



ASSIGNMENT 1

Optimizing Marketing Strategies Through Data-Driven Insights



SEPTEMBER 22, 2024
GOKULAKRISHNAN SELVARAJAN
Student id: 8940140

Contents

Introduction:	2
Objectives:	2
Methodology:	2
Data collection:	3
Web_Souce_data:	3
Social media data:	4
Email Campaign data:	4
Customer transaction data:	5
Data Preprocessing:	5
Exploratory Data Analysis (EDA):	6
Web source data insights:	6
Social media findings:	8
Email Campaign data insights:	9
Customer data insights:	11
Customer Segmentation and Profiling:	11
Predictive Modeling and Forecasting:	13
A/B testing and optimization:	13
Reporting and Visualization:	14
Conclusion:	15
Reference:	17
Appendix:	17

Introduction:

Companies are storing vast amounts of data about the customers and their business, which is further used for analytical purposes and gaining valuable insights out of it. These insights can help the business or company to formula effective strategies for its growth.

This data analysis assignment is more focused to enhance marketing strategies and maximize the revenue generated.

The fictitious e-commerce company called ABC corporation is eagerly looking out for ways to optimize the strategies in order to develop its business. Below are the objectives and methodology used for analyzing ABC corporation.

Objectives:

The major objective of this analysis is to dig deeper on the data collected from the sources like web, social media, email campaigns, and customer database, which can help further to optimize marketing strategies for ABC corporation and can result in financial growth. The major analysis carried out in this process are,

- Prediction revenue from new email campaigns by using the past data.
- Segmentation of their customers by their transaction details
- Formulating A/B testing to enhance the new campaign performance.

Methodology:

As the ABC corporation is a fictional company, the methodology for the analysis involved,

1. **Data Creation** – Created four types of datasets using R
2. **Data preparation** – The created datasets were pre-processed in R to make it analytical ready.
3. **Exploratory analysis** – Several data insights were drawn by exploring data using R.
4. **Data Visualization** – Several insights drawn from the previous steps were visualized in R using ggplot2 package.
5. **Predictive modeling** - We used predictive modeling techniques to predict revenue generation for any new email campaigns by ABC Corporation.
6. **A/B testing** - We used A/B testing to compare the email campaigns to identify the similarity between.
7. **Visual reports** - Used Shiny package from R to create reports

Data collection:

The ABC Corporation is a fictitious company, and therefore I used R to create four datasets. We are creating the data sets for the duration of six months starting March 1st to August 31st 2024. Below are the details about the datasets,

Web_Souce_data:

- **Date:** The date on which the data about the website is collected (6 months).
- **Exit_rate:** It denotes in percentage terms that no. of people exiting the website after visiting just the home page.
- **Potential_views:** It denotes in percentage terms that no. of people bought something when they opened the website.

- **No_of_users:** It denotes the number of users used the website that day.
- **Device_type:** It denotes the type of device users are using to access the website.
- **Old_Vs_New:** It denotes whether the users accessed the website are new or old.

Social media data:

- **Websites:** It denotes the social media platforms of ABC Corporation.
- **No_of_followers:** It denotes the number of followers ABC corporation has on social media platforms.
- **No_of_posts:** It denotes the number of posts ABC corporation posted on these platforms.
- **No_of_interactions:** It denotes the number of interactions on the posts of ABC Corporation across platforms.
- **No_of_reach:** It denotes the number of users the post has reached.
- **No_of_shares:** It denotes the number of shares the users have made for ABC Corporation.

Email Campaign data:

- **Campaign no:** It denotes the Email campaign ids
- **No_of_sent_emails:** It denotes the number of emails sent as part of the email campaigns.
- **Generated_revenue:** It denotes the revenue generated by each campaign.
- **Response_rate:** It denotes the percentage of the users giving positive response to the emails.

- **Delivery_failure_rate:** It denotes the percentage of mails that failed to be delivered.
- **Ignorance_rate:** It denotes the percentage of the users who did not open the emails.

Customer transaction data:

- **Customer_id:** It denotes the id of the customer.
- **Sex:** It denotes the gender of the customer.
- **No_of_products:** It denotes the number of products bought by the user.
- **Amount_spent:** It denotes the total amount spent by the user.
- **Location:** It denotes the location of the user.
- **Payment method:** It denotes the user's mode of payment.

Data Preprocessing:

Below are the data preprocessing methods used to make the data analytical ready,

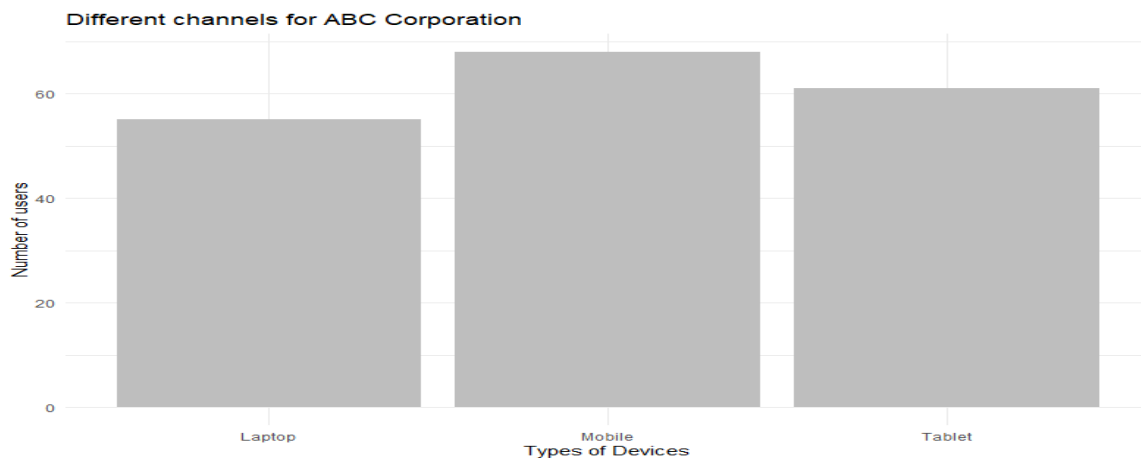
- **Data Cleansing:** Duplicate rows were removed in all the four data sets by using the function called Distinct. The column names were also standardized by using the function `tools::toTitleCase`
- **Datatype conversion:** Converted the datatypes of the columns which had categorical values to factors which will help models to work better.
- **Normalization and Scaling:** All the various rates in the datasets were converted to the percentage terms by multiplying the values by 100.
- **Missing values:** The missing values in the datasets were checked and filled in the by the appropriate values.

Exploratory Data Analysis (EDA):

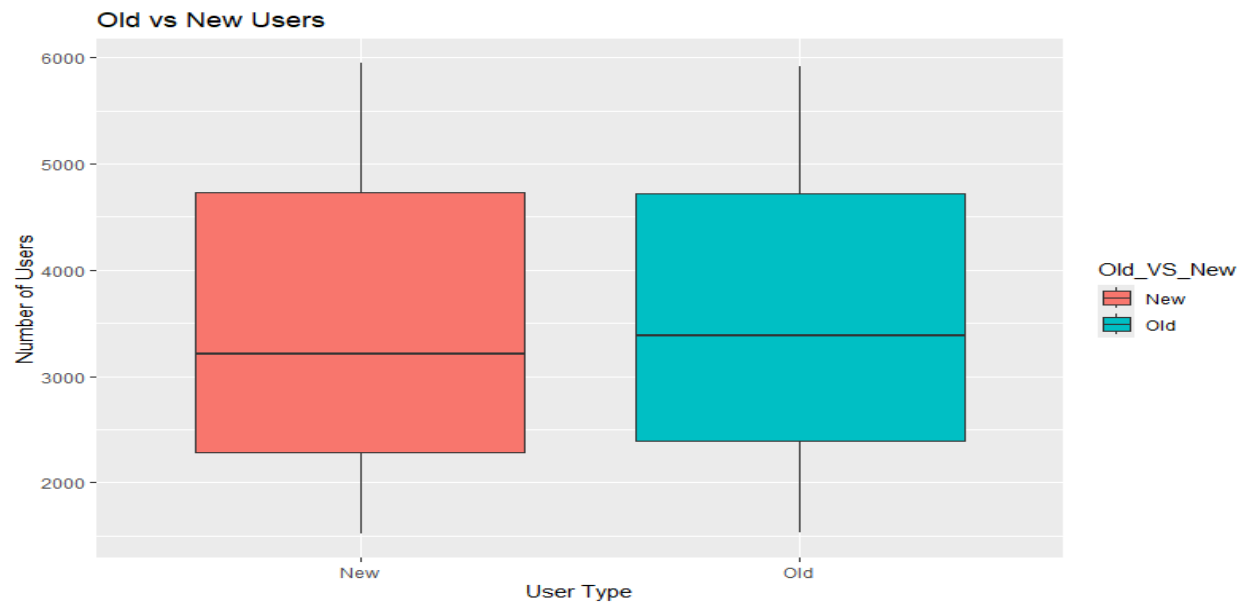
The extensive data analysis was carried out in all the datasets and various visualizations were created to gain insights about the data.

Web source data insights:

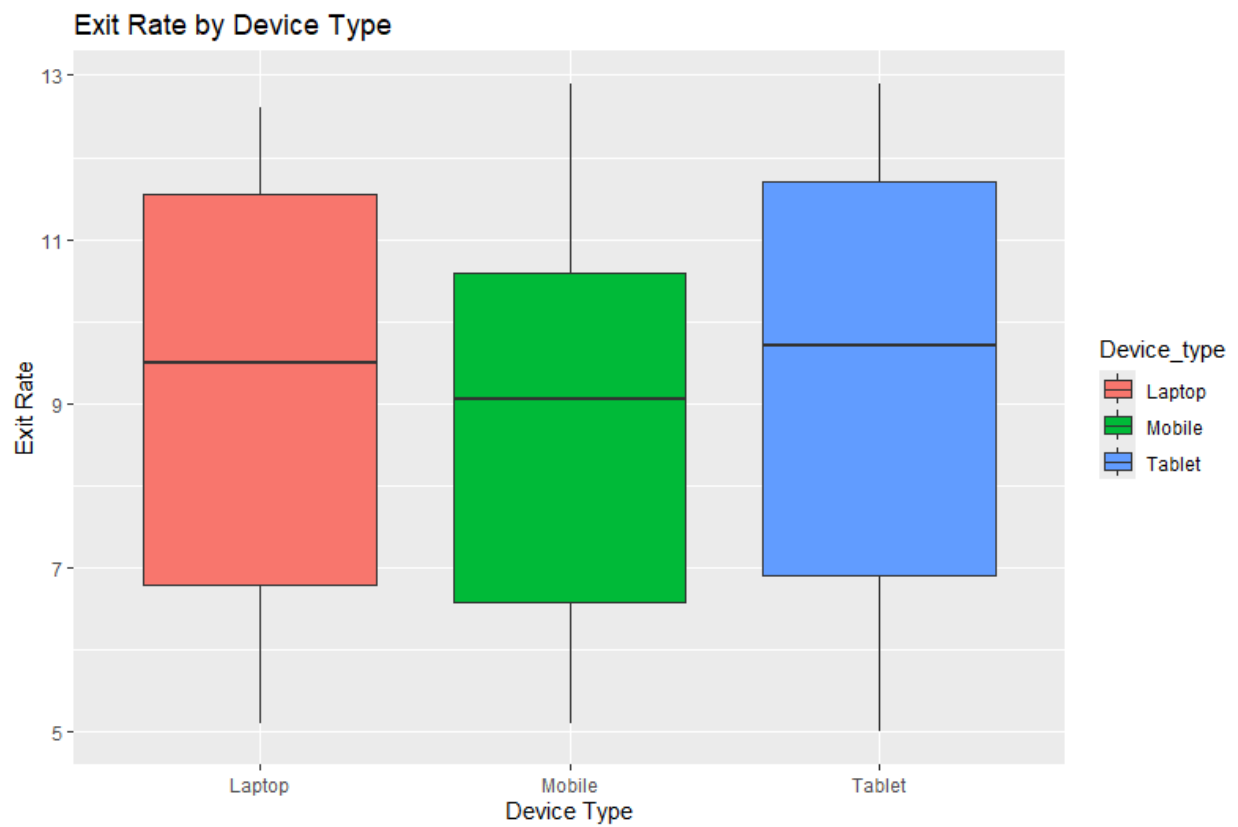
To understand the ABC Corporation better, we used ggplot2 to visualize and understand the nuances of the data,



Based on the above bar graph, it's understood that ABC Corporation has more mobile users and followed by Tablet and lastly Laptop users.

