Data Visualization with R and Google Play Store Data

Open Lab | Digital Summit '18

Miracle Innovation Labs

Miracle Software Systems, Inc.

Data Visualization with R and Google Play Store Data

Introduction

This document contains a step-by-step process of analyzing Google Play Store Apps data with R and will teach you how to create graphs using R language.

This guide was prepared by **Miracle's Innovation Labs**.

Pre-Requisites

All attendees must have their workstation (with Internet) to participate in the lab (Both PC and MAC are compatible). The following pre-requisites will help you to make the Hands-on Lab experience easier.

- Download and install R and R studio

Technology Involved

- R Programming

Lab Steps

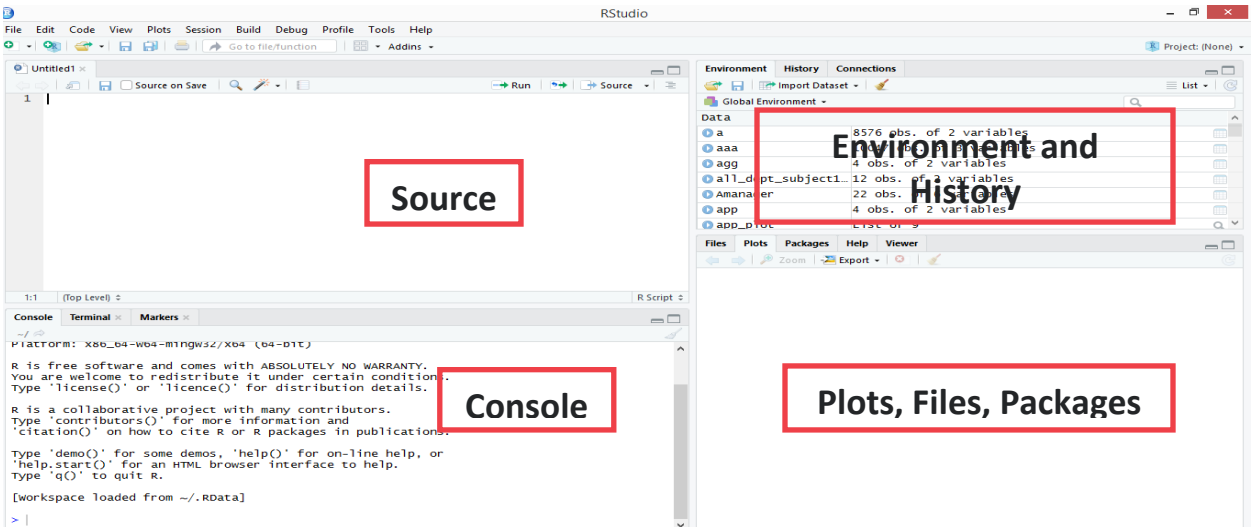
So, let us get started with the analysis!

The following steps will outline how you can Analyze Google Play Store Apps data using R Programming.

Glance on RStudio

RStudio is the IDE to run R scripts. There are four main components in IDE - Source, Console, Environment and History and Plots/Files/Packages.

Source - we write the main program in the source area.
Console - The output and errors are displayed in console.

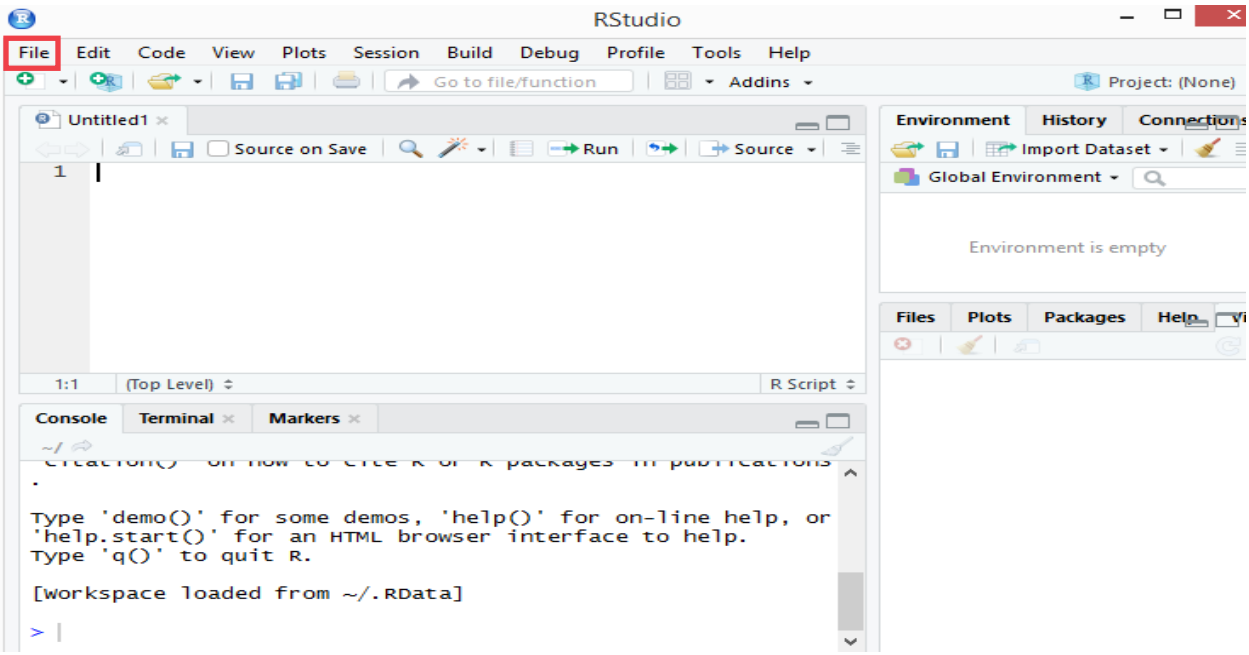


Environment and History - Data Frames and Lists are displayed in environment and provides interactive list of loaded R objects.

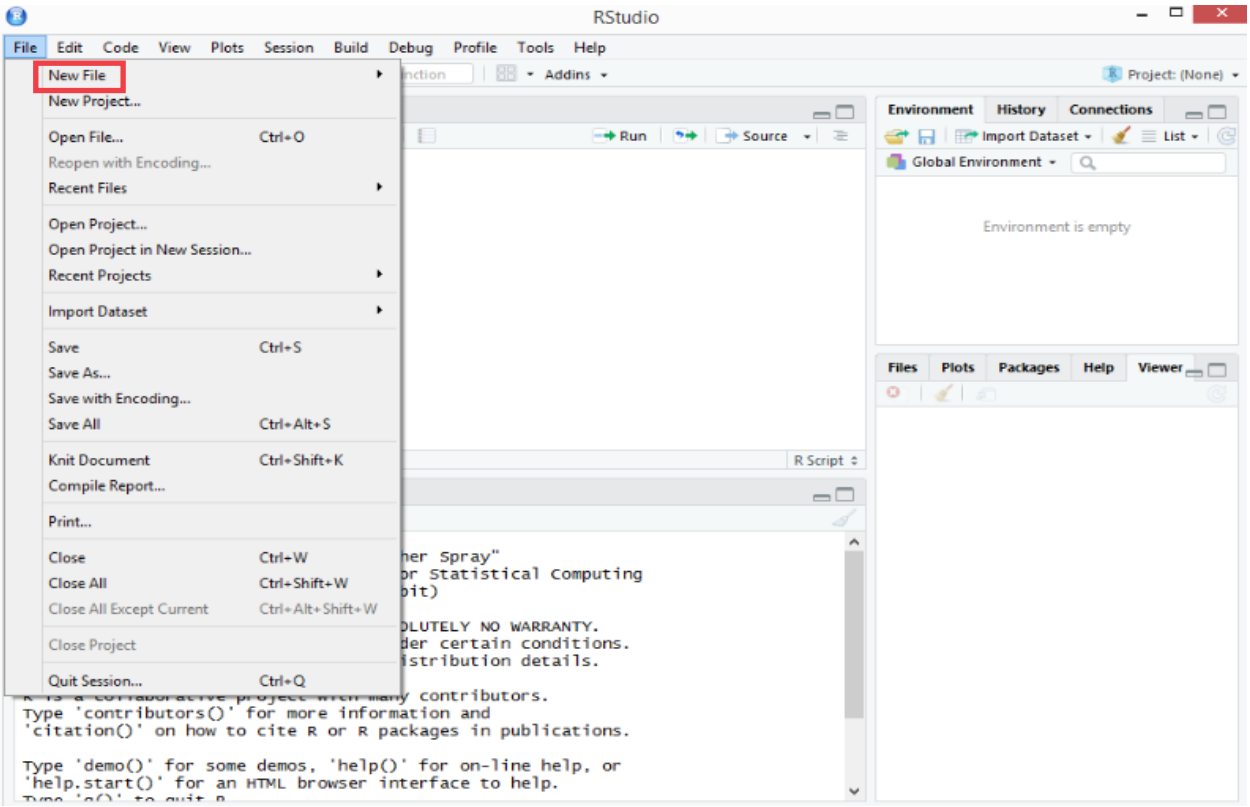
Plots, Files, Packages - Graphs are displayed in the plot section, we can see our installed packages under packages tab and also, we can search for any function with the help tab.

