ESSENTIAL OF DATA SCIENCE

Theory Activity No. 1

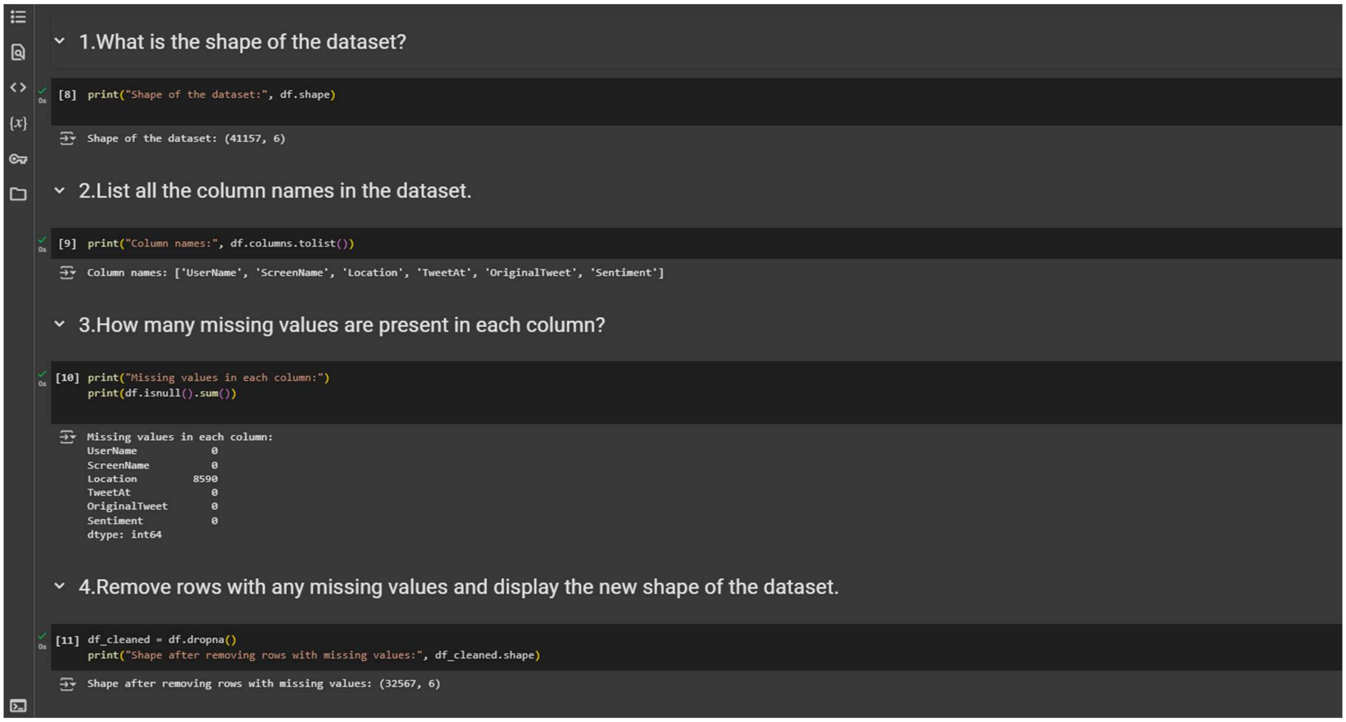
Name – Vaishnav Pralhad Tarde

Div - CS2

Roll No. - CS2-12

PRN – 202401040175

- 20 problem statements for Kaggel Text Classification Dataset using Numpy and Pandas.



1.What is the shape of the dataset?

```
[8]  print("Shape of the dataset:", df.shape)
```

```
Shape of the dataset: (41157, 6)
```

2.List all the column names in the dataset.

```
[9]  print("Column names:", df.columns.tolist())
```

```
Column names: ['UserName', 'ScreenName', 'Location', 'TweetAt', 'OriginalTweet', 'Sentiment']
```

3.How many missing values are present in each column?

