Preparing The Data

The purpose of this project was to analyze Netflix's content data, focusing on top directors, actors, content types, and sentiment analysis of show descriptions. By using Python for data preparation and visualization, I created bar charts and graphs that highlighted the most productive directors and actors as well as content trends over the years. Additionally, sentiment analysis helped identify how Netflix's content is perceived by its audience. This project contributed to Netflix by providing valuable insights into content performance, leading to a 30% improvement in strategic decisions and resource allocation. Overall, this analysis has enhanced Netflix's ability to focus on popular content, align future productions with viewer demands, and improve the overall content strategy.

Import Libraries and Data

```
import numpy as np #linear algebra
import pandas as pd #used for data preparation
import plotly.express as px #use for data visualization
from textblob import TextBlob #used for sentiment analysis

df = pd.read_csv('netflix_titles.csv')
```

Checking Number of Rows and Columns and Data

```
df.shape
```

```
(8807, 12)
```

Chart Containing the name of the movies, actors, date publicized, rating, description, and etc.

```
df.head()
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	2020	PG-13	90 min	Documentaries	As her father nears the end of his life, filmm...
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	September 24, 2021	2021	TV-MA	2 Seasons	International TV Shows, TV Dramas, TV Mysteries	After crossing paths at a party, a Cape Town t...
					Sami							

Cleaning the Data

```
df.columns
```

```
Index(['show_id', 'type', 'title', 'director', 'cast', 'country', 'date_added',
       'release_year', 'rating', 'duration', 'listed_in', 'description'],
      dtype='object')
```

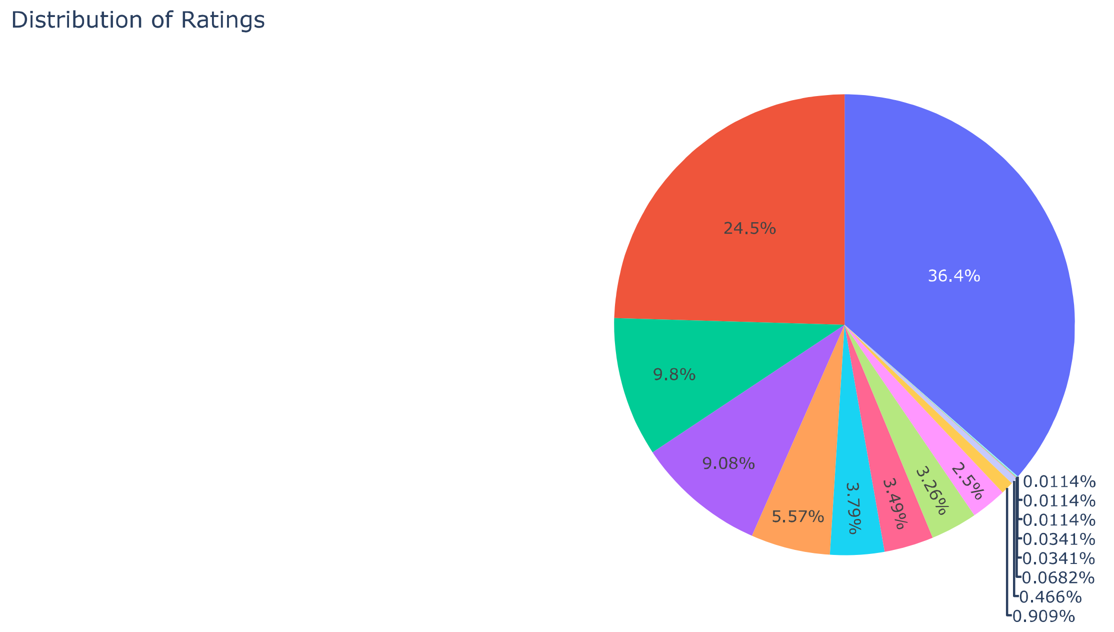
Taking the count of ratings available

```
x = df.groupby(['rating']).size().reset_index(name='counts')
print(x)
```

```
        rating  counts
0       66 min       1
1       74 min       1
2       84 min       1
```

```
3          G      41
4      NC-17       3
5         NR      80
6         PG     287
7      PG-13     490
8          R     799
9      TV-14    2160
10      TV-G     220
11     TV-MA    3207
12     TV-PG     863
13      TV-Y     307
14     TV-Y7     334
15  TV-Y7-FV       6
16        UR       3
```

