WebScraping YouTube's Data For Exploratory Data Analysis

Introduction

In this notebook, I had build an Python Project to Scrape YouTube data using YouTube Data API. Using YouTube API, I extracted the data and then load this data into a Python Pandas DataFrame and then analyze this data. Finally, I had build a simple visualization from this data using the Python Seaborn library.

I have started this project by first creating an YouTube API Key which is required to access the youtube data. Link to create YouTube API Key: https://console.cloud.google.com/

Once the API Key is generated, I had gone through the documentation given by google to use youtube API key to fetch the data.
Link to Google YouTube Data API documentation: https://developers.google.com/youtube/v3

Installing & Importing Required Python Libraries

In [ ]:
```
!pip install google-api-python-client
```

In [118…
```
from googleapiclient.discovery import build
import pandas as pd
import seaborn as sns
```

Extract Channel Details Of Top Data Analysts/Data Scientists

we extract details such as youtube channel name, total no of subscribers, total views and total number of videos posted by each channel. We gather these details for few Data Analyst/Data Scientist kind of channel and then compare these channel data with each other. We shall see who has the highest subscriber and who gets the most views and the amount of videos posted by these channels. We will be loading all of this data into a pandas dataframe and then analyze it. We will also generate some basic visualization using this data so we can easily compare these multiple channels.

In [119…
```
api_key = 'AIzaSyD84F20SxhA00pVk70rtpgg_PlArM7V8lo'
channel_ids =['UCnz-ZXXER4jOvuED5trXfEA',#techTFQ-OK
             'UCLLw7jmFsvfIVaUFsLs8mlQ',#Luke Barousse
             'UC7cs8q-gJRlGwj4A8OmCmXg',#AlexTheAnalyst
             'UC2UXDak6o7rBm23k3Vv5dww' #Tina Huang
             ]

youtube = build('youtube','v3', developerKey=api_key)
```

Comparing The Channel Statistics

In [133…
```
# Function to get the channel statistics
def get_channel_stats(youtube, channel_ids):
    all_data=[]
    request = youtube.channels().list(
               part='snippet,contentDetails,statistics',id=','.join(channel_ids))
    response = request.execute()
    for i in range(len(response['items'])):
        data = dict(Channel_name = response['items'][i]['snippet']['title'],
                    Subscriber = response['items'][i]['statistics']['subscriberCount'],
                    Views = response['items'][i]['statistics']['viewCount'],
                    Total_videos = response['items'][i]['statistics']['videoCount'],
                    Playlist_id = response['items'][i]['contentDetails']['relatedPlaylis
        all_data.append(data)

    return all_data
```

In [121…

```
#print the channel_stats
get_channel_stats(youtube, channel_ids)
```

Out[121]:

```
[{'Channel_name': 'Alex The Analyst',
  'Subscriber': '595000',
  'Views': '24713799',
  'Total_videos': '251',
  'Playlist_id': 'UU7cs8q-gJRlGwj4A8OmCmXg'},
 {'Channel_name': 'Tina Huang',
  'Subscriber': '570000',
  'Views': '25532011',
  'Total_videos': '139',
  'Playlist_id': 'UU2UXDak6o7rBm23k3Vv5dww'},
 {'Channel_name': 'Luke Barousse',
  'Subscriber': '384000',
  'Views': '18091155',
  'Total_videos': '141',
  'Playlist_id': 'UULLw7jmFsvfIVaUFsLs8mlQ'},
 {'Channel_name': 'techTFQ',
  'Subscriber': '240000',
  'Views': '12378634',
  'Total_videos': '98',
  'Playlist_id': 'UUnz-ZXXER4jOvuED5trXfEA'}]
```

In [21]:

```
# Save the output in the varibale
channel_stats = get_channel_stats(youtube, channel_ids)
```

In [22]:

```
# Convert it into PandasDataframe
channel_data = pd.DataFrame(channel_stats)
```

In [122…

```
# See the data in rows and columns format
channel_data
```

Out[122]:

	Channel_name	Subscriber	Views	Total_videos	Playlist_id
0	Luke Barousse	384000	18091155	141	UULLw7jmFsvfIVaUFsLs8mlQ
1	Alex The Analyst	595000	24713799	251	UU7cs8q-gJRlGwj4A8OmCmXg
2	Tina Huang	570000	25532011	139	UU2UXDak6o7rBm23k3Vv5dww
3	techTFQ	240000	12378634	98	UUnz-ZXXER4jOvuED5trXfEA