Step #1 | Create R Script

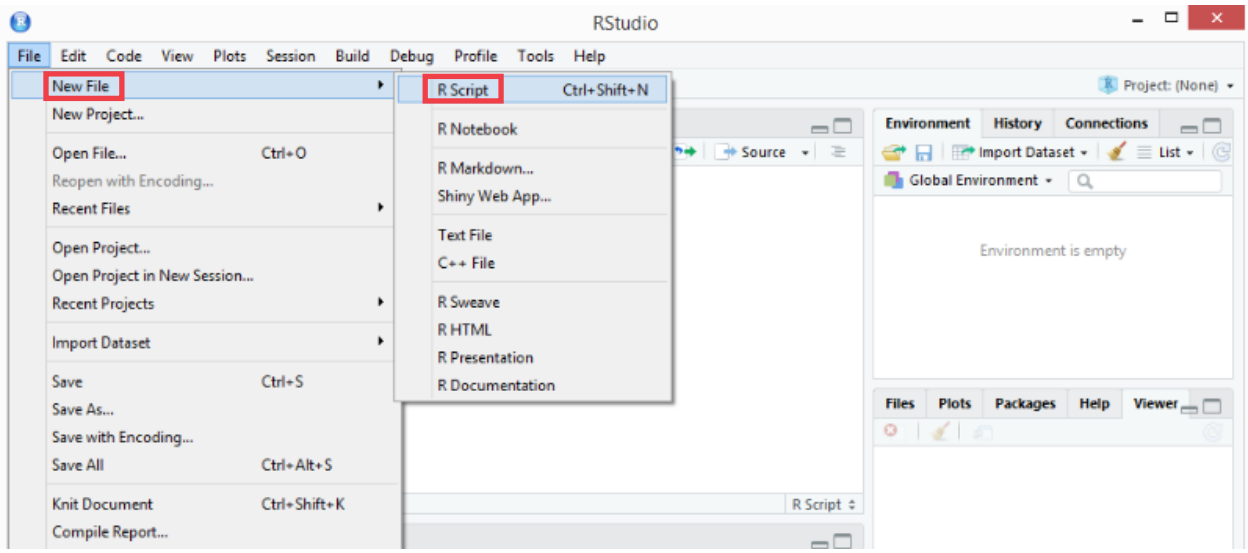
Go to **file** option which is at the top left corner of the RStudio.



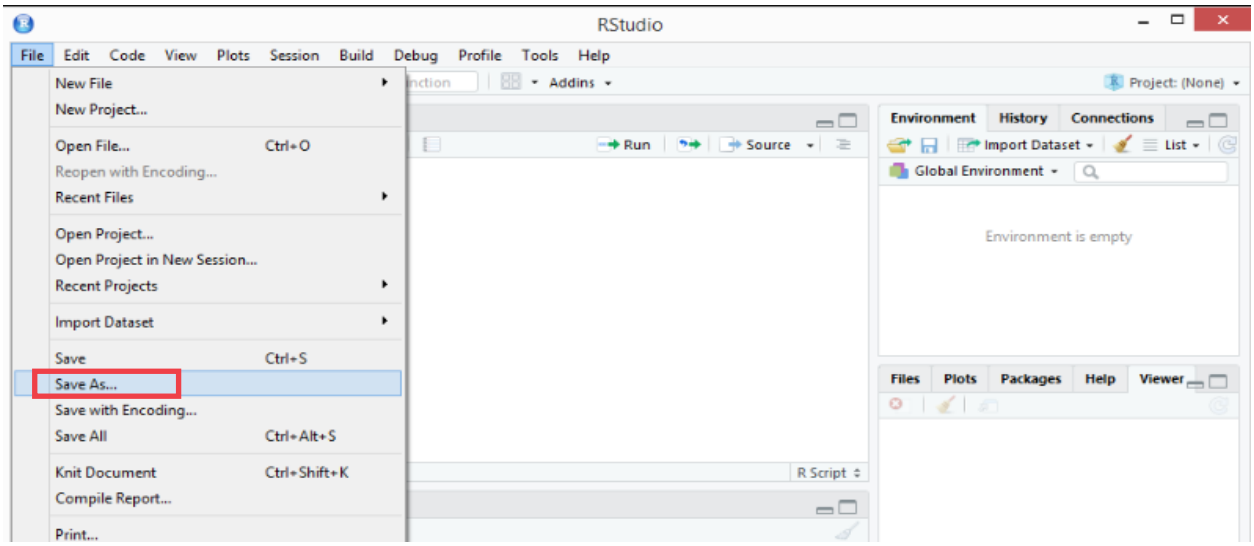
Select **New File** to create R script.



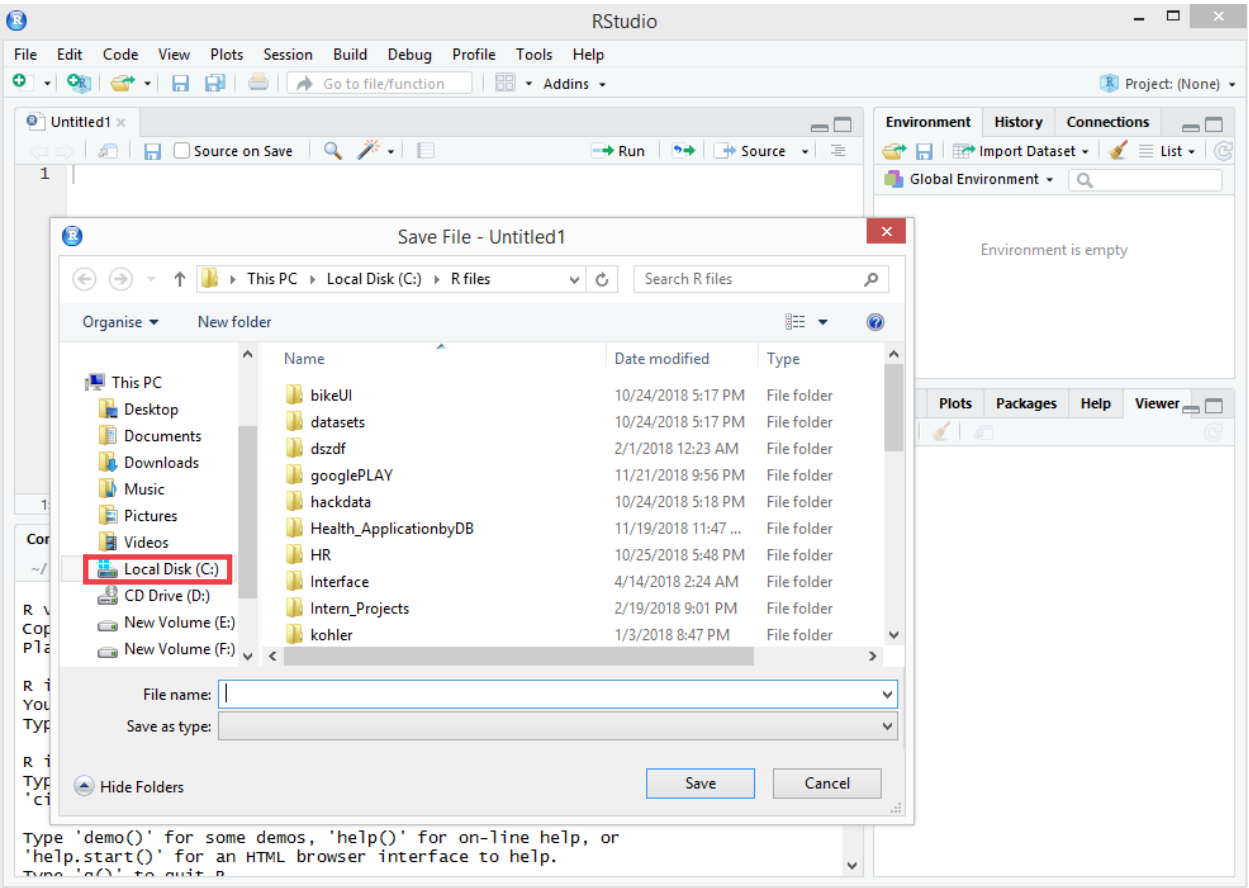
After selecting **New File** there, you will find multiple options. Select **R Script** which is at the top.