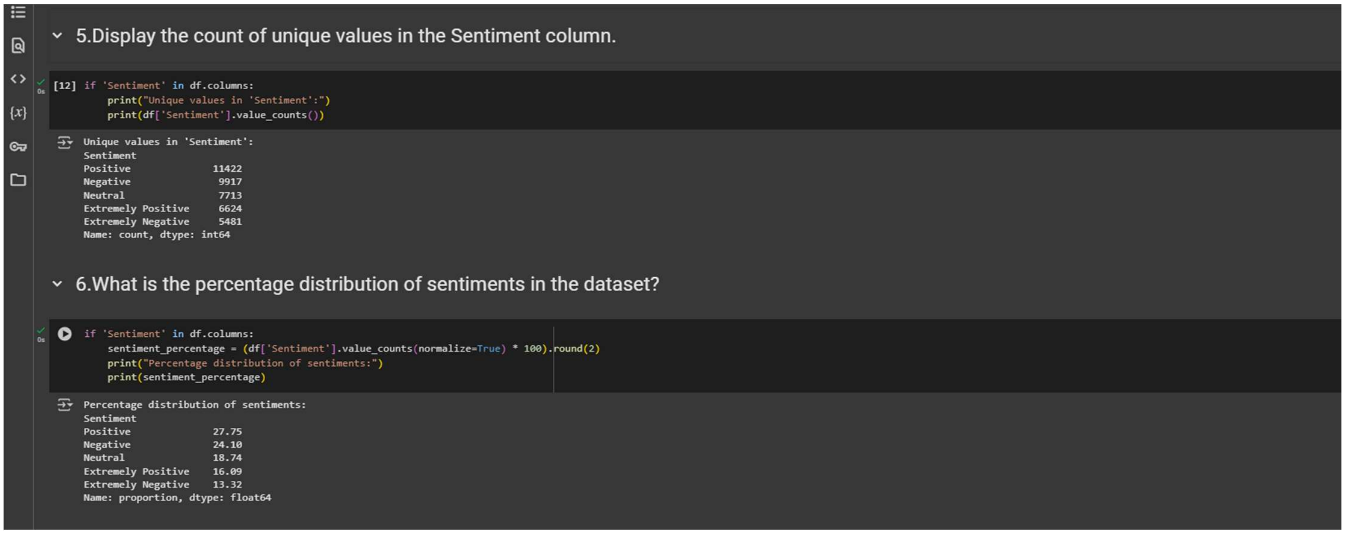
```
[10] print("Missing values in each column:")
     print(df.isnull().sum())
```

```
Missing values in each column:
UserName            0
ScreenName          0
Location         8590
TweetAt             0
OriginalTweet       0
Sentiment           0
dtype: int64
```

4.Remove rows with any missing values and display the new shape of the dataset.

```
[11] df_cleaned = df.dropna()
     print("Shape after removing rows with missing values:", df_cleaned.shape)
```

```
Shape after removing rows with missing values: (32567, 6)
```



5.Display the count of unique values in the Sentiment column.