In [123…

```
# Check the column types
channel_data.dtypes
```

Out[123]:

```
Channel_name    object
Subscriber      object
Views           object
Total_videos    object
Playlist_id     object
dtype: object
```

In [124…

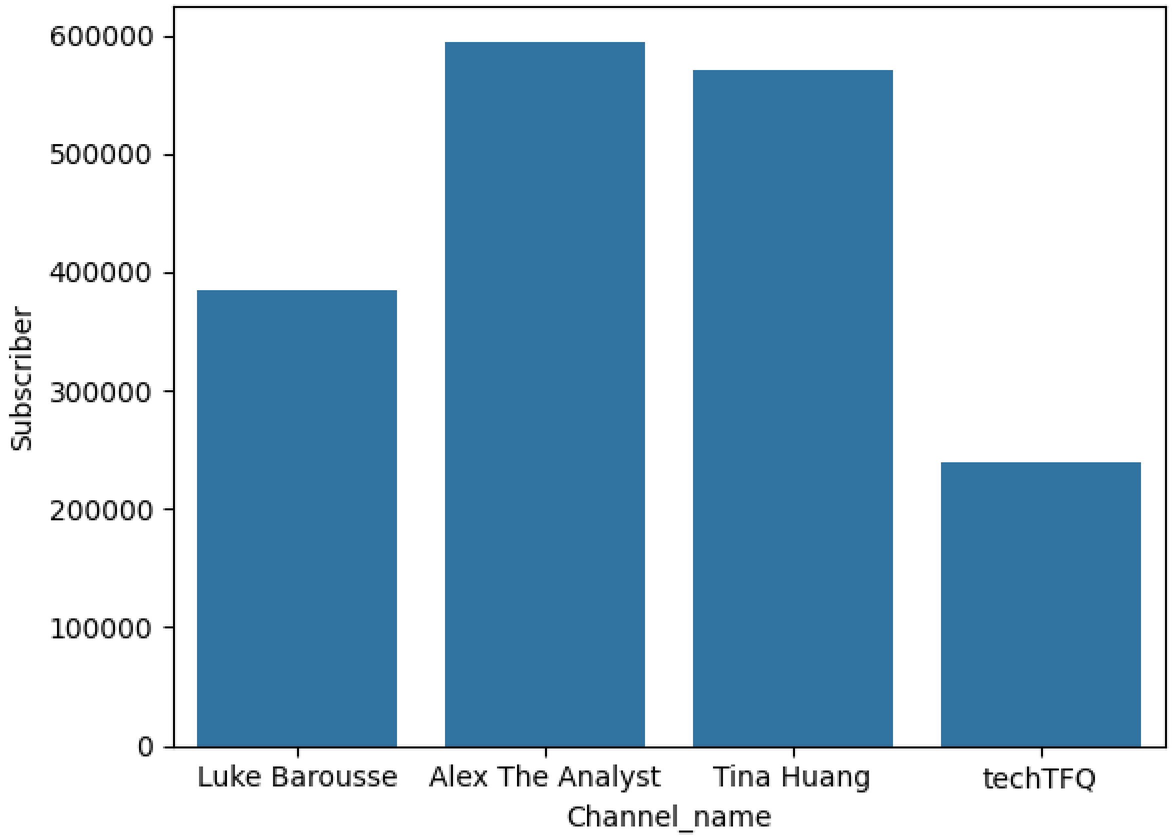
```
# Change the data types
```

```
channel_data['Subscriber'] = pd.to_numeric(channel_data['Subscriber'])
channel_data['Views'] = pd.to_numeric(channel_data['Views'])
channel_data['Total_videos'] = pd.to_numeric(channel_data['Total_videos'])
```

In [125…
```
# Re-check data types after changing data types
channel_data.dtypes
```

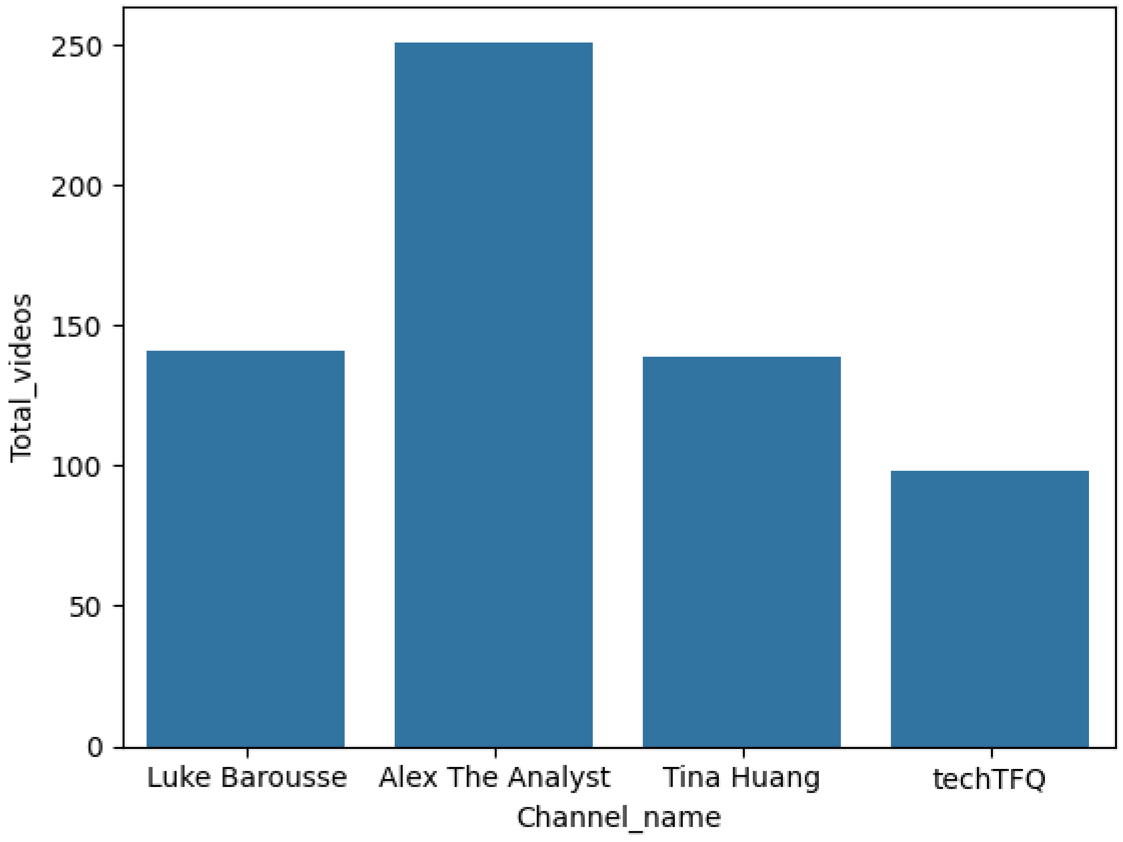
Out[125]:
```
Channel_name     object
Subscriber        int64
Views             int64
Total_videos      int64
Playlist_id      object
dtype: object
```

In [127…
```
# Create an barplot to easily compare the subscriber count
ax = sns.barplot(x='Channel_name',y='Subscriber',data=channel_data)
```



From the above chart we can see that, **'Alex The Analyst'** has the maximum number of subscribers compre to the other channels.

In [128…
```
# Create an barplot to easily compare the video count
ax = sns.barplot(x='Channel_name',y='Total_videos',data=channel_data)
```



From the above chart we can see that, Alex had uploaded maximum number of videos compare to the other youtubers.

Now we will do further analysis of 'Alex The Analyst' channel by analyzing it's all videos data.

Analysis Of 'Alex The Analyst' Channel

Now we will build a logic to extract video details from **'Alex The Analyst'**. We shall extract details such as video title, total views each video has got, total number of likes, and comments each video has got. We will then analyze this data by loading it into a pandas dataframe. At the end we will create some simple visualization using Seaborn python library.

In [129…
```
# Extracting playlist_id for 'Alex The Analyst' channel
playlist_id = channel_data.loc[channel_data['Channel_name']=='Alex The Analyst','Playlis
```

In [130…
```
# Print the playlist_id
playlist_id
```

Out[130]:
```
'UU7cs8q-gJRlGwj4A8OmCmXg'
```

In [134…
```
# Function to get the video statistics for 'Alex The Analyst' channel

def get_video_ids(youtube, playlist_id):

    request = youtube.playlistItems().list(
                part='contentDetails',
                playlistId = playlist_id,
                maxResults = 50)
    response = request.execute()

    video_ids = []

    for i in range(len(response['items'])):
        video_ids.append(response['items'][i]['contentDetails']['videoId'])

    next_page_token = response.get('nextPageToken')
    more_pages = True

    while more_pages:
        if next_page_token is None:
            more_pages = False
        else:
            request = youtube.playlistItems().list(
                part='contentDetails',
                playlistId = playlist_id,
                maxResults = 50,
                pageToken = next_page_token)

            response = request.execute()

            for i in range(len(response['items'])):
                video_ids.append(response['items'][i]['contentDetails']['videoId'])
                next_page_token = response.get('nextPageToken')

    return video_ids
```

In [135…
```
# storing the output in the variable
video_ids = get_video_ids(youtube, playlist_id)
```

In [136…
```
# See the total Videos uploaded by Alex
len(video_ids)
```

Out[136]:
```
251
```

In [137…
```
# Function to extract some stats for each video

def get_video_details(youtube, video_ids):

    all_video_stats = []
    for i in range(0, len(video_ids), 50):
        request = youtube.videos().list(
                    part='snippet,statistics',
                    id =','.join(video_ids[i:i+50]))
        response = request.execute()

        for video in response['items']:
            video_stats = dict(Title = video['snippet']['title'],
                               Published_date = video['snippet']['publishedAt'],
                               Views = video['statistics']['viewCount'],
                               Likes = video['statistics']['likeCount'],
                               Comments = video['statistics']['commentCount'])

            all_video_stats.append(video_stats)

    return all_video_stats
```

In [138…
```
# store the video stats in the variable
video_details = get_video_details(youtube, video_ids)
```

In [139…
```
# store the video data in the DataFrame format
video_data = pd.DataFrame(video_details)
```

In [140…
```
# See the top 5 rows
video_data.head()
```

Out[140]:

	Title	Published_date	Views	Likes	Comments

0	#DataAnalyst #DataCareers #DataAnalysis	2023-10-05T12:13:45Z	3849	292	11
1	Data Analyst vs Data Engineer | Responsibiliti...	2023-10-03T12:00:26Z	21133	473	37
2	Data Analyst Q/A Livestream | September Livest...	2023-09-28T16:09:03Z	7759	268	19
3	#Salary #DataAnalyst #SalaryNegotiation	2023-09-28T12:37:36Z	7162	461	5
4	AI and Analytics with SAS | SAS Explore Recap	2023-09-26T12:00:49Z	8086	267	16

In [141…

```
# See the data types of columns
video_data.dtypes
```

Out[141]:

```
Title             object
Published_date    object
Views             object
Likes             object
Comments          object
dtype: object
```

In [142…

```
# Change the data types
video_data['Published_date'] = pd.to_datetime(video_data['Published_date']).dt.date
video_data['Views'] = pd.to_numeric(video_data['Views'])
video_data['Likes'] = pd.to_numeric(video_data['Likes'])
video_data['Comments'] = pd.to_numeric(video_data['Comments'])
```

In [143…

```
# Again see the data
video_data.head()
```

Out[143]:

	Title	Published_date	Views	Likes	Comments
0	#DataAnalyst #DataCareers #DataAnalysis	2023-10-05	3849	292	11
1	Data Analyst vs Data Engineer | Responsibiliti...	2023-10-03	21133	473	37
2	Data Analyst Q/A Livestream | September Livest...	2023-09-28	7759	268	19
3	#Salary #DataAnalyst #SalaryNegotiation	2023-09-28	7162	461	5
4	AI and Analytics with SAS | SAS Explore Recap	2023-09-26	8086	267	16

In [144…

```
# Re-check the data types
video_data.dtypes
```

Out[144]:

```
Title             object
Published_date    object
Views              int64
Likes              int64
Comments           int64
dtype: object
```

In [146…

```
# Extract top 10 videos by views
top10_videos = video_data.sort_values(by='Views', ascending=False).head(10)
```

In [148…

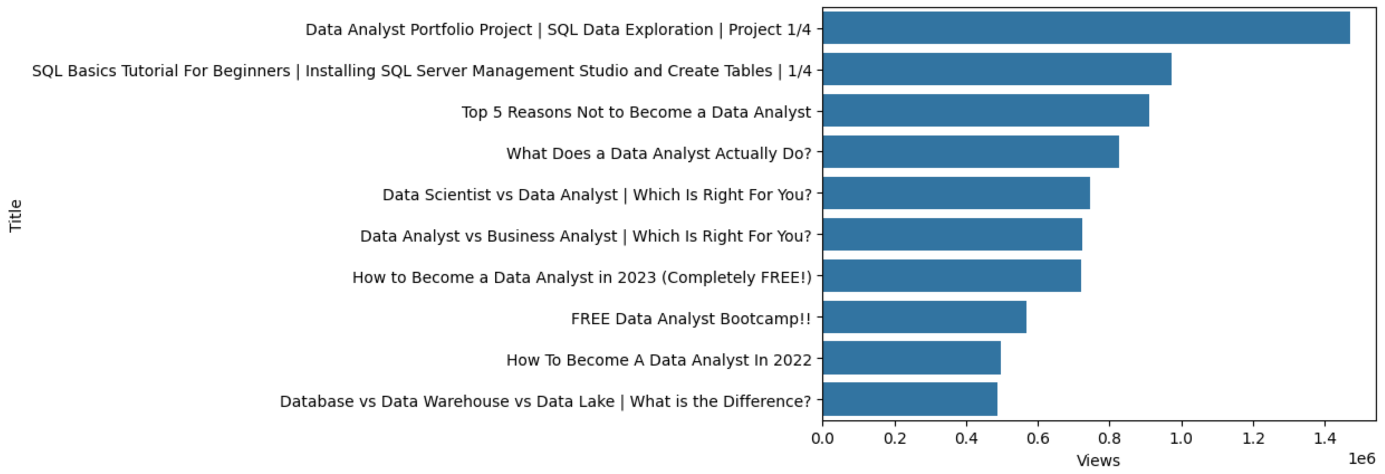
```
# See the top 10 video details
top10_videos
```

Out[148]:

	Title	Published_date	Views	Likes	Comments
156	Data Analyst Portfolio Project | SQL Data Expl...	2021-05-04	1470093	27987	3319
239	SQL Basics Tutorial For Beginners | Installing...	2020-03-01	972906	13829	1562
214	Top 5 Reasons Not to Become a Data Analyst	2020-08-12	911465	19647	1053
216	What Does a Data Analyst Actually Do?	2020-07-29	827151	16583	449

222	Data Scientist vs Data Analyst | Which Is Righ...	2020-06-10	746119	23449	848
219	Data Analyst vs Business Analyst | Which Is Ri...	2020-07-03	724956	18710	570
60	How to Become a Data Analyst in 2023 (Complete...	2023-01-31	721102	27474	844
59	FREE Data Analyst Bootcamp!!	2023-02-07	569807	22190	1135
127	How To Become A Data Analyst In 2022	2022-01-04	495676	22216	642
110	Database vs Data Warehouse vs Data Lake | What...	2022-04-26	488136	13654	229

In [149…
```
# Created an barplot to visually compare Top videos
ax1 = sns.barplot(x='Views',y='Title', data=top10_videos)
```



The 'SQL' related videos uploaded by Alex got the maximum number of views.

In [150…
```
# Extract Month and Year From Date
video_data['Published_Month'] = pd.to_datetime(video_data['Published_date']).dt.strftime
video_data['Published_Year'] = pd.to_datetime(video_data['Published_date']).dt.strftime(
```

In [151…
```
# See the data
video_data.head()
```

Out[151]:

	Title	Published_date	Views	Likes	Comments	Published_Month	Published_Year
0	#DataAnalyst #DataCareers #DataAnalysis	2023-10-05	3849	292	11	Oct	2023
1	Data Analyst vs Data Engineer | Responsibiliti...	2023-10-03	21133	473	37	Oct	2023
2	Data Analyst Q/A Livestream | September Livest...	2023-09-28	7759	268	19	Sep	2023
3	#Salary #DataAnalyst #SalaryNegotiation	2023-09-28	7162	461	5	Sep	2023
4	AI and Analytics with SAS | SAS Explore Recap	2023-09-26	8086	267	16	Sep	2023

In [152…
```
# See the data types
video_data.dtypes
```

Out[152]:
```
Title              object
Published_date     object
Views               int64
Likes               int64
```

```
Comments             int64
Published_Month     object
Published_Year      object
dtype: object
```

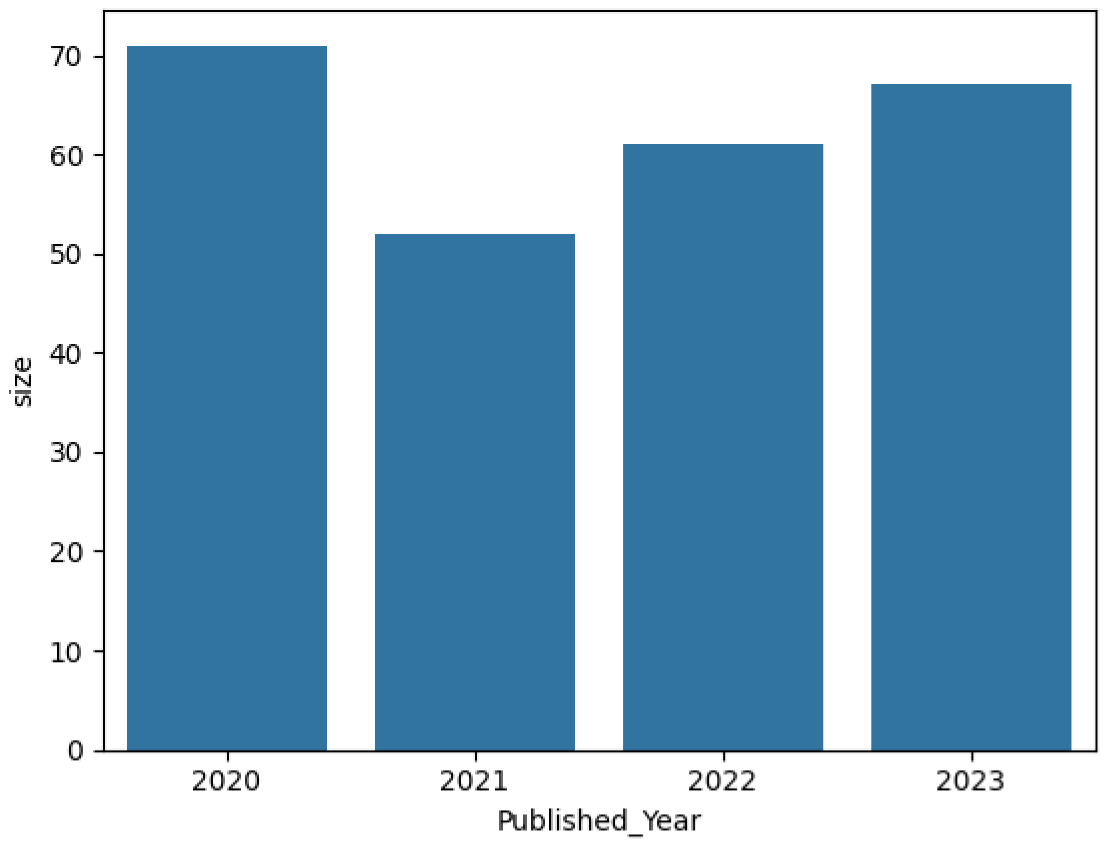
In [153…
```
# Get the year wise video count
videos_per_year = video_data.groupby('Published_Year',as_index=False).size()
```

In [154…
```
videos_per_year
```

Out[154]:

	Published_Year	size
0	2020	71
1	2021	52
2	2022	61
3	2023	67

In [158…
```
# Plot the barplot to easily compare the yearly uploaded video count
ax2 = sns.barplot(x='Published_Year',y='size',data=videos_per_year)
```



In [156…
```
# Get the month wise video count
videos_per_month = video_data.groupby('Published_Month',as_index=False).size()
```

In [160…
```
videos_per_month
```

Out[160]:

	Published_Month	size
0	Apr	19
1	Aug	28
2	Dec	18
3	Feb	19

4	Jan	24
5	Jul	18
6	Jun	18
7	Mar	23
8	May	22
9	Nov	17
10	Oct	22
11	Sep	23

In [161…
```
sort_order = ['Jan','Feb','Mar','Apr','May','Jun','Jul','Aug','Sep','Oct','Nov','Dec']
```

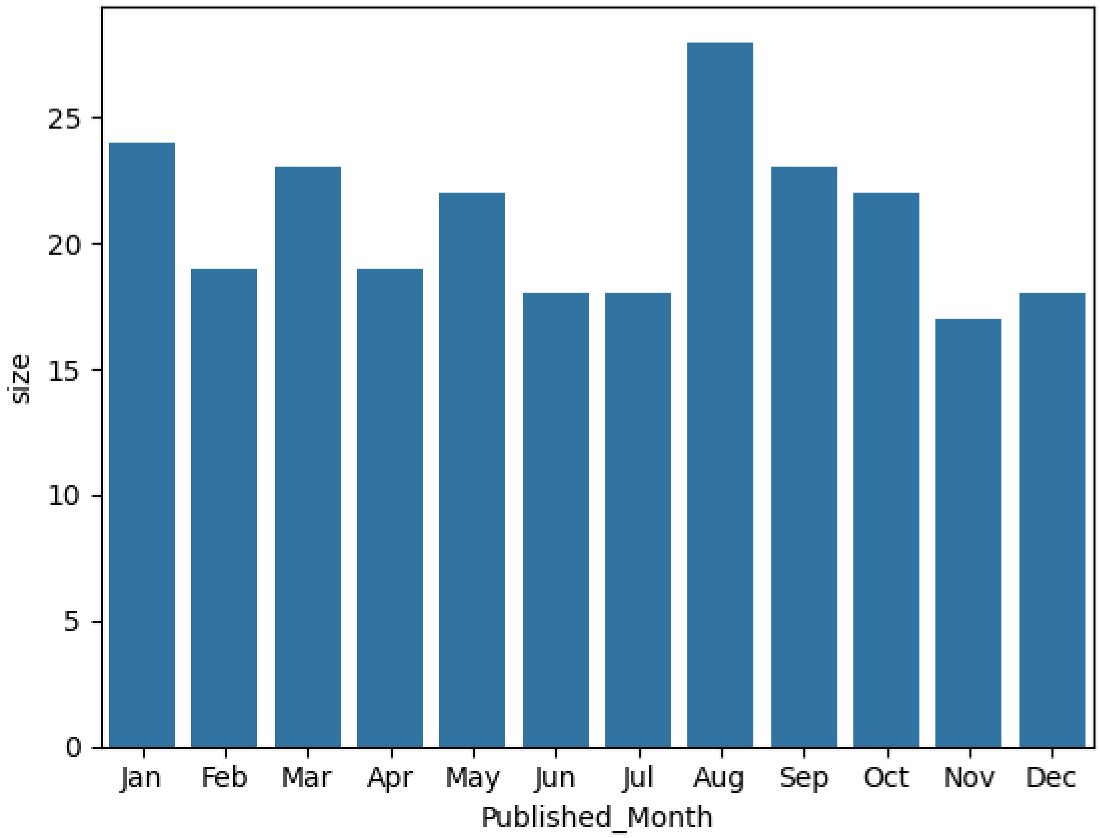
In [162…
```
videos_per_month.index = pd.CategoricalIndex(videos_per_month['Published_Month'],categor
```

In [163…
```
videos_per_month.sort_index()
```

Out[163]:

	Published_Month	**size**
Published_Month		
Jan	Jan	24
Feb	Feb	19
Mar	Mar	23
Apr	Apr	19
May	May	22
Jun	Jun	18
Jul	Jul	18
Aug	Aug	28
Sep	Sep	23
Oct	Oct	22
Nov	Nov	17
Dec	Dec	18

In [164…
```
# Plot the barplot to easily compare the monthly uploaded video count
ax3 = sns.barplot(x='Published_Month',y='size',data=videos_per_month.sort_index())
```


size
0
5
10
15
20
25
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Published_Month

```
In [ ]:
```