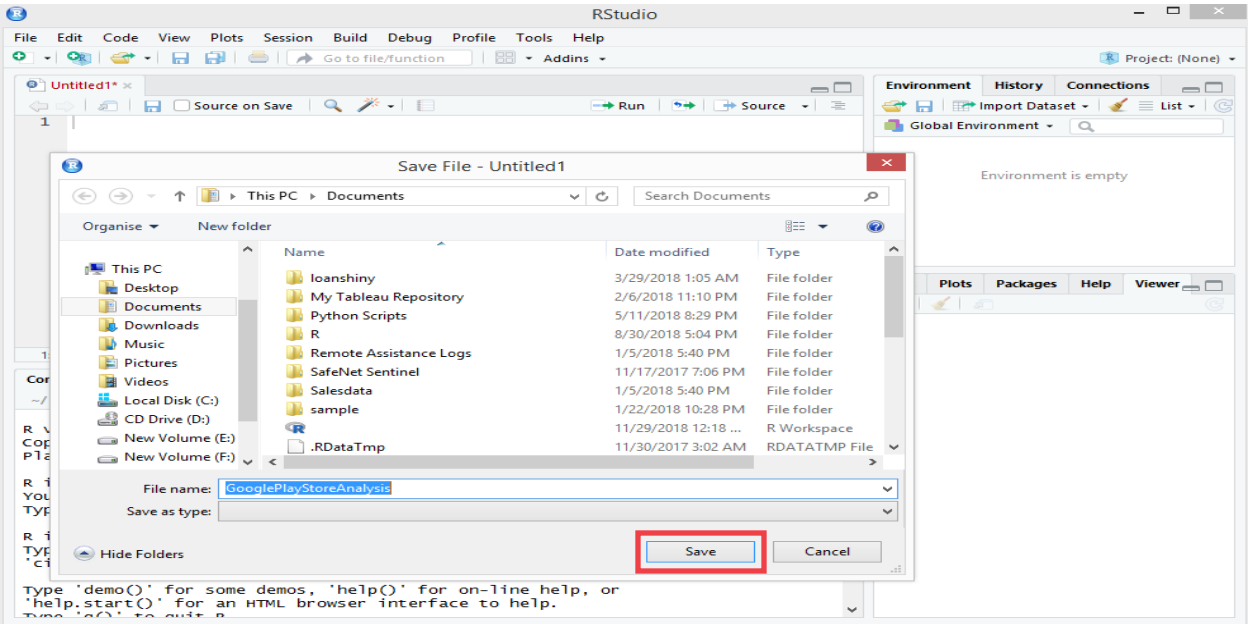
Now your R script will be created with the name **Untitled1.** To save R script, go to file and select **Save As** option.



Choose the location of the directory in which you want to save.

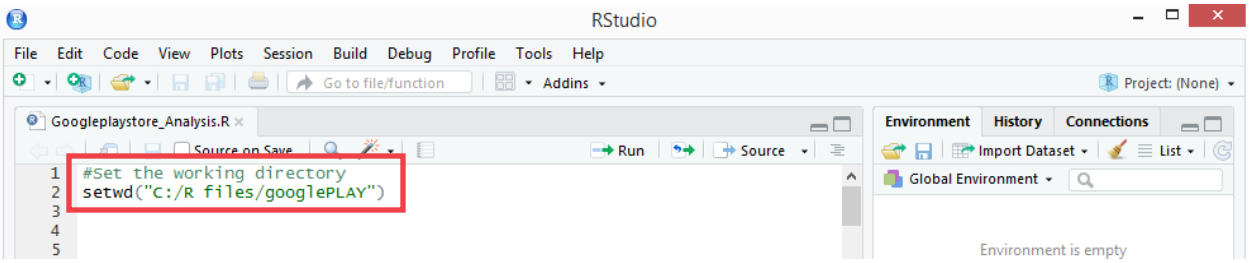


Enter the name you want to give to your R Script in **File name** field and click on **Save** option.



Step #2 | Setting up your Directory

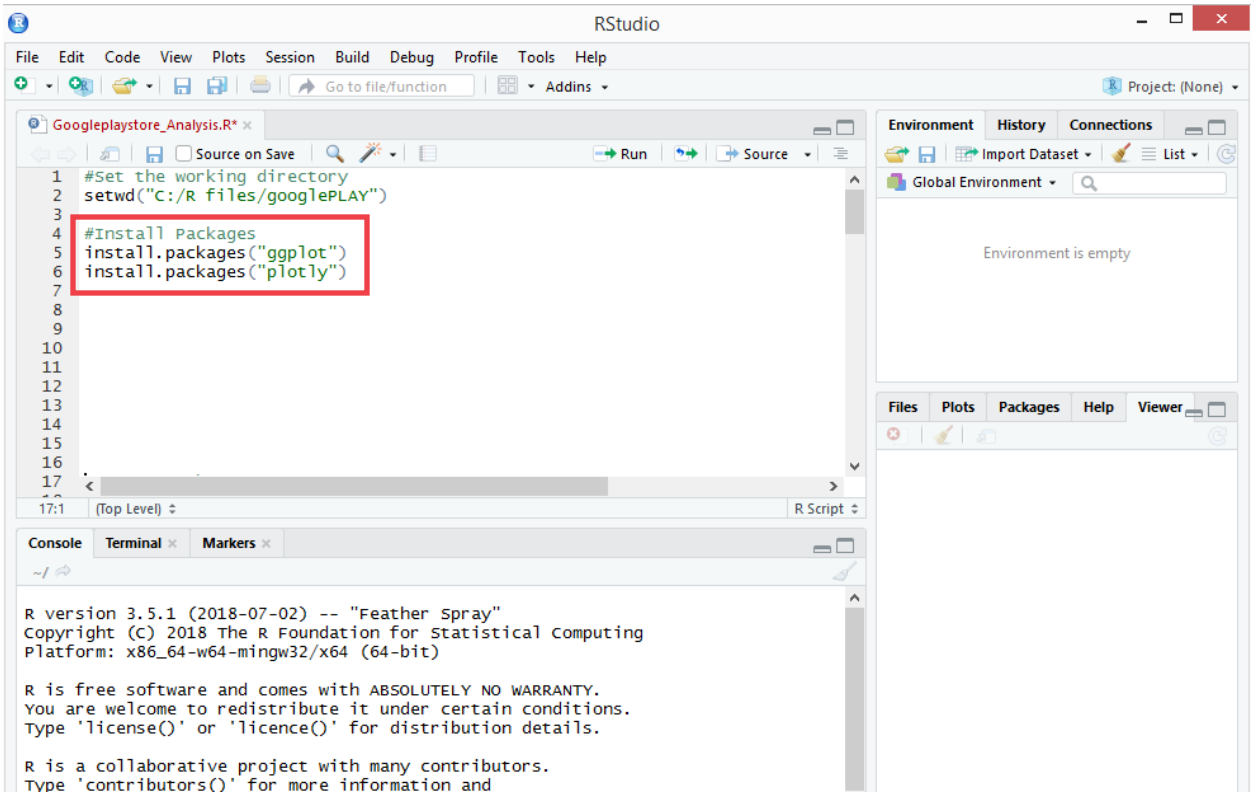
Set your data file location inside setwd function.
setwd("C:/R files/googlePLAY")



Step #3 | Install Packages

Install the libraries which are necessary for our R application.

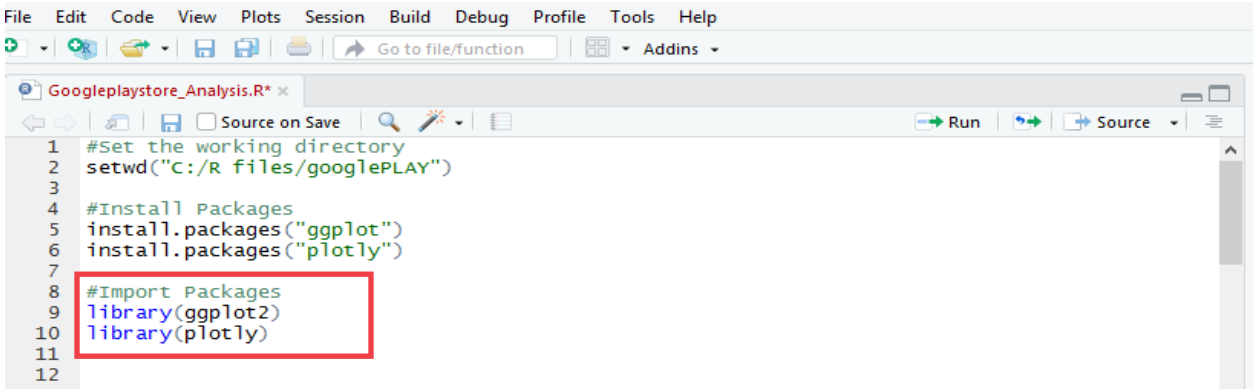
install.packages("ggplot")
install.packages("plotly")



Step #4 | Load Packages

Load the installed packages through libraries.