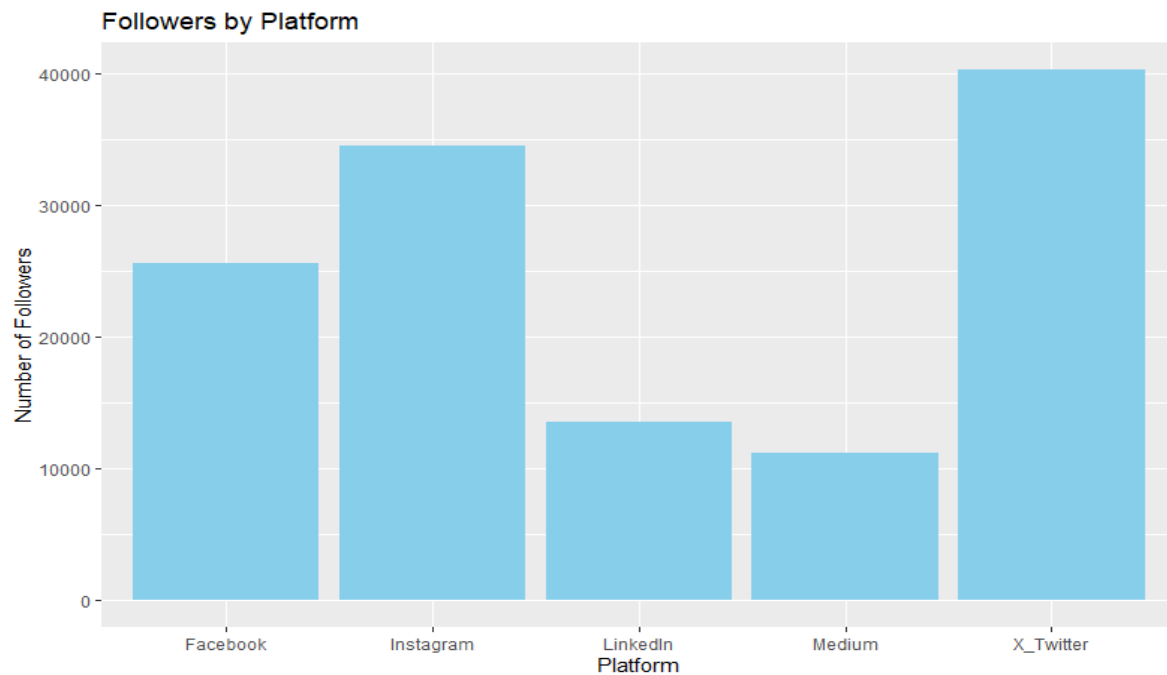


According to the above box plot, it's understood that we have almost equal number of old and new users in the six months data we had.



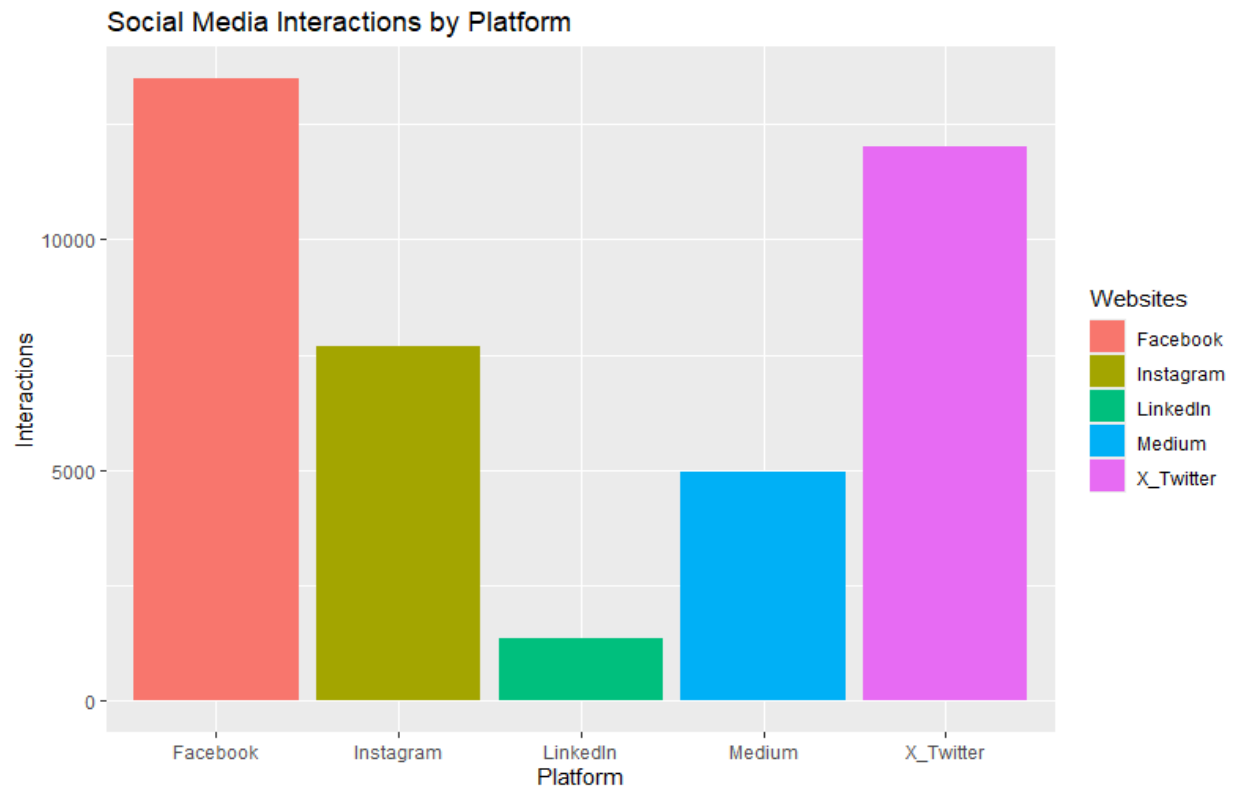
Based on the above visualization, it is evident that tablet users are exiting more frequently and followed by laptop and lastly the mobile users. It is interpreted that mobile users are spending more time on ABC Corporation websites than any other users.

Social media findings:



As per the above visualization, ABC Corporation has more followers on X, which is their official website. Secondly it is Instagram where they post about their new product offerings

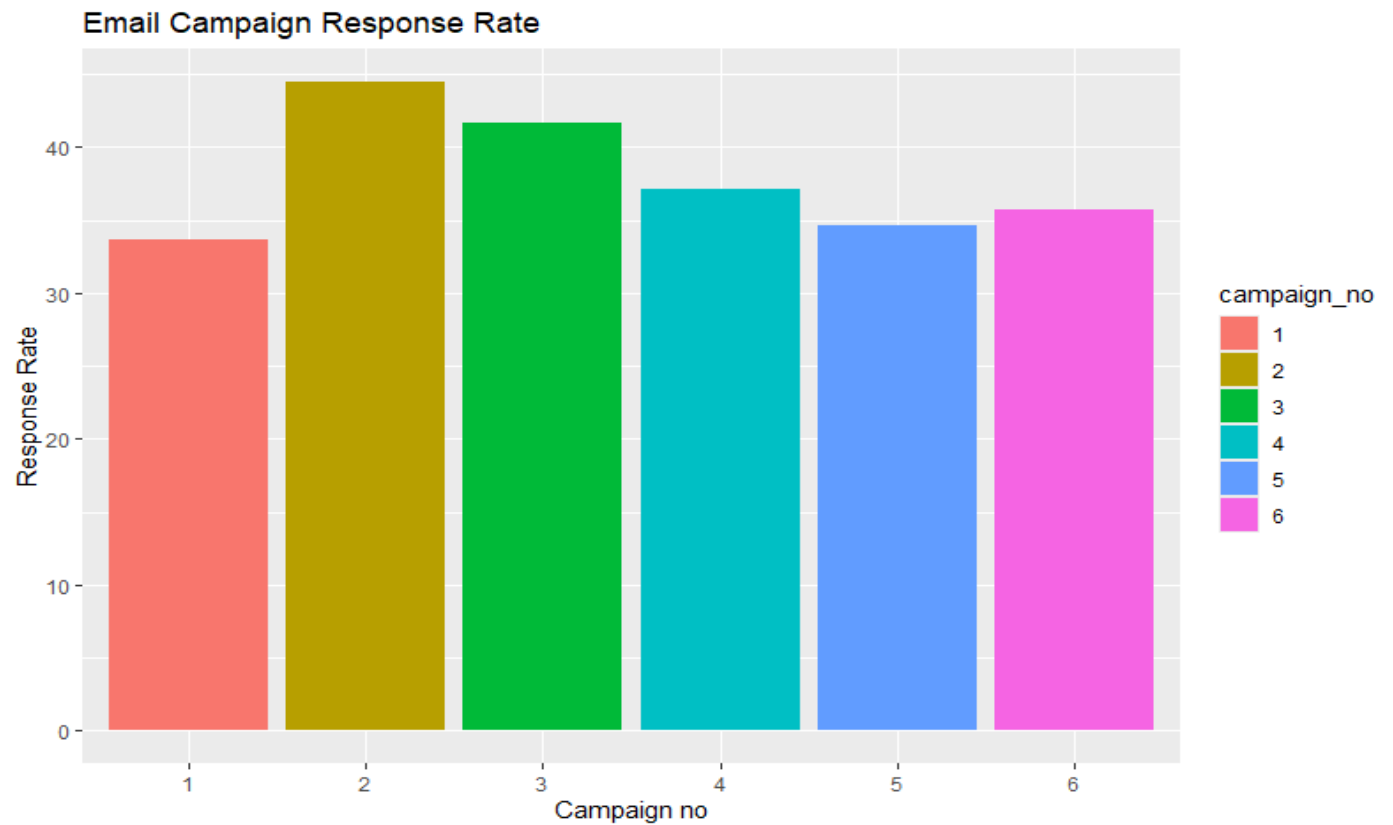
followed by Facebook, LinkedIn, and Medium.



According to the above graph plot, it is interesting to know that users are more engaged with ABC Corporation in Facebook through their likes, comments and shares. On the other hand, ABC Corporation's Linked posts have the least interaction amongst the users.

