##                        Chemical                   Cleantech 
##                               1                           1 
##                       CleanTech               Cloud kitchen 
##                               1                           1
```

```
# Preprocessing of Founded

# Printing unique values in Founded column
year=c(unique(data$Founded))
year
```

```
##  [1] 2016 2018 2020 2021 2014 2015 2017 2019 2011 2009 2008 2007 2012 2006 2010
## [16] 2004 2013 1963 1984 2005 1999 2000 1994 1991 2003 1993 2002 1989 1978 1998
## [31]   NA 2022 1982 1996 1924
```

```
# Preprocessing of Amount(in dollars) column

# Replacing empty cells and cells with 'Undisclosed' value with NA
library(dplyr)
data$`Amount(in dollars)` <- na_if(data$`Amount(in dollars)`, '')
data$`Amount(in dollars)` <- na_if(data$`Amount(in dollars)`, 'Undisclosed')

# Removing comma separator from Amount(in dollars)
data$`Amount(in dollars)`=gsub(",","",data$`Amount(in dollars)`)

# Changing the datatype of the 'Amount(in dollars)' column to numeric
data$`Amount(in dollars)`=as.numeric(data$`Amount(in dollars)`)
```

```
## Warning: NAs introduced by coercion
```

```
summary(data$`Amount(in dollars)`)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.      NA's
## 1.960e+04 1.000e+06 3.000e+06 1.917e+08 1.200e+07 1.500e+11       592
```

```
# Checking how many NA values are there in the column
sum(is.na(data$`Amount(in dollars)`))
```

```
## [1] 592
```

```
# Displaying the first 20 rows of the 'Amount(in dollars)' column
head(data$`Amount(in dollars)`,20)
```

```
##  [1] 2.50e+06 7.00e+06 5.00e+06 1.50e+07 2.00e+06 1.10e+06 1.00e+06 3.00e+06
##  [9] 5.00e+06 6.00e+05       NA 1.50e+06 2.55e+08 2.00e+07 1.00e+06 3.00e+06
## [17] 2.10e+07 1.10e+07 5.20e+06       NA
```

```
# Preprocessing of Type column

# Changing all instances of 'Product' to 'product'
data$Type=gsub("Product","product",data$Type)
# Changing all instances of 'Service' to 'service'
data$Type=gsub("Service","service",data$Type)
```

```
# Displaying the first 10 rows of the 'Type' column
head(data$Type,10)
```

```
##  [1] "product" "product" "product" "service" "product" "service" "product"
##  [8] "product" "product" "service"
```

```
# Preprocessing of Stage
```

```
# Checking the unique values in the 'Stage' column
unique(data$Stage)
```

```
##  [1] NA
##  [2] "Series A"
##  [3] "Series B"
##  [4] "Pre-seed"
##  [5] "Seed"
##  [6] "Series D"
##  [7] "Debt"
##  [8] "Series C"
##  [9] "Series E"
## [10] "Series F"
## [11] "Pre-series"
## [12] "Series G"
## [13] "Early seed"
## [14] "Bridge"
## [15] "Undisclosed"
## [16] "Series H"
## [17] "Debt Financing"
## [18] "series A"
## [19] "Blue Ashva Capital, Supack Industries"
## [20] "Pre-series B"
## [21] "pre-seed"
## [22] "seed"
## [23] "Unknown"
## [24] "Preseed"
## [25] "Pre seed"
## [26] "Angel"
## [27] "Corporate Round"
## [28] "Working Capital to SMEs"
## [29] "undisclosed"
```

```
# Replacing empty cells and unwanted values with NA
library(dplyr)
data$Stage <- na_if(data$Stage, '')
data$Stage <- na_if(data$Stage, 'Unknown')
data$Stage <- na_if(data$Stage, 'Blue Ashva Capital, Supack Industries')
data$Stage <- na_if(data$Stage, 'undisclosed')
data$Stage <- na_if(data$Stage, 'Undisclosed')
data$Stage=gsub(c("Pre-series|seed"),"Seed",data$Stage)
data$Stage=gsub(c("Pre-seed|pre-Seed|PreSeed|Pre-Seed|Pre Seed"),"Pre seed",data$Stage)
data$Stage=gsub(c("Debt Financing"),"Debt",data$Stage)
data$Stage=gsub(c("Seed B"),"Seed",data$Stage)
data$Stage=gsub(c("Early Seed"),"Seed",data$Stage)
data$Stage=gsub(c("series A"),"Series A",data$Stage)
unique(data$Stage)
```

```
##  [1] NA                      "Series A"
##  [3] "Series B"              "Pre seed"
##  [5] "Seed"                  "Series D"
##  [7] "Debt"                  "Series C"
##  [9] "Series E"              "Series F"
## [11] "Series G"              "Bridge"
## [13] "Series H"              "Angel"
## [15] "Corporate Round"       "Working Capital to SMEs"
```

```
# ANALYSIS TASKS
```

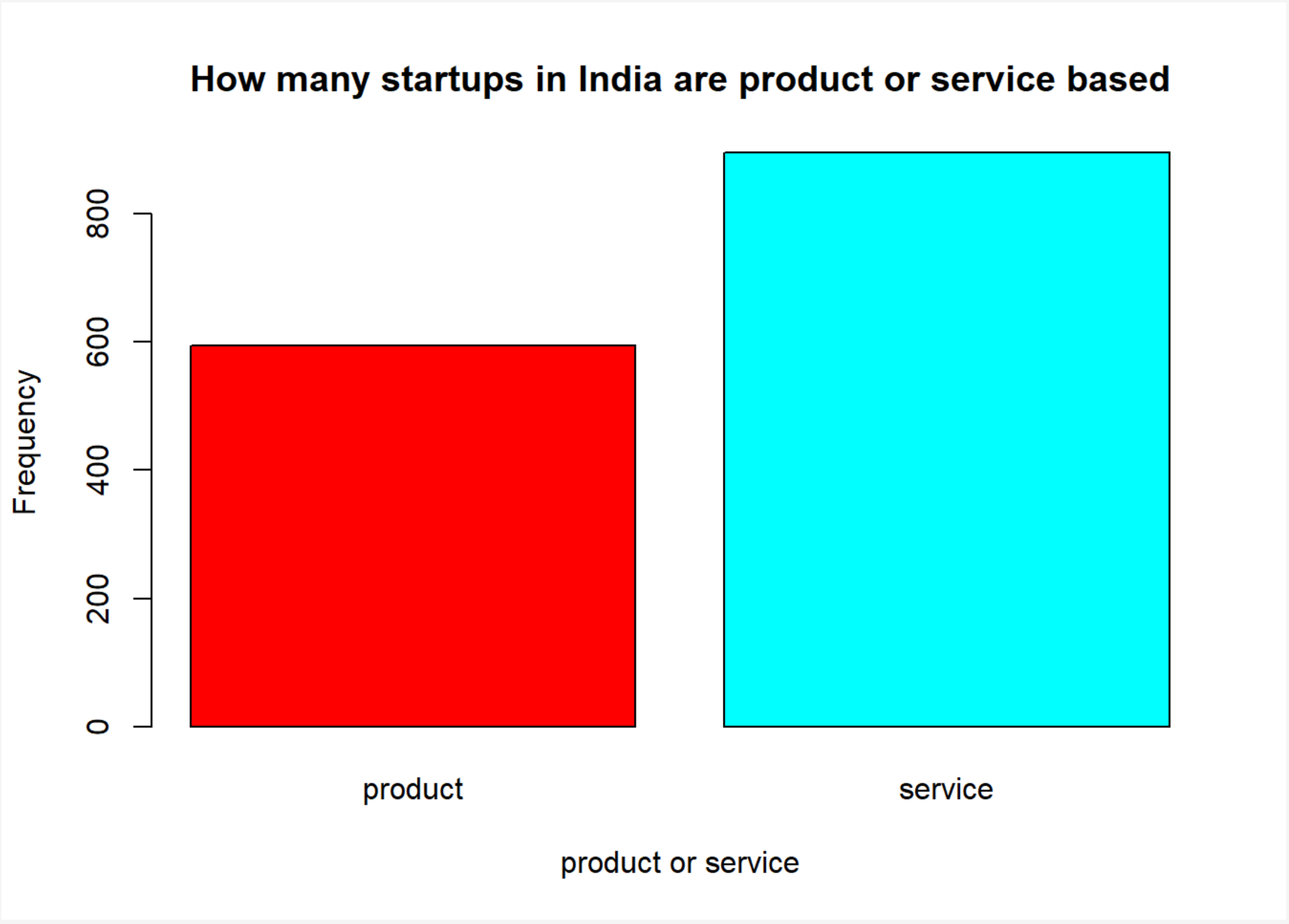
```
# Task 1:
# To showcase the frequency of startups based on their product or service type in India and the top cities
with the highest number of startups - Bangalore, Mumbai, Gurugram, and New Delhi. The first bar chart
displays the frequency of startups based on their product or service type in India. The subsequent bar
charts depict the same analysis for each of the top cities individually.
#The input is a table of frequency (numeric) of Type column (character). It is the respective count of product
and service (for India and a particular city).
```

```
# Plotting bar chart for all rows
freq=table(data$Type) # Counting the frequency of startups based on Type column
freq
```

```
##
```

```
## product service
##     594     896
```

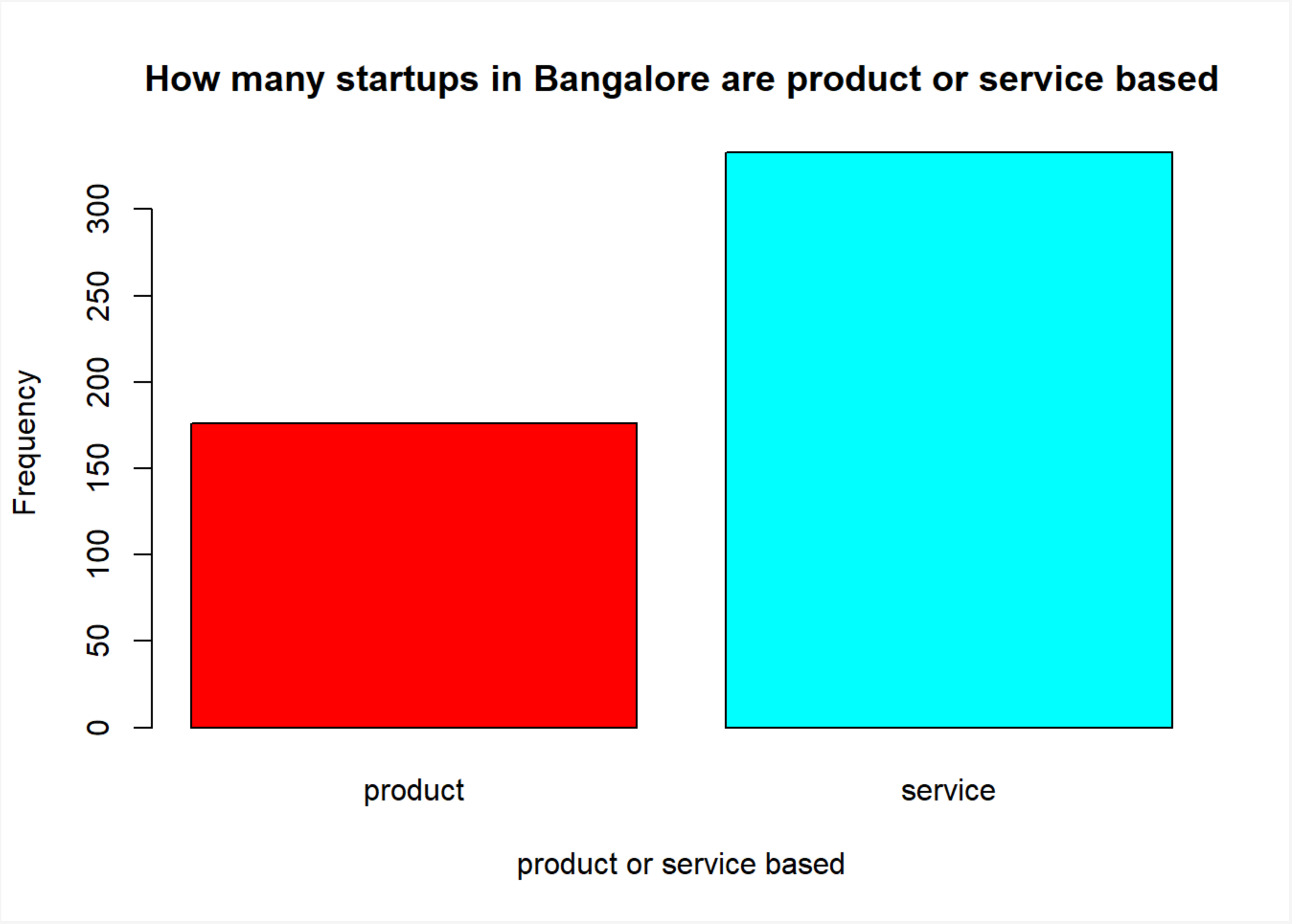
```
barplot(freq, main = "How many startups in India are product or service based", xlab = "product or service",
ylab = "Frequency", col = rainbow(length(freq)))  # Plotting the bar chart for the frequency of startups based
on product or service
```



```
df=data.frame(data) # Creating a data frame from the original data
## Plotting bar chart for Bangalore
B=df[df$cities=="Bangalore",] # Filtering the data for startups in Bangalore
freq1=table(B$Type) # Counting the frequency of startups based on product or service for Bangalore
freq1
```

