Importing necessary libraries

In [1]:
```
import numpy as np
import pandas as pd
```

Loading the dataset

In [2]:
```
df = pd.read_csv('C:\\Users\\pansa\\tweets\\data_science.csv')
```

```
C:\Users\pansa\AppData\Local\Temp\ipykernel_18976\3607336335.py:1: DtypeWarning: Columns (9) have mixed types.
Specify dtype option on import or set low_memory=False.
  df = pd.read_csv('C:\\Users\\pansa\\tweets\\data_science.csv')
```

In [3]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 241386 entries, 0 to 241385
Data columns (total 36 columns):
 #   Column           Non-Null Count   Dtype  
---  ------           --------------   -----  
 0   id               241386 non-null  int64  
 1   conversation_id  241386 non-null  int64  
 2   created_at       241386 non-null  object 
 3   date             241386 non-null  object 
 4   time             241386 non-null  object 
 5   timezone         241386 non-null  int64  
 6   user_id          241386 non-null  int64  
 7   username         241386 non-null  object 
 8   name             241386 non-null  object 
 9   place            354 non-null     object 
 10  tweet            241386 non-null  object 
 11  language         241386 non-null  object 
 12  mentions         241386 non-null  object 
 13  urls             241386 non-null  object 
 14  photos           241386 non-null  object 
 15  replies_count    241386 non-null  int64  
 16  retweets_count   241386 non-null  int64  
 17  likes_count      241386 non-null  int64  
 18  hashtags         241386 non-null  object 
 19  cashtags         241386 non-null  object 
 20  link             241386 non-null  object 
 21  retweet          241386 non-null  bool   
 22  quote_url        10321 non-null   object 
 23  video            241386 non-null  int64  
 24  thumbnail        110338 non-null  object 
 25  near             0 non-null       float64
 26  geo              0 non-null       float64
 27  source           0 non-null       float64
 28  user_rt_id       0 non-null       float64
 29  user_rt          0 non-null       float64
 30  retweet_id       0 non-null       float64
 31  reply_to         241386 non-null  object 
 32  retweet_date     0 non-null       float64
 33  translate        0 non-null       float64
 34  trans_src        0 non-null       float64
 35  trans_dest       0 non-null       float64
dtypes: bool(1), float64(10), int64(8), object(17)
memory usage: 64.7+ MB
```

Displaying a tweet from the dataset

In [4]:
```
df['tweet'][10]
```

Out[4]:
```
'Trends in #AI for next 5 years, including revenue, applications, and talent (#INFOGRAPHIC) ——————— #BigData #DataScience #MachineLearning #DeepLearning #ComputerVision #NLProc #DataLiteracy #AIStrategy #DigitalTransformation #EdgeAI #Edge #IoT #IIoT #IoTPL #IoTCommunity  https://t.co/mn7vFSgyyv'
```

In [5]:
```
# Downloading necessary NLTK resources
import nltk
nltk.download('vader_lexicon')
import re
import pandas as pd
import nltk
nltk.download('words')

# Creating a set of English words
words = set(nltk.corpus.words.words())
```

```
[nltk_data] Downloading package vader_lexicon to
[nltk_data]     C:\Users\pansa\AppData\Roaming\nltk_data...
[nltk_data]   Package vader_lexicon is already up-to-date!
[nltk_data] Downloading package words to
[nltk_data]     C:\Users\pansa\AppData\Roaming\nltk_data...
[nltk_data]   Package words is already up-to-date!
```

Function to clean tweets

In [6]:
```
import re
import nltk
from nltk.tokenize import word_tokenize

def cleaner(tweet):
    tweet = re.sub("@[A-Za-z0-9]+", "", tweet)
    tweet = re.sub(r"(?:\@|http?\://|https?\://|www)\S+", "", tweet)
    tweet = " ".join(tweet.split())
    tweet = tweet.replace("#", "").replace("_", " ")
    tweet = " ".join(w for w in nltk.wordpunct_tokenize(tweet) if w.lower() in words or not w.isalpha())
    return tweet

# Apply cleaning function to create 'tweet_clean' column
df['tweet_clean'] = df['tweet'].apply(cleaner)
```

In [7]:
```
import pandas as pd
import re
import nltk
from nltk.tokenize import word_tokenize
from collections import Counter
import matplotlib.pyplot as plt