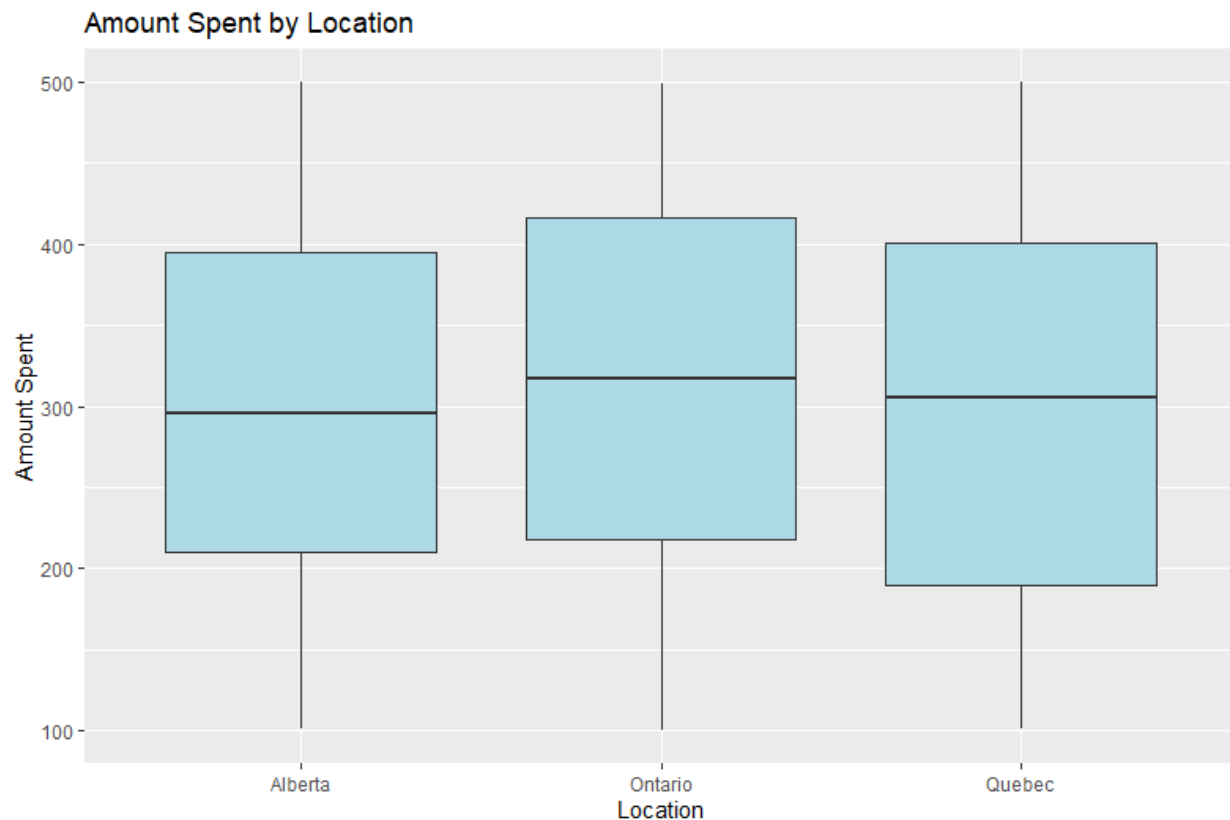
Email Campaign data insights:

ABC Corporation has the practice of sending discounts and new offering mails to their users. They have come across six email campaigns to offer new products and sale events.



The above visualization shows that Email campaign 2 has a higher response rate, (i.e) users have responded to the mails largely in the second email campaign. The first Email campaign has the least response rate than any other campaign.

Customer data insights:



Based on the above visualization, it is understood that ABC Corporation's Ontario users are more likely to spend higher when they make a purchase. Alberta users spend little less compared to other regions.

Customer Segmentation and Profiling:

ABC corporation's customer data is utilized for segmenting the customers based on features like No_of_products and Amount_spent. Furthermore, the three different clusters for the customers are identified as,

- Cluster 1: The users who spend less and buy fewer products in a purchase.

- Cluster 2: The users who spend on a medium scale with multiple products in a purchase.
- Cluster 3: The users who spend more and buy multiple products in a purchase.



Based on the above Kmeans clustering visualization, it is found out that,

Segment 1, who tends to spend less, mostly below \$300, and buy fewer products between 1 to 4.

Segment 2, who tends to buy more products typically between 5 to 8, but spends moderately between \$100 to \$400

Segment 3, who always spends more than \$300 to \$500 and buys a wide range of products.

ABC Corporation can utilize this segmentation and can formulate the marketing strategies for each group based on their purchasing practices.

Predictive Modeling and Forecasting:

I have used a linear regression model to predict the revenue generated from any new email campaigns based on the number of emails sent, response rate, and delivery failure rates.

ABC Corporation can input their desired number of mails that will be sent along with response rate and failure rates which can help them understand their revenue generation from the email campaign.

Model: Generated revenue ~ Emails sent + Response Rate + Delivery failure rate

I have used a random number for the inputs to the model, and it's predicted that the new email campaign has the potential to generate revenue of about \$13,832.

A/B testing and optimization:

I divided the six email campaigns as two sets to identify which campaign set is a winning set. To proceed with the analysis, I have used a statistical test (t-test) to compare the mean revenue between two sets.

The six campaigns are divided into two sets,

- Campaign set1 – Campaign 1, Campaign 2, and Campaign 3.
- Campaign set2 – Campaign 4, Campaign 5, and Campaign 6.

The p-value from the t-test will help us to know if there is a statistically significant difference in the revenue generated by the two campaigns. If the p-value is less than the significance level of 0.05 then we can conclude that there is significant difference between the two campaign sets.

T test result:

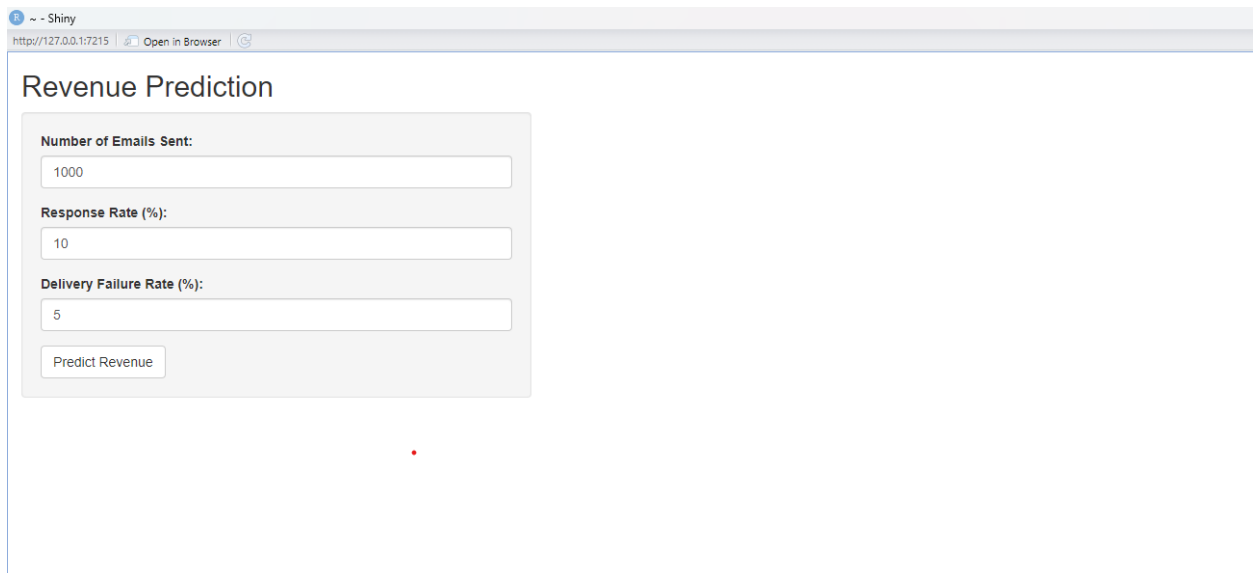
```
welch Two Sample t-test  
data: campaign_set1 and campaign_set2  
t = 0.16623, df = 3.8356, p-value = 0.8764  
alternative hypothesis: true difference in means is not equal to 0  
95 percent confidence interval:  
-6928.346 7795.012  
sample estimates:  
mean of x mean of y  
12266.67 11833.33
```

Based on the above results it is concluded the p-value is 0.8764 and therefore both the campaign sets are similar to each other and not significantly different. Hence, ABC Corporation has to come up with new marketing strategies for their next email campaign.

Reporting and Visualization:

I have used shiny package in R to create an UI interface for ABC Corporation to input requirements of a new email campaigns which can help us to know the revenue generated out of it.

Below is a snapshot of how the interface looks. This can be shared to any of the ABC corporation's stakeholders to understand the email campaign's revenue generation.



The screenshot shows a web browser window with a Shiny application. The browser's address bar displays 'http://127.0.0.1:7215' and includes an 'Open in Browser' button. The application's title bar reads '~ - Shiny'. The main content area is titled 'Revenue Prediction' and contains a form with three input fields: 'Number of Emails Sent:' with the value '1000', 'Response Rate (%):' with the value '10', and 'Delivery Failure Rate (%):' with the value '5'. Below these fields is a 'Predict Revenue' button. The form is set against a light gray background.

The user of this interface should enter the details of the new email campaign such as Number of Emails sent, response rate, and Delivery failure rate in percentage terms. Once the information is entered, the user should press the Predict Revenue button to know the revenue generation prediction from the campaign. The user can use the interface to trial-and-error different inputs and can settle down on the desired revenue they want from the new email campaign.

Conclusion:

Based on the data analysis of ABC corporation, below are the key findings measured.

1. **Revenue Prediction model:** The linear regression model effectively calculates the revenue generation based on the factors inputted. This can help ABC Corporation to

forecast better and can help them to come up with enhanced and effective Email campaign strategies in the future.

2. **A/B testing:** A/B testing helped in understanding which email campaign sets were better in terms of revenue generation. The results show that both the email campaign sets are not significantly different and therefore ABC Corporation has to come up with effective strategies and not to follow the old method to achieve greater growth.
3. **Customer Segmentation:** With the use of clustering analysis, customers were divided based on purchasing behavior. This segmentation can help ABC corporation to formulate effective marketing strategies based on their user's spending habits.

Below are the few recommendations based on the above findings,

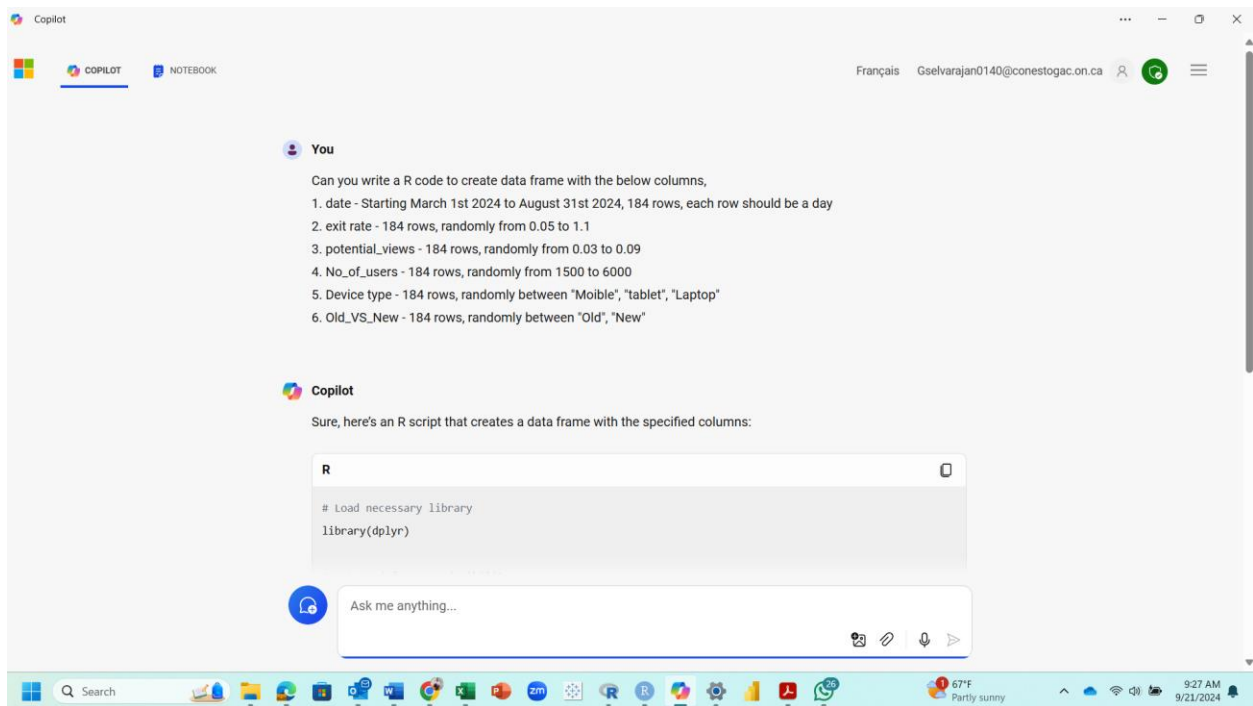
1. **Incline towards effective campaigns:** ABC Corporation should more focus on marketing campaigns that have lower delivery failure rates and higher response rates. This can help them to drive up the revenue.
2. **Marketing strategies based on customer segments:** Based on the customer segments provided to ABC Corporation, they should create marketing strategies pertaining to their buying pattern. Low spenders should be focused more on promoting cheaper products and high spenders should be marketed with high priced products frequently to improve the revenue generation.
3. **Social media strategies:** ABC Corporation should focus more on promoting their products in Facebook rather than any other social websites as they have more interactions compared to others.