```
##
## product service
##     176     333
```

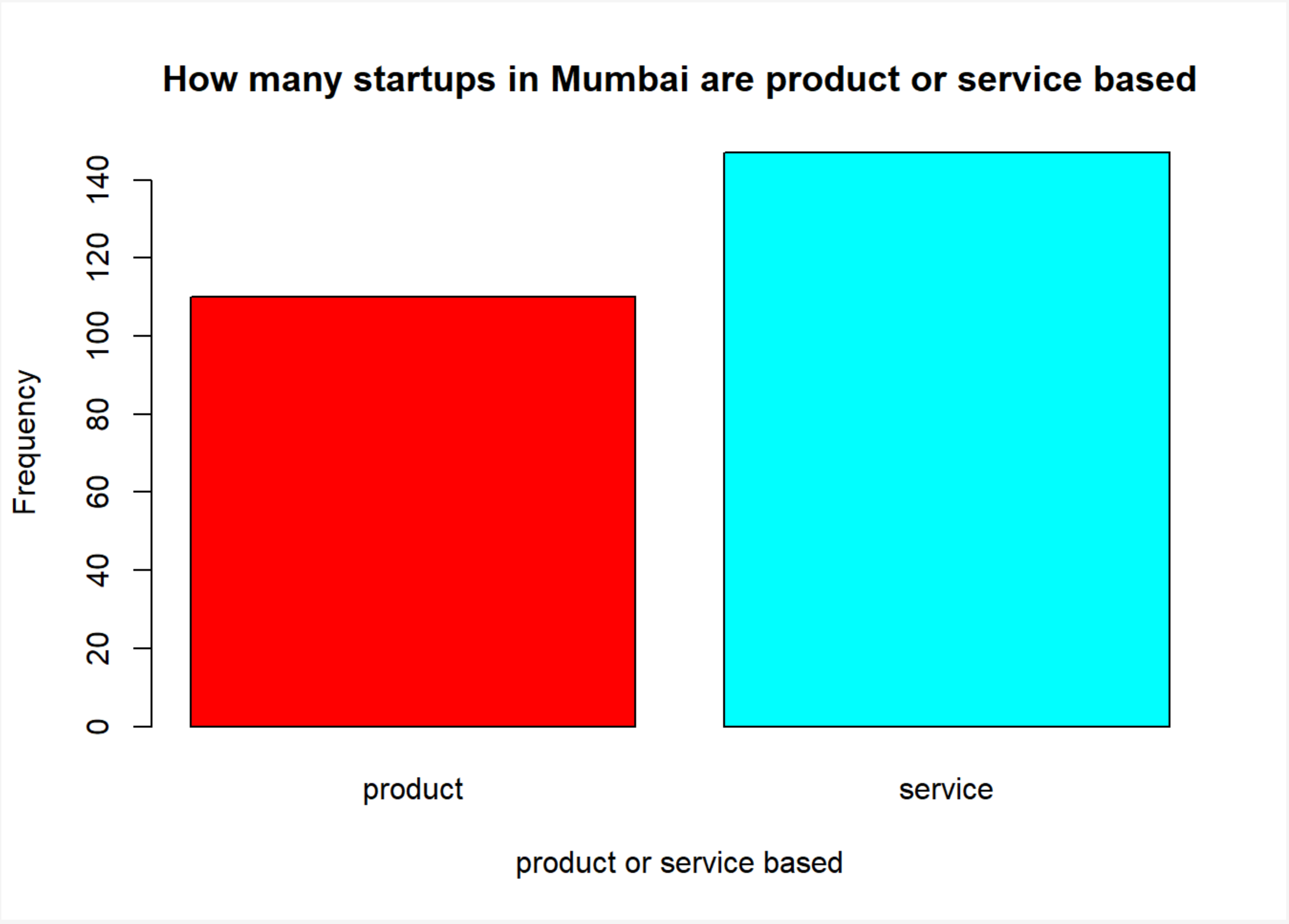
```
barplot.default(freq1,main = "How many startups in Bangalore are product or service based",xlab = "product or service based",ylab ="Frequency", col = rainbow(length(freq1))) # Plotting the bar chart for the frequency of startups based on product or service for Bangalore
```



```
##Plotting bar chart for Mumbai
M=df[df$cities=="Mumbai",] # Filtering the data for startups in Mumbai
freq1=table(M$Type) # Counting the frequency of startups based on product or service for Mumbai
freq1
```

```
##
## product service
##     110     147
```

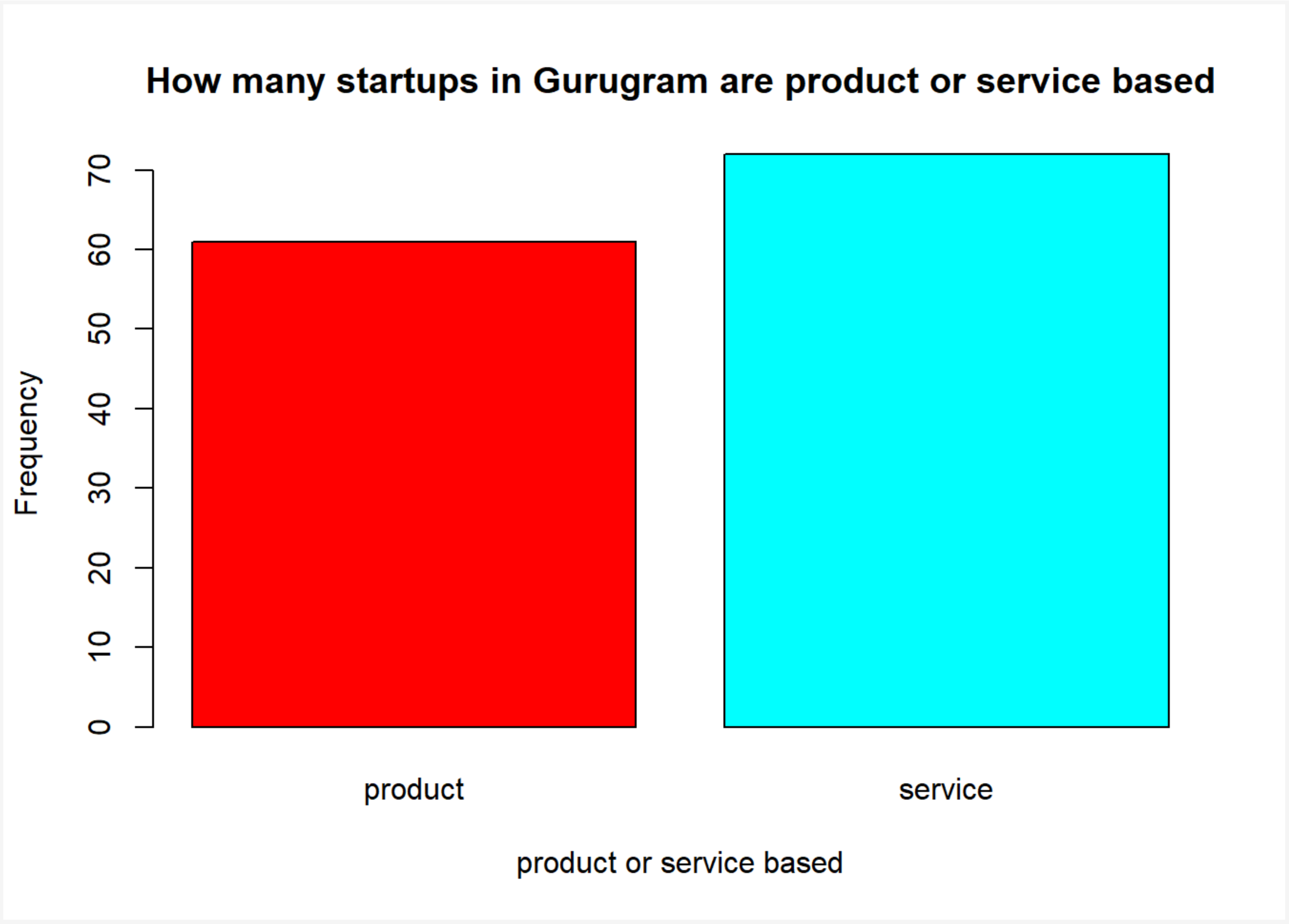
```
barplot.default(freq1,main = "How many startups in Mumbai are product or service based",xlab = "product or service based",ylab ="Frequency", col = rainbow(length(freq1))) # Plotting the bar chart for the frequency of startups based on product or service for Mumbai
```



```
##Plotting bar chart for Gurugram
G=df[df$cities=="Gurugram",] # Filtering the data for startups in Gurugram
freq1=table(G$Type) # Counting the frequency of startups based on product or service for Gurugram
freq1
```

```
##
## product service
##      61      72
```

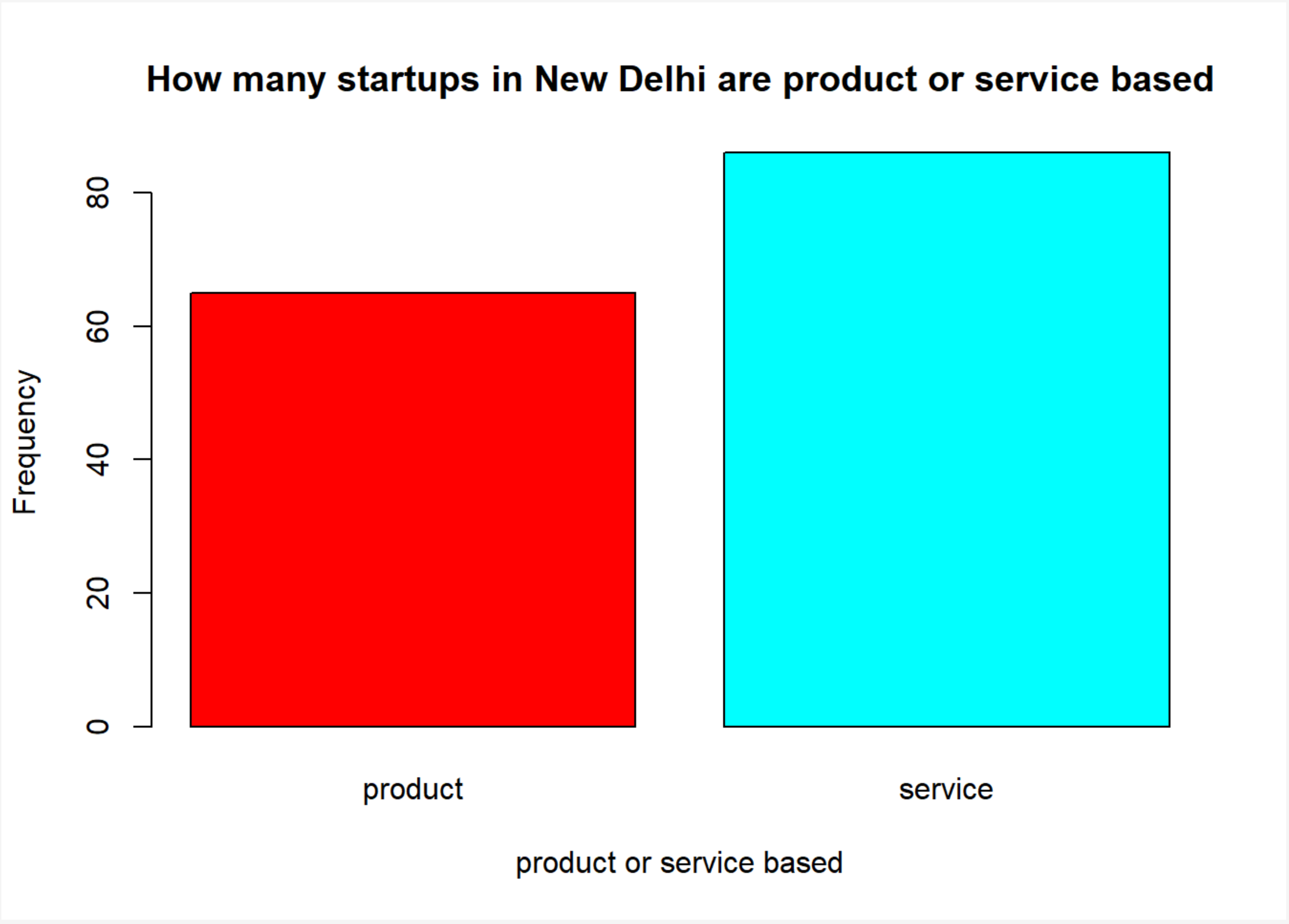
```
barplot.default(freq1,main = "How many startups in Gurugram are product or service based",xlab = "product
or service based",ylab ="Frequency", col = rainbow(length(freq1))) # Plotting the bar chart for the frequency
of startups based on product or service for Gurugram
```



```
##Plotting bar chart for New Delhi
D=df[df$cities=="New Delhi",] # Filtering the data for startups in New Delhi
freq1=table(D$Type) # Counting the frequency of startups based on product or service for New Delhi
freq1
```