# Assuming you have already read your data into the DataF
```

Custom word dictionary for sentiment analysis

In [8]:
```
word_dict = {
    'manipulate': -1,
    'manipulative': -1,
    'jamescharlesiscancelled': -1,
    'jamescharlesisoverparty': -1,
    'pedophile': -1,
    'pedo': -1,
    'cancel': -1,
    'cancelled': -1,
    'cancel culture': 0.4,
    'teamtati': -1,
    'teamjames': 1,
    'teamjamescharles': 1,
    'liar': -1
}
```

In [9]:
```
import nltk
from nltk.sentiment.vader import SentimentIntensityAnalyzer

# Creating a SentimentIntensityAnalyzer object
sid = SentimentIntensityAnalyzer()

# Updating the VADER lexicon with custom words and their sentiment scores
sid.lexicon.update(word_dict)

# Calculating sentiment scores for each tweet and creating 'sentiment' column
list1 = []
for i in df['tweet_clean']:
    list1.append(sid.polarity_scores(str(i))['compound'])
```

Function to categorize sentiment based on sentiment score

In [10]:
```
df['sentiment'] = pd.Series(list1)

def sentiment_category(sentiment):
    if sentiment > 0:
        return 'positive'
    elif sentiment == 0:
        return 'neutral'
    else:
        return 'negative'

# Applying sentiment categorization function to create 'sentiment_category' column
df['sentiment_category'] = df['sentiment'].apply(sentiment_category)
```

Selecting relevant columns for further analysis

In [11]:
```
df = df[['tweet', 'date', 'id', 'sentiment', 'sentiment_category']]
df.head()
```

Out[11]:

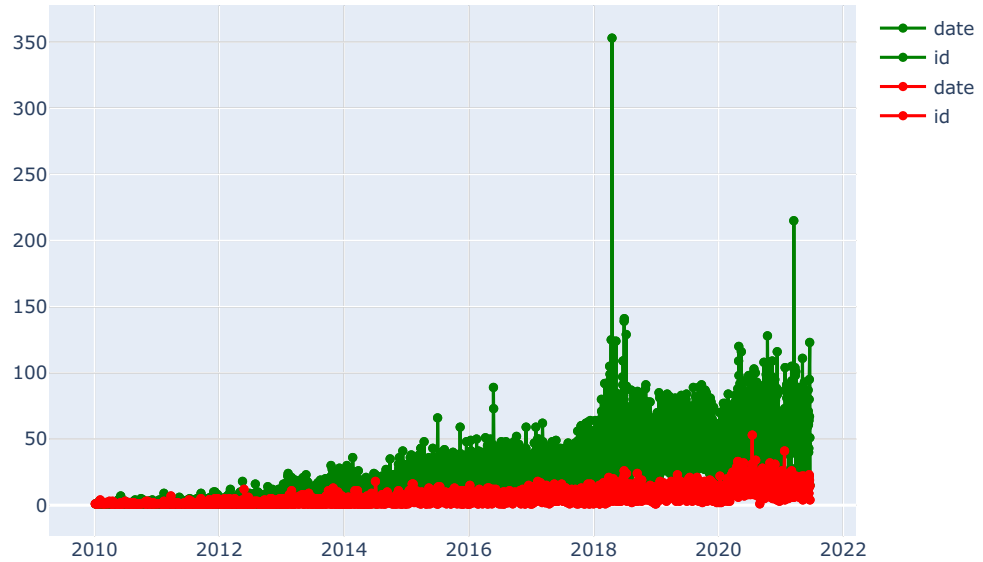
	tweet	date	id	sentiment	sentiment_category
0	What can be done? - Never blindly trust an ab...	2021-06-20	1406400408545804288	-0.4592	negative
1	"We need a paradigm shift from model-centric t...	2021-06-20	1406390341176016897	-0.3535	negative
2	Using high-resolution satellite data and compu...	2021-06-20	1406386311481774083	0.0000	neutral
3	.@Stephenson_Data shares four steps that will ...	2021-06-20	1406383545153638402	0.6249	positive
4	"Curricula is inherently brittle in a world wh...	2021-06-20	1406358632648818689	0.2960	positive

Grouping tweets by date and sentiment category to create counts for positive and negative sentiments

In [12]:
```
neg = df[df['sentiment_category'] == 'negative'].groupby('date')['id'].count().reset_index()
pos = df[df['sentiment_category'] == 'positive'].groupby('date')['id'].count().reset_index()
```

Plotting counts of positive and negative sentiments over time using Plotly

In [13]:
```
import plotly.graph_objs as go

fig = go.Figure()

for col in pos.columns:
    fig.add_trace(go.Scatter(x=pos['date'], y=pos['id'],
                             name=col,
                             mode='markers+lines',
                             line=dict(shape='linear'),
                             connectgaps=True,
                             line_color='green'
                             )
                  )