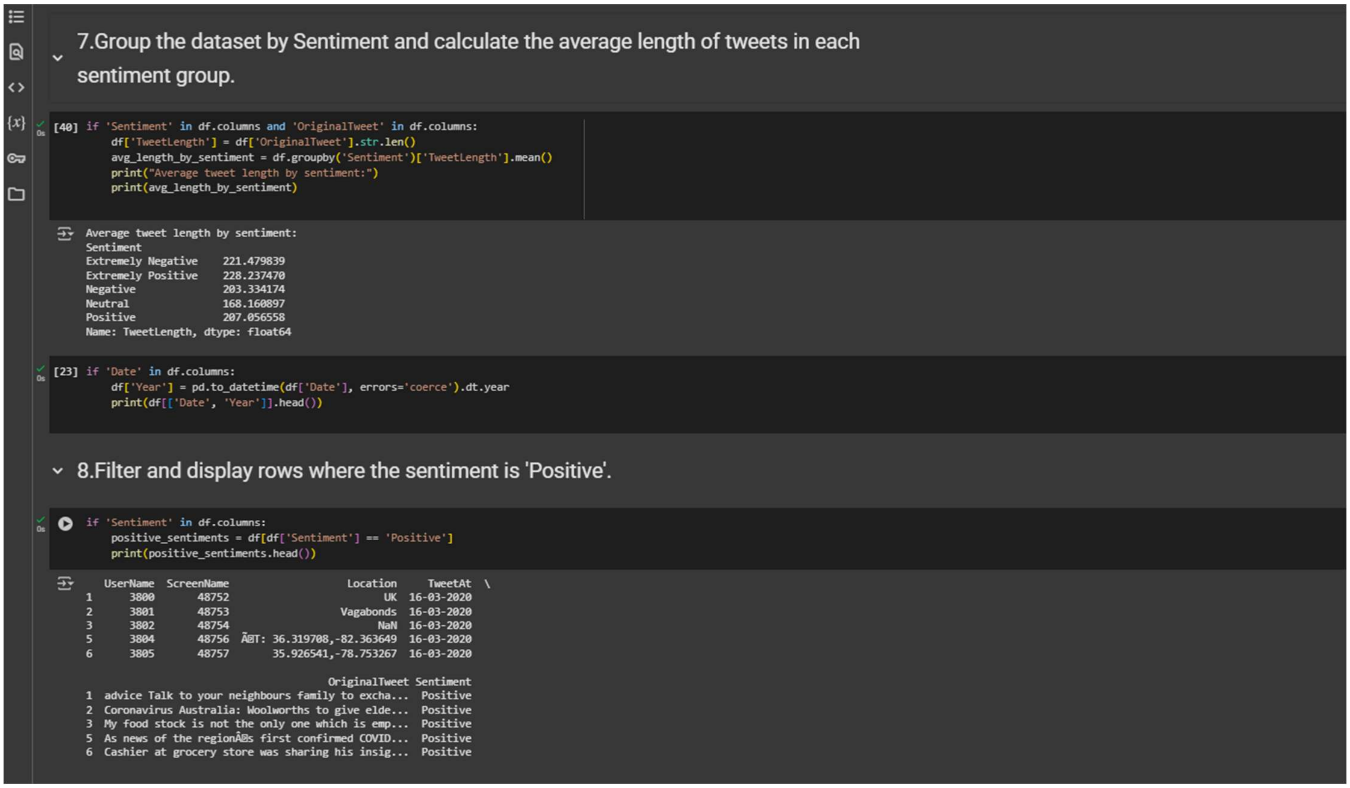
```
[12] if 'Sentiment' in df.columns:
         print("Unique values in 'Sentiment':")
         print(df['Sentiment'].value_counts())
```

```
Unique values in 'Sentiment':
Sentiment
Positive              11422
Negative               9917
Neutral                7713
Extremely Positive     6624
Extremely Negative     5481
Name: count, dtype: int64
```

6.What is the percentage distribution of sentiments in the dataset?

```
if 'Sentiment' in df.columns:
    sentiment_percentage = (df['Sentiment'].value_counts(normalize=True) * 100).round(2)
    print("Percentage distribution of sentiments:")
    print(sentiment_percentage)
```

```
Percentage distribution of sentiments:
Sentiment
Positive              27.75
Negative              24.10
Neutral               18.74
Extremely Positive    16.09
Extremely Negative    13.32
Name: proportion, dtype: float64
```



7.Group the dataset by Sentiment and calculate the average length of tweets in each sentiment group.

```
[40] if 'Sentiment' in df.columns and 'OriginalTweet' in df.columns:
         df['TweetLength'] = df['OriginalTweet'].str.len()
         avg_length_by_sentiment = df.groupby('Sentiment')['TweetLength'].mean()
         print("Average tweet length by sentiment:")
         print(avg_length_by_sentiment)
```