```
##
## product service
##      65      86
```

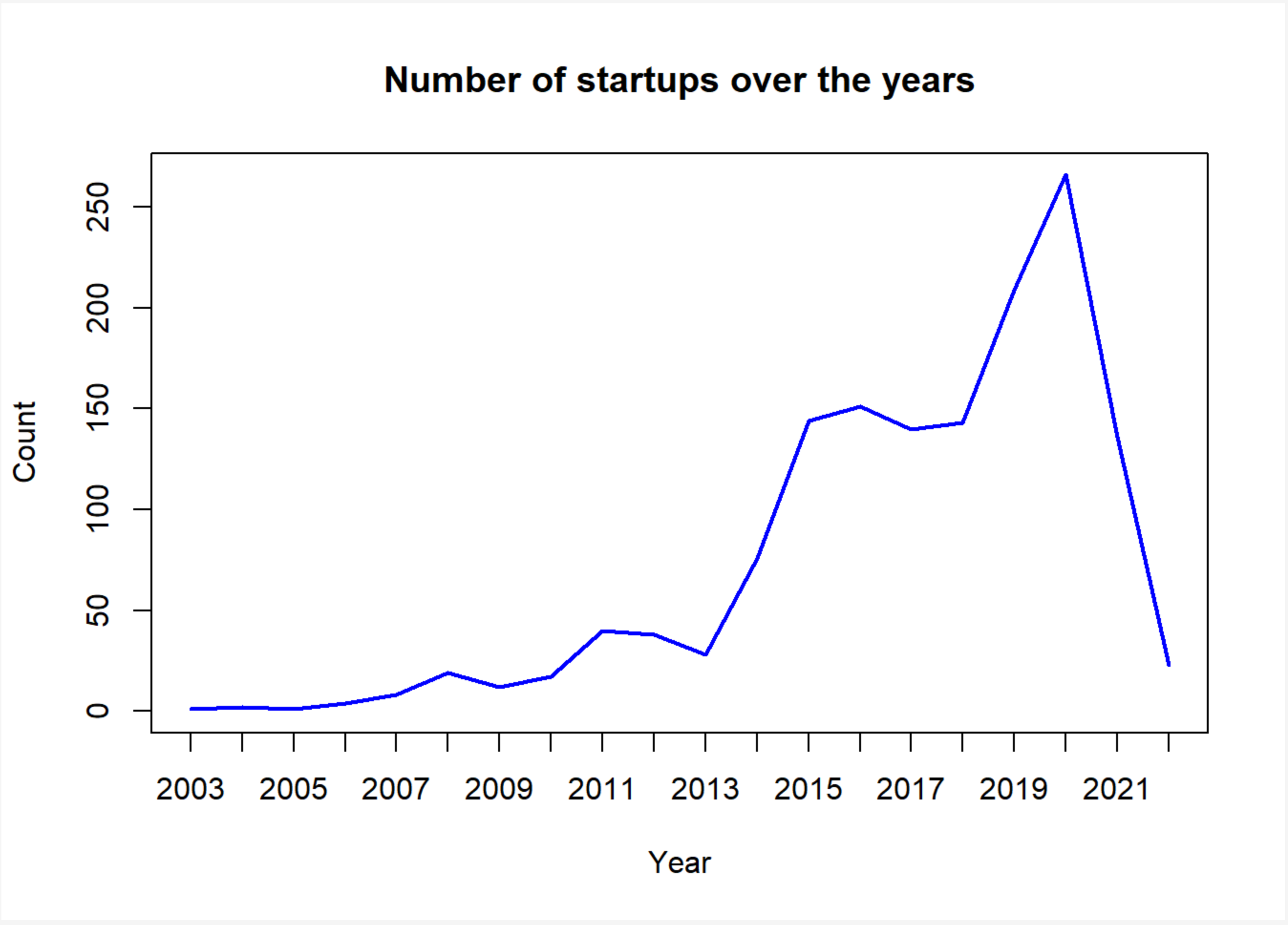
```
barplot.default(freq1,main = "How many startups in New Delhi are product or service based",xlab = "product or service based",ylab ="Frequency", col = rainbow(length(freq1))) # Plotting the bar chart for the frequency of startups based on product or service for New Delhi
```



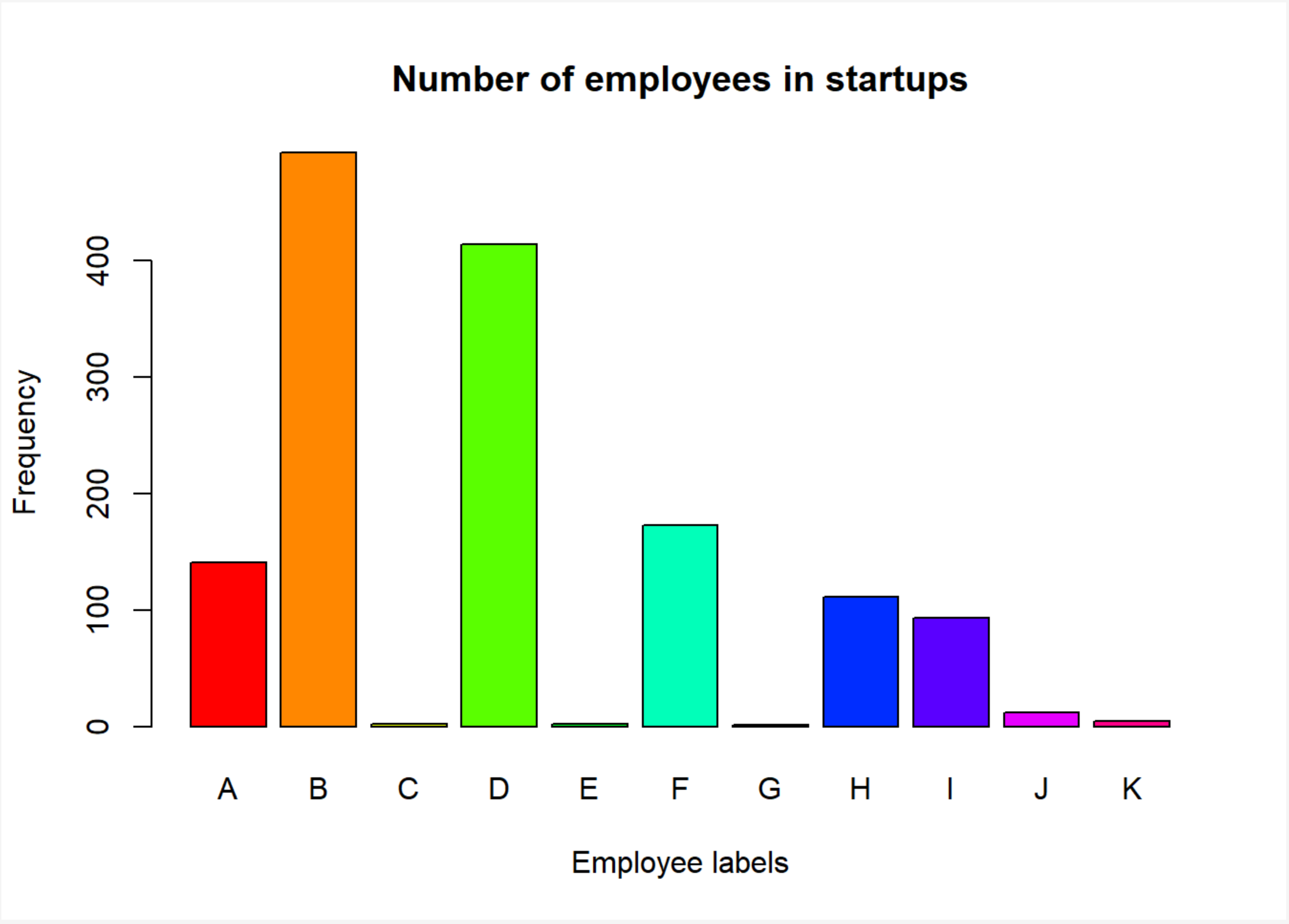
```
# Task 2:
# To visually represent the trend of number of startups founded over the years. The plot shows the trend of the number of startups founded over the past 20 years, with the blue line representing the trend.
# The input is a table of frequencies (numeric) of unique values in Founded column (numeric).

# Line chart showing the trend of number of startups over the years
freq2 = table(df$Founded) # count number of startups founded in each year
freq2
```

```
## 
## 1924 1963 1978 1982 1984 1989 1991 1993 1994 1996 1998 1999 2000 2002 2003 2004 
##    1    1    1    1    1    2    2    1    4    2    1    3    5    3    1    2 
## 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020 
##    1    4    8   19   12   17   40   38   28   76  144  151  140  143  209  266 
## 2021 2022 
##  136   23
```

```
plot(tail(freq2, 20), type = "l", xlab = "Year", ylab = "Count", main = "Number of startups over the years",
col='blue') # plot the trend of number of startups over the years
```



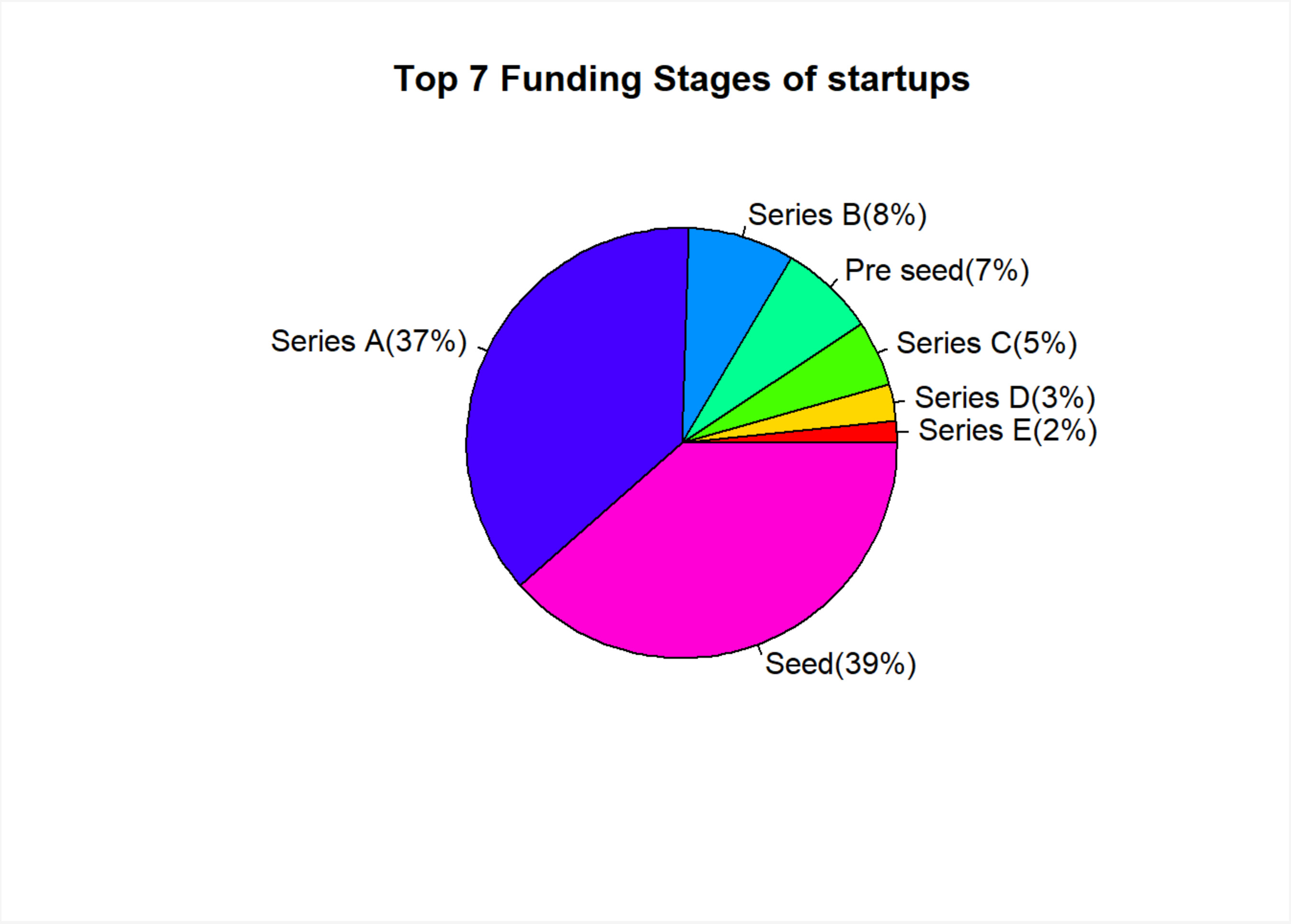
```
# Task 3:
# To visually represent the frequency of different employee count labels in the startups dataset. The result
expected will give us an understanding of the strength among startups in recent years.
# The input is a table of frequencies (numeric) of assigned labels in Employee Count column.