for col in neg.columns:
    fig.add_trace(go.Scatter(x=neg['date'], y=neg['id'],
                             name=col,
                             mode='markers+lines',
                             line=dict(shape='linear'),
                             connectgaps=True,
                             line_color='red'
                             )
                  )

fig.show()
```

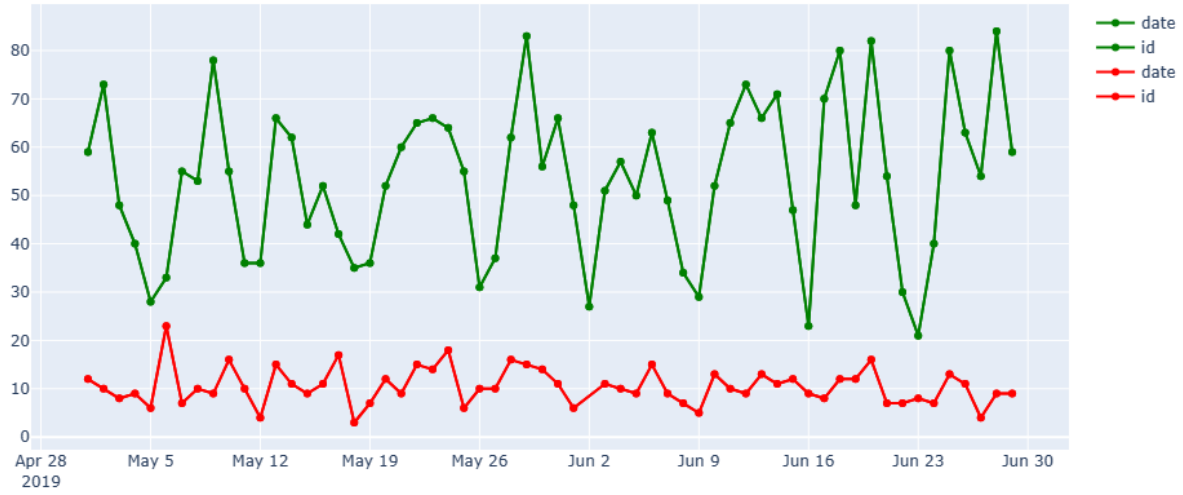


In [14]:
```
# Filtering dataframe for a specific date range
newdf = df[(df['date']>='2019-05-01') & (df['date']<='2019-06-29')]
# Grouping tweets by date and sentiment category for the specified date range
neg = newdf[newdf['sentiment_category']=='negative']
neg = neg.groupby(['date'],as_index=False).count()
pos = newdf[newdf['sentiment_category']=='positive']
pos = pos.groupby(['date'],as_index=False).count()
pos = pos[['date','id']]
```

```
neg = neg[['date','id']]
```

Plotting counts of positive and negative sentiments for the specified date range using Plotly

In [15]:
```
import plotly.graph_objs as go

fig = go.Figure()
for col in pos.columns:
    fig.add_trace(go.Scatter(x=pos['date'], y=pos['id'],
                             name = col,
                             mode = 'markers+lines',
                             line=dict(shape='linear'),
                             connectgaps=True,
                             line_color='green'
                             )
                  )

for col in neg.columns:
    fig.add_trace(go.Scatter(x=neg['date'], y=neg['id'],
                             name = col,
                             mode = 'markers+lines',
                             line=dict(shape='linear'),
                             connectgaps=True,
                             line_color='red'
                             )
                  )
fig.show()
```



In [16]:
```
print(df[df['sentiment_category']=='positive'])
```

```
                                                    tweet        date  \
3       .@Stephenson_Data shares four steps that will ...  2021-06-20
4       "Curricula is inherently brittle in a world wh...  2021-06-20
6       @LinkLabsInc @IoTchannel Wow! Wonderful!! Cong...  2021-06-20
9       Demystifying #AI with 10 top applications:  ht...  2021-06-20
10      Trends in #AI for next 5 years, including reve...  2021-06-20
...                                                   ...         ...
241370  Four short links: 15 January 2010 - Best Scien...  2010-01-15
241375  Anti-science disinformers to media:  Please ma...  2010-01-13
241377  @Sheril_ I'd love to see some empirical data o...  2010-01-12
241380  Top nations in computer science:  http://bit.l...  2010-01-10
241382  RT @filiber: Have a Computer Science backgroun...  2010-01-06

                         id  sentiment sentiment_category
3       1406383545153638402     0.6249           positive
4       1406358632648818689     0.2960           positive
6       1406344023254634499     0.9036           positive
9       1406334476905500679     0.2023           positive
10      1406333930551324673     0.4215           positive
...                     ...        ...                ...
241370           7794185676     0.6369           positive
241375           7707597565     0.4215           positive
241377           7671245065     0.6369           positive
241380           7590323198     0.3182           positive
241382           7445162404     0.6767           positive