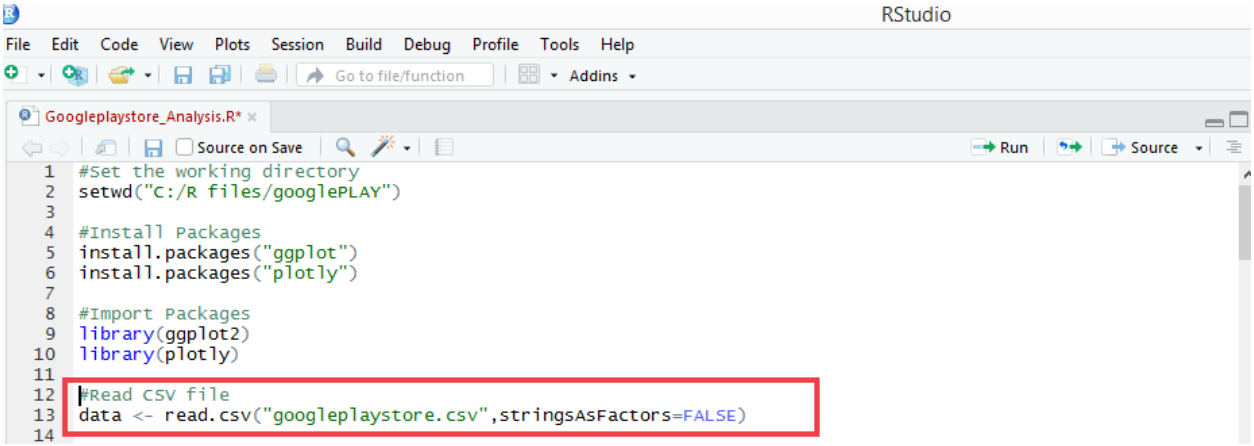
library(ggplot2)
library(plotly)



Step #5 | Import Data into R

Import the dataset to R environment and store it in a variable. Here, we load our data from the csv file into a variable called “data”.

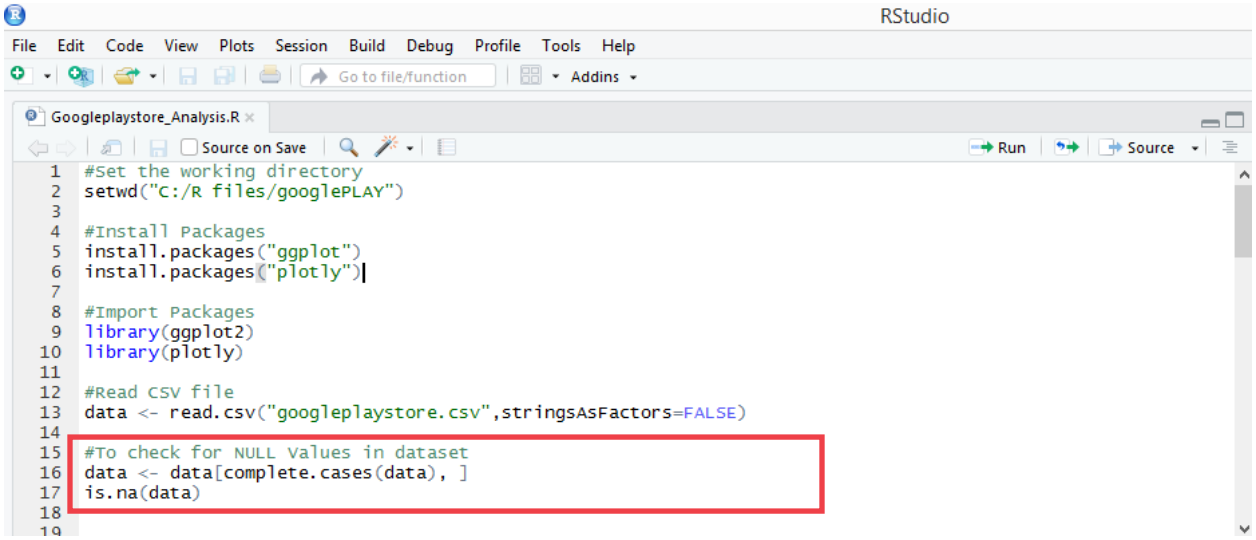
data <- read.csv("googleplaystore.csv",stringsAsFactors=FALSE)



Step #6 | Data Cleansing Process

Check for null values in the data.

data <- data[complete.cases(data),]



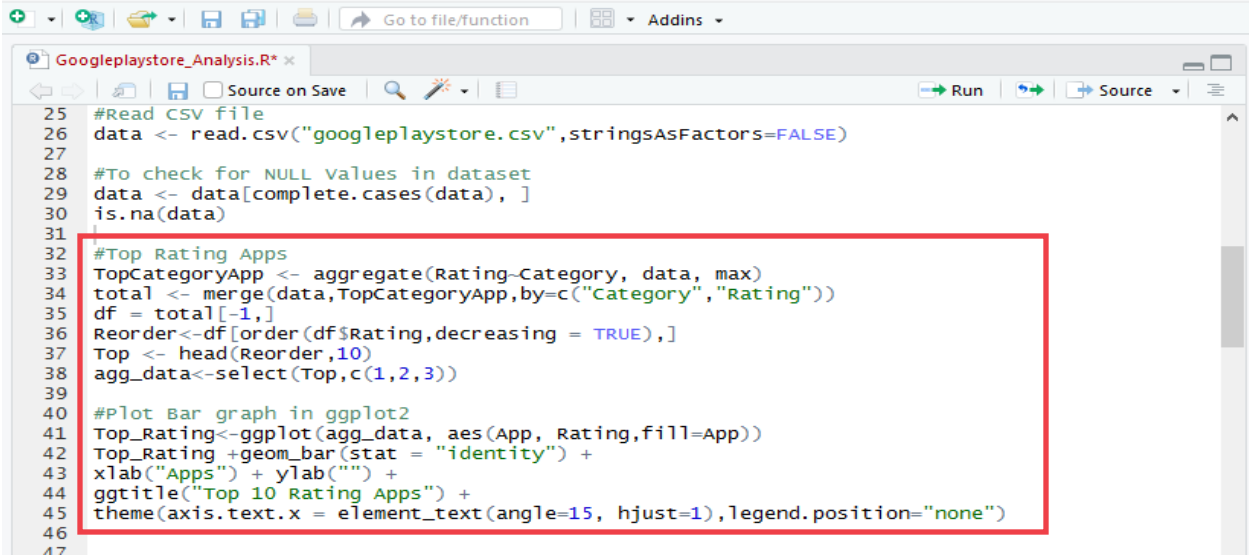
Step #7 | Find out Top Rating Apps

Data Preparation - Inorder to perform data analysis, we need to prepare the data according to our requirements.