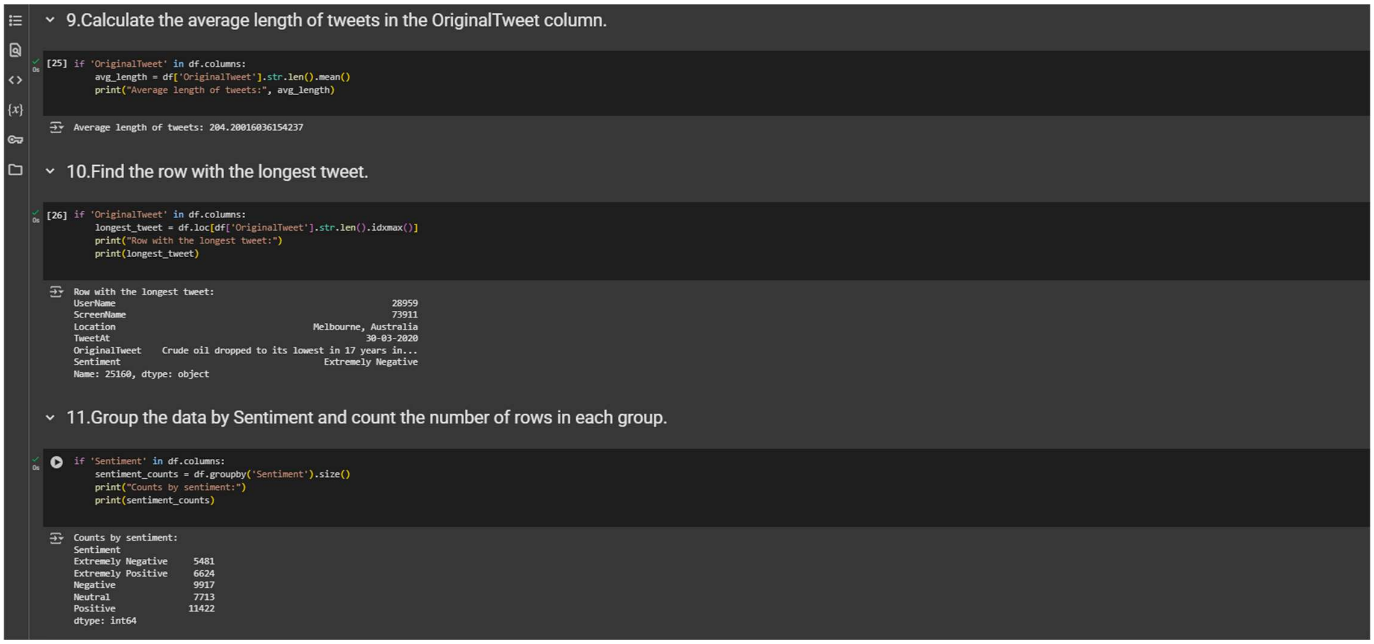
```
Average tweet length by sentiment:
Sentiment
Extremely Negative    221.479839
Extremely Positive    228.237470
Negative              203.334174
Neutral               168.160897
Positive              207.056558
Name: TweetLength, dtype: float64
```

```
[23] if 'Date' in df.columns:
         df['Year'] = pd.to_datetime(df['Date'], errors='coerce').dt.year
         print(df[['Date', 'Year']].head())
```

8.Filter and display rows where the sentiment is 'Positive'.

```
if 'Sentiment' in df.columns:
    positive_sentiments = df[df['Sentiment'] == 'Positive']
    print(positive_sentiments.head())
```

```
   UserName  ScreenName                   Location     TweetAt  \
1      3800       48752                         UK  16-03-2020   
2      3801       48753                  Vagabonds  16-03-2020   
3      3802       48754                        NaN  16-03-2020   
5      3804       48756  Ã�T: 36.319708,-82.363649  16-03-2020   
6      3805       48757       35.926541,-78.753267  16-03-2020   

                                       OriginalTweet Sentiment  
1  advice Talk to your neighbours family to excha...  Positive  
2  Coronavirus Australia: Woolworths to give elde...  Positive  
3  My food stock is not the only one which is emp...  Positive  
5  As news of the regionÂ�s first confirmed COVID...  Positive  
6  Cashier at grocery store was sharing his insig...  Positive  
```



9.Calculate the average length of tweets in the OriginalTweet column.

```
[25] if 'OriginalTweet' in df.columns:
         avg_length = df['OriginalTweet'].str.len().mean()
         print("Average length of tweets:", avg_length)
```

```
Average length of tweets: 204.20016036154237
```

10.Find the row with the longest tweet.