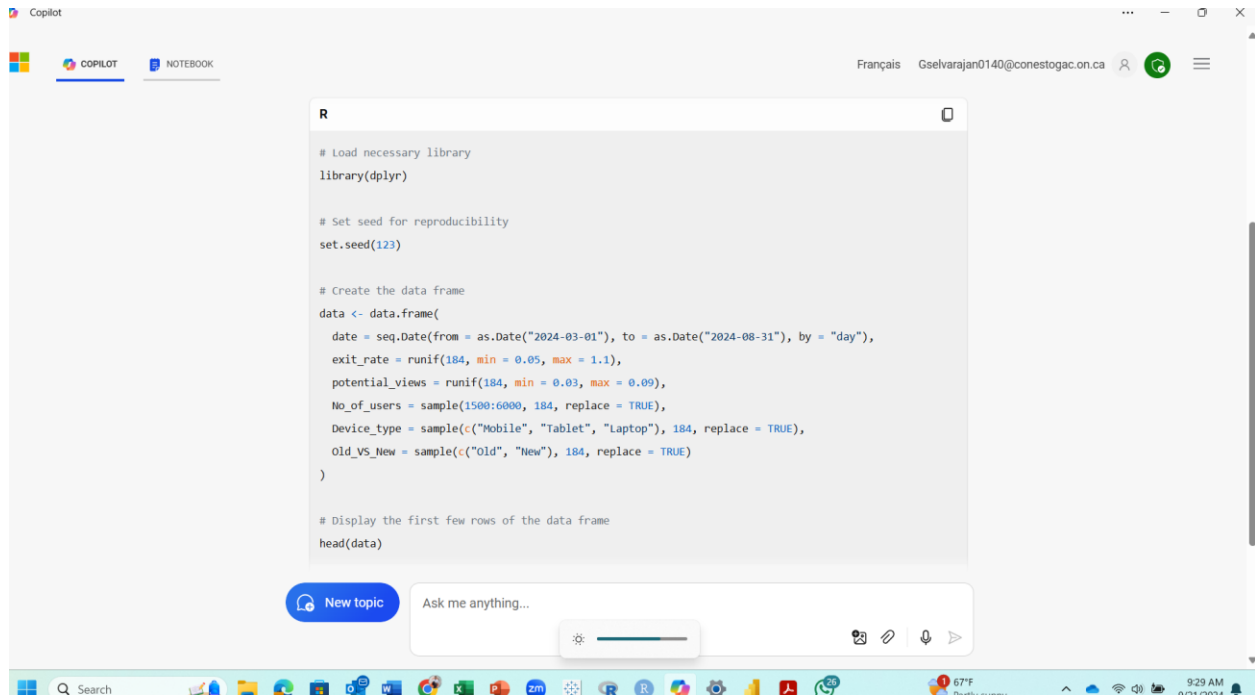
Reference:

1. Banoula, M. (2024) K-means clustering algorithm: Applications, types, and demos
[updated]: Simplilearn, Simplilearn.com. Available at:
<https://www.simplilearn.com/tutorials/machine-learning-tutorial/k-means-clustering-algorithm#:~:text=K%2DMeans%20clustering%20is%20an,objects%20belonging%20to%20another%20cluster>. (Accessed: 22 September 2024).
2. The what, why, and how of A/B and multivariate testing (no date) Oracle Canada. Available at: <https://www.oracle.com/ca-en/cx/marketing/what-is-ab-testing/#:~:text=A%2FB%20testing%E2%80%94also%20called,based%20on%20your%20key%20metrics>. (Accessed: 22 September 2024).

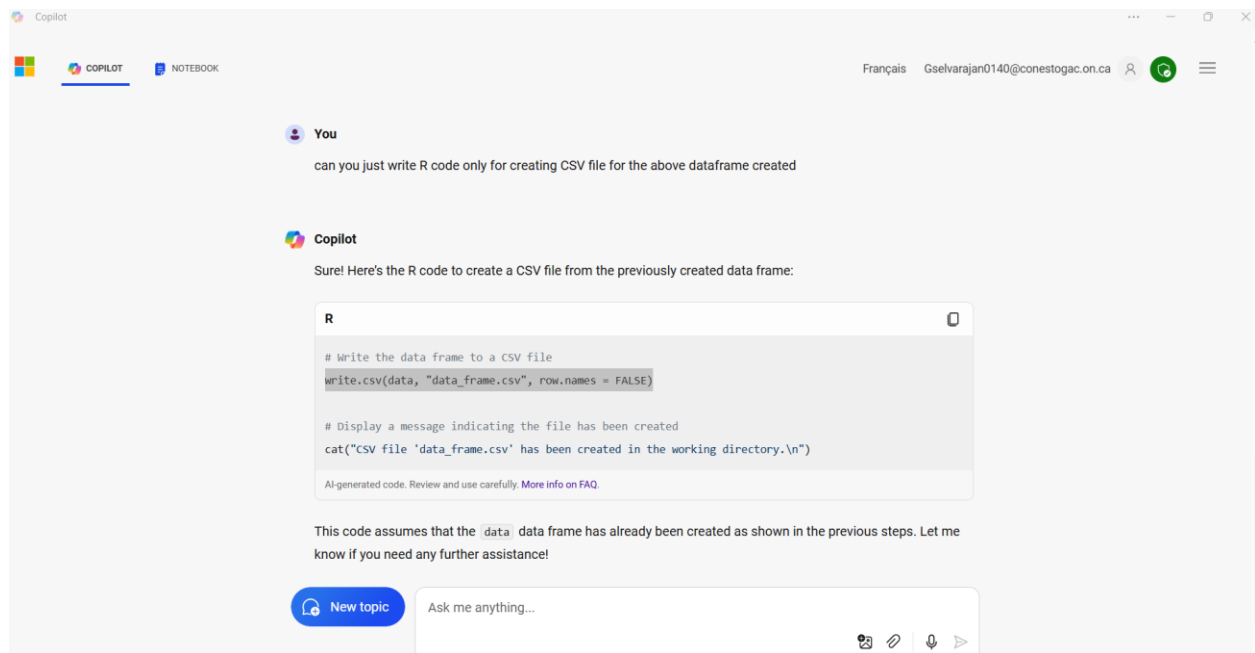
Appendix:

1. Getting R code to create Web_Source data set,





2. Getting R code to create csv file



3. Getting R code to create data frame for Social media data

The screenshot shows the Copilot chat interface. At the top, it says 'Copilot' and 'NOTEBOOK'. The user's message is: 'Can you write R code to create data frame with the below columns, social_websites = "Facebook", "X_Twitter", "LinkedIn", "Instagram", "Medium" no_of_followers = 25566, 40340, 13560, 34511, 11231 no_of_posts = 1233, 1100, 890, 991, 230 no_of_interactions = 13500, 12000, 1350, 7690, 4980 no_of_reach = 11021, 27839, 7890, 18907, 8921 no_of_shares = 750, 982, 230, 340, 120'. The Copilot response starts with 'Sure! Here's the R code to create a data frame with the specified columns and values:' and shows a code block with the following R code:

```
R
# Create the data frame
social_data <- data.frame(
  social_websites = c("Facebook", "X_Twitter", "LinkedIn", "Instagram", "Medium"),
  no_of_followers = c(25566, 40340, 13560, 34511, 11231),
```

Below the code block is a 'New topic' button and a text input field with the placeholder 'Ask me anything...'. The Windows taskbar at the bottom shows the date as 9/21/2024 and the time as 10:36 AM.

This screenshot shows the same Copilot chat interface but with the complete R code response. The code block now includes all the data and a print statement:

```
R
# Create the data frame
social_data <- data.frame(
  social_websites = c("Facebook", "X_Twitter", "LinkedIn", "Instagram", "Medium"),
  no_of_followers = c(25566, 40340, 13560, 34511, 11231),
  no_of_posts = c(1233, 1100, 890, 991, 230),
  no_of_interactions = c(13500, 12000, 1350, 7690, 4980),
  no_of_reach = c(11021, 27839, 7890, 18907, 8921),
  no_of_shares = c(750, 982, 230, 340, 120)
)

# Display the data frame
print(social_data)
```

Below the code block, it says 'AI-generated code. Review and use carefully. [More info on FAQ.](#)'. A message follows: 'This code will create a data frame named `social_data` with the specified columns and values. Let me know if you need any further assistance!'. There are icons for thumbs up, thumbs down, copy, and download, along with a status '4 of 30 responses' and a warning 'AI-generated content may be incorrect'. At the bottom, there is a 'New topic' button and a text input field with the placeholder 'Ask me anything...'. The Windows taskbar at the bottom shows the date as 9/21/2024 and the time as 10:36 AM.

4. Creating table email campaign data from R

The screenshot shows the Copilot interface with a user request and a partial R code response. The user request is as follows:

You

Can you write R code to create data frame with the below columns,
campaign_no = 1 to 6
no_of_sent_emails = 45200, 59000, 31001, 57880, 78330, 43900
generated_revenue = 8900, 15900, 12000, 11000, 15000, 9500
response_rate = randomly generate 0.31 to 0.45
Delivery_failure_rate = randomly generate 0.5 to 0.11
ignorance_rate = randomly generate 0.3 to 0.55

Copilot

Sure! Here's the R code to create a data frame with the specified columns and values:

```
R
# Set seed for reproducibility
set.seed(123)

# Create the data frame
campaign_data <- data.frame(
```

Below the code, there is a text input field with the placeholder text "Ask me anything..."