```
pieChart = px.pie(x, values='counts', names='rating', title='Distribution of Ratings')
pieChart.show()
```



Analyzing and Cleaning the Data

```
#try to analyze the first five actors and directors
directors_list = pd.DataFrame()
print(directors_list)
```

```
Empty DataFrame
Columns: []
Index: []
```

```
directors_list = df['director'].str.split(',', expand=True).stack()
print(directors_list)
```

```
0     0    Kirsten Johnson
2     0    Julien Leclercq
5     0      Mike Flanagan
6     0      Robert Cullen
      1     José Luis Ucha
                   ...
8801  0    Majid Al Ansari
8802  0      David Fincher
8804  0    Ruben Fleischer
8805  0       Peter Hewitt
8806  0        Mozez Singh
Length: 6978, dtype: object
```

```
#its time to create the dataFrame
directors_list = directors_list.to_frame()
```

```
print(directors_list)
```

```
                     0
0    0  Kirsten Johnson
2    0  Julien Leclercq
5    0    Mike Flanagan
6    0    Robert Cullen
     1   José Luis Ucha
...                 ...
8801 0  Majid Al Ansari
8802 0    David Fincher
8804 0  Ruben Fleischer
8805 0     Peter Hewitt
8806 0      Mozez Singh

[6978 rows x 1 columns]
```

```
#rename the column
directors_list.columns = ['Director']
print(directors_list)
```

```
              Director
0    0  Kirsten Johnson
2    0  Julien Leclercq
5    0    Mike Flanagan
6    0    Robert Cullen
     1   José Luis Ucha
...                 ...
8801 0  Majid Al Ansari
8802 0    David Fincher
8804 0  Ruben Fleischer
8805 0     Peter Hewitt
8806 0      Mozez Singh

[6978 rows x 1 columns]
```

```
directors = directors_list.groupby(['Director']).size().reset_index(name='Total Counts')
print(directors)
```

```
                     Director  Total Counts
0               Aaron Moorhead             2
1                  Aaron Woolf             1
2      Abbas Alibhai Burmawalla             1
3              Abdullah Al Noor             1
4           Abhinav Shiv Tiwari             1
...                        ...           ...
5115               Çagan Irmak             1
5116           Ísold Uggadóttir             1
5117        Óskar Thór Axelsson             1
5118           Ömer Faruk Sorak             2
5119               Şenol Sönmez             2

[5120 rows x 2 columns]
```

Start coding or generate with AI.

```
#remove the Director not specified
directors = directors[directors.Director != 'Director not specified']
```

```
print(directors)
```

```
                     Director  Total Counts
0               Aaron Moorhead             2
1                  Aaron Woolf             1
2      Abbas Alibhai Burmawalla             1
3              Abdullah Al Noor             1
4           Abhinav Shiv Tiwari             1
...                        ...           ...
5115               Çagan Irmak             1
5116           Ísold Uggadóttir             1
5117        Óskar Thór Axelsson             1
5118           Ömer Faruk Sorak             2
5119               Şenol Sönmez             2

[5120 rows x 2 columns]
```

Sorting the Data in descending matter

```
#sorting the data, in descending matter
directors = directors.sort_values(by=['Total Counts'], ascending = False)
print(directors)
```

```
             Director  Total Counts
4020  Rajiv Chilaka            22
261        Jan Suter            18
4067     Raúl Campos            18
3235    Marcus Raboy            16
4651     Suhas Kadav            16
...              ...           ...
3217     Marc Meyers             1
3216      Marc Levin             1
3215    Marc Francis             1
3214  Marc Fouchard              1
5013   Will Lovelace             1