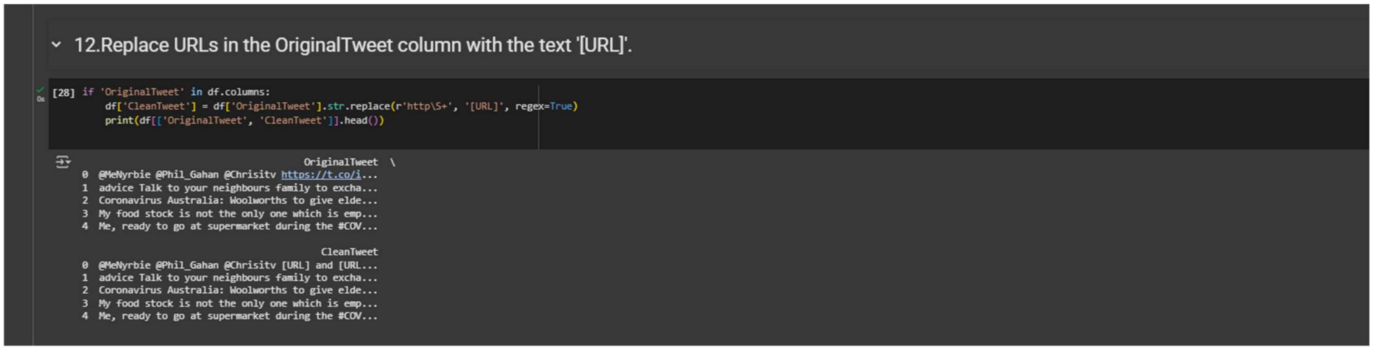
```
[26] if 'OriginalTweet' in df.columns:
         longest_tweet = df.loc[df['OriginalTweet'].str.len().idxmax()]
         print("Row with the longest tweet:")
         print(longest_tweet)
```

```
Row with the longest tweet:
UserName                                                     28959
ScreenName                                                   73911
Location                                       Melbourne, Australia
TweetAt                                                  30-03-2020
OriginalTweet    Crude oil dropped to its lowest in 17 years in...
Sentiment                                        Extremely Negative
Name: 25160, dtype: object
```

11.Group the data by Sentiment and count the number of rows in each group.

```
if 'Sentiment' in df.columns:
    sentiment_counts = df.groupby('Sentiment').size()
    print("Counts by sentiment:")
    print(sentiment_counts)
```

