```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

#importing the dataset
df = pd.read_csv(r'video_id_info.csv', on_bad_lines='skip',
engine='python', quoting=3)

df.head()

{"type":"dataframe","variable_name":"df"}

## lets find out missing values in your data
df.isnull().sum()

video_id          0
comment_text     19
likes           259
replies         300
dtype: int64

## drop missing values as we have very few & lets update dataframe as
well..
df.dropna(inplace=True)

df.isnull().sum()

video_id        0
comment_text    0
likes           0
replies         0
dtype: int64
```

Perform Sentiment Analysis

Sentiment analysis is a way for computers to understand and analyze the emotions expressed in text, like whether it's positive, negative, or neutral. example: 1.This video is quite helpful-->Positive sentiment [0,1] more it will close to 1 it will positve sentiment 2.Uable to understand the topic -->Negative sentiment[-1] 3. I'm attending the lecture this afternoon.-->Neutral sentiment[0] The polarity range refers to the scale used in sentiment analysis to measure the degree of positivity or negativity in text, typically ranging from -1 to 1

TextBlob is a Python library for processing textual data. It provides a simple API for common natural language processing (NLP) tasks

```
#!pip install textblob
import sys #It's called "sys" because it provides access to system-
specific parameters and functions.
!{sys.executable} -m pip install textblob
```

```
Requirement already satisfied: textblob in
/usr/local/lib/python3.10/dist-packages (0.17.1)
Requirement already satisfied: nltk>=3.1 in
/usr/local/lib/python3.10/dist-packages (from textblob) (3.9.1)
Requirement already satisfied: click in
/usr/local/lib/python3.10/dist-packages (from nltk>=3.1->textblob)
(8.1.7)
Requirement already satisfied: joblib in
/usr/local/lib/python3.10/dist-packages (from nltk>=3.1->textblob)
(1.4.2)
Requirement already satisfied: regex>=2021.8.3 in
/usr/local/lib/python3.10/dist-packages (from nltk>=3.1->textblob)
(2024.9.11)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-
packages (from nltk>=3.1->textblob) (4.66.6)
```

```
from textblob import TextBlob
```

```
df.head(6)
```

```
{"type":"dataframe","variable_name":"df"}
```

```
df.shape
```

```
(564993, 4)
```

Creating a new DataFrame (sample_df) by selecting the first 1000 rows of an existing DataFrame (df). This can be useful for working with a smaller subset of data, such as when you want to perform quick analyses or tests without using the entire dataset.

```
sample_df = df[0:1000]
```

```
 # normal text box
TextBlob("Logan Paul it's yo big day !!!!!!")
```

```
TextBlob("Logan Paul it's yo big day !!!!!!")
```

```
#attribute
TextBlob("Logan Paul it's yo big day !!!!!!").sentiment
```

```
Sentiment(polarity=0.0, subjectivity=0.1)
```

```
### its a neutral sentence !
TextBlob("Logan Paul it's yo big day !!!!!!").sentiment.polarity
```

```
0.0
```

```
#performing sentiment for each row of comment_text'
#polarity = [] #--black list
```

```
#for comment in df['comment_text']:
    #TextBlob(comment).sentiment.polarity
```

```
    #polarity.append(TextBlob(comment).sentiment.polarity)

#if there is black txt then will get the exception error . so avoid
the exception we have to use try exception block

#syntax
#try:
    # Code that might raise an exception
 #except :
    # Code to handle the exception

!pip install tqdm
!pip install textblob
from textblob import TextBlob
from tqdm import tqdm  # For progress tracking

# Initialize an empty list to store polarity scores
polarity = []

# Iterate through each comment with a progress bar
for comment in tqdm(df['comment_text'], desc="Processing comments"):
    try:
        # Calculate sentiment polarity for the comment
        polarity_score = TextBlob(comment).sentiment.polarity
        polarity.append(polarity_score)
    except Exception as e:
        # Handle exceptions by appending a default polarity of 0
        polarity.append(0)
        print(f"Error processing comment: {comment}. Exception: {e}")

# Add the polarity list as a new column to the DataFrame (optional)
df['polarity'] = polarity
```

```
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-
packages (4.66.6)
Requirement already satisfied: textblob in
/usr/local/lib/python3.10/dist-packages (0.17.1)
Requirement already satisfied: nltk>=3.1 in
/usr/local/lib/python3.10/dist-packages (from textblob) (3.9.1)
Requirement already satisfied: click in
/usr/local/lib/python3.10/dist-packages (from nltk>=3.1->textblob)
(8.1.7)
Requirement already satisfied: joblib in
/usr/local/lib/python3.10/dist-packages (from nltk>=3.1->textblob)
(1.4.2)
Requirement already satisfied: regex>=2021.8.3 in
/usr/local/lib/python3.10/dist-packages (from nltk>=3.1->textblob)
(2024.9.11)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-
packages (from nltk>=3.1->textblob) (4.66.6)
```

```
Processing comments: 100%|██████████| 564993/564993 [02:28<00:00,
3806.33it/s]
```

```
polarity = []

for comment in df['comment_text']:
    try:
        polarity.append(TextBlob(comment).sentiment.polarity)
    except:
        polarity.append(0);

len(polarity)
```

```
564993
```

Inserting polarity values into comments dataframe while defining feature name as "polarity"

```
df['polarity']  = polarity

df.head(5)
```

```
{"type":"dataframe","variable_name":"df"}
```

Wordcloud Analysis of your data: #Word cloud analysis is a visual representation technique that displays the most frequently occurring words in a text dataset

```
filter1 = df['polarity']==1
comments_positive=df[filter1]

filter2 = df['polarity']==-1
comments_negative= df[filter2]

#!pip install wordcloud
import sys
!{sys.executable} -m pip install wordcloud
```

```
Requirement already satisfied: wordcloud in
/usr/local/lib/python3.10/dist-packages (1.9.4)
Requirement already satisfied: numpy>=1.6.1 in
/usr/local/lib/python3.10/dist-packages (from wordcloud) (1.26.4)
Requirement already satisfied: pillow in
/usr/local/lib/python3.10/dist-packages (from wordcloud) (11.0.0)
Requirement already satisfied: matplotlib in
/usr/local/lib/python3.10/dist-packages (from wordcloud) (3.8.0)
Requirement already satisfied: contourpy>=1.0.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib->wordcloud)
(1.3.1)
Requirement already satisfied: cycler>=0.10 in
/usr/local/lib/python3.10/dist-packages (from matplotlib->wordcloud)
(0.12.1)
```

```
Requirement already satisfied: fonttools>=4.22.0 in
/usr/local/lib/python3.10/dist-packages (from matplotlib->wordcloud)
(4.55.1)
Requirement already satisfied: kiwisolver>=1.0.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib->wordcloud)
(1.4.7)
Requirement already satisfied: packaging>=20.0 in
/usr/local/lib/python3.10/dist-packages (from matplotlib->wordcloud)
(24.2)
Requirement already satisfied: pyparsing>=2.3.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib->wordcloud)
(3.2.0)
Requirement already satisfied: python-dateutil>=2.7 in
/usr/local/lib/python3.10/dist-packages (from matplotlib->wordcloud)
(2.8.2)
Requirement already satisfied: six>=1.5 in
/usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7-
>matplotlib->wordcloud) (1.16.0)
```