# Employee rate visualization
freq3 = table(df$Employee.Count) # count the frequency of different employee count labels
barplot.default(freq3, main = "Number of employees in startups", xlab = "Employee labels", ylab =
"Frequency", col = rainbow(length(freq3))) # plot a bar chart showing the frequency of different employee
count labels
```



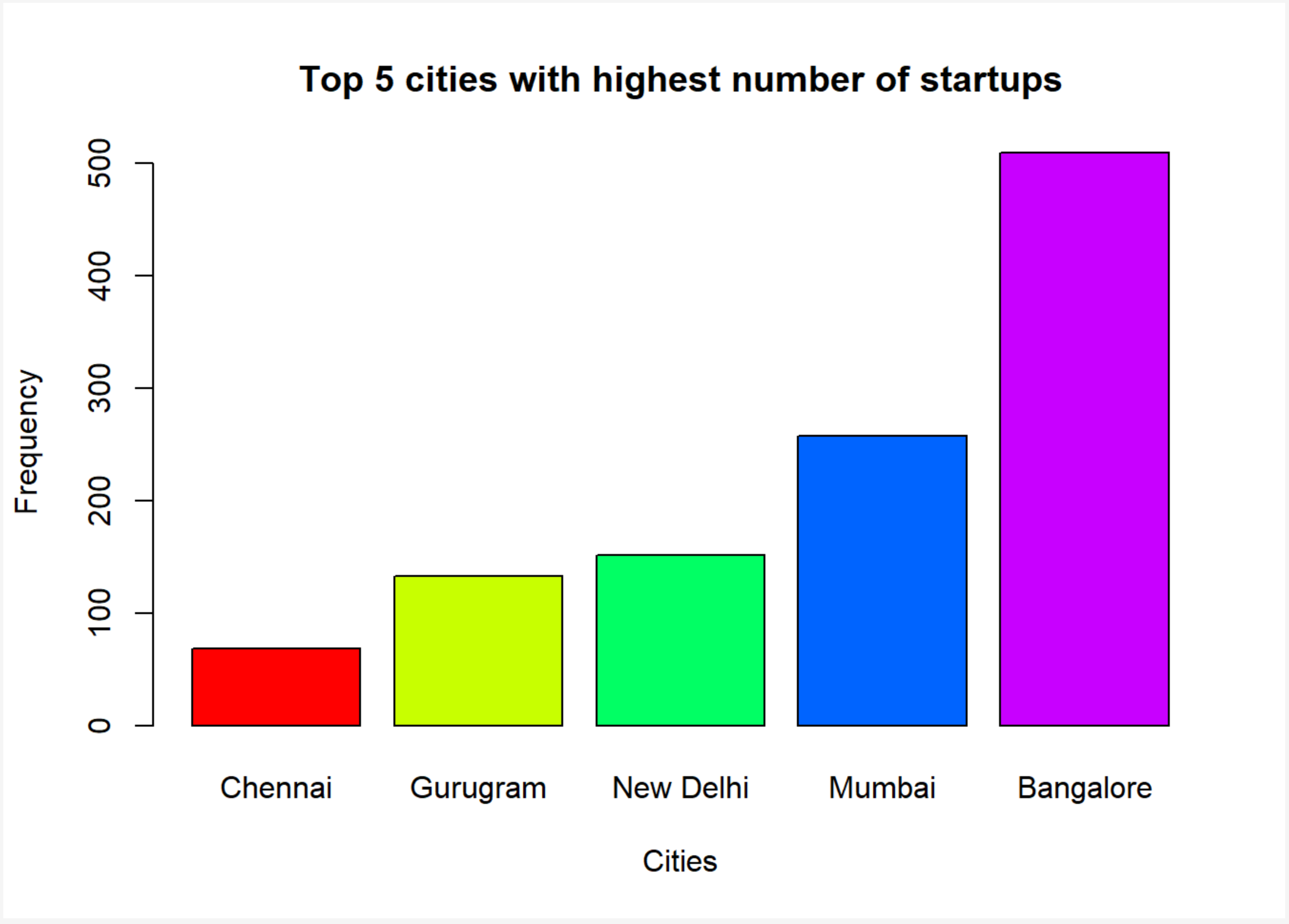
```
# Task 4:
# To visually represent the different funding stages among startups in the dataset. The result expected will
give us a clear understanding of the funding stages in startups these days.
# The input is a table of frequencies (numeric) of unique values in Stage column (character).

# Pie chart showing the different funding stages among startups
freq4 = table(df$Stage) # count the frequency of different funding stages
freq4
```

```
##
##            Angel          Bridge  Corporate Round
##                2               2               1
##             Debt        Pre seed            Seed
##               13              70             376
##         Series A        Series B        Series C
##              361              78              48
##         Series D        Series E        Series F
##                   27                  16                  10
##            Series G            Series H Working Capital to SMEs
##                    3                   4                   1
```

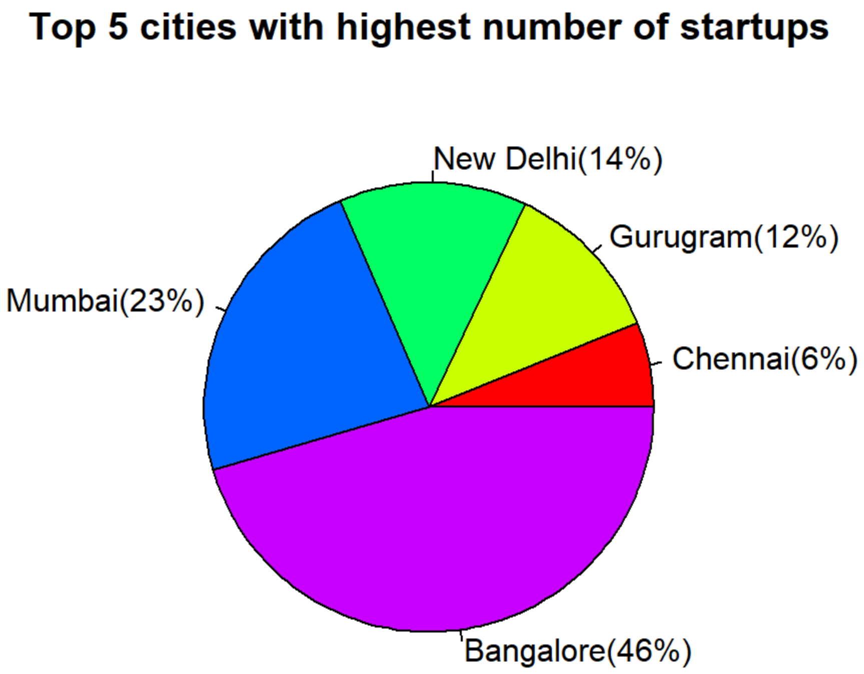
```
freq4 = sort(freq4)
freq4 = tail(freq4, 7) # consider the top 7 funding stages
percentages <- round(100 * prop.table(freq4))
labels <- paste(names(freq4), "(", percentages, "%)", sep = "")
pie(freq4, labels = labels, col = rainbow(length(freq4)), main = "Top 7 Funding Stages of startups") # plot a pie chart showing the top 7 funding stages
```



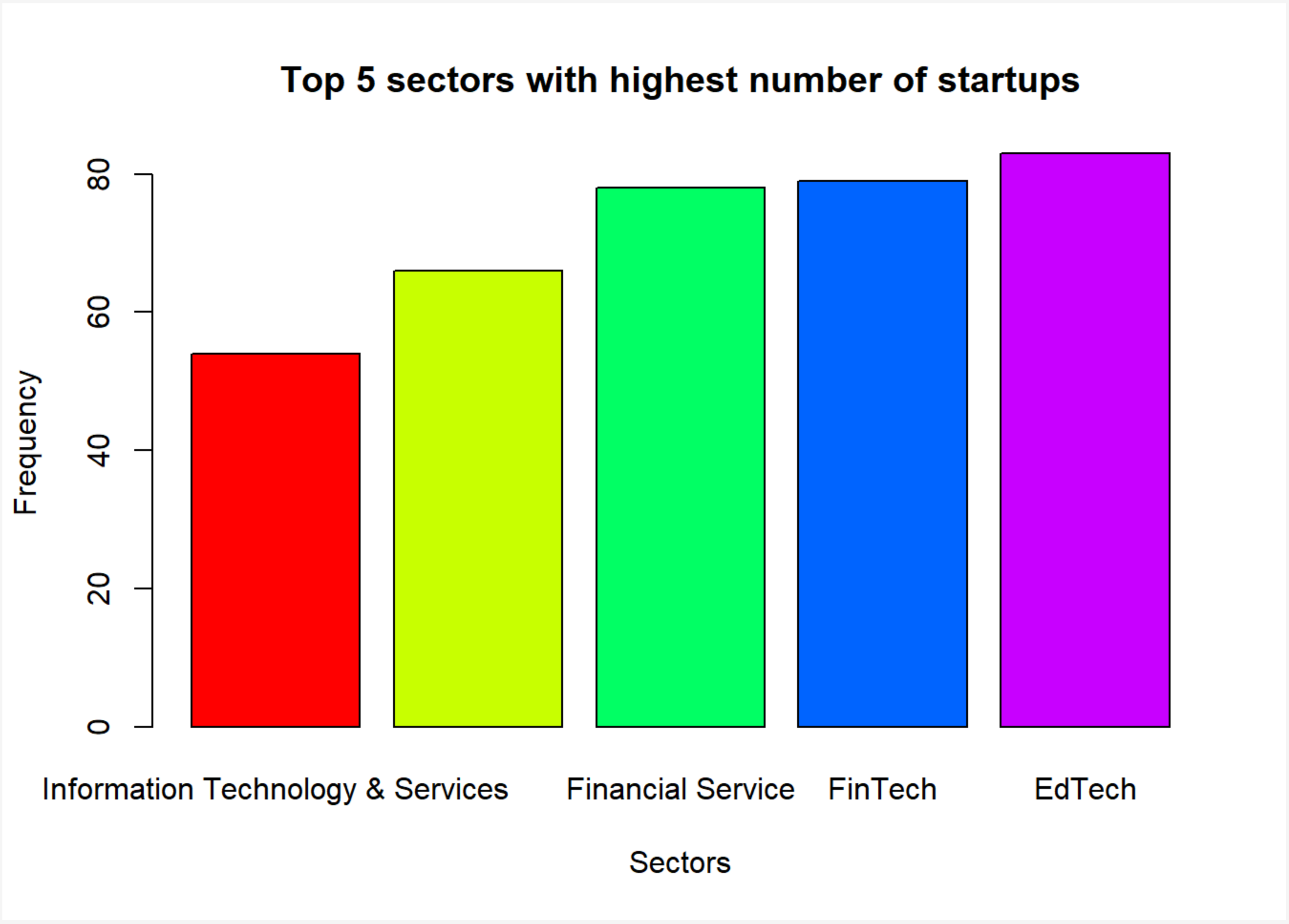
```
# Task 5:
# To analyze the top 5 cities with the highest number of startups in India. It provides valuable insights into the country's startup ecosystem. By visualizing the frequency and percentage of startups in top 5 cities using a bar chart and a pie chart, the analysis helps us understand which cities are leading the charge in terms of entrepreneurial activity and innovation.
# The input is a table of frequencies (numeric) of unique values in Cities column (character).
# Analysis of top 5 cities
freq = table(data$cities) # count the frequency of different cities where startups are located
freq = sort(freq)
freq = tail(freq, 5) # consider the top 5 cities with the highest number of startups
barplot(freq, main = "Top 5 cities with highest number of startups", xlab = "Cities", ylab = "Frequency", col =
rainbow(length(freq))) # plot a bar chart showing the top 5 cities with the highest number of startups
```



```
percentages <- round(100 * prop.table(freq))
labels <- paste(names(freq), "(", percentages, "%)", sep = "")
pie(freq, labels = labels, col = rainbow(length(freq)), main = "Top 5 cities with highest number of startups") #
plot a pie chart showing the top 5 cities with the highest number of startups
```

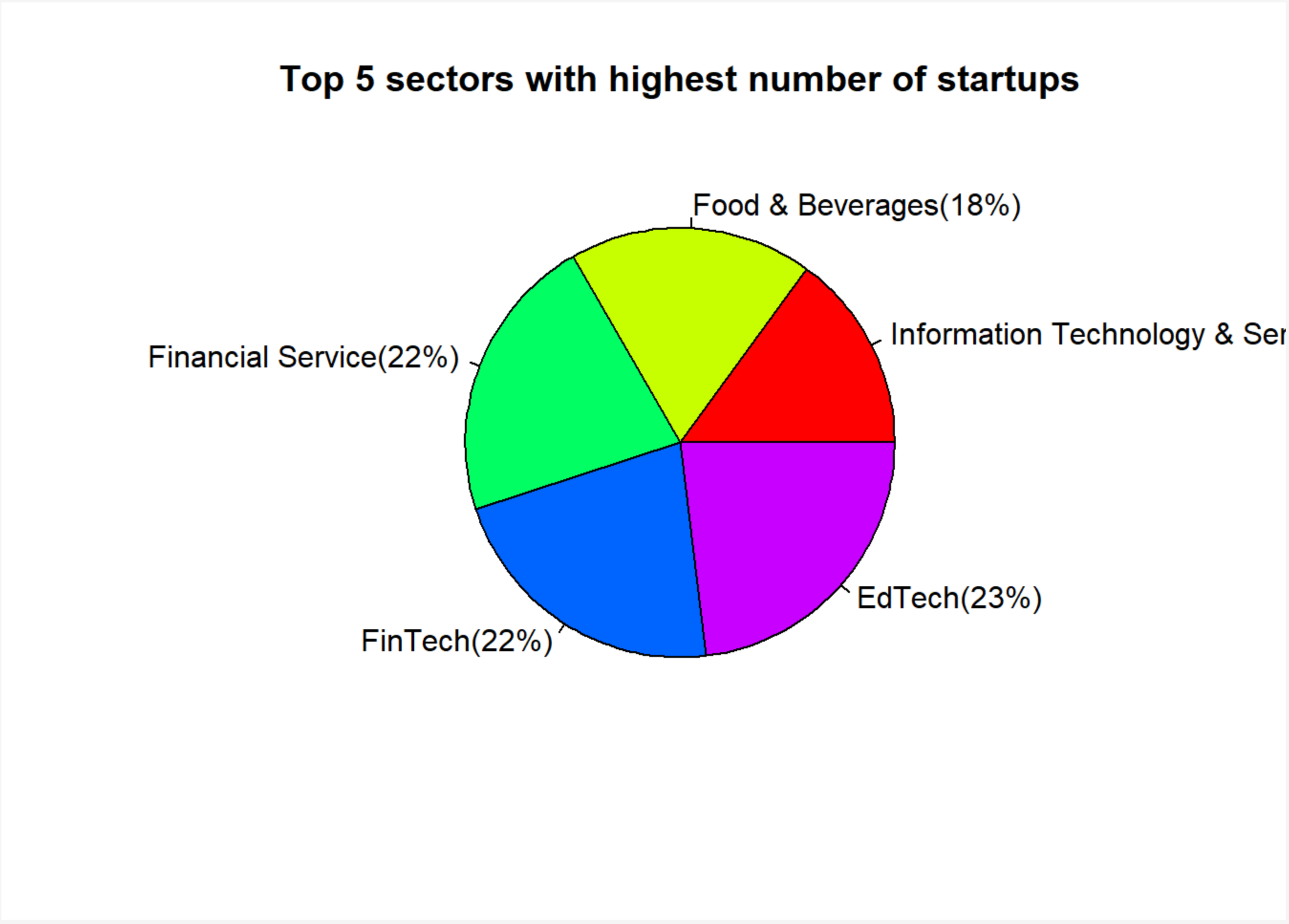


```
# Task 6:
# To analyze the top 5 sectors with the highest number of startups in India. It provides valuable insights into
the country's startup ecosystem. By visualizing the frequency and percentage of startups in top 5 sectors
using a bar chart and a pie chart, the analysis helps us understand which sectors are leading the charge in
terms of entrepreneurial activity and innovation.
# The input is a table of frequencies (numeric) of unique values in Sector column (character).

