ANALYSING WHATSAPP GROUP MESSAGES

```
In [1]: import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        import numpy as np
        import seaborn as sns
        %matplotlib inline
        %matplotlib notebook
        from sklearn import *
        from sklearn.ensemble import RandomForestClassifier, DecisionTreeClassifier
        from sklearn.model_selection import *

In [2]: data = pd.read_csv('Phones.txt', header=None, encoding='utf-8')
        data.head()
```

Out[2]:

	0
0	[8/10/18, 12:00:00] messages and calls encr...
1	[8/10/18, 12:00:00] Jaideep Bansal: Hello
2	[8/10/18, 12:00:00] Jaideep Bansal: Hi
3	[8/10/18, 12:00:00] Jaideep Bansal: Kaise hai...
4	[8/10/18, 12:00:00] Jaideep Bansal: Kaise hai...

```
In [3]: data.shape
```

```
(1048576, 1)
```

Dropping the first 2 rows as they do not contain any messages

```
In [4]: data.drop(range(0,2), inplace=True)
```

```
In [5]: data.shape
```

```
(1048574, 1)
```

Convert the text file into a dataset for further analysis

```
In [6]: chats = []
        for i, line in enumerate(open('Phones.txt', 'r', encoding='utf-8')):
            if i > 2:
                chats.append(line)

        for i in range(5):
            print(chats[i])
            if len(chats[i].split(': ')) > 1:
                print('Date: {0}\nSender: {1}\nMessage: {2}'.format(chats[i].split(', ', maxsplit=1)[0],
                                                                   chats[i].split(': ', maxsplit=1)[1].split(' - ', maxsplit=1)[0],
                                                                   chats[i].split(': ', maxsplit=1)[1].split(' - ', maxsplit=1)[1]))
                print('\n')
            else:
                print('\n')
```

```
In [7]: df = pd.DataFrame(list(zip(chats), columns=['messages']))
        df.head()
```

Out[7]:

	messages
0	[8/10/18, 12:00:00] messages and calls encr...
1	[8/10/18, 12:00:00] Jaideep Bansal: Hello
2	[8/10/18, 12:00:00] Jaideep Bansal: Hi
3	[8/10/18, 12:00:00] Jaideep Bansal: Kaise hai...
4	[8/10/18, 12:00:00] Jaideep Bansal: Kaise hai...

```
In [8]: df.tail()
```

Out[8]:

	messages
1048569	[8/10/18, 23:59:59] [redacted]
1048570	[8/10/18, 23:59:59] [redacted]
1048571	[8/10/18, 23:59:59] [redacted]
1048572	[8/10/18, 23:59:59] [redacted]
1048573	[8/10/18, 23:59:59] [redacted]

```
In [9]: df.iloc[0:5,:]
```

Out[9]:

	messages
0	[8/10/18, 12:00:00] messages and calls encr...
1	[8/10/18, 12:00:00] Jaideep Bansal: Hello
2	[8/10/18, 12:00:00] Jaideep Bansal: Hi
3	[8/10/18, 12:00:00] Jaideep Bansal: Kaise hai...
4	[8/10/18, 12:00:00] Jaideep Bansal: Kaise hai...

Dropping the first 2 rows as they do not contain any messages

```
In [10]: data.drop(range(0,2), inplace=True)
```

```
In [11]: df.shape
```

```
(1048572, 1)
```

Convert the text file into a dataset for further analysis

```
In [12]: df['Date'] = df['messages'].str.split(', ').str[0]
         df['Time'] = df['messages'].str.split(', ').str[1].str.split(' - ').str[0]
         df['Sender'] = df['messages'].str.split(': ').str[1]
         df['Message'] = df['messages'].str.split(': ').str[2]
         df.head()
```

Out[12]:

	Date	Time	Sender	Message	messages
0	[redacted]	[redacted]	NaN	NaN	[8/10/18, 12:00:00] messages and calls encr...
1	8/10/18	12:00:00	[redacted]	Hello	[8/10/18, 12:00:00] Jaideep Bansal: Hello
2	8/10/18	12:00:00	[redacted]	Hi	[8/10/18, 12:00:00] Jaideep Bansal: Hi
3	8/10/18	12:00:00	[redacted]	Kaise hai [redacted]	[8/10/18, 12:00:00] Jaideep Bansal: Kaise hai...
4	8/10/18	12:00:00	[redacted]	Kaise hai [redacted]	[8/10/18, 12:00:00] Jaideep Bansal: Kaise hai...