Stopwords are common words like "the," "is," and "and" that are often removed from text during analysis because they don't carry significant meaning.

```
from wordcloud import WordCloud , STOPWORDS

set(STOPWORDS)
#turns the stopwords list into a unique collection of words for faster
processing.
```

```
{'a',
 'about',
 'above',
 'after',
 'again',
 'against',
 'all',
 'also',
 'am',
 'an',
 'and',
 'any',
 'are',
 "aren't",
 'as',
 'at',
 'be',
 'because',
 'been',
 'before',
 'being',
```

```
 'below',
 'between',
 'both',
 'but',
 'by',
 'can',
 "can't",
 'cannot',
 'com',
 'could',
 "couldn't",
 'did',
 "didn't",
 'do',
 'does',
 "doesn't",
 'doing',
 "don't",
 'down',
 'during',
 'each',
 'else',
 'ever',
 'few',
 'for',
 'from',
 'further',
 'get',
 'had',
 "hadn't",
 'has',
 "hasn't",
 'have',
 "haven't",
 'having',
 'he',
 "he'd",
 "he'll",
 "he's",
 'hence',
 'her',
 'here',
 "here's",
 'hers',
 'herself',
 'him',
 'himself',
 'his',
 'how',
```

```
 "how's",
 'however',
 'http',
 'i',
 "i'd",
 "i'll",
 "i'm",
 "i've",
 'if',
 'in',
 'into',
 'is',
 "isn't",
 'it',
 "it's",
 'its',
 'itself',
 'just',
 'k',
 "let's",
 'like',
 'me',
 'more',
 'most',
 "mustn't",
 'my',
 'myself',
 'no',
 'nor',
 'not',
 'of',
 'off',
 'on',
 'once',
 'only',
 'or',
 'other',
 'otherwise',
 'ought',
 'our',
 'ours',
 'ourselves',
 'out',
 'over',
 'own',
 'r',
 'same',
 'shall',
 "shan't",
```

```
 'she',
 "she'd",
 "she'll",
 "she's",
 'should',
 "shouldn't",
 'since',
 'so',
 'some',
 'such',
 'than',
 'that',
 "that's",
 'the',
 'their',
 'theirs',
 'them',
 'themselves',
 'then',
 'there',
 "there's",
 'therefore',
 'these',
 'they',
 "they'd",
 "they'll",
 "they're",
 "they've",
 'this',
 'those',
 'through',
 'to',
 'too',
 'under',
 'until',
 'up',
 'very',
 'was',
 "wasn't",
 'we',
 "we'd",
 "we'll",
 "we're",
 "we've",
 'were',
 "weren't",
 'what',
 "what's",
 'when',
```

```
 "when's",
 'where',
 "where's",
 'which',
 'while',
 'who',
 "who's",
 'whom',
 'why',
 "why's",
 'with',
 "won't",
 'would',
 "wouldn't",
 'www',
 'you',
 "you'd",
 "you'll",
 "you're",
 "you've",
 'your',
 'yours',
 'yourself',
 'yourselves'}
```

```
df['comment_text']
```

```
0                         "Logan Paul it's yo big day !!!!!"
1         "I've been following you from the start of you...
2                       "Say hi to Kong and maverick for me"
3                                    "MY FAN . attendance"
4                                           "trending ☺"
                                ...
565288                                              "Лучшая"
565289    "qu'est ce que j'aimerais que tu viennes à Roa...
565290                           "Ven a mexico! ☺ te amo LP"
565291                                    "Islığı yeter..."
565292    "Kocham tą piosenkę😍❤❤❤byłam zakochana po uszy...
Name: comment_text, Length: 564993, dtype: object
```

```
type(df['comment_text'])
```

```
pandas.core.series.Series
```

For wordcloud , we need to frame our 'comment_text' feature into string. joins all the text data from the 'comment_text' column in the DataFrame 'comments_positive' into a single string, separated by spaces.

```
total_comments_positive = ' '.join(comments_positive['comment_text'])
```

This line of code creates a word cloud from the text data in `total_comments_positive`, using a predefined set of stopwords to filter out common words that don't carry significant meaning.

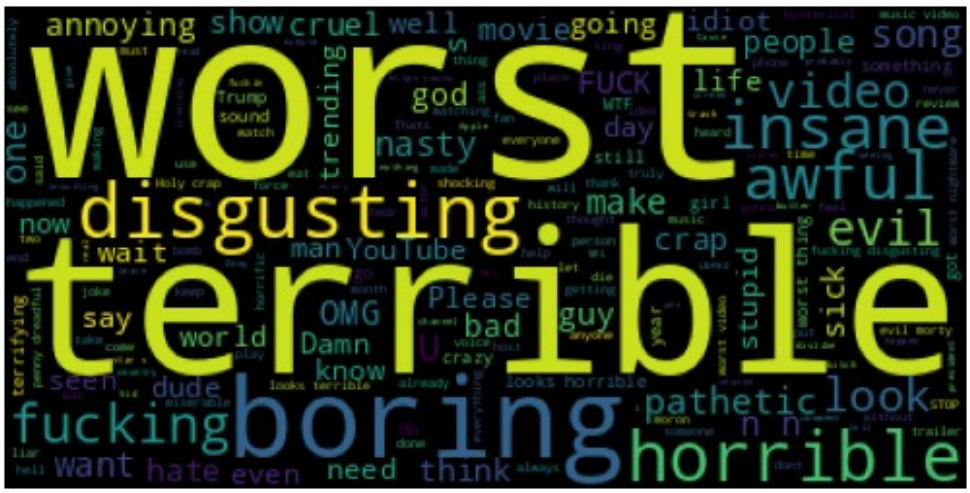
```
wordcloud =
WordCloud(stopwords=set(STOPWORDS)).generate(total_comments_positive)
```

The imshow() function in matplotlib is used to display images, and in this case, it's used to display the word cloud generated by the WordCloud library.

```
plt.imshow(wordcloud)
plt.axis('off')

(-0.5, 399.5, 199.5, -0.5)
```

Conclusion: Positive Users are emphasizing more on best , awesome , perfect , amazing , look , happy etc..

```
total_comments_negative = ' '.join(comments_negative['comment_text'])

wordcloud =
WordCloud(stopwords=set(STOPWORDS)).generate(total_comments_negative)

plt.imshow(wordcloud)
plt.axis('off')

(-0.5, 399.5, 199.5, -0.5)
```



Word cloud analysis is a visual representation technique that displays the most frequently occurring words in a text dataset

Perform Emoji's Analysis

```
#!pip install emoji==2.10.1
import sys
!{sys.executable} -m pip install emoji==2.10.1
## 2.10.0 is a most stable version till date , hence installing this
version makes sense !
```

```
Collecting emoji==2.10.1
  Downloading emoji-2.10.1-py2.py3-none-any.whl.metadata (5.3 kB)
Downloading emoji-2.10.1-py2.py3-none-any.whl (421 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0.0/421.5 kB ? eta -:--:--
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 421.5/421.5 kB 24.1 MB/s eta
0:00:00
oji
Successfully installed emoji-2.10.1
```

```
import emoji

emoji.__version__
```

```
{"type":"string"}
```

```
df['comment_text'].head(6)
```

```
0                  "Logan Paul it's yo big day !!!!!!"
1    "I've been following you from the start of you...
2                 "Say hi to Kong and maverick for me"
3                              "MY FAN . attendance"
4                                       "trending ☺"
5                           "#1 on trending AYYEEEEE"
Name: comment_text, dtype: object
```

```
comment = 'trending 😉'
```

The code snippet you provided is a list comprehension that filters out characters from a string (comment) if they are present in the emoji. EMOJI_DATA dictionary. It's a way to extract emojis from a text string.

```
[char for char in comment if char in emoji.EMOJI_DATA]
```

```
['😉']
```

```
## lets try to write above code in a more simpler & readable way :
emoji_list = []

for char in comment:
    if char in emoji.EMOJI_DATA:
        emoji_list.append(char)

emoji_list
```

```
['😉']
```

```
all_emojis_list = []

for comment in df['comment_text'].dropna(): ## in case u have missing values , call dropna()
    for char in comment:
        if char in emoji.EMOJI_DATA:
            all_emojis_list.append(char)

all_emojis_list[0:10]# 1st 10 emojis
```

```
['‼', '‼', '‼', '😉', '😭', '"', '', '❤', '😍', '']
```

Now we have to compute frequencies of each & every emoji in "all_emojis_list"..

```
from collections import Counter # collection package

Counter(all_emojis_list).most_common(10)
```

```
[('😂', 33283),
 ('❤', 27897),
 ('😍', 26333),
 ('', 8164),
 ('', 7615),
 ('', 5251),
 ('', 4921),
 ('', 4855),
 ('😘', 4852),
 ('', 4461)]
```

```
Counter(all_emojis_list).most_common(10)[0]
```

```
('😂', 33283)
```

```
Counter(all_emojis_list).most_common(10)[0][0]
```

```
{"type":"string"}
```

```
Counter(all_emojis_list).most_common(10)[0][1]
```

```
33283
```

```
Counter(all_emojis_list).most_common(10)[1][0]
```

```
{"type":"string"}
```

```
Counter(all_emojis_list).most_common(10)[2][0]
```

```
{"type":"string"}
```

```
Counter(all_emojis_list).most_common(10)[0][1]
```

```
33283
```

```
Counter(all_emojis_list).most_common(10)[1][1]
```

```
27897
```

```
Counter(all_emojis_list).most_common(10)[2][1]
```

```
26333
```

```
freqs = [Counter(all_emojis_list).most_common(10)[i][1] for i in
range(10)]
freqs
```

```
[33283, 27897, 26333, 8164, 7615, 5251, 4921, 4855, 4852, 4461]
```

```
emojis = [Counter(all_emojis_list).most_common(10)[i][0] for i in
range(10)]
emojis
```

```
['😂', '❤', '😍', '😭 '😘', '□', '□', '□', '□ ', '□']
```

```
#pip install plotly
!pip install plotly==5.24.1
```

```
Collecting plotly==5.24.1
  Downloading plotly-5.24.1-py3-none-any.whl.metadata (7.3 kB)
Requirement already satisfied: tenacity>=6.2.0 in
/usr/local/lib/python3.10/dist-packages (from plotly==5.24.1) (9.0.0)
Requirement already satisfied: packaging in
/usr/local/lib/python3.10/dist-packages (from plotly==5.24.1) (24.2)
Downloading plotly-5.24.1-py3-none-any.whl (19.1 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 19.1/19.1 MB 83.7 MB/s eta
0:00:00
```

```
pting uninstall: plotly
    Found existing installation: plotly 5.14.1
    Uninstalling plotly-5.14.1:
      Successfully uninstalled plotly-5.14.1
Successfully installed plotly-5.24.1
```


```
{"id":"bf1040b8d46f46efb17db819d9b16271","pip_warning":{"packages":
["_plotly_utils","plotly"]}}
```


```
import plotly.io as pio
pio.renderers.default = 'iframe_connected'
```

use this if your chart is not displaying Plotly is configured to display plots correctly.

```
from plotly.offline import init_notebook_mode, iplot
import plotly.graph_objs as go

# Initialize notebook mode for offline plotting
init_notebook_mode(connected=True)

# Example data
emojis = ['😀', '😂', '😍', '', '😇']
freqs = [10, 20, 15, 25, 30]

# Create bar chart
trace = go.Bar(x=emojis, y=freqs)
iplot([trace])
```


```
{"config":{"linkText":"Export to
plot.ly","plotlyServerURL":"https://plot.ly","showLink":false},"data":
[{"type":"bar","x":["😀","😂","😍","","😇"],"y":
[10,20,15,25,30]}],"layout":{"template":{"data":{"bar":[{"error_x":
{"color":"#2a3f5f"},"error_y":{"color":"#2a3f5f"},"marker":{"line":
{"color":"#E5ECF6","width":0.5},"pattern":
{"fillmode":"overlay","size":10,"solidity":0.2}},"type":"bar"}],"barpo
lar":[{"marker":{"line":{"color":"#E5ECF6","width":0.5},"pattern":
{"fillmode":"overlay","size":10,"solidity":0.2}},"type":"barpolar"}],"
carpet":[{"aaxis":
{"endlinecolor":"#2a3f5f","gridcolor":"white","linecolor":"white","min
orgridcolor":"white","startlinecolor":"#2a3f5f"},"baxis":
{"endlinecolor":"#2a3f5f","gridcolor":"white","linecolor":"white","min
orgridcolor":"white","startlinecolor":"#2a3f5f"},"type":"carpet"}],"ch
oropleth":[{"colorbar":
{"outlinewidth":0,"ticks":""},"type":"choropleth"}],"contour":
[{"colorbar":{"outlinewidth":0,"ticks":""},"colorscale":
[[0,"#0d0887"],[0.1111111111111111,"#46039f"],
[0.2222222222222222,"#7201a8"],[0.3333333333333333,"#9c179e"],
[0.4444444444444444,"#bd3786"],[0.5555555555555556,"#d8576b"],
[0.6666666666666666,"#ed7953"],[0.7777777777777778,"#fb9f3a"],
[0.8888888888888888,"#fdca26"],
```

```
[1,"#f0f921"]],"type":"contour"}],"contourcarpet":[{"colorbar":
{"outlinewidth":0,"ticks":""},"type":"contourcarpet"}],"heatmap":
[{"colorbar":{"outlinewidth":0,"ticks":""},"colorscale":
[[0,"#0d0887"],[0.1111111111111111,"#46039f"],
[0.2222222222222222,"#7201a8"],[0.3333333333333333,"#9c179e"],
[0.4444444444444444,"#bd3786"],[0.5555555555555556,"#d8576b"],
[0.6666666666666666,"#ed7953"],[0.7777777777777778,"#fb9f3a"],
[0.8888888888888888,"#fdca26"],
[1,"#f0f921"]],"type":"heatmap"}],"heatmapgl":[{"colorbar":
{"outlinewidth":0,"ticks":""},"colorscale":[[0,"#0d0887"],
[0.1111111111111111,"#46039f"],[0.2222222222222222,"#7201a8"],
[0.3333333333333333,"#9c179e"],[0.4444444444444444,"#bd3786"],
[0.5555555555555556,"#d8576b"],[0.6666666666666666,"#ed7953"],
[0.7777777777777778,"#fb9f3a"],[0.8888888888888888,"#fdca26"],
[1,"#f0f921"]],"type":"heatmapgl"}],"histogram":[{"marker":{"pattern":
{"fillmode":"overlay","size":10,"solidity":0.2}},"type":"histogram"}],
"histogram2d":[{"colorbar":{"outlinewidth":0,"ticks":""},"colorscale":
[[0,"#0d0887"],[0.1111111111111111,"#46039f"],
[0.2222222222222222,"#7201a8"],[0.3333333333333333,"#9c179e"],
[0.4444444444444444,"#bd3786"],[0.5555555555555556,"#d8576b"],
[0.6666666666666666,"#ed7953"],[0.7777777777777778,"#fb9f3a"],
[0.8888888888888888,"#fdca26"],
[1,"#f0f921"]],"type":"histogram2d"}],"histogram2dcontour":
[{"colorbar":{"outlinewidth":0,"ticks":""},"colorscale":
[[0,"#0d0887"],[0.1111111111111111,"#46039f"],
[0.2222222222222222,"#7201a8"],[0.3333333333333333,"#9c179e"],
[0.4444444444444444,"#bd3786"],[0.5555555555555556,"#d8576b"],
[0.6666666666666666,"#ed7953"],[0.7777777777777778,"#fb9f3a"],
[0.8888888888888888,"#fdca26"],
[1,"#f0f921"]],"type":"histogram2dcontour"}],"mesh3d":[{"colorbar":
{"outlinewidth":0,"ticks":""},"type":"mesh3d"}],"parcoords":[{"line":
{"colorbar":{"outlinewidth":0,"ticks":""}},"type":"parcoords"}],"pie":
[{"automargin":true,"type":"pie"}],"scatter":[{"fillpattern":
{"fillmode":"overlay","size":10,"solidity":0.2},"type":"scatter"}],"sc
atter3d":[{"line":{"colorbar":{"outlinewidth":0,"ticks":""}},"marker":
{"colorbar":
{"outlinewidth":0,"ticks":""}},"type":"scatter3d"}],"scattercarpet":
[{"marker":{"colorbar":
{"outlinewidth":0,"ticks":""}},"type":"scattercarpet"}],"scattergeo":
[{"marker":{"colorbar":
{"outlinewidth":0,"ticks":""}},"type":"scattergeo"}],"scattergl":
[{"marker":{"colorbar":
{"outlinewidth":0,"ticks":""}},"type":"scattergl"}],"scattermapbox":
[{"marker":{"colorbar":
{"outlinewidth":0,"ticks":""}},"type":"scattermapbox"}],"scatterpolar"
:[{"marker":{"colorbar":
{"outlinewidth":0,"ticks":""}},"type":"scatterpolar"}],"scatterpolargl
":[{"marker":{"colorbar":
{"outlinewidth":0,"ticks":""}},"type":"scatterpolargl"}],"scatterterna
```

```
ry":[{"marker":{"colorbar":
{"outlinewidth":0,"ticks":""}},"type":"scatterternary"}],"surface":
[{"colorbar":{"outlinewidth":0,"ticks":""},"colorscale":
[[0,"#0d0887"],[0.1111111111111111,"#46039f"],
[0.2222222222222222,"#7201a8"],[0.3333333333333333,"#9c179e"],
[0.4444444444444444,"#bd3786"],[0.5555555555555556,"#d8576b"],
[0.6666666666666666,"#ed7953"],[0.7777777777777778,"#fb9f3a"],
[0.8888888888888888,"#fdca26"],
[1,"#f0f921"]],"type":"surface"}],"table":[{"cells":{"fill":
{"color":"#EBF0F8"},"line":{"color":"white"}},"header":{"fill":
{"color":"#C8D4E3"},"line":
{"color":"white"}},"type":"table"}]},"layout":{"annotationdefaults":
{"arrowcolor":"#2a3f5f","arrowhead":0,"arrowwidth":1},"autotypenumbers
":"strict","coloraxis":{"colorbar":
{"outlinewidth":0,"ticks":""}},"colorscale":{"diverging":
[[0,"#8e0152"],[0.1,"#c51b7d"],[0.2,"#de77ae"],[0.3,"#f1b6da"],
[0.4,"#fde0ef"],[0.5,"#f7f7f7"],[0.6,"#e6f5d0"],[0.7,"#b8e186"],
[0.8,"#7fbc41"],[0.9,"#4d9221"],[1,"#276419"]],"sequential":
[[0,"#0d0887"],[0.1111111111111111,"#46039f"],
[0.2222222222222222,"#7201a8"],[0.3333333333333333,"#9c179e"],
[0.4444444444444444,"#bd3786"],[0.5555555555555556,"#d8576b"],
[0.6666666666666666,"#ed7953"],[0.7777777777777778,"#fb9f3a"],
[0.8888888888888888,"#fdca26"],[1,"#f0f921"]],"sequentialminus":
[[0,"#0d0887"],[0.1111111111111111,"#46039f"],
[0.2222222222222222,"#7201a8"],[0.3333333333333333,"#9c179e"],
[0.4444444444444444,"#bd3786"],[0.5555555555555556,"#d8576b"],
[0.6666666666666666,"#ed7953"],[0.7777777777777778,"#fb9f3a"],
[0.8888888888888888,"#fdca26"],[1,"#f0f921"]]},"colorway":
["#636efa","#EF553B","#00cc96","#ab63fa","#FFA15A","#19d3f3","#FF6692"
,"#B6E880","#FF97FF","#FECB52"],"font":{"color":"#2a3f5f"},"geo":
{"bgcolor":"white","lakecolor":"white","landcolor":"#E5ECF6","showlake
s":true,"showland":true,"subunitcolor":"white"},"hoverlabel":
{"align":"left"},"hovermode":"closest","mapbox":
{"style":"light"},"paper_bgcolor":"white","plot_bgcolor":"#E5ECF6","po
lar":{"angularaxis":
{"gridcolor":"white","linecolor":"white","ticks":""},"bgcolor":"#E5ECF
6","radialaxis":
{"gridcolor":"white","linecolor":"white","ticks":""}},"scene":
{"xaxis":
{"backgroundcolor":"#E5ECF6","gridcolor":"white","gridwidth":2,"lineco
lor":"white","showbackground":true,"ticks":"","zerolinecolor":"white"}
,"yaxis":
{"backgroundcolor":"#E5ECF6","gridcolor":"white","gridwidth":2,"lineco
lor":"white","showbackground":true,"ticks":"","zerolinecolor":"white"}
,"zaxis":
{"backgroundcolor":"#E5ECF6","gridcolor":"white","gridwidth":2,"lineco
lor":"white","showbackground":true,"ticks":"","zerolinecolor":"white"}
},"shapedefaults":{"line":{"color":"#2a3f5f"}},"ternary":{"aaxis":
{"gridcolor":"white","linecolor":"white","ticks":""},"baxis":
```

{"gridcolor":"white","linecolor":"white","ticks":""},"bgcolor":"#E5ECF6","caxis":
{"gridcolor":"white","linecolor":"white","ticks":""}},"title":
{"x":5.0e-2},"xaxis":
{"automargin":true,"gridcolor":"white","linecolor":"white","ticks":"",
"title":
{"standoff":15},"zerolinecolor":"white","zerolinewidth":2},"yaxis":
{"automargin":true,"gridcolor":"white","linecolor":"white","ticks":"",
"title":{"standoff":15},"zerolinecolor":"white","zerolinewidth":2}}}}}

Conclusions : Majority of the customers are happy as most of them are using emojis like: funny , love , heart , outstanding..

Collect Entire data of Youtube !

```
import os

files= os.listdir(r'C:\Users\Desktop\youtube_Project\
YT_additional_data')

files
```

```
['.git',
 'CAvideos.csv',
 'CA_category_id.json',
 'DEvideos.csv',
 'DE_category_id.json',
 'FRvideos.csv',
 'FR_category_id.json',
 'GBvideos.csv',
 'GB_category_id.json',
 'INvideos.csv',
 'IN_category_id.json',
 'JPvideos.csv',
 'JP_category_id.json',
 'KRvideos.csv',
 'KR_category_id.json',
 'MXvideos.csv',
 'MX_category_id.json',
 'README.md',
 'RUvideos.csv',
 'RU_category_id.json',
 'USvideos.csv',
 'US_category_id.json']
```

```
## extracting csv files only from above list ..

files_csv = [file for file in files if '.csv' in file]
files_csv
```

```
['CAvideos.csv',
 'DEvideos.csv',
 'FRvideos.csv',
 'GBvideos.csv',
 'INvideos.csv',
 'JPvideos.csv',
 'KRvideos.csv',
 'MXvideos.csv',
 'RUvideos.csv',
 'USvideos.csv']
```

```
#while colllecting the data if you encounter any kind of warning its always good to consider a warning modules.
import warnings
from warnings import filterwarnings
filterwarnings('ignore')
```

different types of encoding-->>

```
Note : encoding may change depending upon data  , country data , sometimes regional data as well.
Fore more inforation on Encoding -- Follow below
```

https://docs.python.org/3/library/codecs.html#standard-encodings¶

```
#all the csv file i have to store in big data frame

full_df = pd.DataFrame()
path = r'C:\Users\Desktop\youtube_Project\YT_additional_data'
for file in files_csv:
    current_df = pd.read_csv(path+'/'+file, encoding='iso-8859-1')
    full_df = pd.concat([full_df, current_df], ignore_index=True)

full_df.shape
```

```
(375942, 16)
```

```
full_df.duplicated() #True will represent the duplicate rows and False represent the uniques rows.
```

```
0         False
1         False
2         False
3         False
4         False
          ...
375937     True
375938    False
375939    False
375940    False
```

```
375941    False
Length: 375942, dtype: bool
```

```
full_df[full_df.duplicated()].shape
```

```
(36417, 16)
```

```
full_df = full_df.drop_duplicates() ## lets drop duplicate rows ..
```

```
full_df.shape
```

```
(339525, 16)
```

```
#### a... Storing data into csv ..
full_df[0:1000].to_csv(r'C:\Users\Desktop\youtube_Project\
export_data/youtube_sample.csv' , index=False)

#### b... Storing data into json
full_df[0:1000].to_json(r'C:\Users\Desktop\youtube_Project\
youtube_sample.json')
```

Q. Which Category has the maximum likes ?

```
full_df.head(5)
```

```
      video_id trending_date  \
0  n1WpP7iowLc      17.14.11
1  0dBIkQ4Mz1M      17.14.11
2  5qpjK5DgCt4      17.14.11
3  d380meD0W0M      17.14.11
4  2Vv-BfVoq4g      17.14.11

                                                title channel_title  \
0        Eminem - Walk On Water (Audio) ft. BeyoncÃ©    EminemVEVO
1                         PLUSH - Bad Unboxing Fan Mail     iDubbbzTV
2  Racist Superman | Rudy Mancuso, King Bach & Le...  Rudy Mancuso
3                            I Dare You: GOING BALD!?      nigahiga
4        Ed Sheeran - Perfect (Official Music Video)    Ed Sheeran

   category_id              publish_time  \
0           10  2017-11-10T17:00:03.000Z
1           23  2017-11-13T17:00:00.000Z
2           23  2017-11-12T19:05:24.000Z
3           24  2017-11-12T18:01:41.000Z
4           10  2017-11-09T11:04:14.000Z

                                                tags     views
likes  \
0  Eminem|"Walk"|"On"|"Water"|"Aftermath/Shady/In...  17158579
787425
1  plush|"bad unboxing"|"unboxing"|"fan mail"|"id...   1014651
```

```
127794
2  racist superman|"rudy"|"mancuso"|"king"|"bach"...   3191434
146035
3  ryan|"higa"|"higatv"|"nigahiga"|"i dare you"|"...   2095828
132239
4  edsheeran|"ed sheeran"|"acoustic"|"live"|"cove...  33523622
1634130

   dislikes  comment_count
thumbnail_link  \
0     43420         125882
https://i.ytimg.com/vi/n1WpP7iowLc/default.jpg
1      1688          13030
https://i.ytimg.com/vi/0dBIkQ4Mz1M/default.jpg
2      5339           8181
https://i.ytimg.com/vi/5qpjK5DgCt4/default.jpg
3      1989          17518
https://i.ytimg.com/vi/d380meD0W0M/default.jpg
4     21082          85067
https://i.ytimg.com/vi/2Vv-BfVoq4g/default.jpg

   comments_disabled  ratings_disabled  video_error_or_removed  \
0              False             False                   False
1              False             False                   False
2              False             False                   False
3              False             False                   False
4              False             False                   False

                                          description
0  Eminem's new track Walk on Water ft. BeyoncÃ© ...
1  STill got a lot of packages. Probably will las...
2  WATCH MY PREVIOUS VIDEO â□¶ \n\nSUBSCRIBE â□º ...
3  I know it's been a while since we did this sho...
4  ð□□§: https://ad.gt/yt-perfect\nð□□°: https://...
```

```
full_df['category_id'].unique() #returns an array containing the
unique values of the category_id
```

```
array([10, 23, 24, 25, 22, 26,  1, 28, 20, 17, 29, 15, 19,  2, 27, 43,
30,
       44], dtype=int64)
```

```
## lets read json file ..
json_df = pd.read_json(r'C:\Users\Desktop\youtube_Project\
YT_additional_data/US_category_id.json')
```

```
json_df
```

```
                                  kind  \
0   youtube#videoCategoryListResponse
1   youtube#videoCategoryListResponse
```

```
2   youtube#videoCategoryListResponse
3   youtube#videoCategoryListResponse
4   youtube#videoCategoryListResponse
5   youtube#videoCategoryListResponse
6   youtube#videoCategoryListResponse
7   youtube#videoCategoryListResponse
8   youtube#videoCategoryListResponse
9   youtube#videoCategoryListResponse
10  youtube#videoCategoryListResponse
11  youtube#videoCategoryListResponse
12  youtube#videoCategoryListResponse
13  youtube#videoCategoryListResponse
14  youtube#videoCategoryListResponse
15  youtube#videoCategoryListResponse
16  youtube#videoCategoryListResponse
17  youtube#videoCategoryListResponse
18  youtube#videoCategoryListResponse
19  youtube#videoCategoryListResponse
20  youtube#videoCategoryListResponse
21  youtube#videoCategoryListResponse
22  youtube#videoCategoryListResponse
23  youtube#videoCategoryListResponse
24  youtube#videoCategoryListResponse
25  youtube#videoCategoryListResponse
26  youtube#videoCategoryListResponse
27  youtube#videoCategoryListResponse
28  youtube#videoCategoryListResponse
29  youtube#videoCategoryListResponse
30  youtube#videoCategoryListResponse
31  youtube#videoCategoryListResponse

                                                  etag  \
0   "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
1   "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
2   "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
3   "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
4   "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
5   "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
6   "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
7   "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
8   "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
9   "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
10  "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
11  "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
12  "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
13  "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
14  "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
15  "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
16  "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
```

```
17  "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
18  "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
19  "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
20  "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
21  "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
22  "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
23  "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
24  "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
25  "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
26  "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
27  "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
28  "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
29  "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
30  "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
31  "m2yskBQFythfE4irbTIeOgYYfBU/S730Ilt-Fi-emsQJv...
```

```
                                                items
0   {'kind': 'youtube#videoCategory', 'etag': '"m2...
1   {'kind': 'youtube#videoCategory', 'etag': '"m2...
2   {'kind': 'youtube#videoCategory', 'etag': '"m2...
3   {'kind': 'youtube#videoCategory', 'etag': '"m2...
4   {'kind': 'youtube#videoCategory', 'etag': '"m2...
5   {'kind': 'youtube#videoCategory', 'etag': '"m2...
6   {'kind': 'youtube#videoCategory', 'etag': '"m2...
7   {'kind': 'youtube#videoCategory', 'etag': '"m2...
8   {'kind': 'youtube#videoCategory', 'etag': '"m2...
9   {'kind': 'youtube#videoCategory', 'etag': '"m2...
10  {'kind': 'youtube#videoCategory', 'etag': '"m2...
11  {'kind': 'youtube#videoCategory', 'etag': '"m2...
12  {'kind': 'youtube#videoCategory', 'etag': '"m2...
13  {'kind': 'youtube#videoCategory', 'etag': '"m2...
14  {'kind': 'youtube#videoCategory', 'etag': '"m2...
15  {'kind': 'youtube#videoCategory', 'etag': '"m2...
16  {'kind': 'youtube#videoCategory', 'etag': '"m2...
17  {'kind': 'youtube#videoCategory', 'etag': '"m2...
18  {'kind': 'youtube#videoCategory', 'etag': '"m2...
19  {'kind': 'youtube#videoCategory', 'etag': '"m2...
20  {'kind': 'youtube#videoCategory', 'etag': '"m2...
21  {'kind': 'youtube#videoCategory', 'etag': '"m2...
22  {'kind': 'youtube#videoCategory', 'etag': '"m2...
23  {'kind': 'youtube#videoCategory', 'etag': '"m2...
24  {'kind': 'youtube#videoCategory', 'etag': '"m2...
25  {'kind': 'youtube#videoCategory', 'etag': '"m2...
26  {'kind': 'youtube#videoCategory', 'etag': '"m2...
27  {'kind': 'youtube#videoCategory', 'etag': '"m2...
28  {'kind': 'youtube#videoCategory', 'etag': '"m2...
29  {'kind': 'youtube#videoCategory', 'etag': '"m2...
30  {'kind': 'youtube#videoCategory', 'etag': '"m2...
31  {'kind': 'youtube#videoCategory', 'etag': '"m2...
```

Retrieves the first item (index 0) from the 'items' column of the DataFrame

```
json_df['items'][0]
```

```
{'kind': 'youtube#videoCategory',
 'etag': '"m2yskBQFythfE4irbTIeOgYYfBU/Xy1mB4_yLrHy_BmKmPBggty2mZQ"',
 'id': '1',
 'snippet': {'channelId': 'UCBR8-60-B28hp2BmDPdntcQ',
  'title': 'Film & Animation',
  'assignable': True}}
```

```
#now i want id and title in a dictionary
cat_dict = {} #empty dict


for item in json_df['items'].values:     #values here return the array representation
    ## cat_dict[key] = value (Syntax to insert key:value in dictionary)
    cat_dict[int(item['id'])] = item['snippet']['title'] # snippet here is the sub dict so we have to write this way  ['snippet']['title']
```

```
cat_dict
```

```
{1: 'Film & Animation',
 2: 'Autos & Vehicles',
 10: 'Music',
 15: 'Pets & Animals',
 17: 'Sports',
 18: 'Short Movies',
 19: 'Travel & Events',
 20: 'Gaming',
 21: 'Videoblogging',
 22: 'People & Blogs',
 23: 'Comedy',
 24: 'Entertainment',
 25: 'News & Politics',
 26: 'Howto & Style',
 27: 'Education',
 28: 'Science & Technology',
 29: 'Nonprofits & Activism',
 30: 'Movies',
 31: 'Anime/Animation',
 32: 'Action/Adventure',
 33: 'Classics',
 34: 'Comedy',
 35: 'Documentary',
 36: 'Drama',
 37: 'Family',
 38: 'Foreign',
```

```
 39: 'Horror',
 40: 'Sci-Fi/Fantasy',
 41: 'Thriller',
 42: 'Shorts',
 43: 'Shows',
 44: 'Trailers'}
```

Maps category IDs in the 'category_id' column of full_df DataFrame to their corresponding category titles using the cat_dict dictionary.

```
full_df['category_name'] = full_df['category_id'].map(cat_dict)
full_df['category_name']
```

```
0                    Music
1                   Comedy
2                   Comedy
3            Entertainment
4                    Music
                ...
375936      People & Blogs
375938      People & Blogs
375939       Entertainment
375940    Film & Animation
375941              Gaming
Name: category_name, Length: 339525, dtype: object
```

Now you can notice that you have a new feature which is a category name

```
full_df.head(4)
```

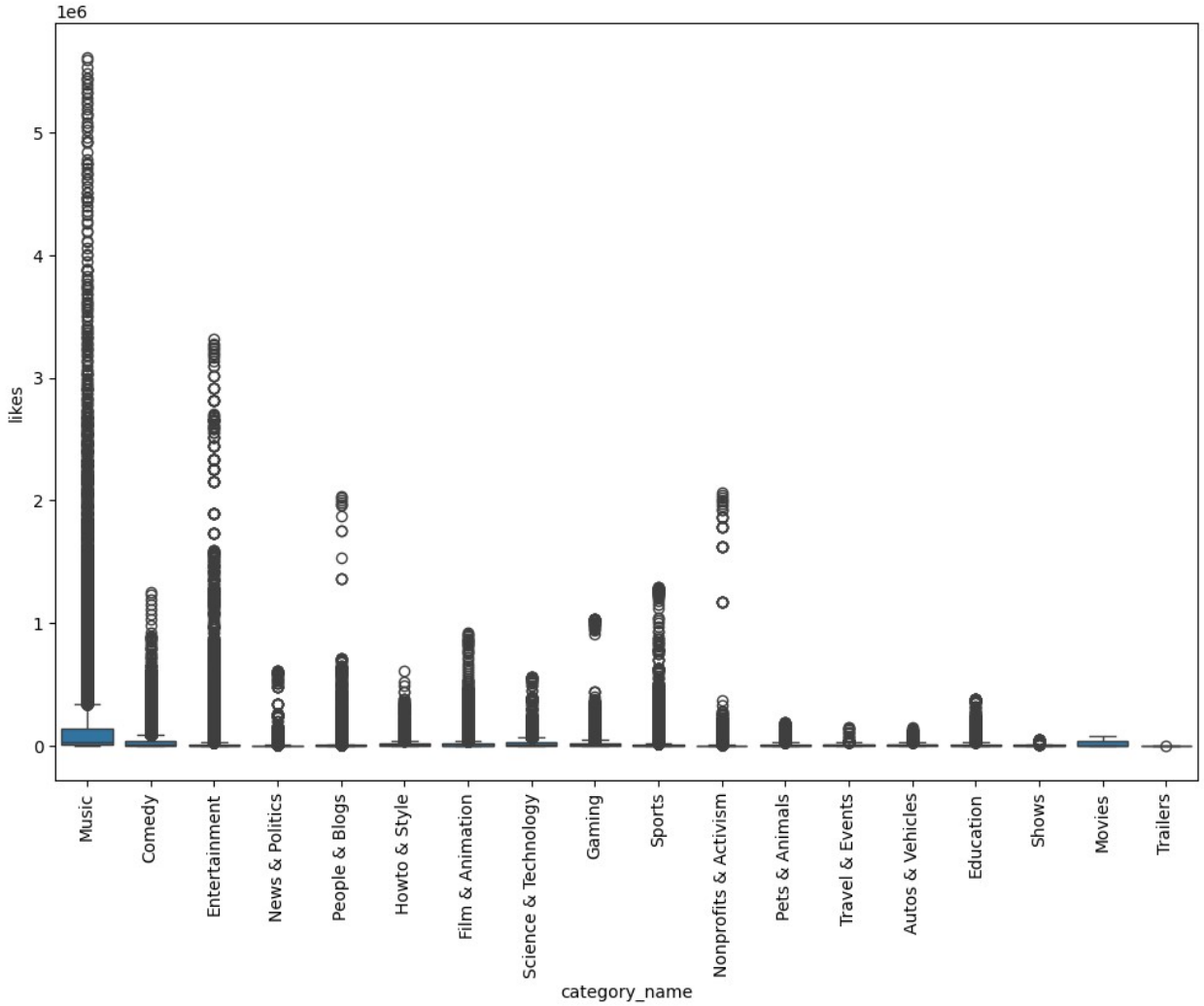
```
      video_id trending_date  \
0  n1WpP7iowLc      17.14.11
1  0dBIkQ4Mz1M      17.14.11
2  5qpjK5DgCt4      17.14.11
3  d380meD0W0M      17.14.11

                                                title channel_title  \
0        Eminem - Walk On Water (Audio) ft. BeyoncÃ©    EminemVEVO
1                        PLUSH - Bad Unboxing Fan Mail     iDubbbzTV
2  Racist Superman | Rudy Mancuso, King Bach & Le...  Rudy Mancuso
3                           I Dare You: GOING BALD!?      nigahiga

   category_id              publish_time  \
0           10  2017-11-10T17:00:03.000Z
1           23  2017-11-13T17:00:00.000Z
2           23  2017-11-12T19:05:24.000Z
3           24  2017-11-12T18:01:41.000Z

                                                tags     views   likes
```

```
\
0  Eminem|"Walk"|"On"|"Water"|"Aftermath/Shady/In...  17158579  787425

1  plush|"bad unboxing"|"unboxing"|"fan mail"|"id...   1014651  127794

2  racist superman|"rudy"|"mancuso"|"king"|"bach"...   3191434  146035

3  ryan|"higa"|"higatv"|"nigahiga"|"i dare you"|"...   2095828  132239


   dislikes  comment_count
thumbnail_link  \
0     43420         125882
https://i.ytimg.com/vi/n1WpP7iowLc/default.jpg
1      1688          13030
https://i.ytimg.com/vi/0dBIkQ4Mz1M/default.jpg
2      5339           8181
https://i.ytimg.com/vi/5qpjK5DgCt4/default.jpg
3      1989          17518
https://i.ytimg.com/vi/d380meD0W0M/default.jpg

   comments_disabled  ratings_disabled  video_error_or_removed  \
0              False             False                   False
1              False             False                   False
2              False             False                   False
3              False             False                   False

                                          description  category_name
0  Eminem's new track Walk on Water ft. BeyoncÃ© ...          Music
1  STill got a lot of packages. Probably will las...         Comedy
2  WATCH MY PREVIOUS VIDEO â□¶ \n\nSUBSCRIBE â□º ...         Comedy
3  I know it's been a while since we did this sho...  Entertainment
```

Q. which category has the maximum likes ?

```
plt.figure(figsize=(12,8))#Creates a new figure with a specified size
of 12 inches by 8 inches for better visualization.
sns.boxplot(x='category_name' , y='likes' , data=full_df)
plt.xticks(rotation='vertical')#Rotates the x-axis labels vertically
for better readability.

([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17],
 [Text(0, 0, 'Music'),
  Text(1, 0, 'Comedy'),
  Text(2, 0, 'Entertainment'),
  Text(3, 0, 'News & Politics'),
  Text(4, 0, 'People & Blogs'),
  Text(5, 0, 'Howto & Style'),
  Text(6, 0, 'Film & Animation'),
  Text(7, 0, 'Science & Technology'),
```

```
  Text(8, 0, 'Gaming'),
  Text(9, 0, 'Sports'),
  Text(10, 0, 'Nonprofits & Activism'),
  Text(11, 0, 'Pets & Animals'),
  Text(12, 0, 'Travel & Events'),
  Text(13, 0, 'Autos & Vehicles'),
  Text(14, 0, 'Education'),
  Text(15, 0, 'Shows'),
  Text(16, 0, 'Movies'),
  Text(17, 0, 'Trailers')])
```



Find out whether audience is engaged or not like rate ,dislike , comment_count_rate

```
(full_df['likes']/full_df['views'])*100
```

```
0          4.589104
1         12.594873
2          4.575843
```

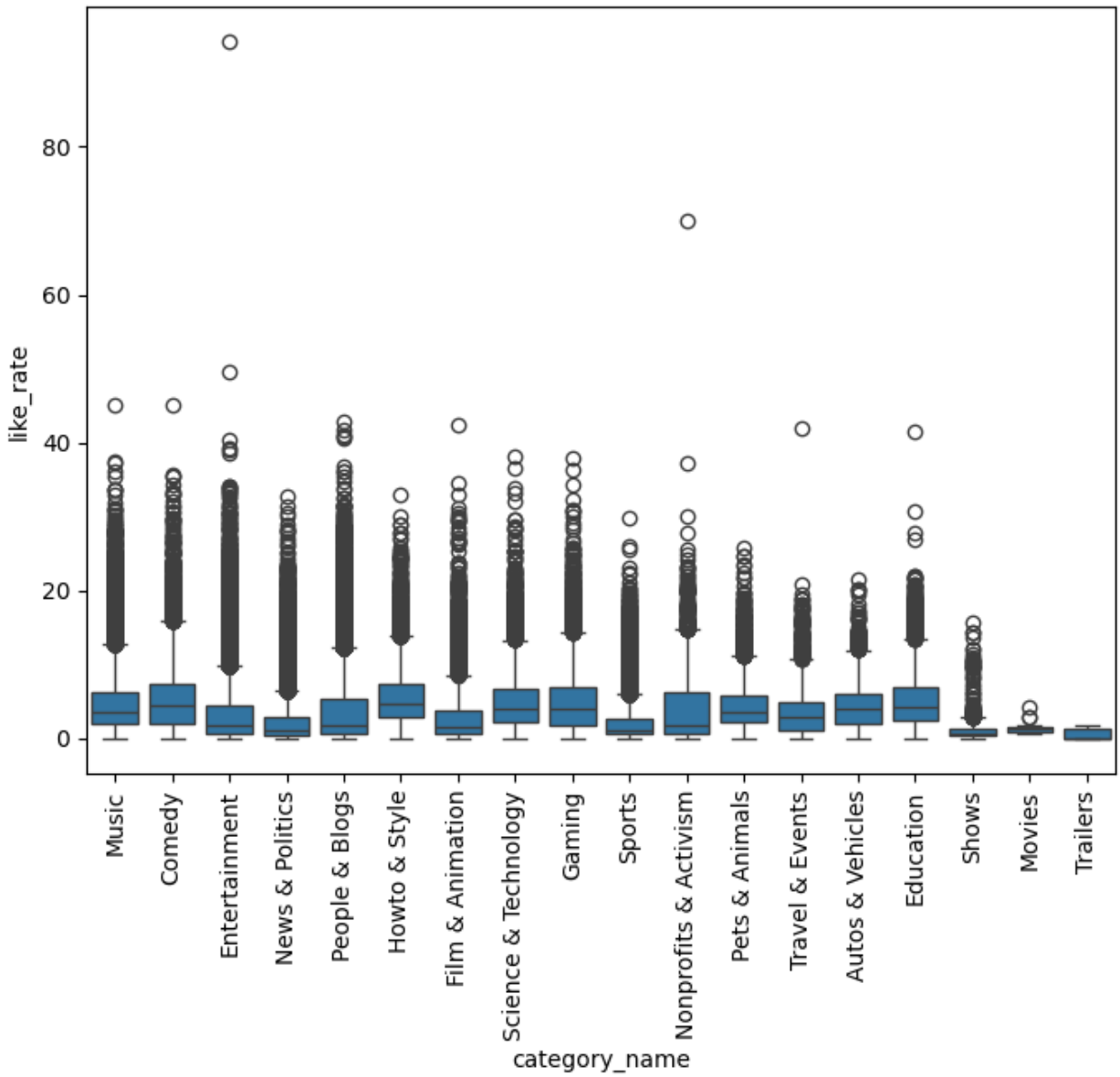
```
3          6.309630
4          4.874563
            ...
375936     7.820293
375938     5.635623
375939     4.507286
375940     3.408645
375941     3.464728
Length: 339525, dtype: float64
```

```
full_df['like_rate'] = (full_df['likes']/full_df['views'])*100
full_df['dislike_rate'] = (full_df['dislikes']/full_df['views'])*100
full_df['comment_count_rate'] =
(full_df['comment_count']/full_df['views'])*100
```

```
full_df.columns # three things added 'like_rate','dislike_rate',
'comment_count_rate'
```

```
Index(['video_id', 'trending_date', 'title', 'channel_title',
'category_id',
       'publish_time', 'tags', 'views', 'likes', 'dislikes',
'comment_count',
       'thumbnail_link', 'comments_disabled', 'ratings_disabled',
       'video_error_or_removed', 'description', 'category_name',
'like_rate',
       'dislike_rate', 'comment_count_rate'],
      dtype='object')
```

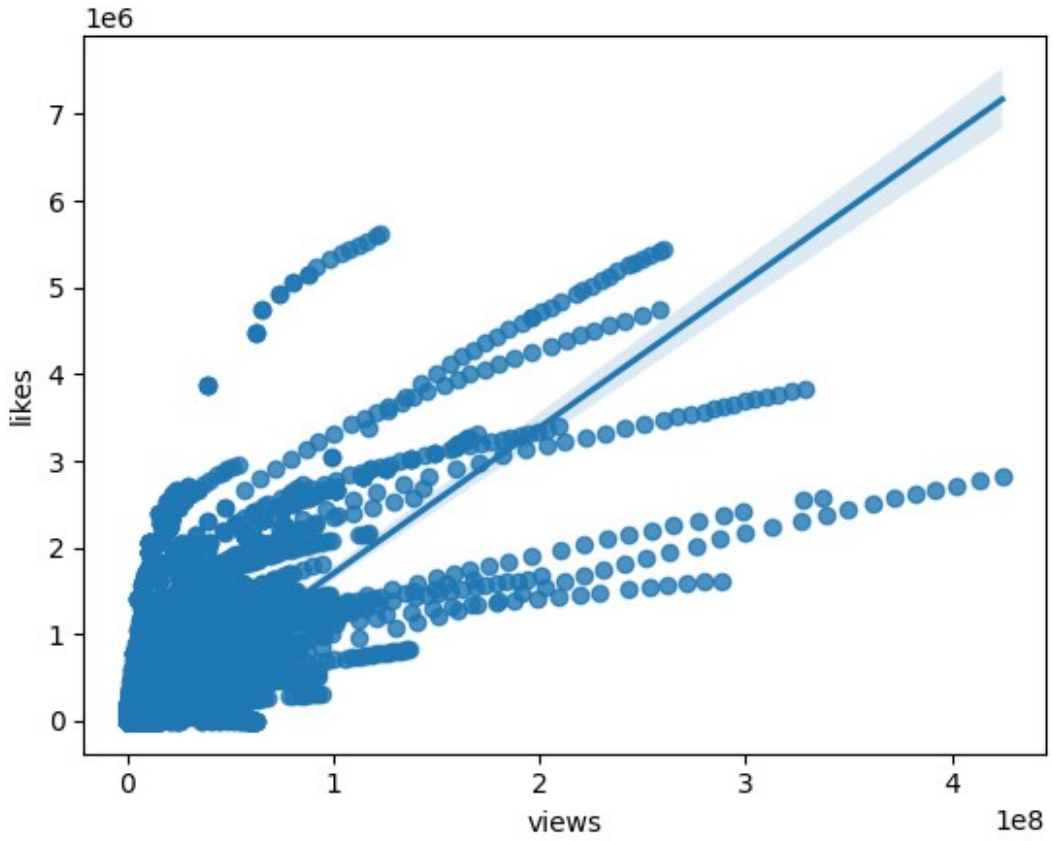
```
#creating box plot for like rate
plt.figure(figsize=(8,6))
sns.boxplot(x='category_name' , y='like_rate' , data=full_df)
plt.xticks(rotation='vertical')
plt.show()
```



Analysing relationship between views & likes

```
#using Regression plot
#regression plot is nothing but it is the combination of a scatter
plot + a regression kine on top of that
sns.regplot(x='views' , y='likes' , data = full_df)
```

```
<Axes: xlabel='views', ylabel='likes'>
```




```
full_df.columns

Index(['video_id', 'trending_date', 'title', 'channel_title',
'category_id',
       'publish_time', 'tags', 'views', 'likes', 'dislikes',
'comment_count',
       'thumbnail_link', 'comments_disabled', 'ratings_disabled',
       'video_error_or_removed', 'description', 'category_name',
'like_rate',
       'dislike_rate', 'comment_count_rate'],
      dtype='object')

full_df[['views', 'likes', 'dislikes']]

           views    likes  dislikes
0       17158579   787425     43420
1        1014651   127794      1688
2        3191434   146035      5339
3        2095828   132239      1989
4       33523622  1634130     21082
...          ...      ...       ...
375936   8259128   645888      4052
375938   1064798    60008       382
375939   1066451    48068      1032
```

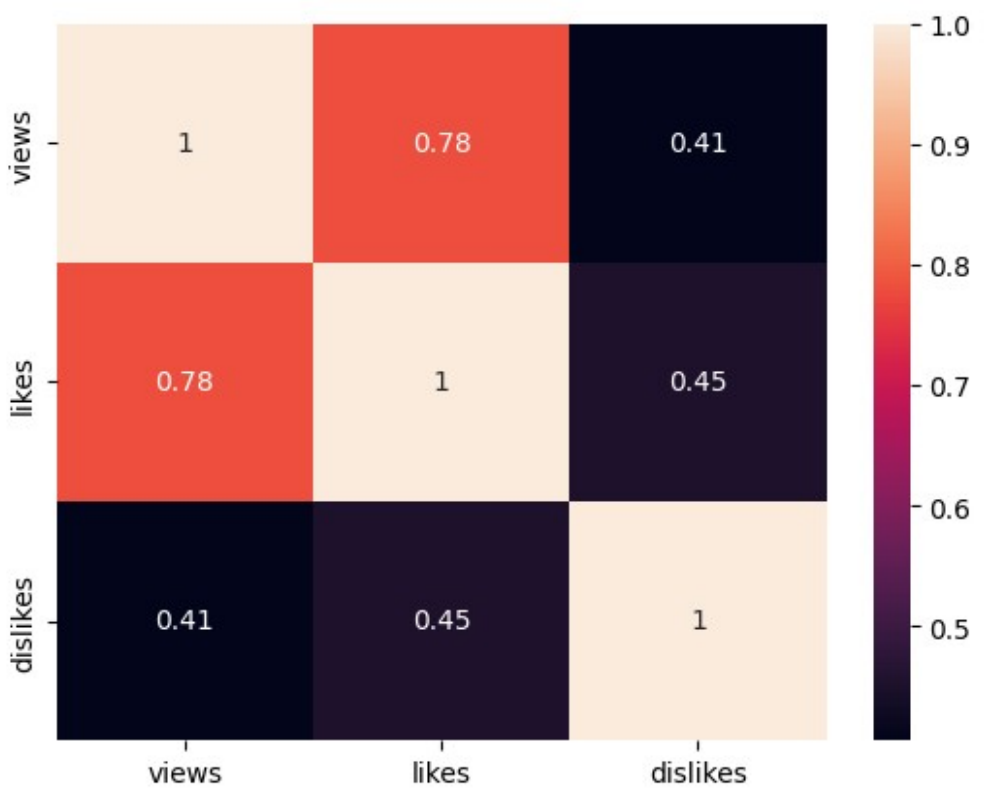
```
375940   5660813   192957      2846
375941  10306119   357079    212976

[339525 rows x 3 columns]
```

```
full_df[['views', 'likes', 'dislikes']].corr() ### finding co-relation values between ['views', 'likes', 'dislikes']
```

```
             views     likes  dislikes
views     1.000000  0.779531  0.405428
likes     0.779531  1.000000  0.451809
dislikes  0.405428  0.451809  1.000000
```

```
#Now if you want to showcase this correlation table in a vsiualized way , you can use the heatmap
sns.heatmap(full_df[['views', 'likes', 'dislikes']].corr() , annot=True)
#When annot=True, numerical values are displayed on the heatmap cells
```

```
<Axes: >
```



Q. Which channels have the largest number of trending videos?

```
full_df.head(6)
```

```
      video_id trending_date  \
0  n1WpP7iowLc      17.14.11
1  0dBIkQ4Mz1M      17.14.11
2  5qpjK5DgCt4      17.14.11
3  d380meD0W0M      17.14.11
4  2Vv-BfVoq4g      17.14.11
5  0yIWz1XEeyc      17.14.11

                                                title channel_title  \
0        Eminem - Walk On Water (Audio) ft. BeyoncÃ©    EminemVEVO
1                         PLUSH - Bad Unboxing Fan Mail     iDubbbzTV
2  Racist Superman | Rudy Mancuso, King Bach & Le...  Rudy Mancuso
3                              I Dare You: GOING BALD!?      nigahiga
4        Ed Sheeran - Perfect (Official Music Video)    Ed Sheeran
5  Jake Paul Says Alissa Violet CHEATED with LOGA...    DramaAlert

   category_id              publish_time  \
0           10  2017-11-10T17:00:03.000Z
1           23  2017-11-13T17:00:00.000Z
2           23  2017-11-12T19:05:24.000Z
3           24  2017-11-12T18:01:41.000Z
4           10  2017-11-09T11:04:14.000Z
5           25  2017-11-13T07:37:51.000Z

                                                tags     views
likes  \
0  Eminem|"Walk"|"On"|"Water"|"Aftermath/Shady/In...  17158579
787425
1  plush|"bad unboxing"|"unboxing"|"fan mail"|"id...   1014651
127794
2  racist superman|"rudy"|"mancuso"|"king"|"bach"...   3191434
146035
3  ryan|"higa"|"higatv"|"nigahiga"|"i dare you"|"...   2095828
132239
4  edsheeran|"ed sheeran"|"acoustic"|"live"|"cove...  33523622
1634130
5  #DramaAlert|"Drama"|"Alert"|"DramaAlert"|"keem...   1309699
103755

   dislikes  comment_count
thumbnail_link  \
0     43420         125882
https://i.ytimg.com/vi/n1WpP7iowLc/default.jpg
1      1688          13030
https://i.ytimg.com/vi/0dBIkQ4Mz1M/default.jpg
2      5339           8181
https://i.ytimg.com/vi/5qpjK5DgCt4/default.jpg
3      1989          17518
https://i.ytimg.com/vi/d380meD0W0M/default.jpg
4     21082          85067
```

```
https://i.ytimg.com/vi/2Vv-BfVoq4g/default.jpg
5      4613          12143
https://i.ytimg.com/vi/0yIWz1XEeyc/default.jpg

   comments_disabled  ratings_disabled  video_error_or_removed  \
0              False             False                   False
1              False             False                   False
2              False             False                   False
3              False             False                   False
4              False             False                   False
5              False             False                   False

                                          description    category_name
\
0  Eminem's new track Walk on Water ft. BeyoncÃ© ...             Music

1  STill got a lot of packages. Probably will las...           Comedy

2  WATCH MY PREVIOUS VIDEO â¶ \n\nSUBSCRIBE â º ...           Comedy

3  I know it's been a while since we did this sho...    Entertainment

4  ð§: https://ad.gt/yt-perfect\nð°: https://...             Music

5  â º Follow for News! - https://twitter.com/KEE...  News & Politics


   like_rate  dislike_rate  comment_count_rate
0   4.589104      0.253051            0.733639
1  12.594873      0.166363            1.284185
2   4.575843      0.167292            0.256342
3   6.309630      0.094903            0.835851
4   4.874563      0.062887            0.253752
5   7.922049      0.352218            0.927160
```

```
full_df['channel_title'].value_counts()
# returns the count of unique values in a Series, providing a
frequency distribution of the values.

channel_title
The Late Show with Stephen Colbert    710
WWE                                   643
Late Night with Seth Meyers           592
TheEllenShow                          555
Jimmy Kimmel Live                     528
                                     ...
Daas                                    1
YT Industries                           1
BTLV Le mÃ©dia complÃ©mentaire           1
Quem Sabia ?                            1
```

```
Jessi Osorno                                        1
Name: count, Length: 37824, dtype: int64
```

```python
### lets obtain above frequency table using groupby approach :
full_df.groupby(['channel_title']).size()
```

```
channel_title
! ì¸ì ì ë¬´ì ì¼ì ´                               7
!!8æ ã é ç½ã ã ¿å¤§é                       1
!BTSã »TWICE ã ¾ã ¨ã                                 1
!Los amorosos ViralesÂ¡                                2
!t Live                                               3
                                                     ..
ï¼ ã ã §ã §ã ã  DIY                                 5
ï¼¢ï¼¡ï¼³ï¼¨ï½ ï½                                          2
ï¼«ã ®ã ã £ã ã ·ã ³ã °ã ¡ã ã ã ã     1
ï¼·ï¼¡ï¼ªï¼¯ï¼©ï¼«ï¼¡                                      2
ð Sandrea                                          2
Length: 37824, dtype: int64
```

Reset_index() is a pandas DataFrame method used to reset the index of a DataFrame. It converts the index labels into a new column and assigns a default numeric index to the DataFrame.

```python
cdf =
full_df.groupby(['channel_title']).size().sort_values(ascending=False)
.reset_index()

cdf
```

```
                            channel_title    0
0      The Late Show with Stephen Colbert  710
1                                     WWE  643
2             Late Night with Seth Meyers  592
3                            TheEllenShow  555
4                       Jimmy Kimmel Live  528
...                                   ...  ...
37819                            Kd Malts    1
37820                            Zedan TV    1
37821          Kc Kelly - Rocketprenuer    1
37822                               Kbaby    1
37823                     Pavel Sidorik TV    1

[37824 rows x 2 columns]
```

```python
cdf = cdf.rename(columns={0:'total_videos'})

cdf
```

```
                            channel_title  total_videos
0      The Late Show with Stephen Colbert           710
```

```
1                                      WWE           643
2              Late Night with Seth Meyers           592
3                              TheEllenShow          555
4                        Jimmy Kimmel Live           528
...                                    ...           ...
37819                            Kd Malts              1
37820                            Zedan TV              1
37821            Kc Kelly - Rocketprenuer              1
37822                                Kbaby             1
37823                     Pavel Sidorik TV             1

[37824 rows x 2 columns]
```

```
import plotly.express as px
```

Q. Which channels have the largest number of trending videos?

```
px.bar(data_frame=cdf[0:20] , x='channel_title' , y='total_videos')
```