# Analysis of the top 5 sectors
freq2 = table(data$Sector) # count the frequency of different sectors
freq2 = sort(freq2)
freq2 = tail(freq2, 5) # consider the top 5 sectors with the highest number of startups
barplot(freq2, main = "Top 5 sectors with highest number of startups", xlab = "Sectors", ylab = "Frequency",
col = rainbow(length(freq))) # plot a bar chart showing the top 5 sectors with the highest number of startups
```

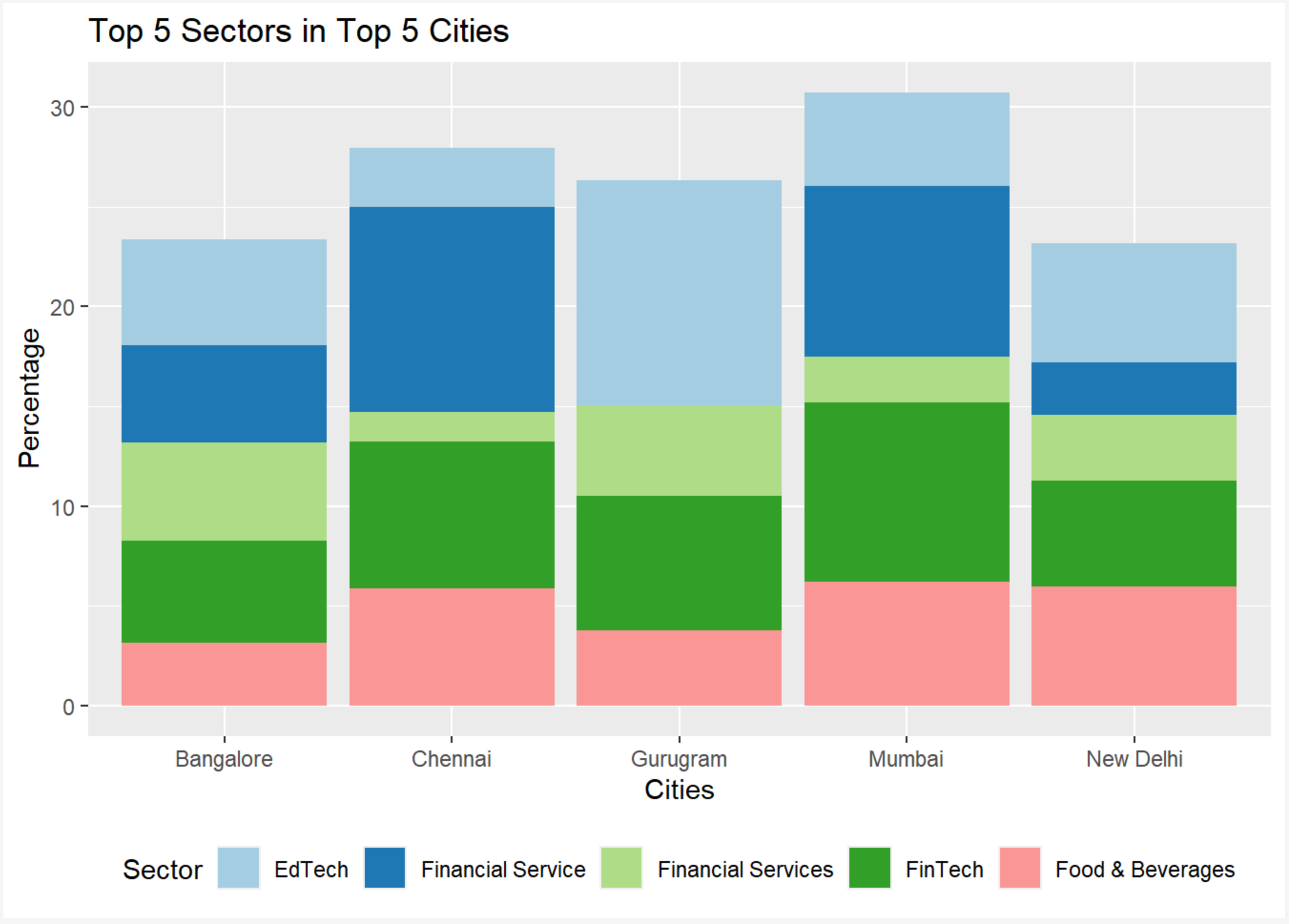


```
percentages <- round(100 * prop.table(freq2))
labels <- paste(names(freq2), "(", percentages, "%)", sep = "")
pie(freq2, labels = labels, col = rainbow(length(freq2)), main = "Top 5 sectors with highest number of
startups") # plot a pie chart showing the top 5 sectors with the highest number of startups
```

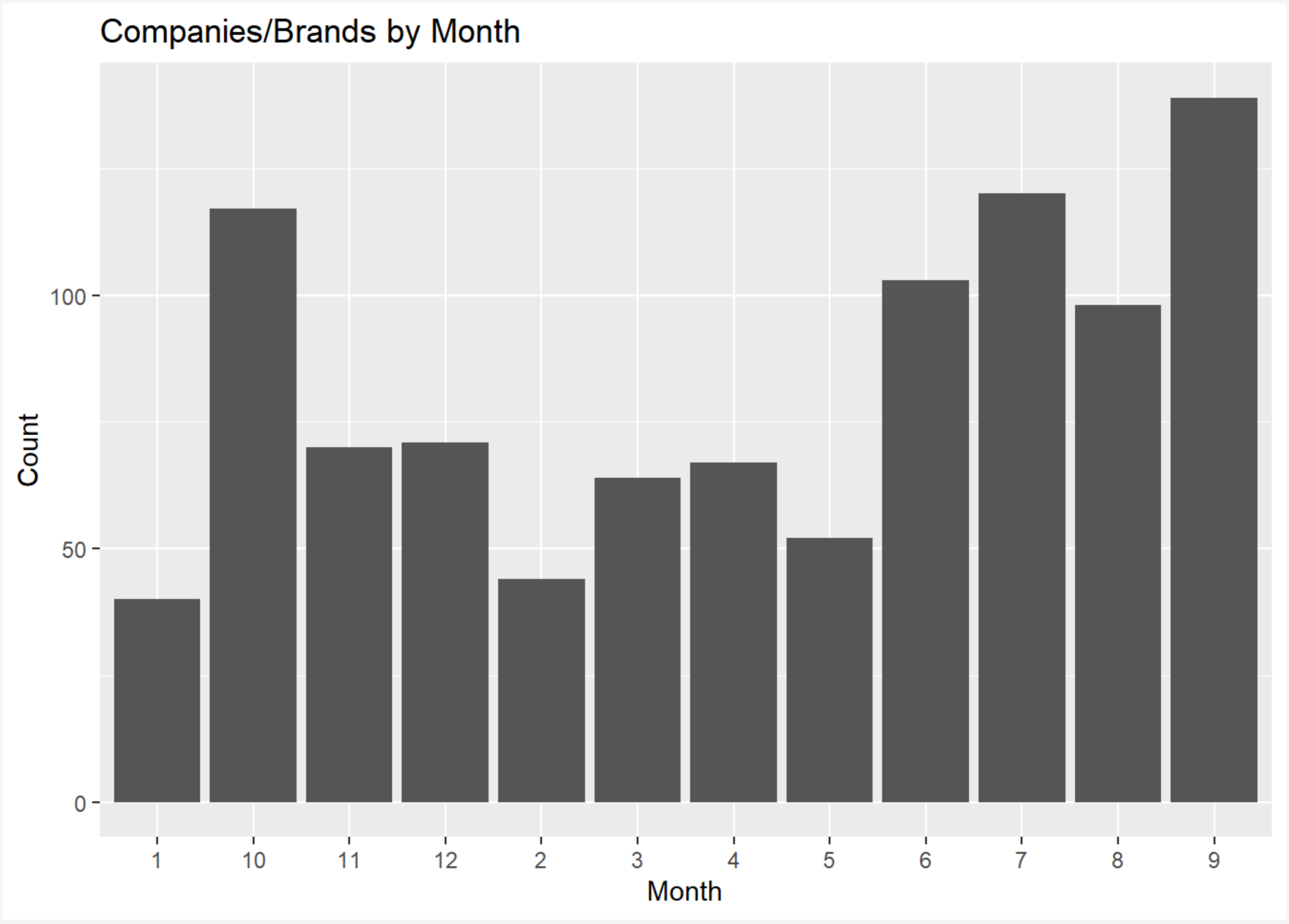
```
# Task 7:
# To visualize the distribution of top 5 startup sectors across the top 5 cities. It can help identify which of the
top sectors are more prominent in certain cities and how they compare to other cities.

# Stacked bar chart showing overall top 5 sectors in the top 5 cities
library(ggplot2)
```



```
## Get the top 5 Headquarters
top_headquarters <- head(sort(table(data$cities), decreasing = TRUE), 5) # count the frequency of different
cities and get the top 5

## Get the top 5 Sectors
top_sectors <- head(sort(table(data$Sector), decreasing = TRUE), 5) # count the frequency of different
sectors and get the top 5

## Create an empty data frame to store the results
result_df <- data.frame(Cities = character(),
                        Sector = character(),
                        Percentage = numeric())

## For each of the top 5 Headquarters and top 5 Sectors, calculate the percentage
for (hq in names(top_headquarters)) {
  for (sector in names(top_sectors)) {
    ## Calculate the percentage of startups in the current headquarters and sector combination
    percentage <- 100 * sum(data$cities == hq & data$Sector == sector) / top_headquarters[hq]
    ## Add the results to the data frame
    result_df <- rbind(result_df, data.frame(Cities = hq, Sector = sector, Percentage = percentage))
  }
}

## Create a stacked bar plot using ggplot2
ggplot(result_df, aes(x = Cities, y = Percentage, fill = Sector)) +
  geom_bar(stat = "identity") +
  labs(title = "Top 5 Sectors in Top 5 Cities", x = "Cities", y = "Percentage") +
  scale_fill_brewer(palette = "Paired") +
  theme(legend.position = "bottom")
```



```
# Task 8:
# To visualize the funding months across startups. Provides insights into the seasonality of funding for startups.
# The input is a table of frequency (numeric) of Month column (numeric).
# Aggregate the counts by month
counts <- table(data$Month)
# Convert the counts to a data frame
df <- data.frame(Month = names(counts), Count = as.numeric(counts))
# Plot the data in a bar chart
ggplot(df, aes(x = Month, y = Count)) +
  geom_bar(stat = "identity") +
  xlab("Month") +
  ylab("Count") +
  ggtitle("Companies/Brands by Month")
```



```
# Task 9:
# To provide insights into the revenue distribution for the top sectors in the dataset with the help of a
piechart.
# The inputs are the top sectors (character) with respect to their average of Amount(in dollars) column
(numeric).
```