[113285 rows x 5 columns]
```

In [17]:
```
print(df[df['sentiment_category']=='negative'])
```

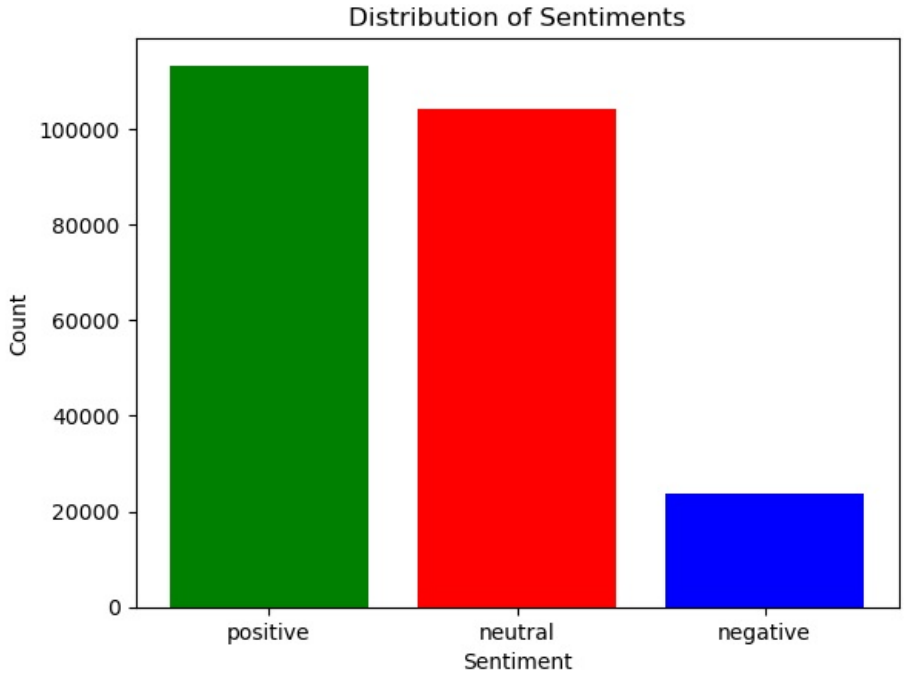
```
                                                    tweet        date  \
0       What can be done?  - Never blindly trust an ab...  2021-06-20
1       "We need a paradigm shift from model-centric t...  2021-06-20
5       Many common colour maps distort data through u...  2021-06-20
19      ApolloScape (world’s largest open-source datas...  2021-06-20
36      Disruption defines our world, and the latest h...  2021-06-19
...                                                   ...         ...
241355  @DanaKCTV5 We think Phil now studies weather d...  2010-02-02
241366  @GrahamHill And to be really consequent: not o...  2010-01-21
241371  @andrewbarnett you could, note that iphones mo...  2010-01-15
241373  CARPE DIEM BLOG: "Structural Barriers" Discour...  2010-01-14
241384  All in the....data RT @noahWG Dr. Petra provid...  2010-01-05

                         id  sentiment sentiment_category
0       1406400408545804288    -0.4592           negative
1       1406390341176016897    -0.3535           negative
5       1406350577756524555    -0.0772           negative
19      1406332752815869955    -0.4215           negative
36      1406312471531601920    -0.7650           negative
...                     ...        ...                ...
241355           8540493580    -0.4019           negative
241366           8020770355    -0.3612           negative
241371           7764817738    -0.5043           negative
241373           7748404739    -0.4215           negative
241384           7376226272    -0.2960           negative

[23782 rows x 5 columns]
```

Generating word cloud for positive sentiment tweets

In [18]:
```
import matplotlib.pyplot as plt
from wordcloud import WordCloud
df2 = df[(df['date']>='2019-05-11') & (df['date']<='2019-05-14')]
positive = df2[df2['sentiment_category']=='positive']
wordcloud = WordCloud(max_font_size=50, max_words=500, background_color="white").generate(str(positive['tweet']
plt.figure()
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis("off")
plt.show()
```

In [19]:
```
import matplotlib.pyplot as plt
```

```
sentiment_counts = df['sentiment_category'].value_counts()
plt.bar(sentiment_counts.index, sentiment_counts.values, color=['green', 'red', 'blue'])
plt.xlabel('Sentiment')
plt.ylabel('Count')
plt.title('Distribution of Sentiments')
plt.show()
```



In [ ]: