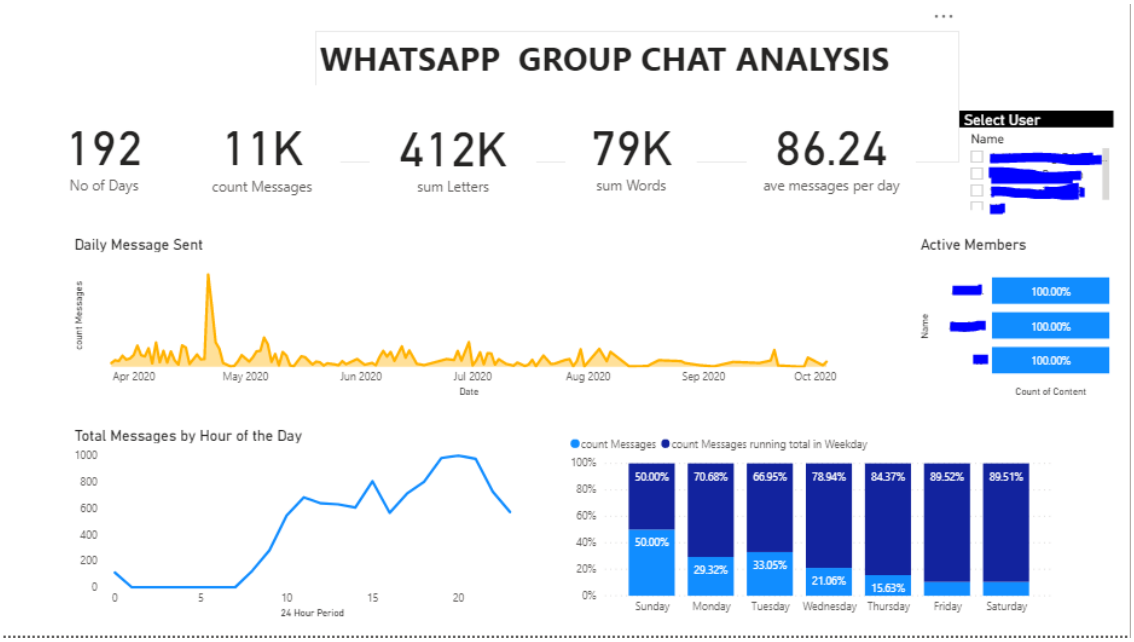
WhatsApp group chat analysis using Python & PowerBI



Introduction:

Nowadays, WhatsApp became the preferred communication mode, which allows us to create groups and share our thought. Sometimes, we might be in a group for a long time as a silent member. Similarly, there could be many other members who don't contribute. This analysis helps us to get insight into the numbers, active members, and the message distribution in a week. In order to achieve this, I have used Python for data preparation and PowerBI to create the visualization.

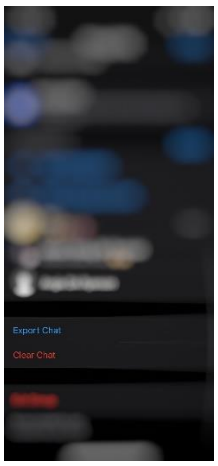
Data Preparation:

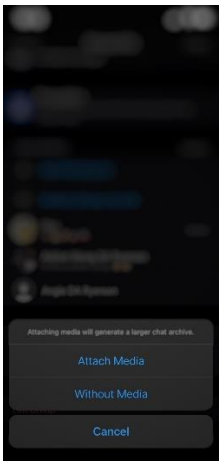
Our WhatsApp data can be used to perform this analysis. Below sections explains the steps to export chat and prepare the data for visualization.

Exporting chat:

Depends on the device (IOS or Android) the export option varies. Based on the available option, "export chat" will allow us to get the chat history in a text file. Make sure to select "without media" while exporting chat. Below screenshot is taken from IOS device,

Vijay Jayachandran





Cleaning data:

First the data (text file) is loaded into the Python environment. Code used to perform this operation is mentioned below,

```
import pandas as pd
import numpy as np
import re
import dateparser
from collections import Counter

Groupchat = read_file('Groupchat.txt')
```

Once the data is loaded into Python, below shared sequence of code will be executed to clean the chat contents.

```
 #Remove new lines
Groupchat = [line.strip() for line in Groupchat]

#Cleaning the notification for newly joined members
clean_Groupchat = [line for line in Groupchat if not "joined using this" in line]

#Similarly cleaning the notification for the members who left the group
clean_Groupchat = [line for line in clean_Groupchat if not line.endswith("left")]
```

Creating consistent data for visualization:

This is an important step in all analysis process, as we need to prepare the consistent data which can be used for visualization. So, the date, time, name and content from the text file are extracted using the below syntaxes.

```
#Extracting Date, Time and Name from the chat text
GroupChatdate = [clean_Groupchat[i].split('[')[1].split(',')[0] for i in range(len(clean_Groupchat))]

GroupChattime = [clean_Groupchat[i].split(',')[1].split(']')[0] for i in range(len(clean_Groupchat))]
GroupChattime = [s.strip(' ') for s in GroupChattime]

GroupChatname = [clean_Groupchat[i].split(']')[1].split(':')[0] for i in range(len(clean_Groupchat))]
```

There could be some missing contents in the chat while extracting, so we should replace the text with the “missing text” value, which further will be removed in the coming steps.

```
#Code to append the missing text
GroupChatcontent = []
for i in range(len(clean_Groupchat)):
  try:
    GroupChatcontent.append(clean_Groupchat[i].split(']')[1].split(':')[1])
  except IndexError:
    GroupChatcontent.append('Missing Text')
```

A data frame is created to hold these extracted values in order to export them into csv for visualization.

```
#Creating a dataframe to finaly export to csv for visualization
df_GroupChat = pd.DataFrame(list(zip(GroupChatdate, GroupChattime, GroupChatname, GroupChatcontent)),
                            columns = ['Date', 'Time', 'Name', 'Content'])
```

As mentioned, while creating the “missing text” filed, below code is executed to remove those as they will not have impact to the analysis.

```
#Removing the missing text details
df_GroupChat = df_GroupChat[df_GroupChat["Content"]!='Missing Text']
df_GroupChat.reset_index(inplace=True, drop=True)
```

New fields are created in the data frame to hold the Date time, weekday, letter, and word count and hours. These are required for data visualization.

```
#Combining Date and time; creating datatime column
df_GroupChat['DateTime'] = df_GroupChat.to_datetime(df_GroupChat['Date'] + ' ' + df_GroupChat['Time'])

#Creating a new column to hold the weekday based on datetime column
df_GroupChat['weekday'] = df_GroupChat['DateTime'].apply(lambda x: x.day_name())

#creating a new columns to hold the letter & word counts to use in visualization based on the chat content
df_GroupChat['Letter_Count'] = df_GroupChat['Content'].apply(lambda s : len(s))
df_GroupChat['Word_Count'] = df_GroupChat['Content'].apply(lambda s : len(s.split(' ')))

#new column to hold the hours in the chat
df_GroupChat['Hour'] = df_GroupChat['Time'].apply(lambda x : x.split(':')[0])
```

Finally, the clean and consistent data will be extracted to csv to create visualization in PowerBI

```
#Finally saving the dataframe into csv file for visalualization
df_GroupChat.to_csv("\WhatsappChat.csv")
```

Data Visualization:

Note: I have not included the steps in detail to create PowerBI visualization. Expected to have some basic knowledge in the tool.

Final Group Chat CSV data is loaded into PowerBI. Once it is loaded, in the “Transform Data”, data type for all fields are manually changed.

Date – Date

Time – Time

Name, Content, Weekday - Text

Date/Time – Datetime

Letter_Count, Word_Count and Hours – Whole Numbers

Created a conditional column to identify the “media” contents in addition to a table to hold the weekdays. Measure tables are created using DAX to calculate the Avg letter per message, message count, number of days, letters count, Words count.

Finally using the toolbox options, the below visualization is created. With the filter options, the data can be seen based on user selection.