```
data4=data
names(data4)[10]="Amount"
library(dplyr)
sector_amount_summary <- data4 %>%
  group_by(Sector) %>%
  summarize(total_amount = sum(Amount))
sector_amount_summary
```

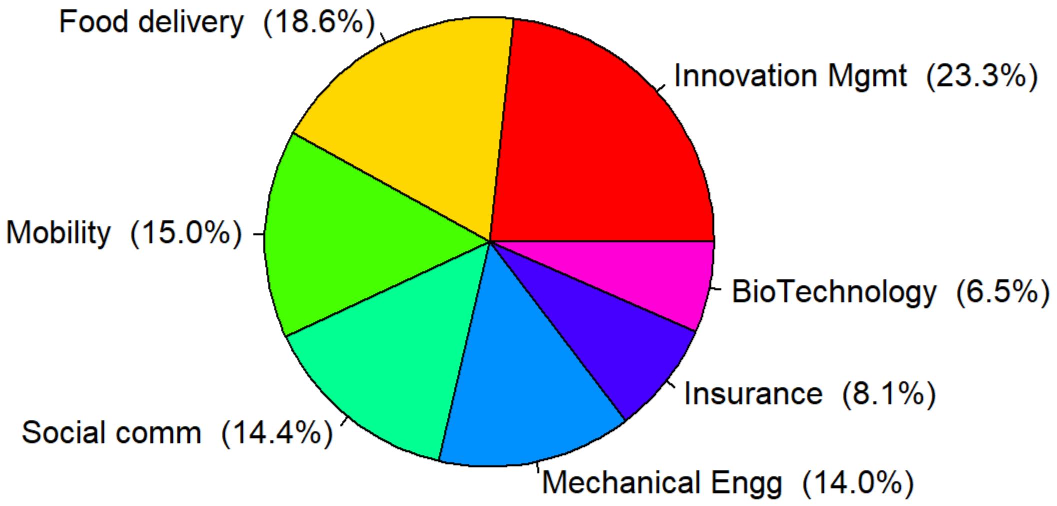
```
## # A tibble: 319 × 2
##    Sector                   total_amount
##    <chr>                           <dbl>
##  1 3D AI company                      NA
##  2 AI Chatbot                    7500000
##  3 AI company                         NA
##  4 AI startup                         NA
##  5 AR startup                     300000
##  6 Advertisement                 1800000
##  7 Aeorspace                     1800000
##  8 Aerospace, Manufacturing           NA
##  9 AgriTech                           NA
## 10 Agriculture                        NA
## # ℹ 309 more rows
```

```
sector_amount_summary <- sector_amount_summary %>%
  arrange(desc(total_amount))
sector_amount_summary
```

```
## # A tibble: 319 × 2
##    Sector                               total_amount
##    <chr>                                       <dbl>
##  1 Innovation Management                  1000000000
##  2 Food delivery                           800000000
##  3 Mobility                                643840000
##  4 Social commerce                         617900000
##  5 Mechanical Or Industrial Engineering    600400000
##  6 Insurance                               347925000
##  7 BioTechnology                           280600000
##  8 Social media                            266000000
##  9 Home services                           250000000
## 10 Venture Capital                         225000000
## # i 309 more rows
```

```
top7_sectors <- sector_amount_summary[1:7,]
```

```
top7_sectors <- top7_sectors %>%
  mutate(Sector = ifelse(Sector == "Innovation Management", "Innovation Mgmt",
                         ifelse(Sector == "Mechanical Or Industrial Engineering", "Mechanical Engg",
                                ifelse(Sector == "Social commerce", "Social comm", Sector))))
# Create a pie chart with percentage labels
pct <- round(100 * top7_sectors$total_amount / sum(top7_sectors$total_amount), 1)
labels <- paste(top7_sectors$Sector, sprintf(" (%.1f%%)", pct))
pie(top7_sectors$total_amount, labels = labels, col = rainbow(length(top7_sectors$Sector)))
```



```
# Task 10:
# To provide insights into the revenue distribution for the top cities in the dataset with the help of a piechart.
# The inputs are the top cities (character) with respect to their average of Amount(in dollars) column
(numeric).
```

```
data5=data
names(data5)[10]="Amount"
library(dplyr)
cities_amount_summary <- data5 %>%
  group_by(cities) %>%
  summarize(total_amount = sum(Amount))
cities_amount_summary
```

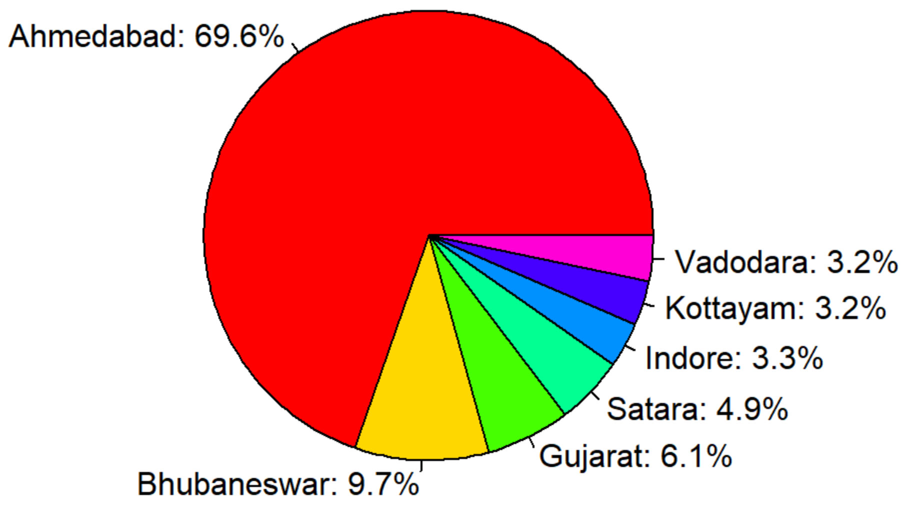
```
## # A tibble: 72 × 2
##    cities       total_amount
##    <chr>               <dbl>
##  1 Ahemdabad              NA
##  2 Ahmedabad       215955000
##  3 Ambernath          200000
##  4 Andheri           1000000
##  5 Bangalore              NA
##  6 Bhilwara          8000000
##  7 Bhopal                 NA
##  8 Bhubaneswar      30000000
##  9 Bhubhneshwar           NA
## 10 Chandigarh             NA
## # ℹ 62 more rows
```

```
cities_amount_summary <- cities_amount_summary %>%
  arrange(desc(total_amount))
```

```
cities_amount_summary
```

```
## # A tibble: 72 × 2
##    cities      total_amount
##    <chr>              <dbl>
##  1 Ahmedabad      215955000
##  2 Bhubaneswar     30000000
##  3 Gujarat         18800000
##  4 Satara          15140000
##  5 Indore          10200000
##  6 Kottayam        10000000
##  7 Vadodara        10000000
##  8 Bhilwara         8000000
##  9 Orissa           5000000
## 10 Ghaziabad        4150000
## # i 62 more rows
```

```
top7_cities <- cities_amount_summary[1:7,]
# Calculate the percentages
percent <- round(top7_cities$total_amount / sum(top7_cities$total_amount) * 100, 1)

# Create the pie chart
pie(top7_cities$total_amount, labels = paste0(top7_cities$cities, ": ", percent, "%"), col =
rainbow(length(top7_cities$total_amount)))
```

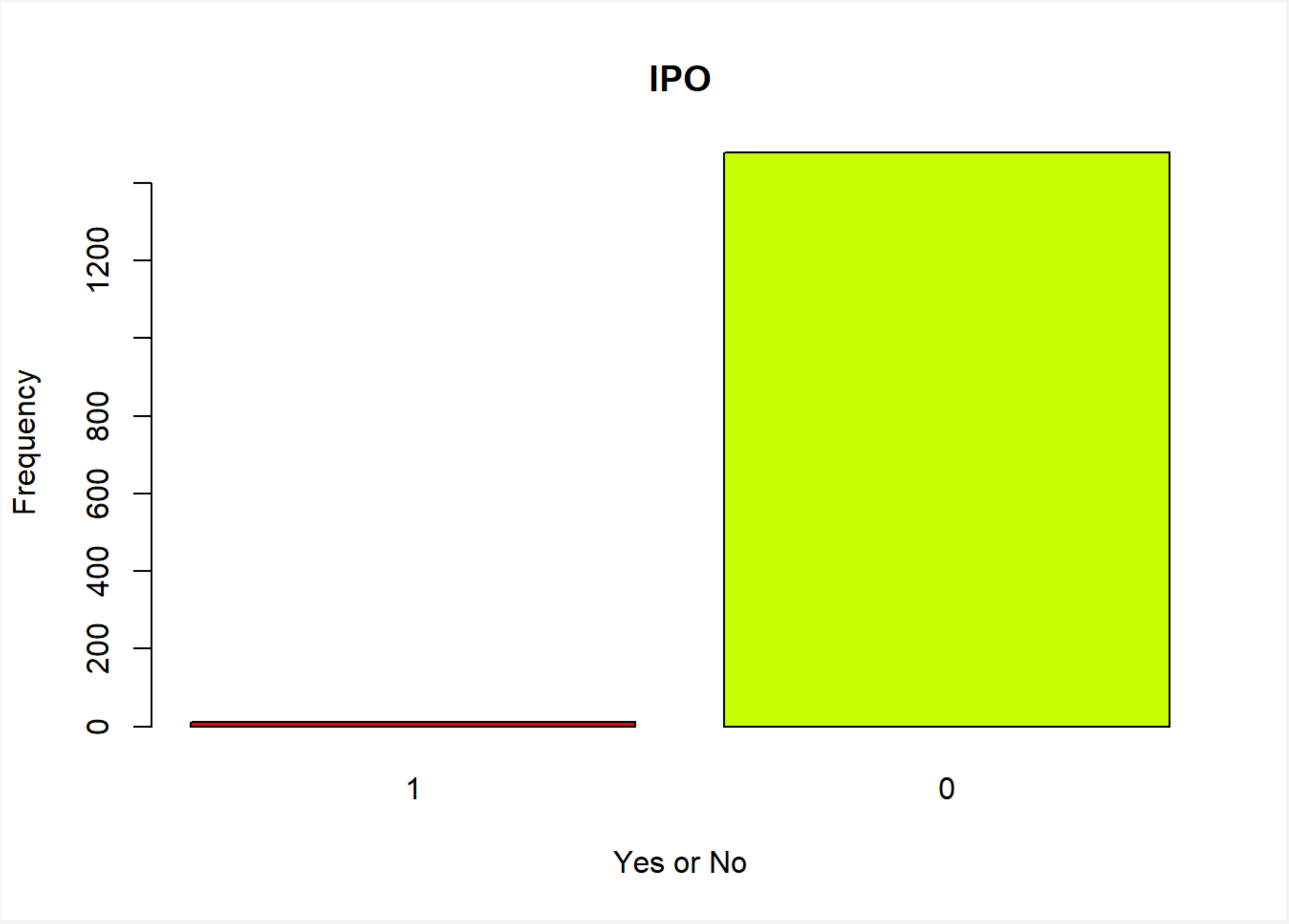


```
# Task 11:
# To provide insights into the distribution of startups that have had IPOs versus those that have not. The
frequency table shows the number of startups with IPOs and without IPOs.
# The input is a table of frequencies (numeric) of unique values in IPO column (numeric).

# Create a frequency table for the IPO column
```

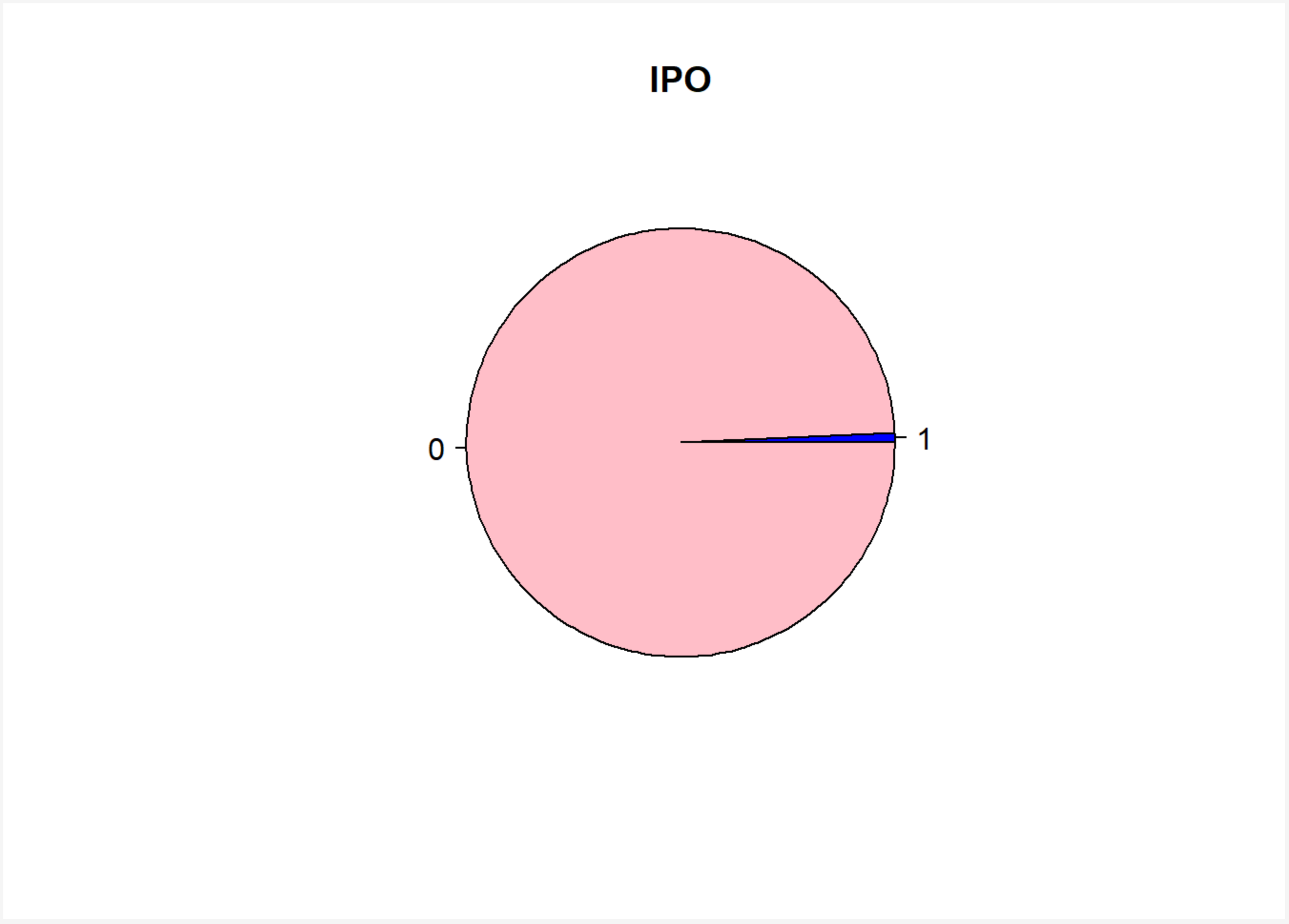
```
freq1=table(data$IPO)
# Sort the table in ascending order
freq1=sort(freq1)
# Display the frequency table
freq1
```