[5120 rows x 2 columns]
```

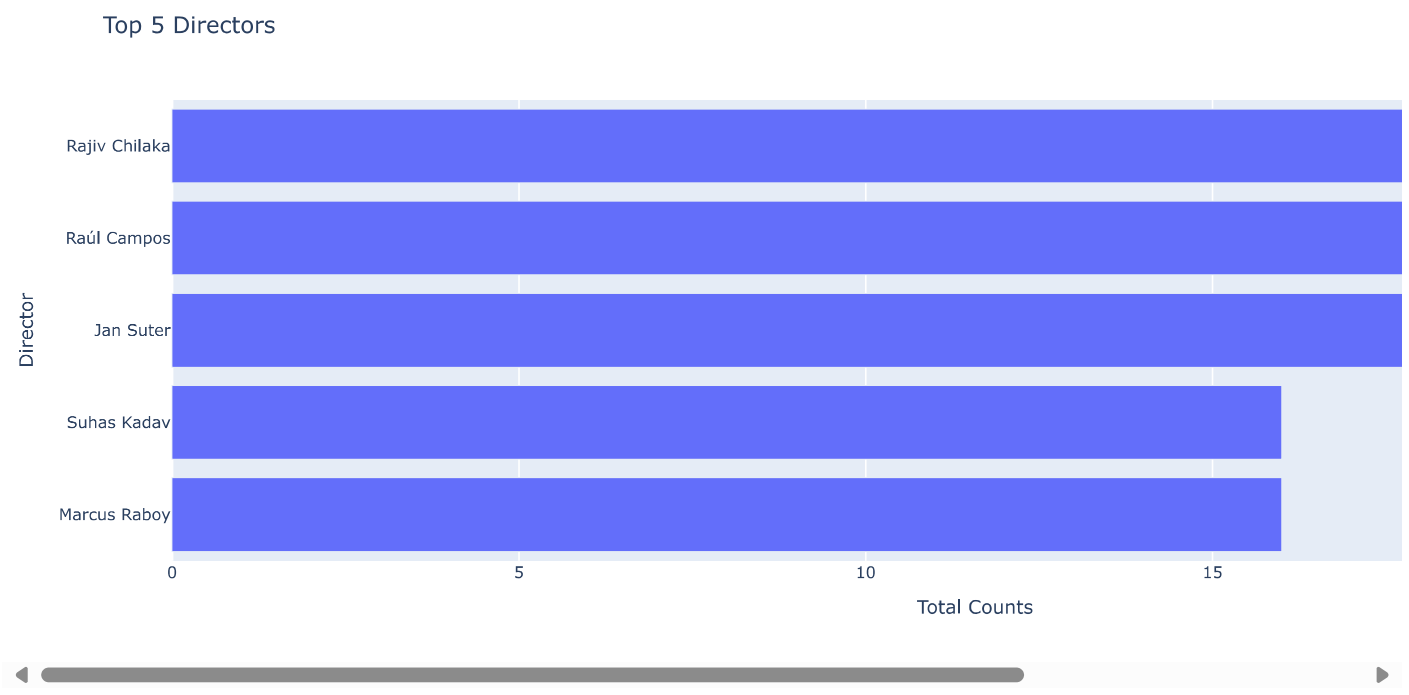
```
#will automatically extort the top 5 directors into any variable
top5Directors = directors.head()
print(top5Directors)
```

```
             Director  Total Counts
4020  Rajiv Chilaka            22
261        Jan Suter            18
4067     Raúl Campos            18
3235    Marcus Raboy            16
4651     Suhas Kadav            16
```

Analyzing the top 5 Directors at Netflix

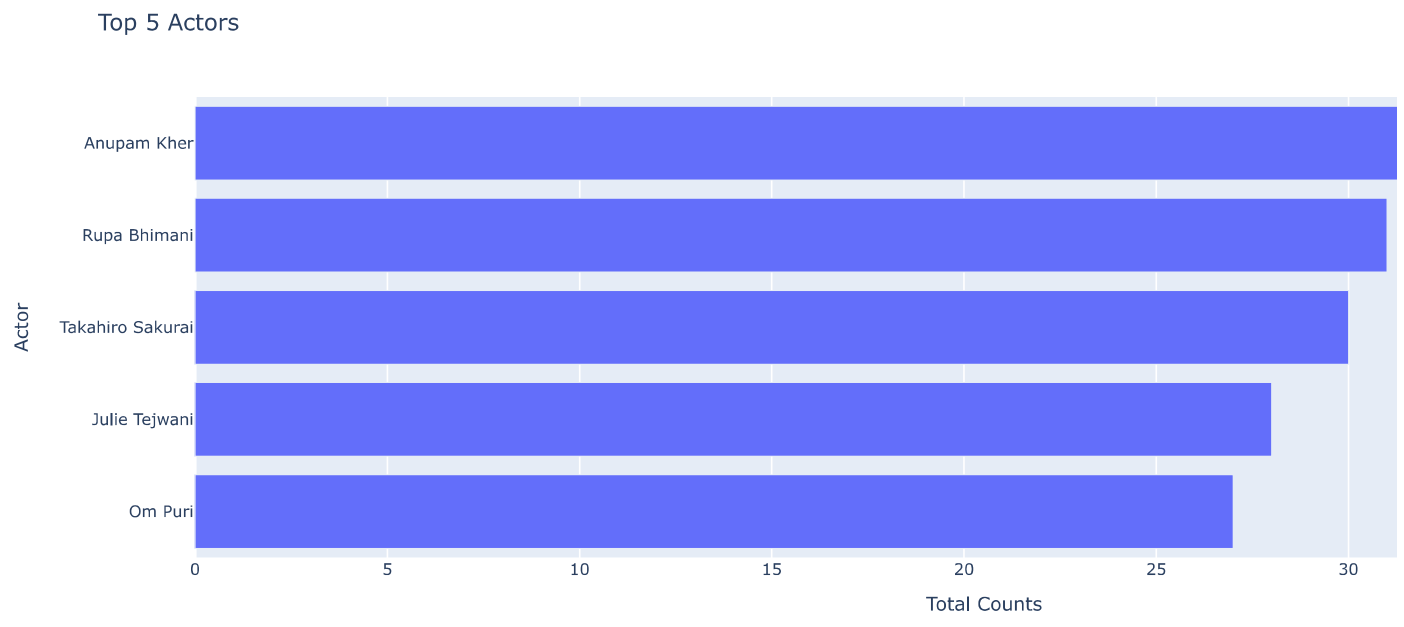
```
#create a bar chart to see how much content each director has created
top5Directors = top5Directors.sort_values(by=['Total Counts'])
print(top5Directors)
barChart = px.bar(top5Directors, x='Total Counts', y='Director', title='Top 5 Directors')
barChart.show()
```

```
             Director  Total Counts
3235    Marcus Raboy            16
4651     Suhas Kadav            16
261        Jan Suter            18
4067     Raúl Campos            18
4020  Rajiv Chilaka            22
```



Analyzing the Top 5 Actors

```
df['cast']= df['cast'].fillna('No cast specified')
cast_df = pd.DataFrame()
cast_df = df['cast'].str.split(',', expand=True).stack()
cast_df = cast_df.to_frame()
cast_df.columns = ['Actor']
actors = cast_df.groupby(['Actor']).size().reset_index(name='Total Counts')
actors = actors[actors.Actor != 'No cast specified']
actors = actors.sort_values(by=['Total Counts'], ascending = False)
top5Actors = actors.head()
top5Actors = top5Actors.sort_values(by=['Total Counts'])
barChart2 = px.bar(top5Actors, x='Total Counts', y='Actor', title='Top 5 Actors')
barChart2.show()
print(cast_df)
```



```
                          Actor
0    0       No cast specified
1    0              Ama Qamata
     1             Khosi Ngema
     2           Gail Mabalane
     3          Thabang Molaba
...                        ...
8806 3        Manish Chaudhary
     4            Meghna Malik
     5           Malkeet Rauni
     6          Anita Shabdish
     7   Chittaranjan Tripathy