The screenshot shows the Copilot interface with the complete R code response. The code is as follows:

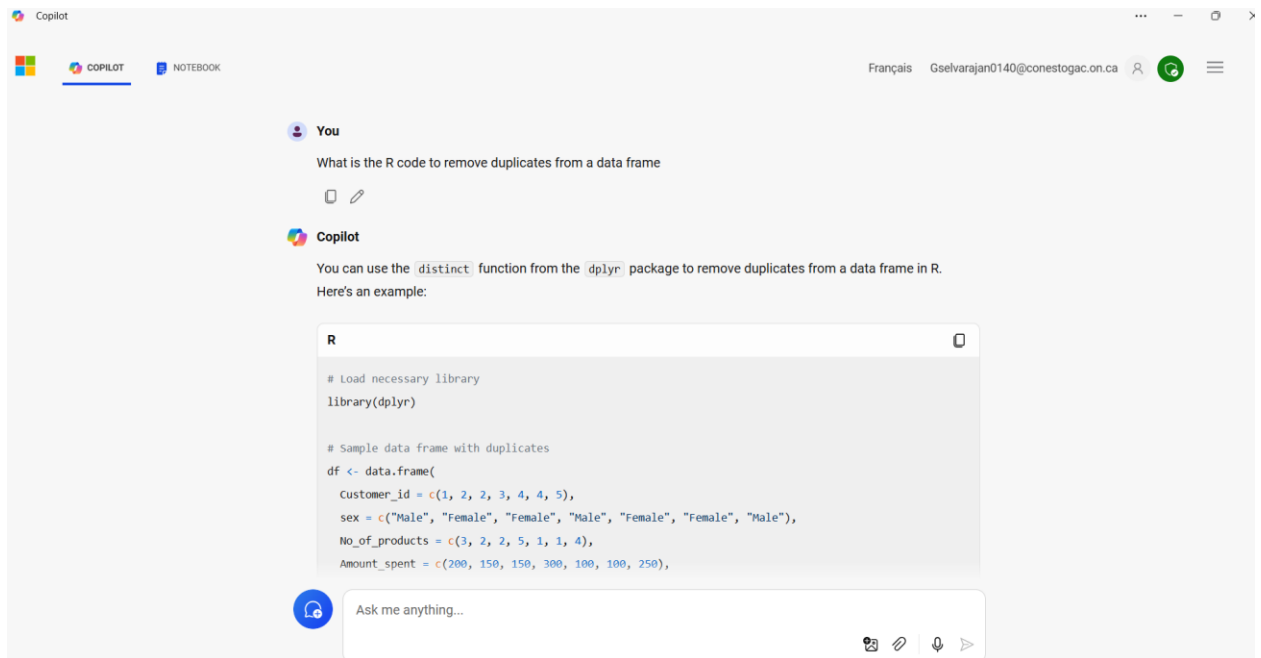
```
# Create the data frame
campaign_data <- data.frame(
  campaign_no = 1:6,
  no_of_sent_emails = c(45200, 59000, 31001, 57880, 78330, 43900),
  generated_revenue = c(8900, 15900, 12000, 11000, 15000, 9500),
  response_rate = runif(6, min = 0.31, max = 0.45),
  Delivery_failure_rate = runif(6, min = 0.05, max = 0.11),
  ignorance_rate = runif(6, min = 0.3, max = 0.55)
)

# Display the data frame
print(campaign_data)
```

Below the code, there is a text input field with the placeholder text "Ask me anything..."

This code will create a data frame named `campaign_data` with the specified columns and values, including randomly generated values for `response_rate`, `Delivery_failure_rate`, and `ignorance_rate`. Let me know if you need any further assistance!

5. Getting R code to remove any duplicates



Copilot

FRANÇAIS Gselvarajan0140@conestogac.on.ca

You

What is the R code to remove duplicates from a data frame

Copilot

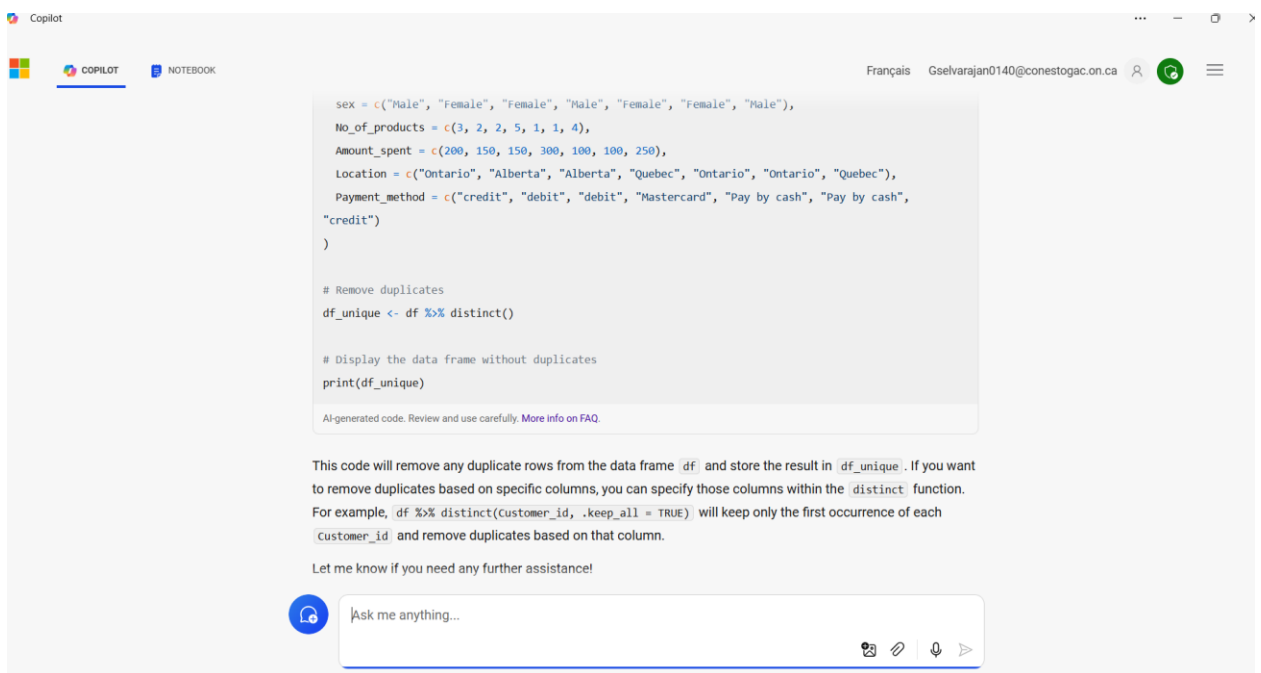
You can use the `distinct` function from the `dplyr` package to remove duplicates from a data frame in R. Here's an example:

```
R
# Load necessary library
library(dplyr)

# Sample data frame with duplicates
df <- data.frame(
  Customer_id = c(1, 2, 2, 3, 4, 4, 5),
  sex = c("Male", "Female", "Female", "Male", "Female", "Female", "Male"),
  No_of_products = c(3, 2, 2, 5, 1, 1, 4),
  Amount_spent = c(200, 150, 150, 300, 100, 100, 250),

```

Ask me anything...



```
sex = c("Male", "Female", "Female", "Male", "Female", "Female", "Male"),
No_of_products = c(3, 2, 2, 5, 1, 1, 4),
Amount_spent = c(200, 150, 150, 300, 100, 100, 250),
Location = c("Ontario", "Alberta", "Alberta", "Quebec", "Ontario", "Ontario", "Quebec"),
Payment_method = c("credit", "debit", "debit", "Mastercard", "Pay by cash", "Pay by cash",
"credit")
)

# Remove duplicates
df_unique <- df %>% distinct()

# Display the data frame without duplicates
print(df_unique)
```

AI-generated code. Review and use carefully. [More info on FAQ.](#)

This code will remove any duplicate rows from the data frame `df` and store the result in `df_unique`. If you want to remove duplicates based on specific columns, you can specify those columns within the `distinct` function. For example, `df %>% distinct(customer_id, .keep_all = TRUE)` will keep only the first occurrence of each `customer_id` and remove duplicates based on that column.

Let me know if you need any further assistance!

Ask me anything...

6. Standardizing the column names

The screenshot shows a Copilot chat window. The user asks: "what is the R code to capitalize the first letter in the column names of data frame". Copilot responds: "You can use the `tools::toTitleCase` function along with `names` to capitalize the first letter of each column name in a data frame. Here's how you can do it:".

The R code provided is as follows:

```
R
# Sample data frame
df <- data.frame(
  customer_id = 1:5,
  sex = c("Male", "Female", "Female", "Male", "Female"),
  no_of_products = c(3, 2, 2, 5, 1),
  amount_spent = c(200, 150, 150, 300, 100),
  location = c("Ontario", "Alberta", "Alberta", "Quebec", "Ontario"),
  payment_method = c("credit", "debit", "debit", "Mastercard", "Pay by cash")
)
```

Below the code, there is a "New topic" button and a text input field with the placeholder "Ask me anything...".

The second screenshot shows the same Copilot chat window with the R code updated to include the capitalization step:

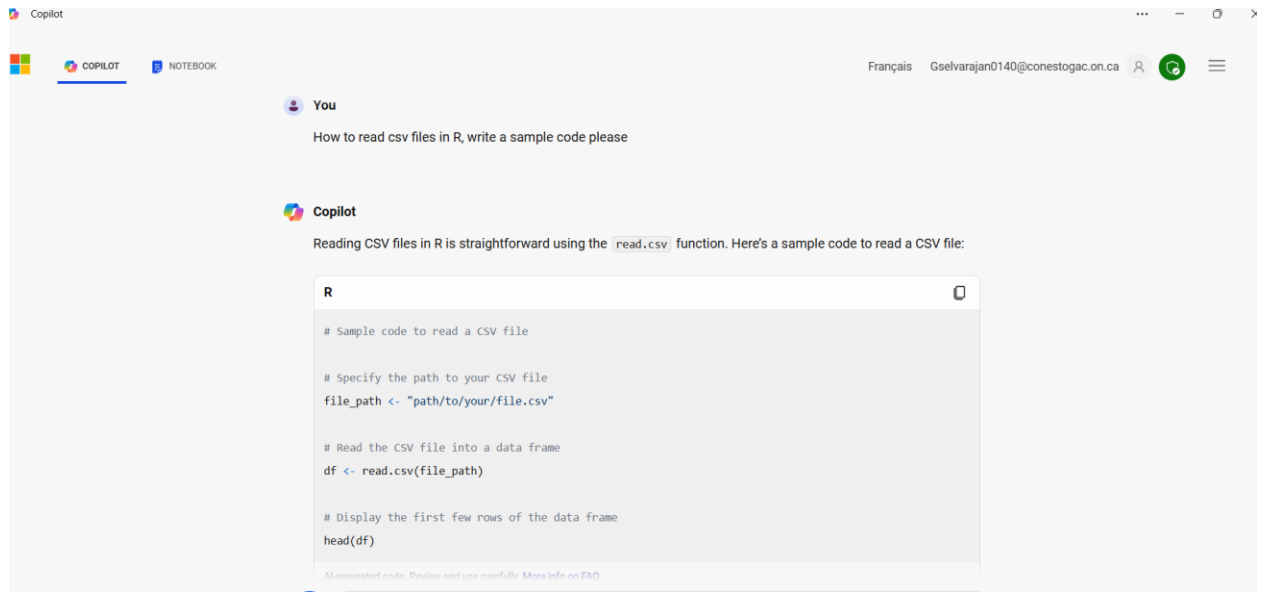
```
amount_spent = c(200, 150, 150, 300, 100),
location = c("Ontario", "Alberta", "Alberta", "Quebec", "Ontario"),
payment_method = c("credit", "debit", "debit", "Mastercard", "Pay by cash")
)

# Capitalize the first letter of each column name
names(df) <- tools::toTitleCase(names(df))

# Display the data frame with updated column names
print(df)
```

At the bottom of the code block, it says: "AI-generated code. Review and use carefully. [More info on FAQ](#)."

7. Reading csv file in R and storing it as data frames



The screenshot shows the Copilot web interface. At the top, there's a header with the Copilot logo, a language selector set to 'Français', and a user profile. Below the header, a user message asks: 'How to read csv files in R, write a sample code please'. Copilot's response follows, stating: 'Reading CSV files in R is straightforward using the `read.csv` function. Here's a sample code to read a CSV file:'. A code block is then displayed with the following R code:

```
R

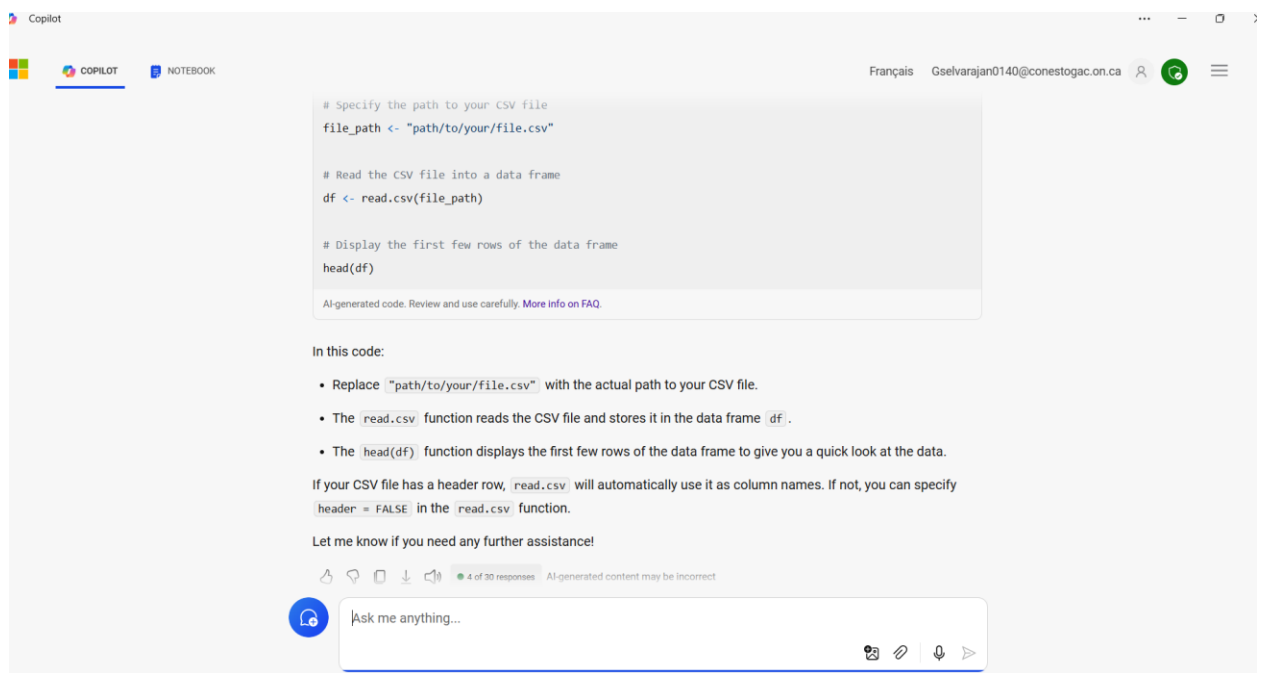
# Sample code to read a CSV file

# Specify the path to your CSV file
file_path <- "path/to/your/file.csv"

# Read the CSV file into a data frame
df <- read.csv(file_path)

# Display the first few rows of the data frame
head(df)
```

Below the code block, a small note says: 'AI-generated code. Review and use carefully. More info on FAQ.'



This screenshot shows the continuation of the Copilot interface. The same R code block is visible. Below it, the text 'In this code:' is followed by a bulleted list:

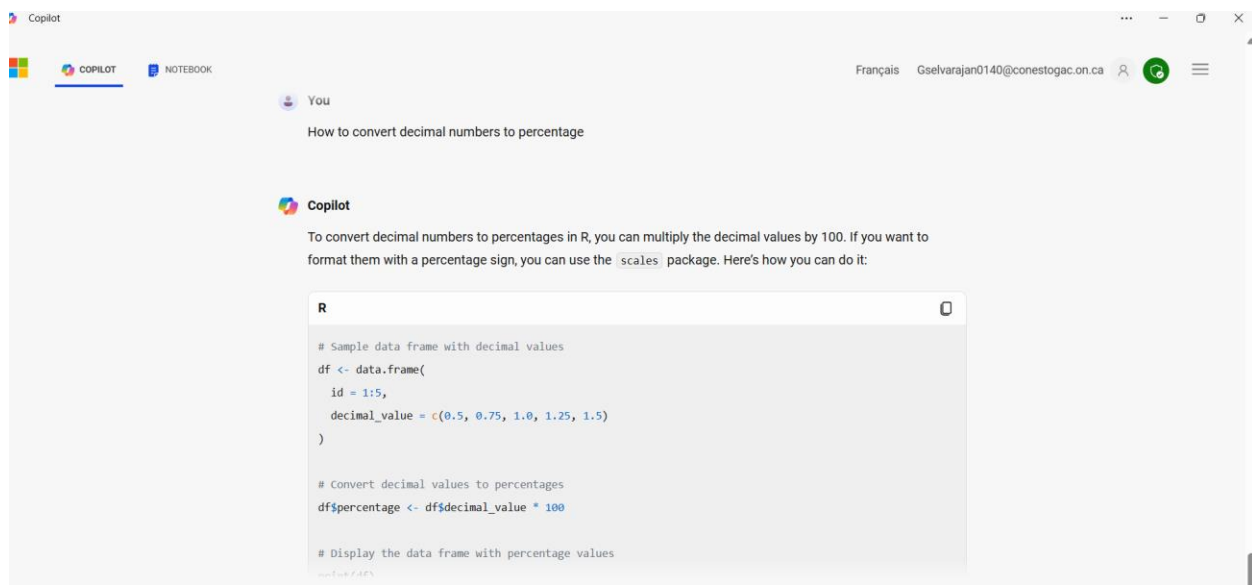
- Replace `"path/to/your/file.csv"` with the actual path to your CSV file.
- The `read.csv` function reads the CSV file and stores it in the data frame `df`.
- The `head(df)` function displays the first few rows of the data frame to give you a quick look at the data.

Below the list, it says: 'If your CSV file has a header row, `read.csv` will automatically use it as column names. If not, you can specify `header = FALSE` in the `read.csv` function.'

Then it says: 'Let me know if you need any further assistance!'

At the bottom, there's a chat input area with a microphone icon, a text box containing 'Ask me anything...', and a send button. Above the input area, there are icons for thumbs up, thumbs down, copy, and a status indicator showing '4 of 30 responses' and a warning 'AI-generated content may be incorrect'.

8. How to convert numbers to percentage



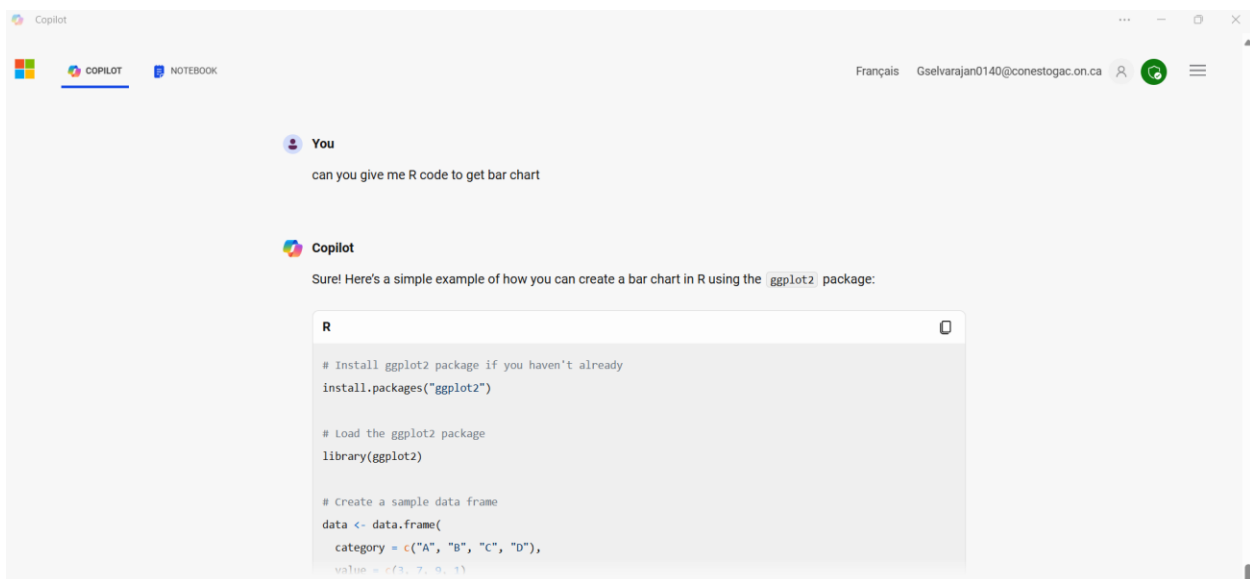
The screenshot shows a Copilot chat window. At the top, there's a header with the Copilot logo, a language selector set to 'Français', and a user profile. Below the header, the chat history shows a user asking 'How to convert decimal numbers to percentage'. Copilot responds with a text explanation: 'To convert decimal numbers to percentages in R, you can multiply the decimal values by 100. If you want to format them with a percentage sign, you can use the `scales` package. Here's how you can do it:'. Below the text, there's a code block titled 'R' containing the following R code:

```
R
# Sample data frame with decimal values
df <- data.frame(
  id = 1:5,
  decimal_value = c(0.5, 0.75, 1.0, 1.25, 1.5)
)

# Convert decimal values to percentages
df$percentage <- df$decimal_value * 100

# Display the data frame with percentage values
df
```

9. Getting sample R code for creating bar chart



The screenshot shows a Copilot chat window. At the top, there's a header with the Copilot logo, a language selector set to 'Français', and a user profile. Below the header, the chat history shows a user asking 'can you give me R code to get bar chart'. Copilot responds with a text explanation: 'Sure! Here's a simple example of how you can create a bar chart in R using the `ggplot2` package:'. Below the text, there's a code block titled 'R' containing the following R code:

```
R
# Install ggplot2 package if you haven't already
install.packages("ggplot2")

# Load the ggplot2 package
library(ggplot2)

# Create a sample data frame
data <- data.frame(
  category = c("A", "B", "C", "D"),
  value = c(3, 7, 0, 1)
```

Copilot

Fransais Gselvarajan0140@conestogac.on.ca

```
# Create a sample data frame
data <- data.frame(
  category = c("A", "B", "C", "D"),
  value = c(3, 7, 9, 1)
)

# Create a bar chart
ggplot(data, aes(x = category, y = value)) +
  geom_bar(stat = "identity") +
  theme_minimal() +
  labs(title = "Sample Bar Chart", x = "Category", y = "Value")
```

AI-generated code. Review and use carefully. [More info on FAQ](#).

This code will generate a bar chart with categories A, B, C, and D on the x-axis and their corresponding values on the y-axis. You can customize it further by modifying the `data` frame or adding more `ggplot2` functions.

Feel free to ask if you need any more help with R or data visualization!

8 of 30 responses AI-generated content may be incorrect

Can you show me how to create a stacked bar chart? What other types of plots can I create in R?

New topic Ask me anything...

10. Getting sample code for boxplot

Copilot

Fransais Gselvarajan0140@conestogac.on.ca

You

can you give me the sample code for boxplot

Copilot

Absolutely! Here's a simple example of how you can create a boxplot in R using the `ggplot2` package:

```
R

# Install ggplot2 package if you haven't already
install.packages("ggplot2")

# Load the ggplot2 package
library(ggplot2)

# Create a sample data frame
data <- data.frame(
  category = c("A", "A", "A", "B", "B", "B", "C", "C", "C"),
  value = c(1, 2, 3, 4, 5, 6, 7, 8, 9)
```

Ask me anything...

Copilot

FRANÇAIS Gselvarajan0140@conestogac.on.ca

```
# Create a sample data frame
data <- data.frame(
  category = c("A", "A", "A", "B", "B", "B", "C", "C", "C"),
  value = c(3, 5, 2, 7, 8, 6, 9, 10, 8)
)

# Create a boxplot
ggplot(data, aes(x = category, y = value)) +
  geom_boxplot() +
  theme_minimal() +
  labs(title = "Sample Boxplot", x = "category", y = "value")
```

AI-generated code. Review and use carefully. [More info on FAQ.](#)

This code will generate a boxplot with categories A, B, and C on the x-axis and their corresponding values on the y-axis. You can customize it further by modifying the `data` frame or adding more `ggplot2` functions.

If you have any specific requirements or need further assistance, feel free to ask!

9 of 30 responses AI-generated content may be incorrect

Can you show me how to create a violin plot? What other types of plots can I create in R?

New topic Ask me anything...

11. Getting R code for Kmeans clustering

Copilot

FRANÇAIS Gselvarajan0140@conestogac.on.ca

You

Can you let me know to write codes for Kmeans clustering

Copilot

Certainly! Here's a simple example of how you can perform K-means clustering in R:

```
R

# Install ggplot2 package if you haven't already
install.packages("ggplot2")

# Load the ggplot2 package
library(ggplot2)

# Create a sample data frame
data <- data.frame(
  x = c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10),
  y = c(2, 1, 4, 3, 5, 6, 8, 7, 9, 10)
)
```

New topic Ask me anything...

Copilot

COPILOT NOTEBOOK

Français Gselvarajan0140@conestogac.on.ca

```
# Perform K-means clustering
set.seed(123) # Set seed for reproducibility
kmeans_result <- kmeans(data, centers = 3)

# Add cluster assignment to the data frame
data$cluster <- as.factor(kmeans_result$cluster)

# Plot the clusters
ggplot(data, aes(x = x, y = y, color = cluster)) +
  geom_point(size = 3) +
  theme_minimal() +
  labs(title = "K-means Clustering", x = "X-axis", y = "Y-axis")
```

AI-generated code. Review and use carefully. [More info on FAQ.](#)

This code will:

1. Create a sample data frame with two variables, `x` and `y`.
2. Perform K-means clustering with 3 clusters.
3. Add the cluster assignments to the data frame.

New topic Ask me anything...

12. Getting R code for linear model

Copilot

COPILOT NOTEBOOK

Français Gselvarajan0140@conestogac.on.ca

You

Can you help me to write R code for a linear regression model?

Copilot

Of course! Here's a simple example of how you can create a linear regression model in R:

```
R

# Create a sample data frame
data <- data.frame(
  x = c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10),
  y = c(2, 3, 5, 7, 11, 13, 17, 19, 23, 29)
)

# Fit a linear regression model
model <- lm(y ~ x, data = data)

# Display the summary of the model
summary(model)
```

Ask me anything...

The screenshot shows the Microsoft Copilot web interface. At the top, there's a header with the Copilot logo, a language selector set to 'Français', and a user profile 'Gselvarajan0140@conestogac.on.ca'. Below the header, the main area displays R code for plotting data and a regression line:

```
# Plot the data and the regression line
plot(data$x, data$y, main = "Linear Regression", xlab = "X", ylab = "Y", pch = 19)
abline(model, col = "blue")
```

Below the code, it says 'AI-generated code. Review and use carefully. [More info on FAQ.](#)'

Then, it states 'This code will:' followed by a numbered list:

1. Create a sample data frame with two variables, `x` and `y`.
2. Fit a linear regression model where `y` is predicted by `x`.
3. Display the summary of the model, which includes coefficients, R-squared value, and other statistics.
4. Plot the data points and the regression line.

Below the list, it says: 'You can replace the sample data with your own dataset by modifying the `data` frame. If you have any specific requirements or need further assistance, feel free to ask!'

At the bottom, there are three suggested prompts in rounded rectangles:

- Can you show me how to perform multiple linear regression?
- What other types of regression models are available in R?
- How do I interpret the coefficients of a linear regression model?

At the very bottom, there's a 'New topic' button and a text input field 'Ask me anything...'.

13. What is A/B test in R?

The screenshot shows the Microsoft Copilot web interface. At the top, there's a header with the Copilot logo, a language selector set to 'Français', and a user profile 'Gselvarajan0140@conestogac.on.ca'. Below the header, the main area shows a conversation:

You
What is A/B test in R?

Copilot
An A/B test, also known as split testing, is a method used to compare two versions of a variable to determine which one performs better. In R, you can conduct an A/B test by comparing the means of two groups using statistical tests like the t-test.

Here's a simple example of how to perform an A/B test in R:

```
R
# Create sample data for two groups (A and B)
set.seed(123) # Set seed for reproducibility
group_A <- rnorm(50, mean = 5, sd = 2) # Group A with mean 5 and standard deviation 2
group_B <- rnorm(50, mean = 6, sd = 2) # Group B with mean 6 and standard deviation 2

# Perform a t-test to compare the means of the two groups
t_test_result <- t.test(group_A, group_B)
```

At the bottom, there's a 'New topic' button and a text input field 'Ask me anything...'.

Copilot

COPILOT NOTEBOOK Français Gselvarajan0140@conestogac.on.ca

```
# Display the results of the t-test
print(t_test_result)
```

AI-generated code. Review and use carefully. [More info on FAQ.](#)

This code will:

1. Create two sample groups, `group_A` and `group_B`, with different means.
2. Perform a t-test to compare the means of the two groups.
3. Display the results of the t-test, including the p-value and confidence interval.

The p-value will help you determine if there is a statistically significant difference between the two groups. If the p-value is less than your chosen significance level (commonly 0.05), you can conclude that there is a significant difference between the groups.

If you have specific data or need further assistance with A/B testing in R, feel free to ask!

👍 🗑️ ⬇️ 🔊 12 of 30 responses AI-generated content may be incorrect

Can you show me how to visualize A/B test results? What other statistical tests can I use for A/B testing? How do I calculate the effect size in an A/B test?

New topic Ask me anything...

14. R code for A/B testing,

Copilot

COPILOT NOTEBOOK Français Gselvarajan0140@conestogac.on.ca

Can you write R code for A/B testing for two Email campaigns to see which is the winning campaign based on the revenue generated

Copilot


Sure! Here's an example of how you can perform an A/B test in R to compare the revenue generated by two email campaigns:

```
R
# Create sample data for two email campaigns (A and B)
set.seed(123) # Set seed for reproducibility
campaign_A <- rnorm(100, mean = 50, sd = 10) # Campaign A with mean revenue of 50 and standard deviation of 10
campaign_B <- rnorm(100, mean = 55, sd = 10) # Campaign B with mean revenue of 55 and standard deviation of 10

# Perform a t-test to compare the means of the two campaigns
t_test_result <- t.test(campaign_A, campaign_B)
```

New topic Ask me anything...

Copilot

 COPILOT

NOTEBOOK

Français Gselvarajan0140@conestogac.on.ca

```
ggplot(data, aes(x = Campaign, y = revenue, fill = Campaign)) +
  geom_boxplot() +
  theme_minimal() +
  labs(title = "Revenue Comparison of Email Campaigns", x = "Campaign", y = "Revenue")
```






AI-generated code. Review and use carefully. [More info on FAQ.](#)

This code will:

1. Create sample data for two email campaigns, `campaign_A` and `campaign_B`, with different mean revenues.
2. Perform a t-test to compare the mean revenues of the two campaigns.
3. Display the results of the t-test, including the p-value and confidence interval.
4. Visualize the revenue distributions of the two campaigns using a boxplot.

The p-value from the t-test will help you determine if there is a statistically significant difference in the revenue generated by the two campaigns. If the p-value is less than your chosen significance level (commonly 0.05), you can conclude that there is a significant difference between the campaigns.

Feel free to modify the sample data with your actual revenue data from the email campaigns. If you need further assistance, just let me know!


    

13 of 30 responses AI-generated content may be incorrect

New topic

Ask me anything...

Copilot

 COPILOT

NOTEBOOK

Français Gselvarajan0140@conestogac.on.ca

```
# Display the results of the t-test
print(t_test_result)

# Visualize the revenue distributions of the two campaigns
data <- data.frame(
  revenue = c(campaign_A, campaign_B),
  campaign = rep(c("A", "B"), each = 100)
)

ggplot(data, aes(x = campaign, y = revenue, fill = campaign)) +
  geom_boxplot() +
  theme_minimal() +
  labs(title = "Revenue Comparison of Email Campaigns", x = "Campaign", y = "Revenue")
```

AI-generated code. Review and use carefully. [More info on FAQ.](#)

This code will:

1. Create sample data for two email campaigns, `campaign_A` and `campaign_B`, with different mean revenues.
2. Perform a t-test to compare the mean revenues of the two campaigns.

New topic

Ask me anything...

15. What is shiny apps in R

Copilot


Franglais Gselvarajan0140@conestogac.on.ca

You

What is shiny apps in R

Copilot

Shiny is an R package that allows you to build interactive web applications directly from R. Here are some key points about Shiny apps:



1. **Interactive Web Applications:** Shiny apps are web pages that can interact with R code running on a server. This means users can manipulate the app's interface, and the server will update the display in real-time based on user inputs ¹.

2. **Components:** A Shiny app typically consists of two main components:

- **UI (User Interface):** This defines the layout and appearance of the app. It can be built using R functions or directly with HTML, CSS, and JavaScript for more flexibility.
- **Server:** This contains the instructions needed to build the app's outputs. It handles the reactive programming that makes Shiny apps interactive ¹.

[Stop Responding](#)

[New topic](#) Ask me anything...

Copilot

Franglais Gselvarajan0140@conestogac.on.ca

- **Server:** This contains the instructions needed to build the app's outputs. It handles the reactive programming that makes Shiny apps interactive ¹.

3. **Reactive Programming:** Shiny uses a reactive programming model, which means that outputs automatically update when inputs change. This is similar to how spreadsheets work, where changing a cell value updates related cells instantly ².

4. **Examples and Use Cases:** Shiny is used for a wide range of applications, from simple data visualizations to complex dashboards and interactive reports. The Shiny Gallery showcases various examples and demos to help you get started ³.

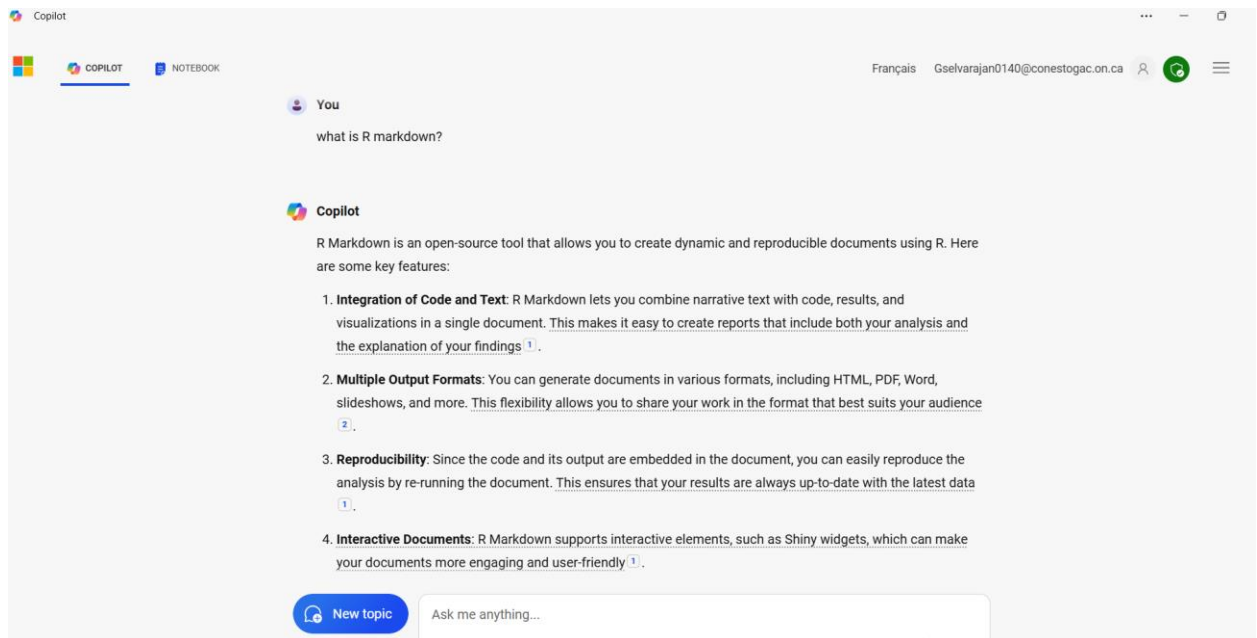
5. **Installation and Setup:** To get started with Shiny, you need to install the package in R using `install.packages("shiny")`. You can then create a new Shiny app by defining the UI and server components and running the app with `shinyApp(ui, server)` ¹.

If you're interested in learning more or seeing some examples, you can check out the [Shiny Gallery](#) for inspiration ³.

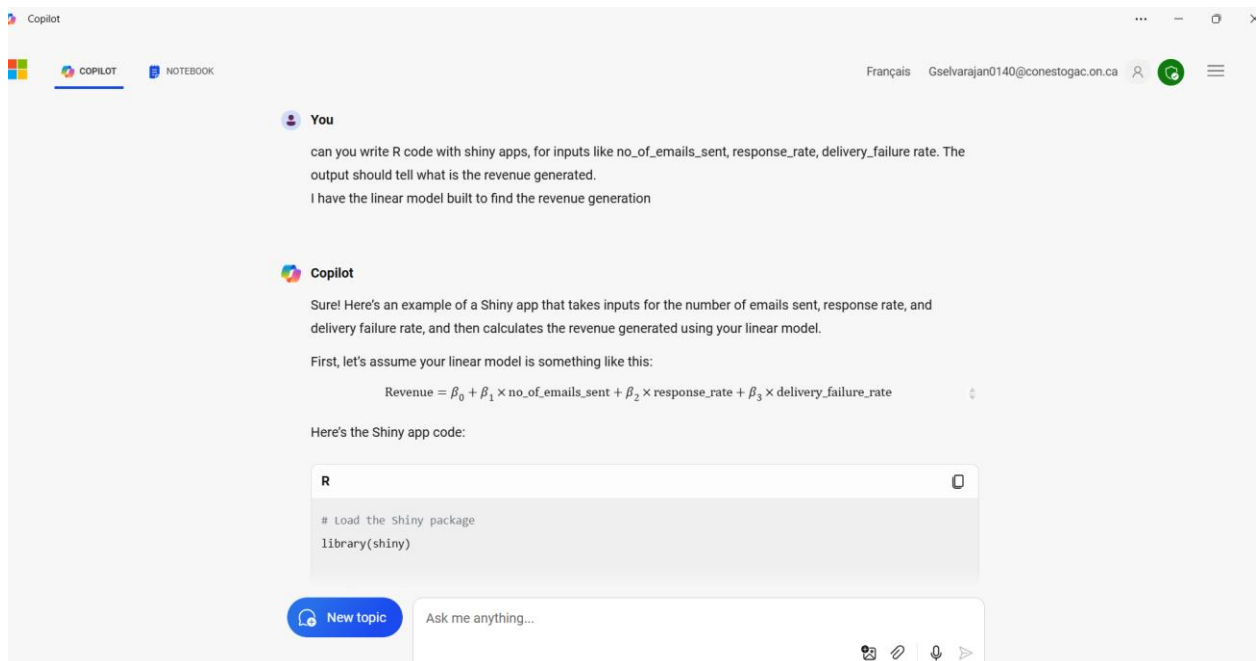
Would you like to know how to create a simple Shiny app, or do you have a specific project in mind?

[New topic](#) Ask me anything...

16. What is R markdown?



17. Getting R code for shiny apps



Copilot

COPILOT NOTEBOOK

Français Gselvarajan0140@conestogac.on.ca

```
# Define UI for the app
ui <- fluidPage(
  titlePanel("Revenue Prediction"),

  sidebarLayout(
    sidebarPanel(
      numericInput("emails", "Number of Emails Sent:", value = 1000, min = 0),
      numericInput("response", "Response Rate (%)", value = 10, min = 0, max = 100),
      numericInput("failure", "Delivery Failure Rate (%)", value = 5, min = 0, max = 100),
      actionButton("predict", "Predict Revenue")
    ),

    mainPanel(
      textOutput("revenue")
    )
  )
)

# Define server logic for the app
server <- function(input, output) {
```

New topic Ask me anything...

Copilot

COPILOT NOTEBOOK

Français Gselvarajan0140@conestogac.on.ca

```
# Define the coefficients of the linear model
beta_0 <- 1000 # Intercept
beta_1 <- 2 # Coefficient for no_of_emails_sent
beta_2 <- 50 # Coefficient for response_rate
beta_3 <- -30 # Coefficient for delivery_failure_rate

observeEvent(input$predict, {
  # Calculate the revenue based on the inputs and the linear model
  revenue <- beta_0 + beta_1 * input$emails + beta_2 * (input$response / 100) + beta_3 *
(input$failure / 100)

  # Display the revenue
  output$revenue <- renderText({
    paste("Estimated Revenue: $", round(revenue, 2))
  })
})

# Run the application
shinyApp(ui = ui, server = server)
```

81 generated code. Download and use mindfully. More info on ESO

Copilot

COPILOT NOTEBOOK Français Gselvarajan0140@conestogac.on.ca

```

    })
  }

# Run the application
shinyApp(ui = ui, server = server)

```

AI-generated code. Review and use carefully. [More info on FAQ.](#)

In this example:

- The `ui` function defines the user interface, including input fields for the number of emails sent, response rate, and delivery failure rate, and a button to trigger the prediction.
- The `server` function contains the logic to calculate the revenue based on the inputs and the linear model coefficients.
- The `shinyApp` function runs the app.

You can adjust the coefficients (`beta_0`, `beta_1`, `beta_2`, `beta_3`) to match your actual linear model. When you run this app, it will allow you to input the values and see the predicted revenue.

Would you like any further customization or additional features for this app?

👍 🗑️ ⬇️ 🔊 16 of 30 responses

Copilot


COPILOT NOTEBOOK Français Gselvarajan0140@conestogac.on.ca

You

What is shiny apps in R

Copilot

Shiny is an R package that allows you to build interactive web applications directly from R. Here are some key points about Shiny apps:



Explore

- 1. Interactive Web Applications:** Shiny apps are web pages that can interact with R code running on a server. This means users can manipulate the app's interface, and the server will update the display in real-time based on user inputs ¹.
- 2. Components:** A Shiny app typically consists of two main components:
 - **UI (User Interface):** This defines the layout and appearance of the app. It can be built using R functions or directly with HTML, CSS, and JavaScript for more flexibility.
 - **Server:** This contains the instructions needed to build the app's

Ask me anything...

🗑️ 🔗 🔊 ➡️

COPILLOT

NOTEBOOK

FrançaisGselvarajan0140@conestogac.on.ca

CSS, and JavaScript for more flexibility.


- **Server:** This contains the instructions needed to build the app's outputs. It handles the reactive programming that makes Shiny apps interactive ¹.

3. **Reactive Programming:** Shiny uses a reactive programming model, which means that outputs automatically update when inputs change. This is similar to how spreadsheets work, where changing a cell value updates related cells instantly ².

4. **Examples and Use Cases:** Shiny is used for a wide range of applications, from simple data visualizations to complex dashboards and interactive reports. The Shiny Gallery showcases various examples and demos to help you get started ³.

5. **Installation and Setup:** To get started with Shiny, you need to install the package in R using `install.packages("shiny")`. You can then [create a new Shiny app](#) and [run it](#).

If you're interested in learning more about Shiny, you can visit the [Shiny website](#).

 Shiny - Welcome to Shiny

<https://shiny.posit.co/r/getstarted/shiny-basics/lesson1/index.html>

Shiny is an R package that makes it easy to build interactive web applications (apps) straight from R.

COPILLOT

NOTEBOOK

FrançaisGselvarajan0140@conestogac.on.ca

CSS, and JavaScript for more flexibility.


- **Server:** This contains the instructions needed to build the app's outputs. It handles the reactive programming that makes Shiny apps interactive ¹.

3. **Reactive Programming:** Shiny uses a reactive programming model, which means that outputs automatically update when inputs change. This is similar to how spreadsheets work, where changing a cell value updates related cells instantly ².

4. **Examples and Use Cases:** Shiny is used for a wide range of applications, from simple data visualizations to complex dashboards and interactive reports. The Shiny Gallery showcases various examples and demos to help you get started ³.

5. **Installation and Setup:** To get started with Shiny, you need to install the package in R using `install.packages("shiny")`. You can then [create a new Shiny app](#) and [run it](#).

If you're interested in learning more about Shiny, you can visit the [Shiny website](#).

 Shiny - Welcome to Shiny

<https://shiny.posit.co/r/getstarted/shiny-basics/lesson1/index.html>

Shiny is an R package that makes it easy to build interactive web applications (apps) straight from R.

Copilot

FRANÇAIS Gselvarajan0140@conestogac.on.ca

5. **Ease of Use:** Writing in R Markdown is straightforward. You use a simple plain text format called Markdown, which is easy to learn and use. You can include R code chunks within the Markdown text, and these chunks will be executed when you "knit" the document [\[1\]](#).

Here's a basic example of an R Markdown document:

```

---
title: "My First R Markdown Document"
output: html_document
---

## Introduction

This is an R Markdown document. Below is an example of an R code chunk:

```{r}
summary(cars)

```

You can also include plots:

Copilot

FRANÇAIS Gselvarajan0140@conestogac.on.ca

**You**

can you write R code with shiny apps, for inputs like no\_of\_emails\_sent, response\_rate, delivery\_failure rate. The output should tell what is the revenue generated. I have the linear model built to find the revenue generation

**Copilot**

Sure! Here's an example of a Shiny app that takes inputs for the number of emails sent, response rate, and delivery failure rate, and then calculates the revenue generated using your linear model.

First, let's assume your linear model is something like this:

$$\text{revenue} = p_0 + p_1 \times \text{no\_of\_emails\_sent} + p_2 \times \text{response\_rate} + p_3 \times \text{delivery\_failure\_rate}$$

Here's the Shiny app code:

```

R
Load the shiny package
library(shiny)

```

[New topic](#) Ask me anything...

Copilot

COPILOT NOTEBOOK

Français Gselvarajan0140@conestogac.on.ca

```
Define UI for the app
ui <- fluidPage(
 titlePanel("Revenue Prediction"),

 sidebarLayout(
 sidebarPanel(
 numericInput("emails", "Number of Emails Sent:", value = 1000, min = 0),
 numericInput("response", "Response Rate (%):", value = 10, min = 0, max = 100),
 numericInput("failure", "Delivery Failure Rate (%):", value = 5, min = 0, max = 100),
 actionButton("predict", "Predict Revenue")
),

 mainPanel(
 textOutput("revenue")
)
)
)

Define server logic for the app
server <- function(input, output) {
 # Define the coefficients of the linear model
```

New topic Ask me anything...

Copilot

COPILOT NOTEBOOK

Français Gselvarajan0140@conestogac.on.ca

```
server <- function(input, output) {
 # Define the coefficients of the linear model
 beta_0 <- 1000 # Intercept
 beta_1 <- 2 # Coefficient for no_of_emails_sent
 beta_2 <- 50 # Coefficient for response_rate
 beta_3 <- -30 # Coefficient for delivery_failure_rate

 observeEvent(input$predict, {
 # Calculate the revenue based on the inputs and the linear model
 revenue <- beta_0 + beta_1 * input$emails + beta_2 * (input$response / 100) + beta_3 *
 (input$failure / 100)

 # Display the revenue
 output$revenue <- renderText({
 paste("Estimated Revenue: $", round(revenue, 2))
 })
 })
}

Run the application
shinyApp(ui = ui, server = server)
```

New topic Ask me anything...

```

 })
 })
}

Run the application
shinyApp(ui = ui, server = server)

```

AI-generated code. Review and use carefully. [More info on FAQ](#).

In this example:

- The `ui` function defines the user interface, including input fields for the number of emails sent, response rate, and delivery failure rate, and a button to trigger the prediction.
- The `server` function contains the logic to calculate the revenue based on the inputs and the linear model coefficients.
- The `shinyApp` function runs the app.

You can adjust the coefficients ( `beta_0` , `beta_1` , `beta_2` , `beta_3` ) to match your actual linear model. When you run this app, it will allow you to input the values and see the predicted revenue.

Would you like any further customization or additional features for this app?