```
##
##    1    0
##   11 1479
```

```
# Create a bar plot of the top 5 frequencies, with specified properties
barplot(tail(freq1,5),
        main = "IPO",
        xlab = "Yes or No",
        ylab = "Frequency",
        col = rainbow(length(freq)))
```

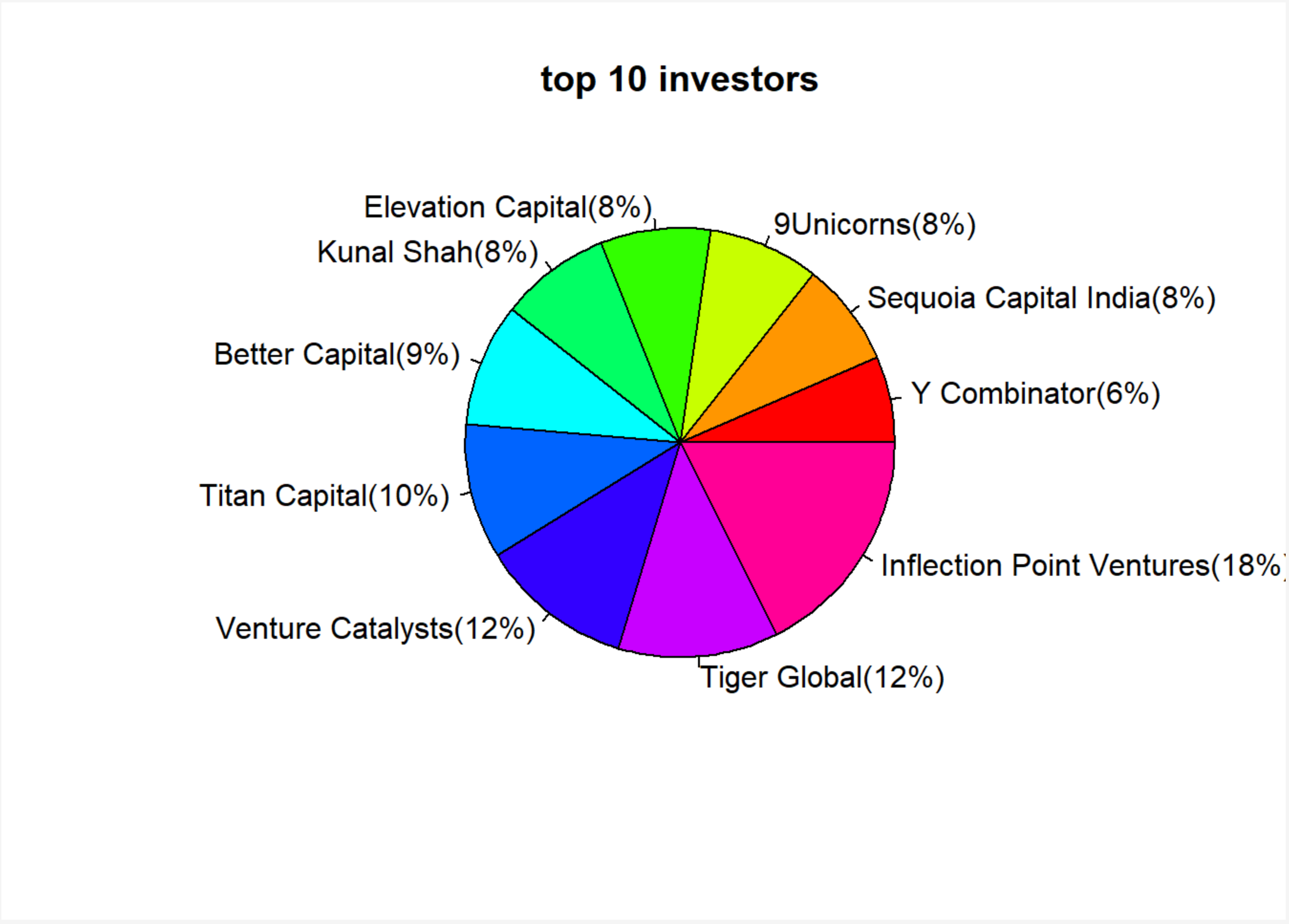


```
# Create a pie chart of the top 5 frequencies, with specified properties
pie(tail(freq1,5),
   main = "IPO",
   col =c('blue','pink'))
```



```
# Task 12:
# To identify the top 10 investors and visualize a pie chart to show their relative frequencies.
# The inputs are the investors (character) and their respective Amount(in dollars) (numeric) values.

data1=data
# Convert the Investment column to numeric
colnames(data1)[colnames(data1) == "Amount(in dollars)"] = "Investment"
data1$Investment <- as.numeric(data1$Investment)

# Create a new data frame to store the individual investments
individual_investments <- data.frame(Investor = character(),
                                     Investment = numeric())
```

```
# Loop through each row in the original data frame
for (i in 1:nrow(data1)) {
  # Get the list of investors and split them by comma
  investors <- strsplit(data1$Investors[i], ",")[[1]]
  # Calculate the investment per investor
  investment_per_investor <- data1$Investment[i]
  # Loop through each investor and add a new row to the individual_investments data frame
  for (j in 1:length(investors)) {
    individual_investments <- rbind(individual_investments,
                                    data.frame(Investor = investors[j],
                                               Investment = investment_per_investor))
  }
}

# Write the individual investments data frame to a new CSV file
library(openxlsx)
# Write the data frame to an Excel file
write.xlsx(individual_investments, "individual_investments.xlsx", rowNames = FALSE)
d=read_excel("D:\\individual_investments.xlsx")
library(dplyr)
d$Investor <- na_if(d$Investor, '')
freq4=table(d$Investor)
freq4=sort(freq4)
freq4=tail(freq4,10)
percentages <- round(100 * prop.table(freq4))
labels <- paste(names(freq4), "(", percentages, "%)", sep = "")
pie(freq4, labels = labels, main = "top 10 investors",col =rainbow(length(freq4)))
```



```
# Task 13:
# To  perform a classification task using a Naive Bayes model to predict the Type column of the dataset
based on the Sector column. A confusion matrix is created and the correlation coefficient is calculated.

#Load necessary packages
library(e1071)
library(caTools)
library(rcompanion)
library(caret)

## Loading required package: lattice

# Subset the data to only include columns 6 and 13, then create a new data frame
df4=data[,c(6,13)]
df4=data.frame(df4)

# Rename the second column to 'type'
```

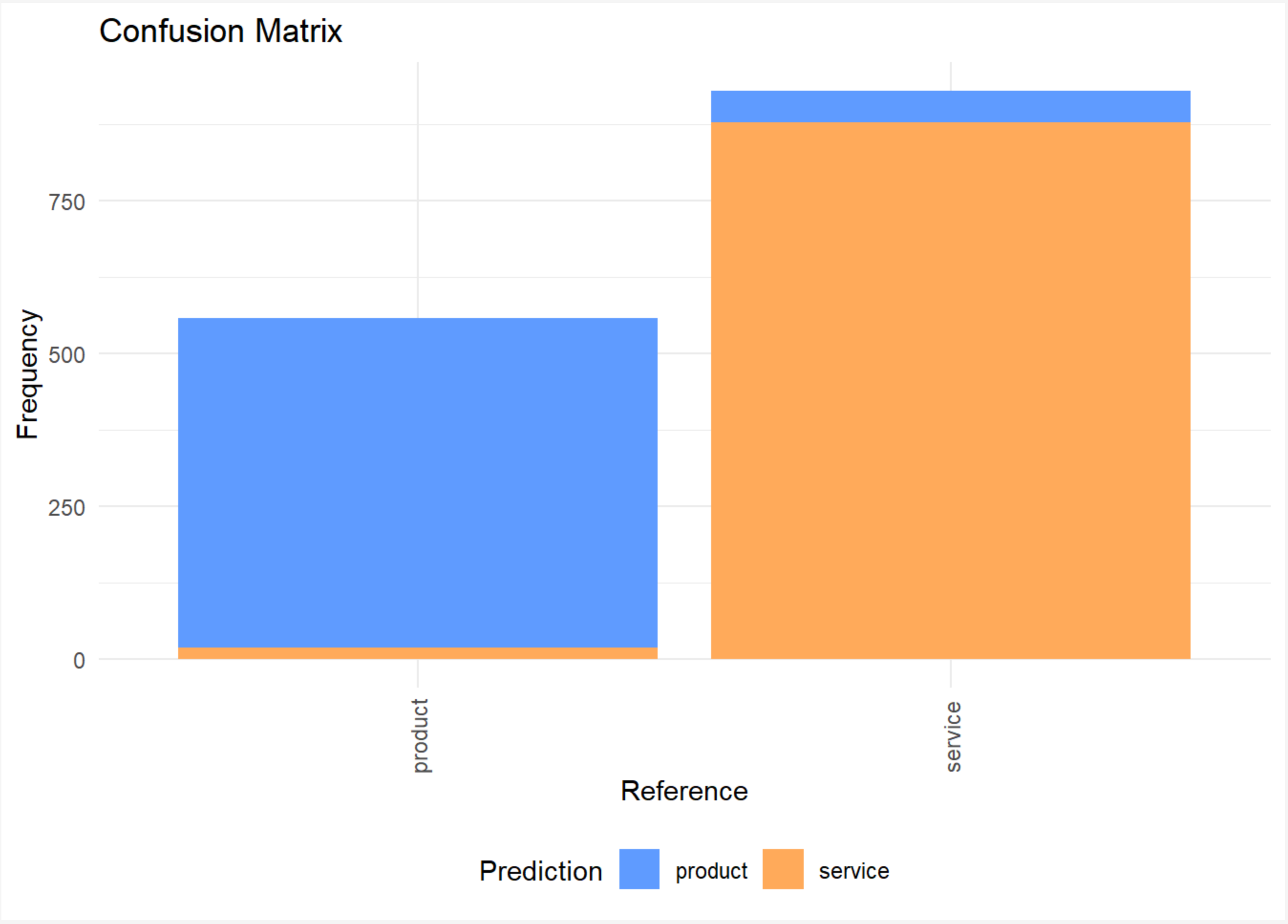
```
colnames(df4)[2]="type"

# Convert the 'Sector' and 'type' columns to factors
df4$Sector=factor(df4$Sector)
df4$type=factor(df4$type)

# Set a random seed for reproducibility, then split the data into a training and test set
set.seed(123)
trainIndex <- createDataPartition(df4, p = 0.7, list = FALSE)

## Warning in createDataPartition(df4, p = 0.7, list = FALSE): Some classes have
## no records ( ) and these will be ignored

## Warning in createDataPartition(df4, p = 0.7, list = FALSE): Some classes have a
## single record ( ) and these will be selected for the sample

train <- df4[trainIndex, ]
test <- df4[-trainIndex, ]

# Fit a Naive Bayes model to the training data
model <- naiveBayes(type~ Sector, data = df4)

# Make predictions on the test set using the Naive Bayes model
predictions <- predict(model, newdata = test)

# Calculate the accuracy of the classifier by comparing predicted values to actual values
accuracy <- mean(predictions == test$type)

# Print the accuracy of the classifier
accuracy

## [1] 0.9516129

# Create a confusion matrix to evaluate the performance of the classifier
confusionMatrix(predictions, test$type)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction product service
##    product     539      19
##    service      53     877
##
##                Accuracy : 0.9516
##                  95% CI : (0.9394, 0.962)
##     No Information Rate : 0.6022
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.898
##
##  Mcnemar's Test P-Value : 0.0001006
##
##             Sensitivity : 0.9105
##             Specificity : 0.9788
##          Pos Pred Value : 0.9659
##          Neg Pred Value : 0.9430
##              Prevalence : 0.3978
##          Detection Rate : 0.3622
##    Detection Prevalence : 0.3750
##       Balanced Accuracy : 0.9446
##
##        'Positive' Class : product
##
```

```
# Calculate the correlation coefficient (Cramer's V) between 'Sector' and 'type' using the rcompanion package
cramerV(df4$Sector,df4$type)
```

```
## Cramer V
##   0.9267
```

```
conf_mat <- confusionMatrix(predictions, test$type)