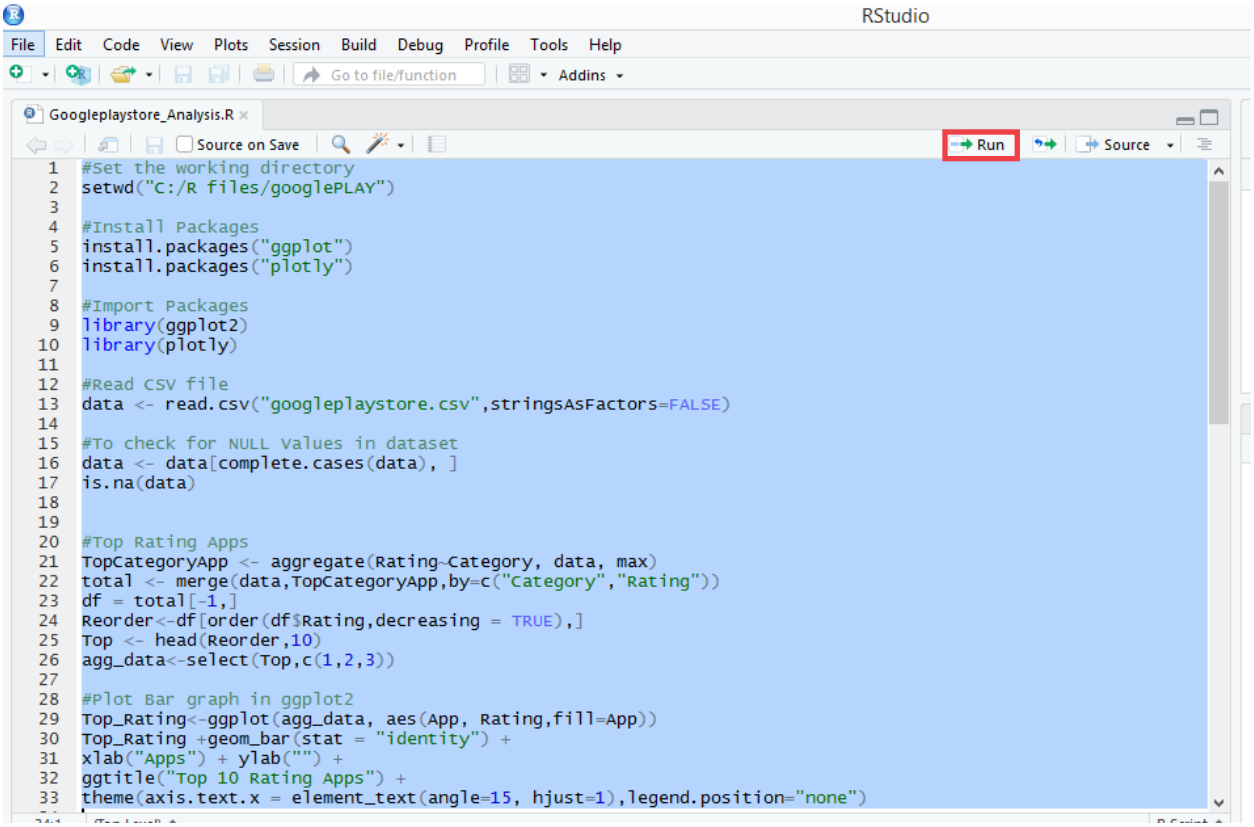
```
TopCategoryApp <- aggregate(Rating~Category, data, max)
total <- merge(data,TopCategoryApp,by=c("Category","Rating"))
df = total[-1,]
Reorder<-df[order(df$Rating,decreasing = TRUE),]
Top <- head(Reorder,10)
agg_data<-select(Top,c(1,2,3))
```

Visualize the resulted data in the form of graphs.

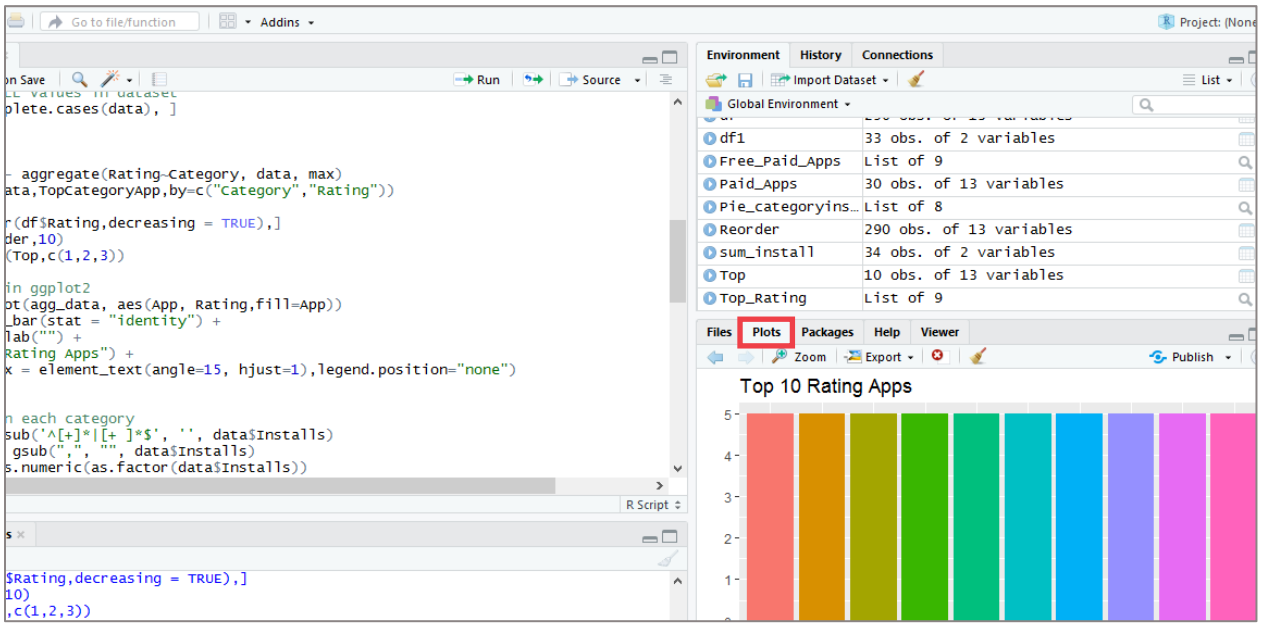
```
Top_Rating<-ggplot(agg_data, aes(App, Rating,fill=App))
Top_Rating +geom_bar(stat = "identity") +
xlab("Apps") + ylab("") +
ggtitle("Top 10 Rating Apps") +
theme(axis.text.x = element_text(angle=15, hjust=1),legend.position="none")
```



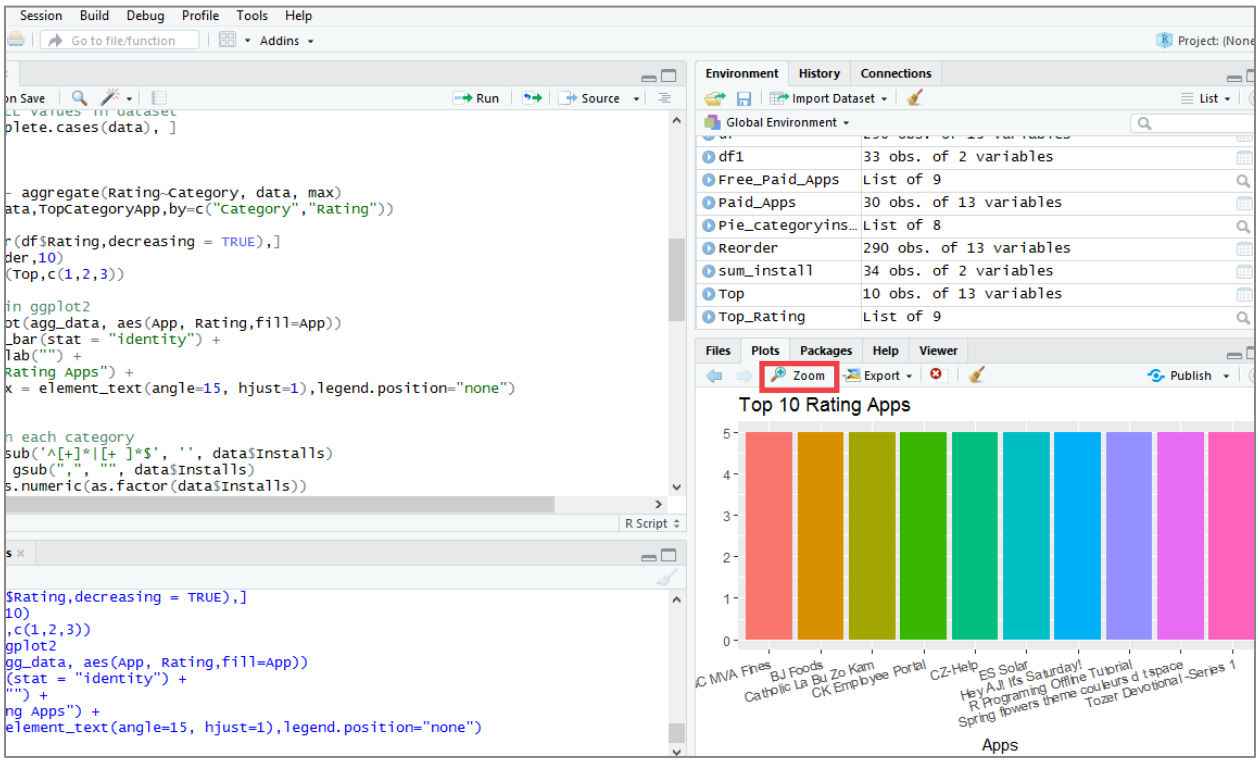
To run commands in R script, select the commands you wish to execute and click on **run** button.