Changing the Date to a date so that we can extract year, weekdays

```
In [13]: df['Date'] = df['Date'].str.replace('/', '-')
         df.head()
```

Out[13]:

	Date	Time	Sender	Message	messages
0	[redacted]	[redacted]	NaN	NaN	[8/10/18, 12:00:00] messages and calls encr...
1	8-10-18	12:00:00	[redacted]	Hello	[8/10/18, 12:00:00] Jaideep Bansal: Hello
2	8-10-18	12:00:00	[redacted]	Hi	[8/10/18, 12:00:00] Jaideep Bansal: Hi
3	8-10-18	12:00:00	[redacted]	Kaise hai [redacted]	[8/10/18, 12:00:00] Jaideep Bansal: Kaise hai...
4	8-10-18	12:00:00	[redacted]	Kaise hai [redacted]	[8/10/18, 12:00:00] Jaideep Bansal: Kaise hai...

```
In [14]: df.dropna(inplace=True)
         df.shape
```

```
(1048572, 5)
```

Converting the Date column into a datetime format

```
In [15]: df['Date'] = pd.to_datetime(df['Date'])
         df.head()
```

Out[15]:

	Date	Time	Sender	Message	messages
0	[redacted]	[redacted]	NaN	NaN	[8/10/18, 12:00:00] messages and calls encr...
1	2018-08-10	12:00:00	[redacted]	Hello	[8/10/18, 12:00:00] Jaideep Bansal: Hello
2	2018-08-10	12:00:00	[redacted]	Hi	[8/10/18, 12:00:00] Jaideep Bansal: Hi
3	2018-08-10	12:00:00	[redacted]	Kaise hai [redacted]	[8/10/18, 12:00:00] Jaideep Bansal: Kaise hai...
4	2018-08-10	12:00:00	[redacted]	Kaise hai [redacted]	[8/10/18, 12:00:00] Jaideep Bansal: Kaise hai...

```
In [16]: df['Sender'].unique()
```

```
array[redacted]Sender', [redacted]Sender', [redacted]Sender', [redacted]Sender', [redacted]Sender',
       [
```

```
In [48]: dialog_df = df[df['Speaker'] == 'Tuesday']
         dialog_df = dialog_df[
         dialog_df['Dialogue'] != '']
         for_dialog = dialog_df.groupby('Dialogue').
             sort_values(
             ['Speaker', 'Episode', 'Line'], ascending=[
             True, True, True])
         for_dialog['prev_speaker'] = for_dialog.groupby('Episode')['Speaker'].
         shift(1)
         for_dialog['prev_dialogue'] = for_dialog.groupby('Episode')['Dialogue'].
         shift(1)
         for_dialog.dropna(subset=['prev_speaker', 'prev_dialogue'], inplace=True)
         for_dialog['prev_speaker'] = for_dialog['prev_speaker'].
         astype(int)
         for_dialog['prev_dialogue'] = for_dialog['prev_dialogue'].
         astype(int)
         for_dialog.reset_index(drop=True, inplace=True)
         for_dialog.head()
```

Out[48]:

	Dialogue	Episode	Line	Speaker	prev_speaker	prev_dialogue
0	4	1	1	4	1	3
1	5	1	2	5	4	4
2	6	1	3	6	5	5
3	7	1	4	7	6	6
4	8	1	5	8	7	7

```
In [49]: dialog_top_10 = df[df[
         'Dialogue'] == 4]
         dialog_top_10.head()
```

Out[49]:

	Dialogue	Episode	Line	Speaker
3	4	1	4	1
9	4	2	4	1
15	4	3	4	1
21	4	4	4	1
27	4	5	4	1

```
In [50]: fig = px.bar(dialog_frame=dialog_top_10_speaker, x='Speaker',
         y='Episode',
         fig.show()
```

Wilk's top 10 emojis

```
In [51]: dialog_df = df[df['Speaker'] == 'Tuesday']
         dialog_df = dialog_df[
         dialog_df['Dialogue'] != '']
         for_dialog = dialog_df.groupby('Dialogue').
             sort_values(
             ['Speaker', 'Episode', 'Line'], ascending=[
             True, True, True])
         for_dialog['prev_speaker'] = for_dialog.groupby('Episode')['Speaker'].
         shift(1)
         for_dialog['prev_dialogue'] = for_dialog.groupby('Episode')['Dialogue'].
         shift(1)
         for_dialog.dropna(subset=['prev_speaker', 'prev_dialogue'], inplace=True)
         for_dialog['prev_speaker'] = for_dialog['prev_speaker'].
         astype(int)
         for_dialog['prev_dialogue'] = for_dialog['prev_dialogue'].
         astype(int)
         for_dialog.reset_index(drop=True, inplace=True)
         for_dialog.head()
```

Out[51]:

	Dialogue	Episode	Line	Speaker	prev_speaker	prev_dialogue
0	4	1	1	4	1	3
1	5	1	2	5	4	4
2	6	1	3	6	5	5
3	7	1	4	7	6	6
4	8	1	5	8	7	7

```
In [52]: dialog_top_10_emoji
```

Out[52]:

	Dialogue	Episode	Line	Speaker
3	4	1	4	1
9	4	2	4	1
15	4	3	4	1
21	4	4	4	1
27	4	5	4	1

```
In [53]: fig = px.bar(dialog_frame=dialog_top_10_emoji, x='Speaker',
         y='Episode',
         fig.show()
```