# Create data frame from confusion matrix
conf_mat_df <- as.data.frame.matrix(conf_mat$table)

# Convert row names to a variable
conf_mat_df$Reference <- rownames(conf_mat_df)

# Reshape data from wide to long format
library(tidyr)
conf_mat_long <- gather(conf_mat_df, key = "Prediction", value = "Frequency", -Reference)
```

```
# Plot stacked bar chart
library(ggplot2)
ggplot(conf_mat_long, aes(x = Reference, y = Frequency, fill = Prediction)) +
  geom_bar(stat = "identity") +
  scale_fill_manual(values = c("#619CFF", "#FFAA5E")) +
  labs(title = "Confusion Matrix", x = "Reference", y = "Frequency", fill = "Prediction") +
  theme_minimal() +
  theme(legend.position = "bottom", axis.text.x = element_text(angle = 90, vjust = 0.5))
```



```
# Task 14:
# Random Forest Implementation. We predict the Type (whether its a product or service) with the help of
predictors- Sector. A confusion matrix is built and plotted to get a better understanding and the accuracy of
the model is printed.

library(dplyr)
library(randomForest)
```

```
## randomForest 4.7-1.1

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':
##
##     margin

## The following object is masked from 'package:dplyr':
##
##     combine
```

```
# Convert variables to numeric where applicable
# Remove non-numeric variables
datarf <- data %>% select(-`Company Name`,-IPO, -cities, -Description, -Founders, -Investors, -Stage,
-Month,-`Amount(in dollars)`,-Founded,-`Employee Count`)
datarf$Type <- as.factor(datarf$Type)
# Remove rows with missing values
datarf <- na.omit(datarf)

# Split the data into training and testing sets
set.seed(123)
trainIndex <- createDataPartition(datarf$Type, p = 0.7, list = FALSE)
trainData <- datarf[trainIndex, ]
testData <- datarf[-trainIndex, ]

# Train the random forest model
model <- randomForest(Type ~ ., data = trainData, ntree = 100, mtry = 2, type = "class")
```

```
## Warning in randomForest.default(m, y, ...): invalid mtry: reset to within valid
## range
```

```
print(model)
```

```
##
## Call:
##  randomForest(formula = Type ~ ., data = trainData, ntree = 100,      mtry = 2, type = "class")
##                Type of random forest: classification
##                      Number of trees: 100
## No. of variables tried at each split: 1
##
##         OOB estimate of  error rate: 12.16%
## Confusion matrix:
##         product service class.error
## product     337      79  0.18990385
## service      48     580  0.07643312
```

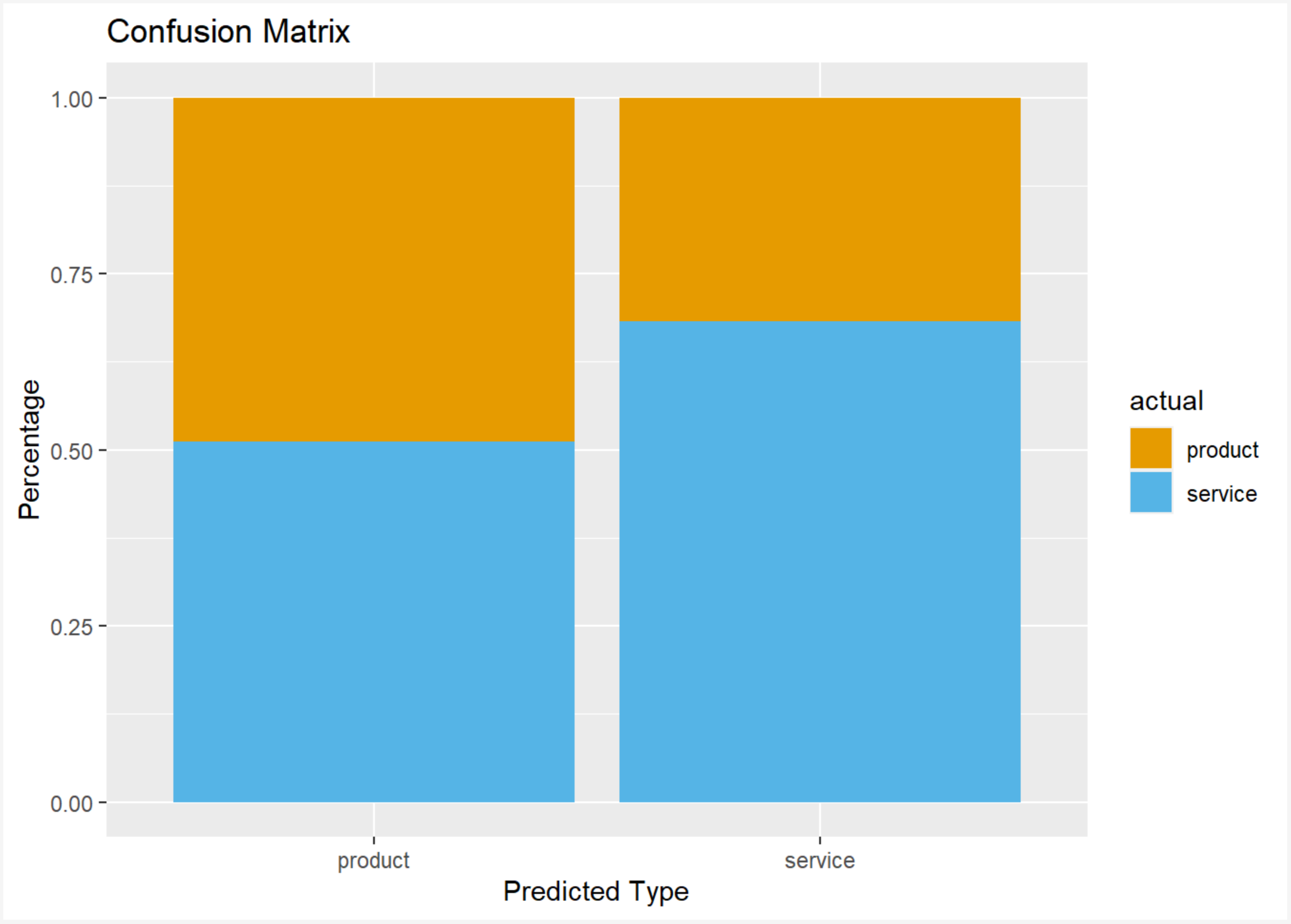
```
# Make predictions on the test data
predictions <- predict(model, testData)

# Compute the confusion matrix
conf_mat <- table(predictions, testData$Type)
conf_mat_pct <- prop.table(conf_mat, margin = 1)

# Print the confusion matrix
print(conf_mat_pct)
```

```
##
## predictions   product   service
##     product 0.4882629 0.5117371
##     service 0.3175966 0.6824034
```

```
accuracy <- sum(diag(conf_mat)) / sum(conf_mat)
accuracy
```

```
## [1] 0.5896861
```

```
library(ggplot2)
# Convert the confusion matrix to a data frame
conf_mat_df <- as.data.frame.matrix(conf_mat_pct)
conf_mat_df$predicted <- rownames(conf_mat_df)
conf_mat_df <- tidyr::gather(conf_mat_df, actual, value, -predicted)

# Plot the stacked bar chart
ggplot(conf_mat_df, aes(x = predicted, y = value, fill = actual)) +
  geom_col(position = "stack") +
  scale_fill_manual(values = c("#E69F00", "#56B4E9")) +
 labs(x = "Predicted Type", y = "Percentage", title = "Confusion Matrix")
```



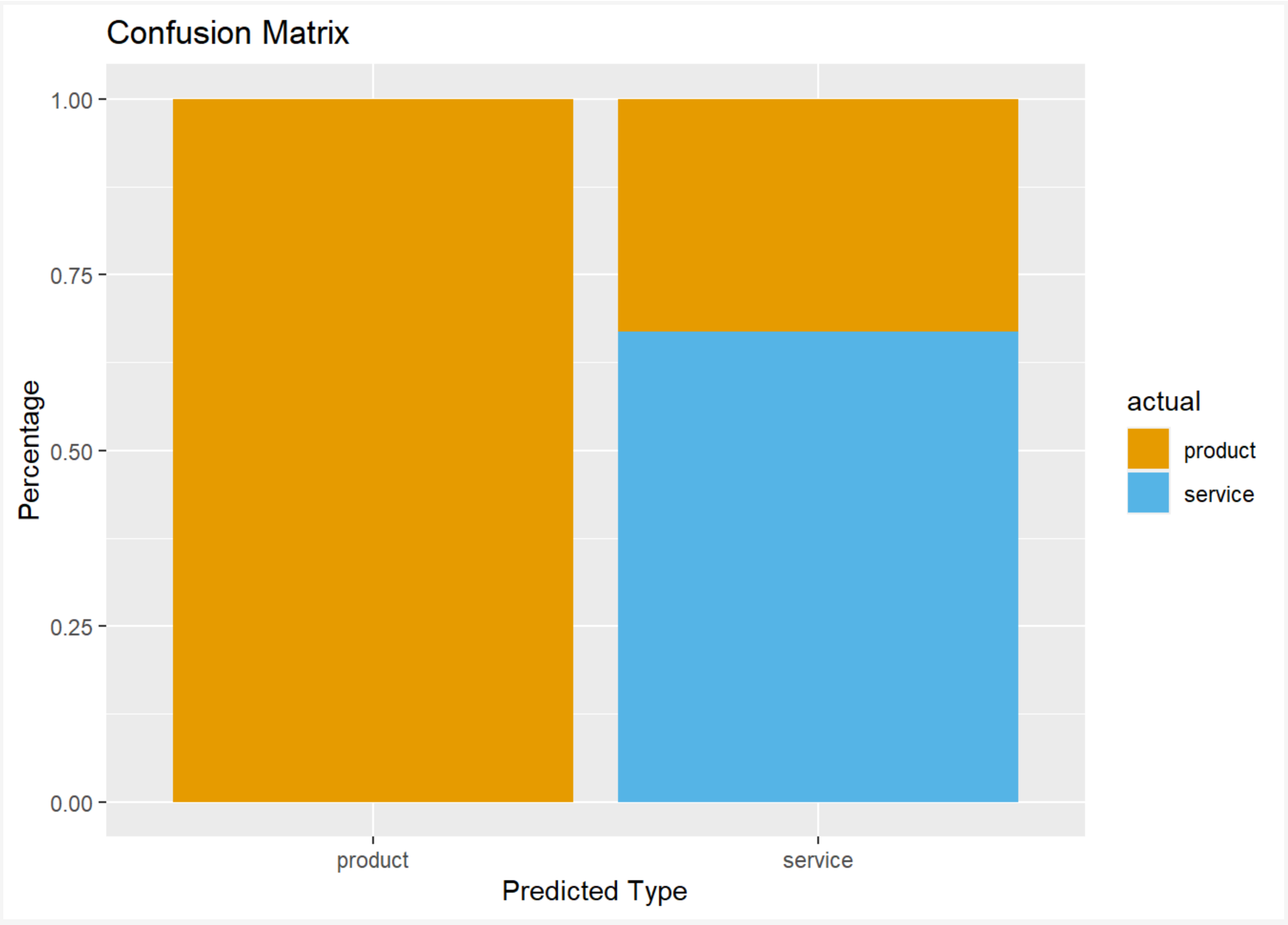
```
# Task 15:
# Decision Tree Implementation. We predict the Type (whether its a product or service) with the help of
predictors- Sector. The accuracy of the model and the confusion matrix is printed.

# Load the dataset
df <- data

# Preprocess the data
df$Sector <- as.factor(df$Sector)
df$Type <- as.factor(df$Type)

# Split the data into train and test sets
set.seed(123)
trainIndex <- createDataPartition(df$Type, p = .8, list = FALSE)
train <- df[trainIndex,]
```

```
test <- df[-trainIndex,]

# Fit a Naive Bayes model to the training data
model <- train(Type ~ Sector, data = train, method = "rpart")
print(model)

## CART
##
## 1193 samples
##    1 predictor
##    2 classes: 'product', 'service'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 1193, 1193, 1193, 1193, 1193, 1193, ...
## Resampling results across tuning parameters:
##
##   cp          Accuracy   Kappa
##   0.05252101  0.7129112  0.31720052
##   0.10924370  0.6834218  0.23603886
##   0.15546218  0.6327943  0.08885404
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.05252101.

# Make predictions on the test data
predictions <- predict(model, testData)

# Compute the confusion matrix
conf_mat <- table(predictions, testData$Type)
conf_mat_pct <- prop.table(conf_mat, margin = 1)

# Print the confusion matrix
print(conf_mat_pct)

##
## predictions   product   service
##     product 1.0000000 0.0000000
##     service 0.3316708 0.6683292

accuracy <- sum(diag(conf_mat)) / sum(conf_mat)
```

```
accuracy
```

```
## [1] 0.7017937
```

```
library(ggplot2)
# Convert the confusion matrix to a data frame
conf_mat_df <- as.data.frame.matrix(conf_mat_pct)
conf_mat_df$predicted <- rownames(conf_mat_df)
conf_mat_df <- tidyr::gather(conf_mat_df, actual, value, -predicted)

# Plot the stacked bar chart
ggplot(conf_mat_df, aes(x = predicted, y = value, fill = actual)) +
 geom_col(position = "stack") +
 scale_fill_manual(values = c("#E69F00", "#56B4E9")) +
 labs(x = "Predicted Type", y = "Percentage", title = "Confusion Matrix")
```


Confusion Matrix
Percentage
product
service
Predicted Type
actual
product
service