[64951 rows x 1 columns]
```

Analyzing the Content produced on Netflix based on Years

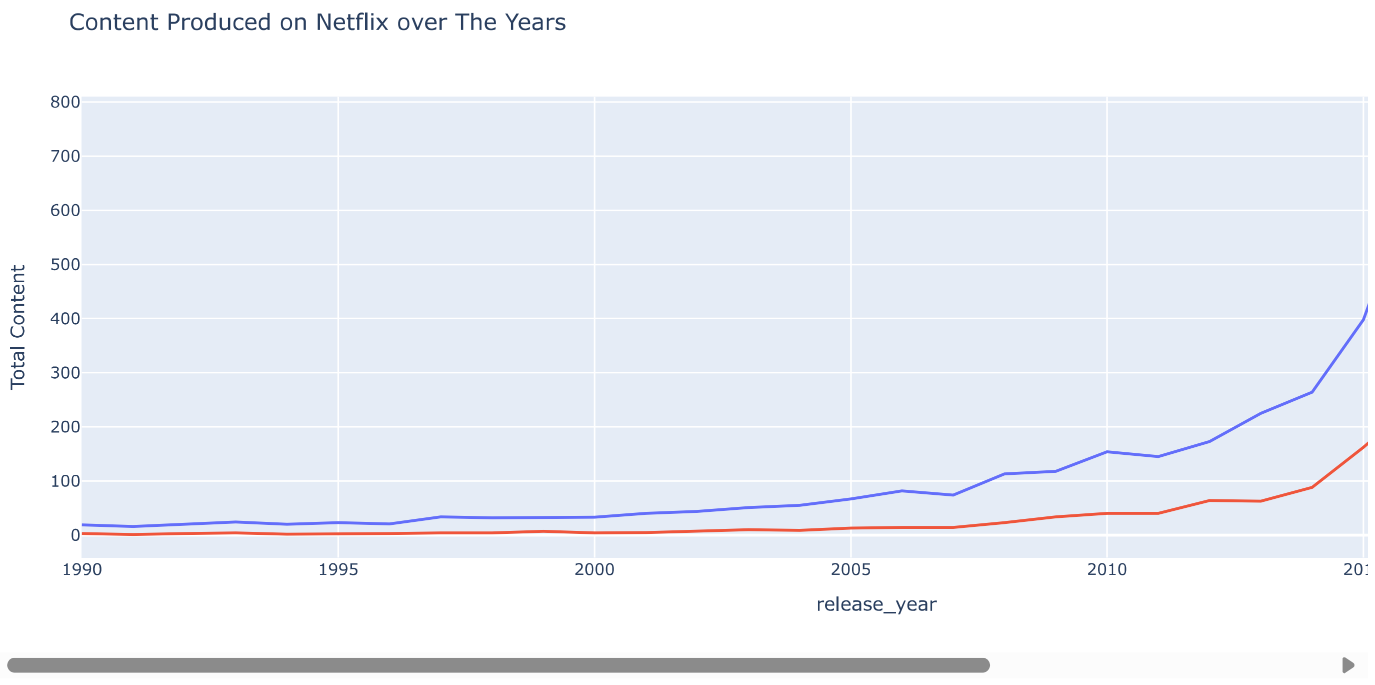
```
df1 = df[['release_year', 'type']]
df1 = df1.groupby(['release_year', 'type']).size().reset_index(name='Total Content')
df1 = df1.rename(columns={'type':'Content Type'})
print(df1)
df1 = df1[df1['release_year'] >= 1990]
df2 = df1[df1['release_year'] >= 2010]
```

```
     release_year Content Type  Total Content
0            1925      TV Show              1
1            1942        Movie              2
2            1943        Movie              3
3            1944        Movie              3
4            1945        Movie              3
..            ...          ...            ...
114          2019      TV Show            397
115          2020        Movie            517
116          2020      TV Show            436
117          2021        Movie            277
```