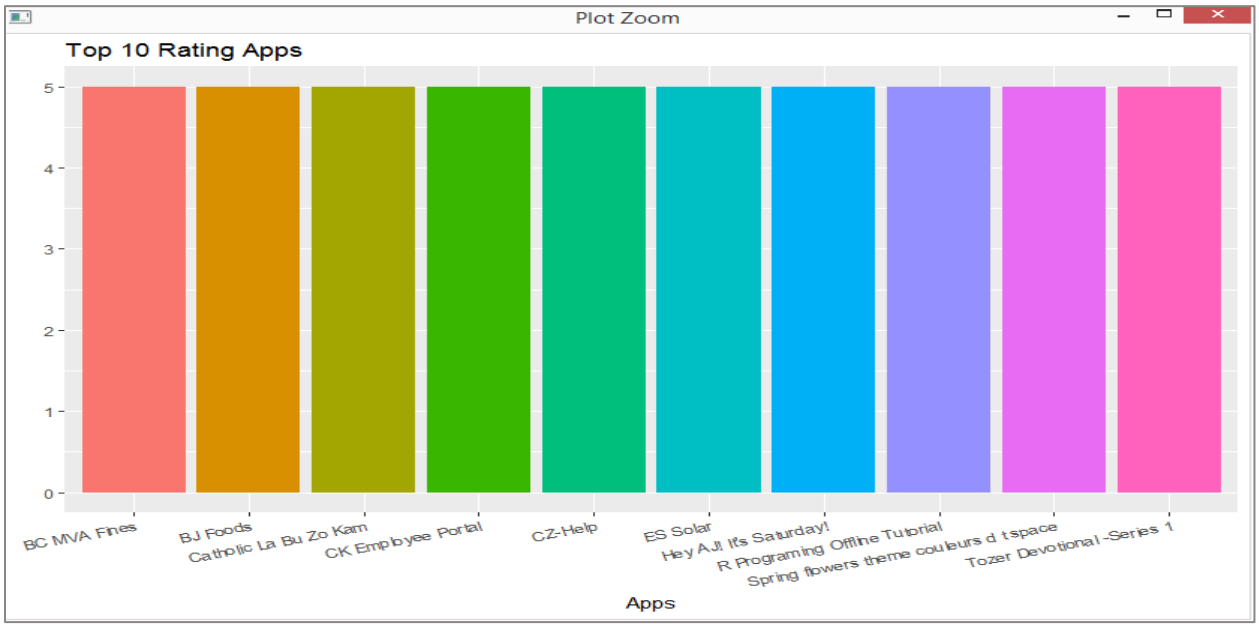
The graphs will be displayed under **Plots** tab which is at the bottom right corner of the R Studio.



We can choose **Zoom** option to view the graph in a separate window.



Below is the resulted ggplot graph for Top Rating Apps,

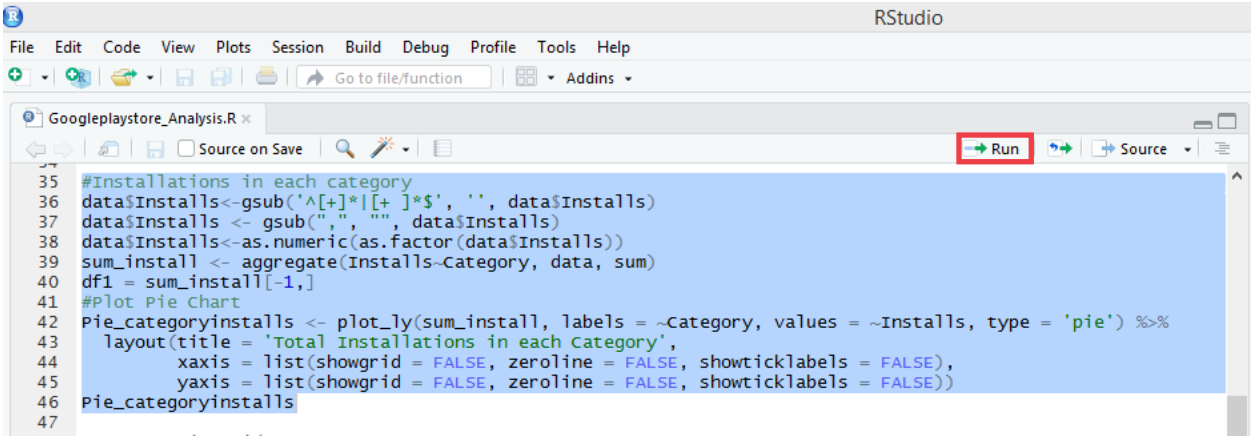


Step #8 | Find out Installations Count in each Category

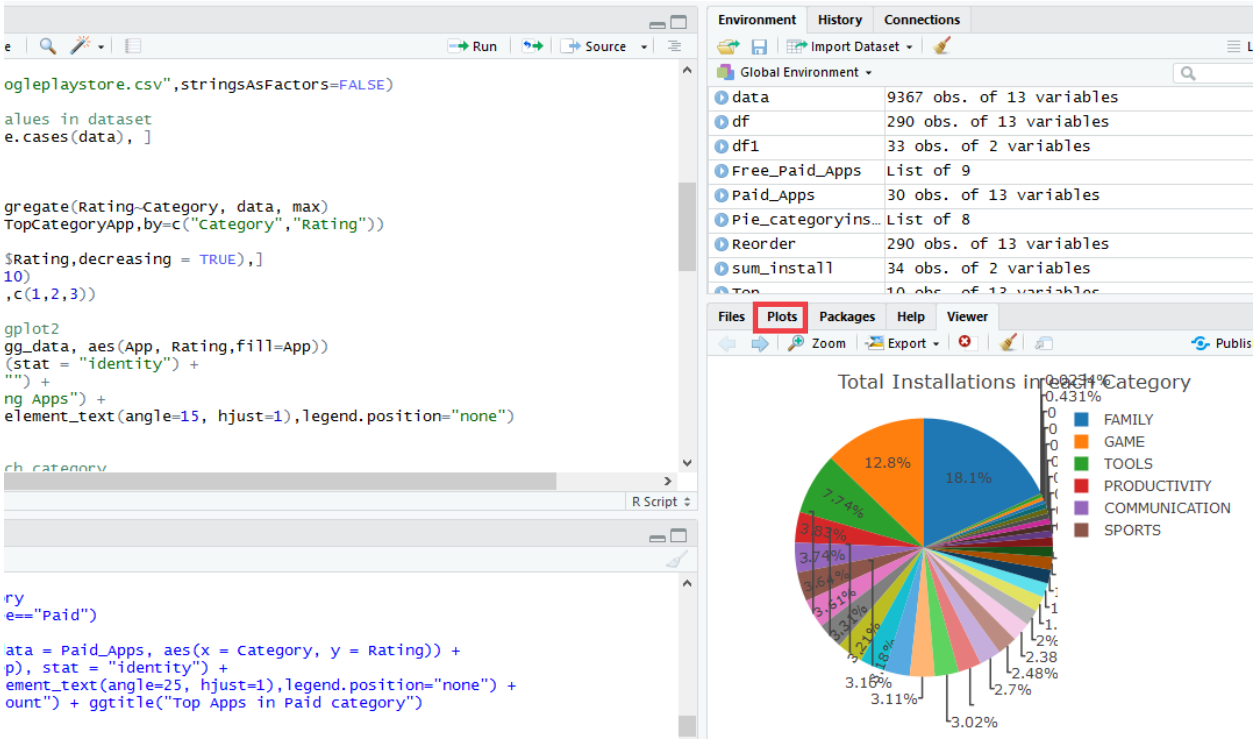
Calculate number of installations in each category.

```
data$Installs<-gsub('^[+]*|[+ ]*$', '', data$Installs)
data$Installs <- gsub(",", "", data$Installs)
data$Installs<-as.numeric(as.factor(data$Installs))
sum_install <- aggregate(Installs~Category, data, sum)
df1 = sum_install[-1,]
#Plot Pie Chart
Pie_categoryinstalls <- plot_ly(sum_install, labels = ~Category, values = ~Installs,
type = 'pie') %>%
layout(title = 'Total Installations in each Category',
xaxis = list(showgrid = FALSE, zeroline = FALSE, showticklabels = FALSE),
yaxis = list(showgrid = FALSE, zeroline = FALSE, showticklabels = FALSE))
Pie_categoryinstalls
```

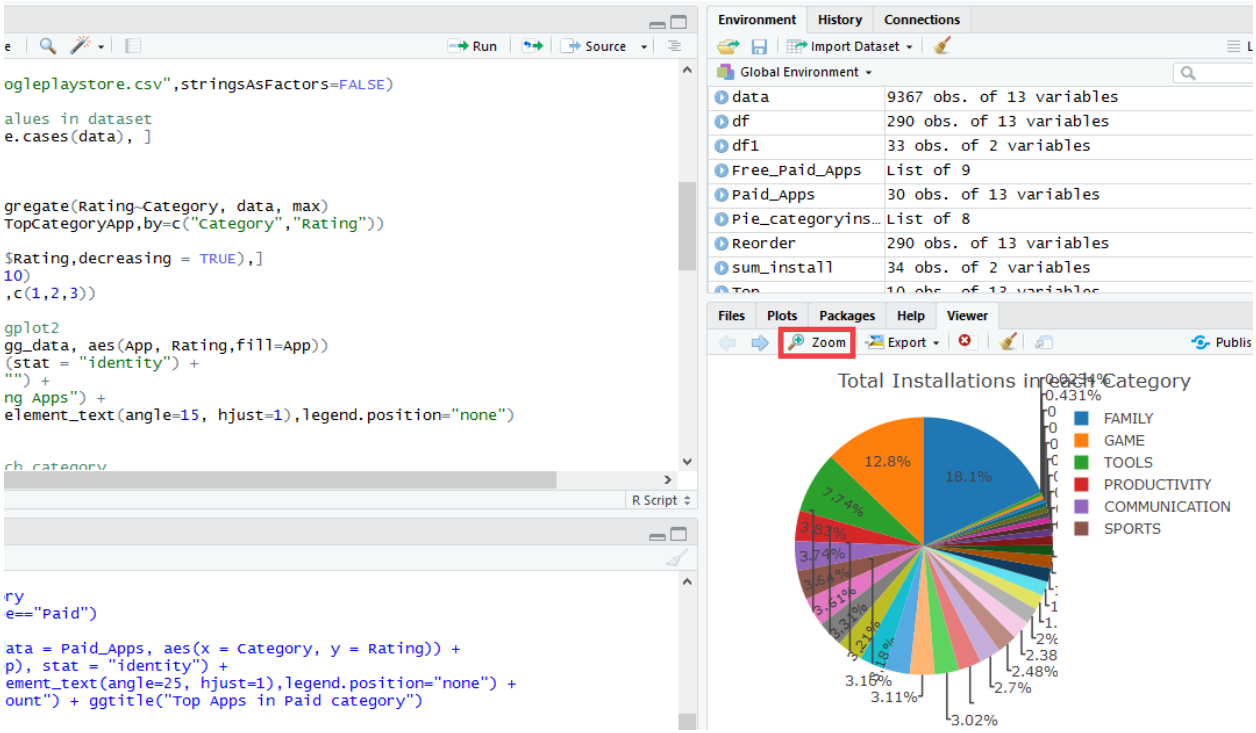
To run commands in R script, select the commands you wish to execute and click on **run** button.



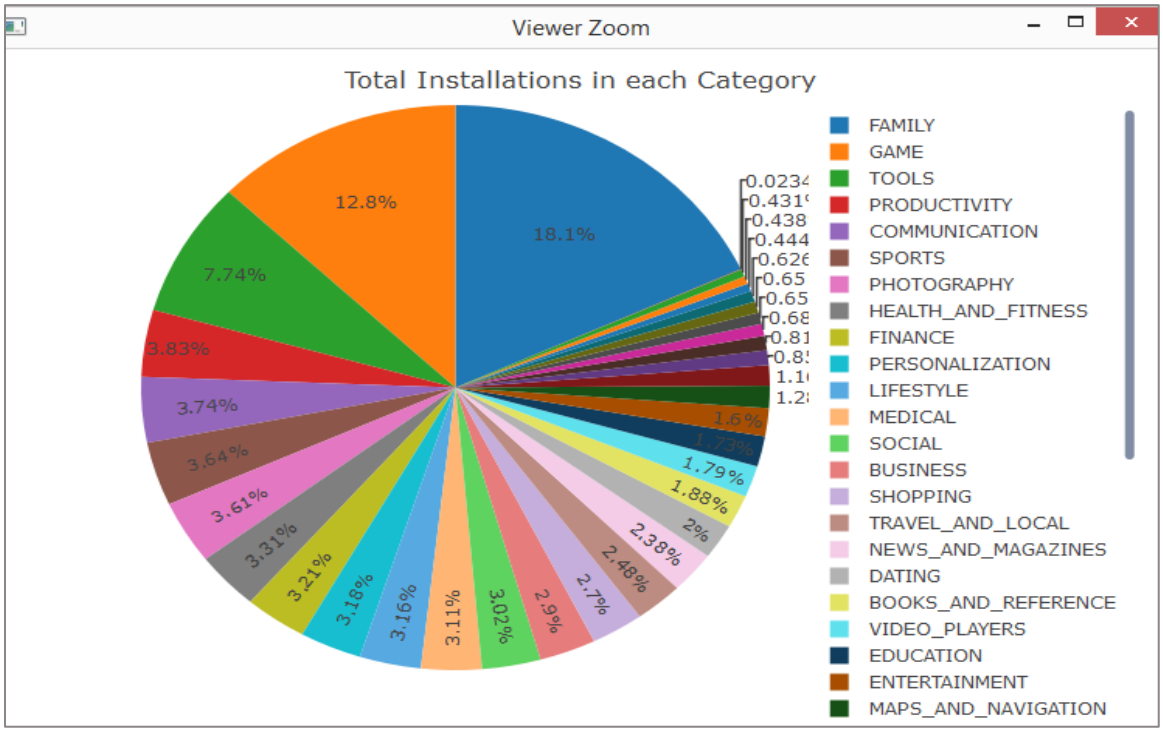
The graphs will be displayed under **Plots** tab which is at the bottom right corner of the R Studio.



Select **Zoom** option to view the graph in a separate window.



Below is the Pie chart for Number of Installations in each Category,

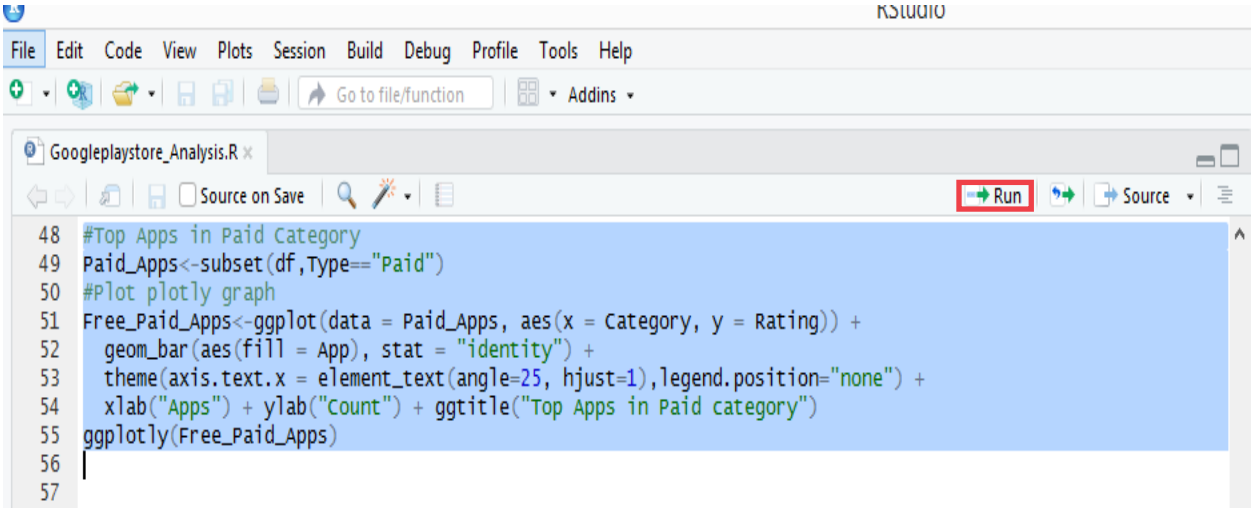


Step #9 | Find out Top Paid Apps

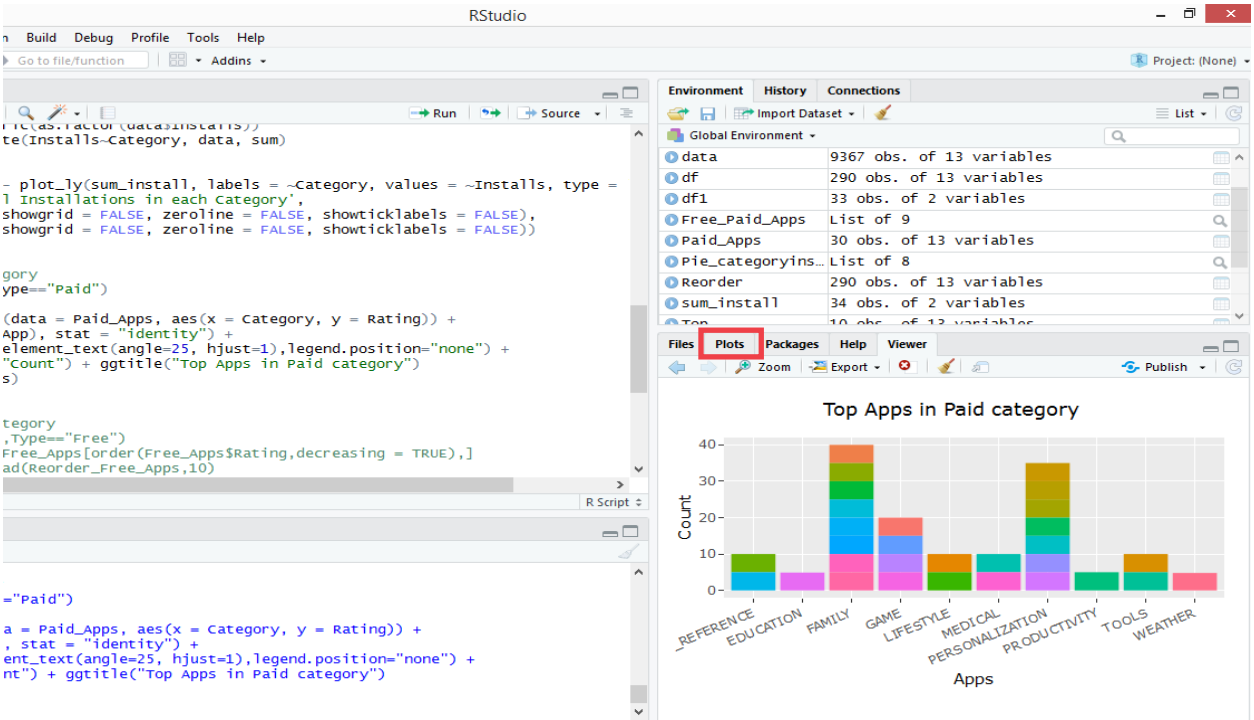
Analyze top paid apps.

```
Paid_Apps<-subset(df,Type=="Paid")
TopPaid_Apps<-ggplot(data = Paid_Apps, aes(x = Category, y = Rating)) +
 geom_bar(aes(fill = App), stat = "identity") +
 theme(axis.text.x = element_text(angle=25, hjust=1),legend.position="none") +
 xlab("Apps") + ylab("Count") + ggtitle("Top Apps in Paid Category")
ggplotly(TopPaid_Apps)
```

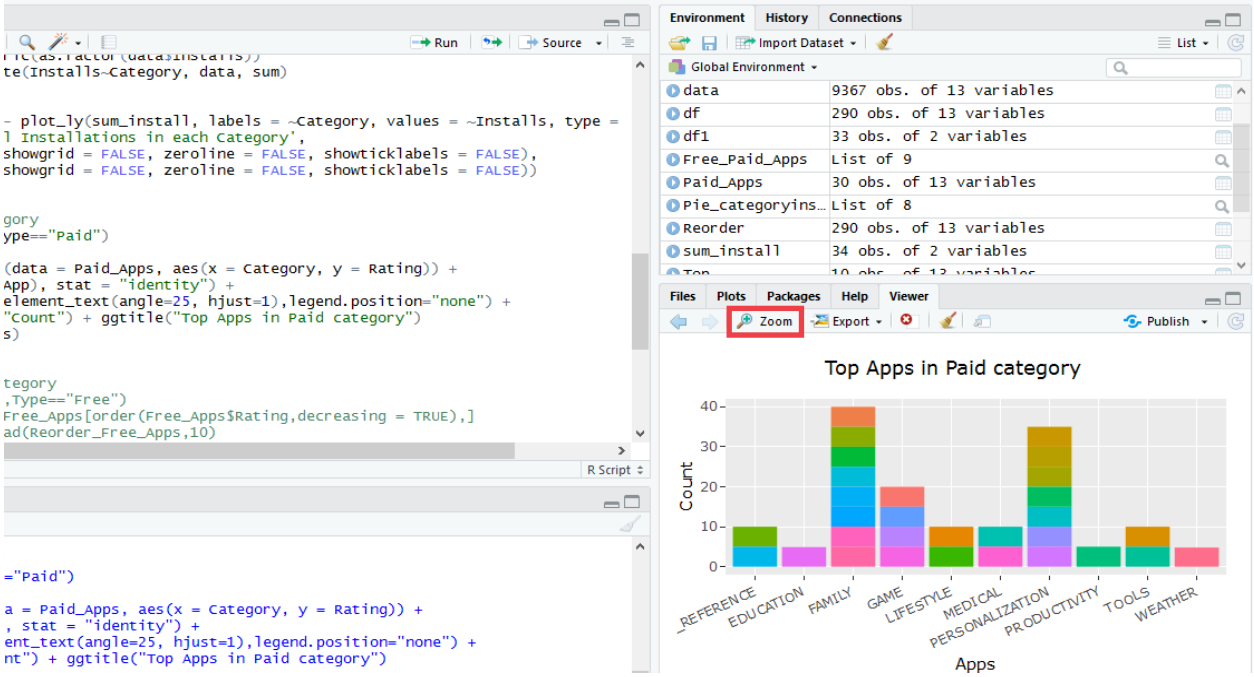
To run commands in R script, select the commands you wish to execute and click on **run** button.



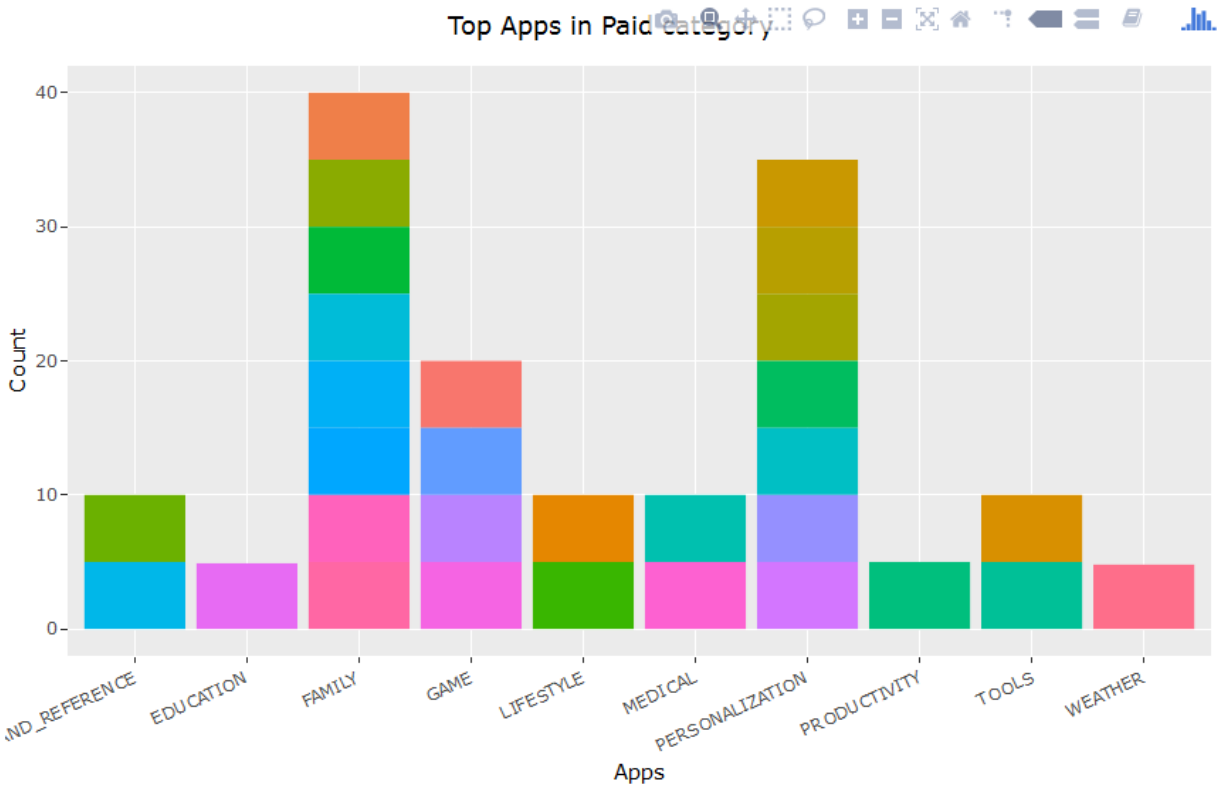
The graphs will be displayed under **Plots** tab which is at the bottom right corner of the R Studio.



Select **Zoom** option to view the graph in a separate window.



Below is the plotly graph for Top Apps in Paid Category,



Hurrah!! With this lab you were able to analyze and visualize the **Google Play Store Apps Data using R.**

For any questions regarding the lab please feel free to reach out to innoation@miraclesoft.com. **We hope you enjoyed analyzing data with us ☺**