Challenge-4

Chen Zi Xin

2023-09-04

```
knitr::opts_chunk$set(echo = FALSE)
```

Questions

Load the "CommQuest2023.csv" dataset using the `read_csv()` command and assign it to a variable named "comm_data."

```
library(tidyverse)
comm_data <- read.csv("CommQuest2023_Larger.csv")
comm_data
```

Question-1: Communication Chronicles Using the select command, create a new dataframe containing only the "date," "channel," and "message" columns from the "comm_data" dataset.

Solution:

```
comm_data %>% select(date,channel,message)
```

Question-2: Channel Selection Use the filter command to create a new dataframe that includes messages sent through the "Twitter" channel on August 2nd.

Solution:

```
comm_data %>% filter(channel == "Twitter") %>% filter(date == "2023-08-02")
```

Question-3: Chronological Order Utilizing the arrange command, arrange the "comm_data" dataframe in ascending order based on the "date" column.

Solution:

```
comm_data %>% arrange(date)
```

Question-4: Distinct Discovery Apply the distinct command to find the unique senders in the "comm_data" dataframe.

Solution:

```
comm_data %>% distinct(sender)
```

Question-5: Sender Stats Employ the count and group_by commands to generate a summary table that shows the count of messages sent by each sender in the "comm_data" dataframe.

Solution:

```
comm_data %>% group_by(sender) %>% count(message)
```

Question-6: Channel Chatter Insights Using the group_by and count commands, create a summary table that displays the count of messages sent through each communication channel in the "comm_data" dataframe.

Solution:

```
comm_data %>% group_by(channel) %>% count(message)
```

Question-7: Positive Pioneers Utilize the filter, select, and arrange commands to identify the top three senders with the highest average positive sentiment scores. Display their usernames and corresponding sentiment averages.

Solution:

```
comm_data%>% group_by(sender)%>% summarise(mean_sentiment = mean(sentiment)) %>% arrange(desc(mean_senti
```

Question-8: Message Mood Over Time With the group_by, summarise, and arrange commands, calculate the average sentiment score for each day in the "comm_data" dataframe.

Solution:

```
comm_data %>% group_by(date) %>% summarise(avg_sentiment = mean(sentiment))
```

Question-9: Selective Sentiments Use the filter and select commands to extract messages with a negative sentiment score (less than 0) and create a new dataframe.

Solution:

```
negative_sentiment <- comm_data %>% filter(sentiment < 0) %>% select(message,sentiment)
negative_sentiment
```

Question-10: Enhancing Engagement Apply the mutate command to add a new column to the "comm_data" dataframe, representing a sentiment label: "Positive," "Neutral," or "Negative," based on the sentiment score.

Solution:

```
comm_data %>%
  mutate(level_of_sentiment= case_when(sentiment >0 ~ "Positive", sentiment == 0 ~ "Neutral" , sentimen
```

Question-11: Message Impact Create a new dataframe using the mutate and arrange commands that calculates the product of the sentiment score and the length of each message. Arrange the results in descending order.

Solution:

```
New_dataframe <- comm_data %>% mutate(new_sentiment_score= sentiment*nchar(message)) %>% arrange(desc(n

New_dataframe
```

Question-12: Daily Message Challenge Use the group_by, summarise, and arrange commands to find the day with the highest total number of characters sent across all messages in the "comm_data" dataframe.

Solution:

```
comm_data %>% group_by(date) %>% summarise(sum(nchar(message))) %>% arrange(date)
```

Question-13: Untidy data Can you list at least two reasons why the dataset illustrated in slide 10 is non-tidy? How can it be made Tidy?

Solution: It has multiple variables and the observations are in columns. It can have a standard structure and the codes can be further simplified.