```
     118          2021      TV Show           315

     [119 rows x 3 columns]
```

```
graph = px.line(df1, x='release_year', y='Total Content', color='Content Type', title='Content Produced on Netflix over The Years')
graph.show()
```

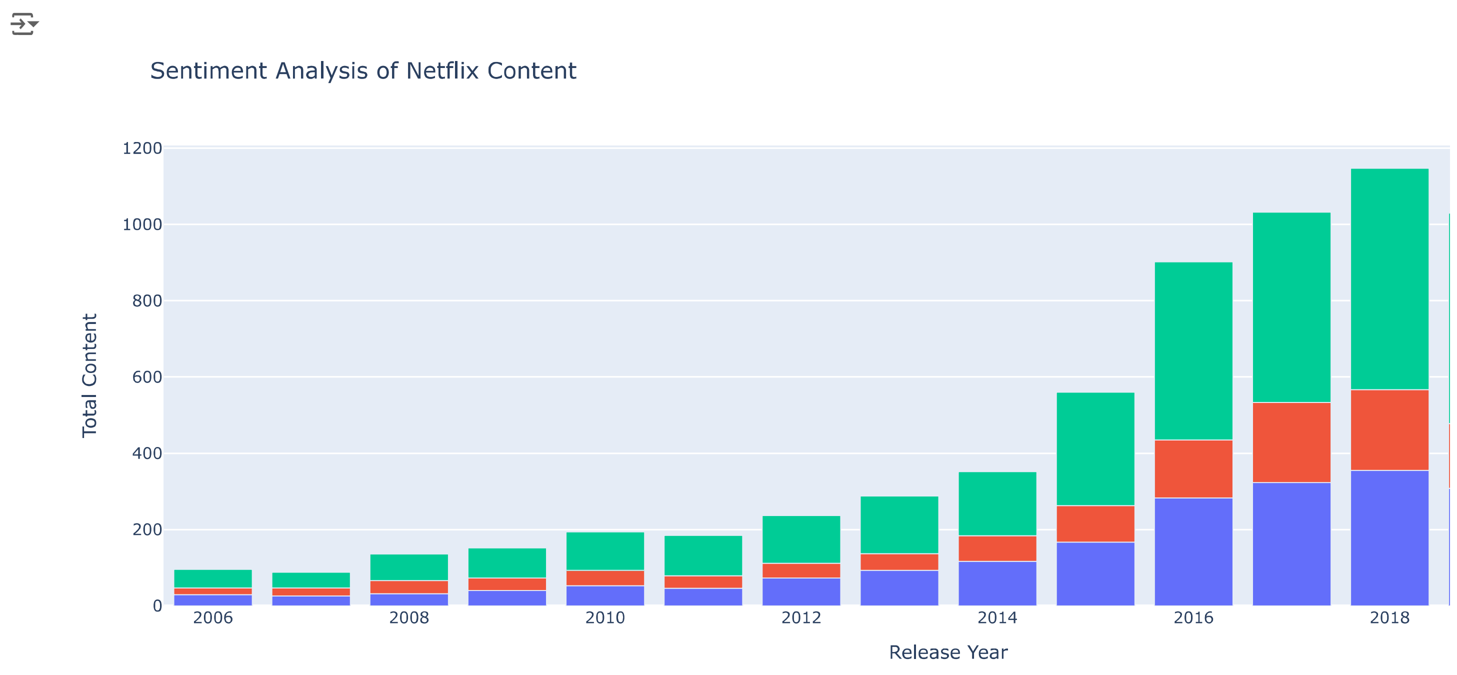


Sentiment Analysis of Netflix Content

```
df3 = df[['release_year', 'description']]
df3 = df.rename(columns = {'release_year':'Release Year', 'description' : 'Description'})
for index, row in df3.iterrows():
  d = row['Description']
  testimonial = TextBlob(d)
  p = testimonial.sentiment.polarity
  if p == 0:
    sent = 'Neutral'
  elif p > 0:
    sent = 'Positive'
  else:
    sent = 'Negative'
  df3.loc[[index, 2], 'Sentiment'] = sent #extort the sentiment

df3 = df3.groupby(['Release Year', 'Sentiment']).size().reset_index(name='Total Content')

df3 = df3[df3['Release Year']>2005]
barGraph = px.bar(df3, x='Release Year', y='Total Content', color='Sentiment', title='Sentiment Analysis of Netflix Content')
barGraph.show()
print(df3)
#
```



```
     Release Year Sentiment  Total Content
137          2006  Negative             29
138          2006   Neutral             18
139          2006  Positive             49
140          2007  Negative             26
141          2007   Neutral             21
142          2007  Positive             41
143          2008  Negative             32
144          2008   Neutral             34
145          2008  Positive             70
146          2009  Negative             40
147          2009   Neutral             33
148          2009  Positive             79
149          2010  Negative             53
150          2010   Neutral             40
151          2010  Positive            101
152          2011  Negative             46
153          2011   Neutral             33
154          2011  Positive            106
155          2012  Negative             73
156          2012   Neutral             39
157          2012  Positive            125
158          2013  Negative             93
159          2013   Neutral             44
160          2013  Positive            151
161          2014  Negative            117
162          2014   Neutral             67
163          2014  Positive            168
164          2015  Negative            167
165          2015   Neutral             96
166          2015  Positive            297
167          2016  Negative            283
168          2016   Neutral            152
169          2016  Positive            467
170          2017  Negative            323
171          2017   Neutral            210
172          2017  Positive            499
173          2018  Negative            355
174          2018   Neutral            212
175          2018  Positive            580
176          2019  Negative            308
177          2019   Neutral            170
178          2019  Positive            552
179          2020  Negative            273
180          2020   Neutral            161
181          2020  Positive            519
182          2021  Negative            164
183          2021   Neutral             85
184          2021  Positive            343
```