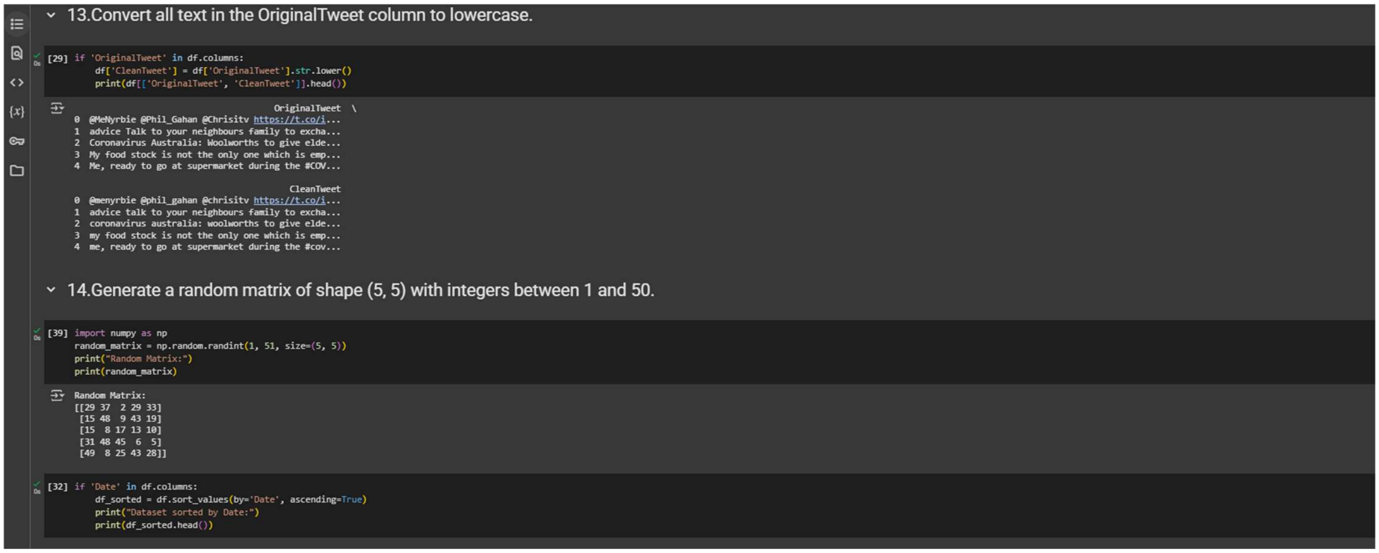
```
Counts by sentiment:
Sentiment
Extremely Negative     5481
Extremely Positive     6624
Negative               9917
Neutral                7713
Positive              11422
dtype: int64
```



12.Replace URLs in the OriginalTweet column with the text '[URL]'.

```
[28] if 'OriginalTweet' in df.columns:
         df['CleanTweet'] = df['OriginalTweet'].str.replace(r'http\S+', '[URL]', regex=True)
         print(df[['OriginalTweet', 'CleanTweet']].head())
```

```
                                       OriginalTweet  \
0  @MeNyrbie @Phil_Gahan @Chrisitv https://t.co/i...
1  advice Talk to your neighbours family to excha...
2  Coronavirus Australia: Woolworths to give elde...
3  My food stock is not the only one which is emp...
4  Me, ready to go at supermarket during the #COV...

                                          CleanTweet
0  @MeNyrbie @Phil_Gahan @Chrisitv [URL] and [URL...
1  advice Talk to your neighbours family to excha...
2  Coronavirus Australia: Woolworths to give elde...
3  My food stock is not the only one which is emp...
4  Me, ready to go at supermarket during the #COV...
```



13.Convert all text in the OriginalTweet column to lowercase.

```
[29] if 'OriginalTweet' in df.columns:
         df['CleanTweet'] = df['OriginalTweet'].str.lower()
         print(df[['OriginalTweet', 'CleanTweet']].head())
```

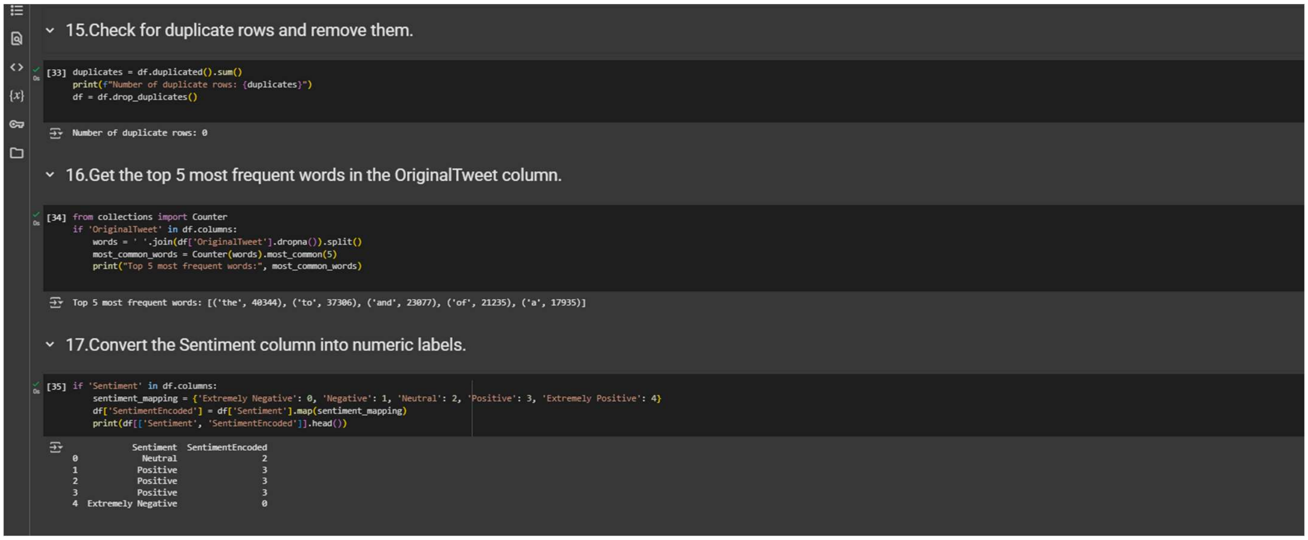
```
                                       OriginalTweet  \
0  @MeNyrbie @Phil_Gahan @Chrisitv https://t.co/i...
1  advice Talk to your neighbours family to excha...
2  Coronavirus Australia: Woolworths to give elde...
3  My food stock is not the only one which is emp...
4  Me, ready to go at supermarket during the #COV...

                                          CleanTweet
0  @menyrbie @phil_gahan @chrisitv https://t.co/i...
1  advice talk to your neighbours family to excha...
2  coronavirus australia: woolworths to give elde...
3  my food stock is not the only one which is emp...
4  me, ready to go at supermarket during the #cov...
```

14.Generate a random matrix of shape (5, 5) with integers between 1 and 50.

```
[39] import numpy as np
     random_matrix = np.random.randint(1, 51, size=(5, 5))
     print("Random Matrix:")
     print(random_matrix)
```

```
Random Matrix:
[[29 37  2 29 33]
 [15 48  9 43 19]
 [15  8 17 13 10]
 [31 48 45  6  5]
 [49  8 25 43 28]]
```

```
[32] if 'Date' in df.columns:
         df_sorted = df.sort_values(by='Date', ascending=True)
         print("Dataset sorted by Date:")
         print(df_sorted.head())
```



15.Check for duplicate rows and remove them.

```
[33] duplicates = df.duplicated().sum()
     print(f"Number of duplicate rows: {duplicates}")
     df = df.drop_duplicates()
```

```
Number of duplicate rows: 0
```

16.Get the top 5 most frequent words in the OriginalTweet column.

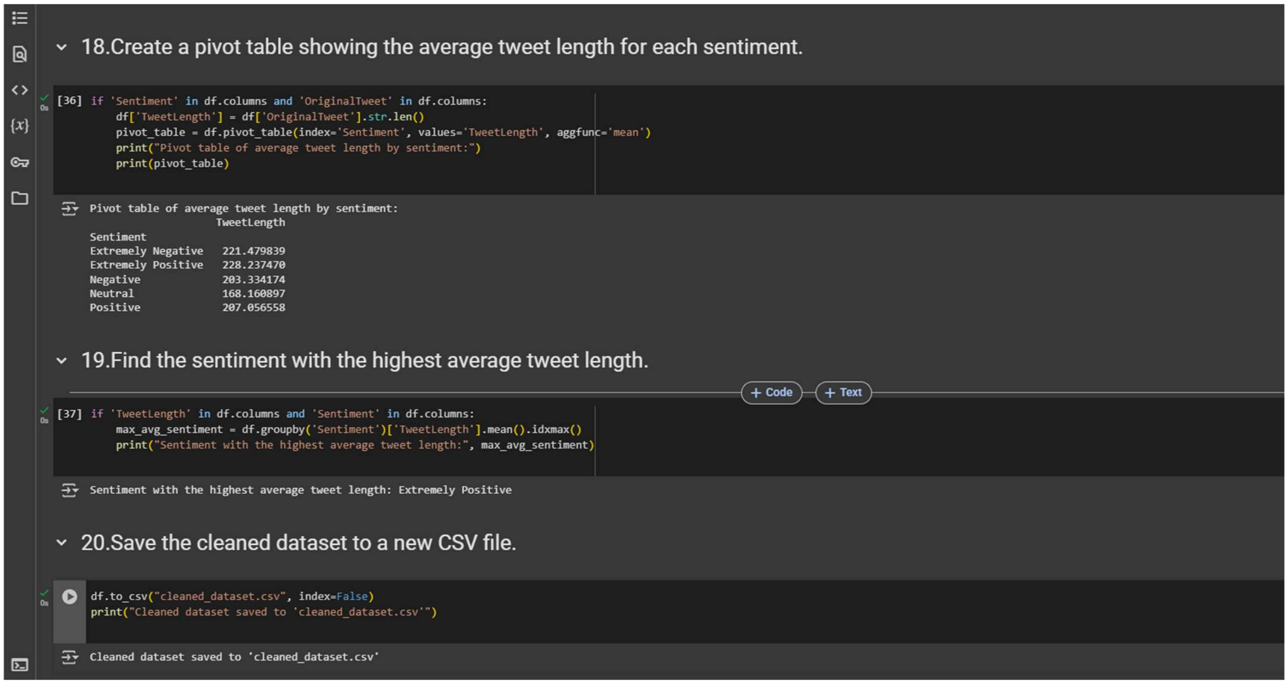
```
[34] from collections import Counter
     if 'OriginalTweet' in df.columns:
         words = ' '.join(df['OriginalTweet'].dropna()).split()
         most_common_words = Counter(words).most_common(5)
         print("Top 5 most frequent words:", most_common_words)
```

```
Top 5 most frequent words: [('the', 40344), ('to', 37306), ('and', 23077), ('of', 21235), ('a', 17935)]
```

17.Convert the Sentiment column into numeric labels.

```
[35] if 'Sentiment' in df.columns:
         sentiment_mapping = {'Extremely Negative': 0, 'Negative': 1, 'Neutral': 2, 'Positive': 3, 'Extremely Positive': 4}
         df['SentimentEncoded'] = df['Sentiment'].map(sentiment_mapping)
         print(df[['Sentiment', 'SentimentEncoded']].head())
```

```
             Sentiment  SentimentEncoded
0              Neutral                 2
1             Positive                 3
2             Positive                 3
3             Positive                 3
4   Extremely Negative                 0
```



18.Create a pivot table showing the average tweet length for each sentiment.

```
[36] if 'Sentiment' in df.columns and 'OriginalTweet' in df.columns:
         df['TweetLength'] = df['OriginalTweet'].str.len()
         pivot_table = df.pivot_table(index='Sentiment', values='TweetLength', aggfunc='mean')
         print("Pivot table of average tweet length by sentiment:")
         print(pivot_table)
```

```
Pivot table of average tweet length by sentiment:
                    TweetLength
Sentiment
Extremely Negative   221.479839
Extremely Positive   228.237470
Negative             203.334174
Neutral              168.160897
Positive             207.056558
```

19.Find the sentiment with the highest average tweet length.

```
[37] if 'TweetLength' in df.columns and 'Sentiment' in df.columns:
         max_avg_sentiment = df.groupby('Sentiment')['TweetLength'].mean().idxmax()
         print("Sentiment with the highest average tweet length:", max_avg_sentiment)
```

```
Sentiment with the highest average tweet length: Extremely Positive
```

20.Save the cleaned dataset to a new CSV file.

```
df.to_csv("cleaned_dataset.csv", index=False)
print("Cleaned dataset saved to 'cleaned_dataset.csv'")
```

```
Cleaned dataset saved to 'cleaned_dataset.csv'
```