MACHINE LEARNING

LAB EXPERIMENT-2

TOPIC CHOSEN: YouTube Trending Videos Analysis

DATASET LINK: https://www.kaggle.com/datasnaek/youtube-new/

DATASET NAME: USvideos.csv

SOURCE: Kaggle

DATASET DETAILS:

The attributes in the dataset are category_id, views, likes, dislikes and comment_count.

IMPORTING LIBRARIES:

```
import pandas as pd
import numpy as np
import matplotlib as mpl
from matplotlib import pyplot as plt
import seaborn as sns
import warnings
from collections import Counter
import datetime
import wordcloud
import json
```

READ THE DATA:

```
df = pd.read_csv("USvideos.csv")
PLOT_COLORS = ["#268bd2", "#0052CC", "#FF5722", "#b58900", "#003f5c"]
pd.options.display.float_format = '{:.2f}'.format
```

```
sns.set(style="ticks")
plt.rc('figure', figsize=(8, 5), dpi=100)
plt.rc('axes', labelpad=20, facecolor="#ffffff", linewidth=0.4, grid=Tr
ue, labelsize=14)
plt.rc('patch', linewidth=0)
plt.rc('xtick.major', width=0.2)
plt.rc('ytick.major', width=0.2)
plt.rc('grid', color='#9E9E9E', linewidth=0.4)
plt.rc('font', family='Arial', weight='400', size=10)
plt.rc('text', color='#282828')
plt.rc('savefig', pad_inches=0.3, dpi=300)
```

DATA EXPLORATION:

```
df.describe()
```

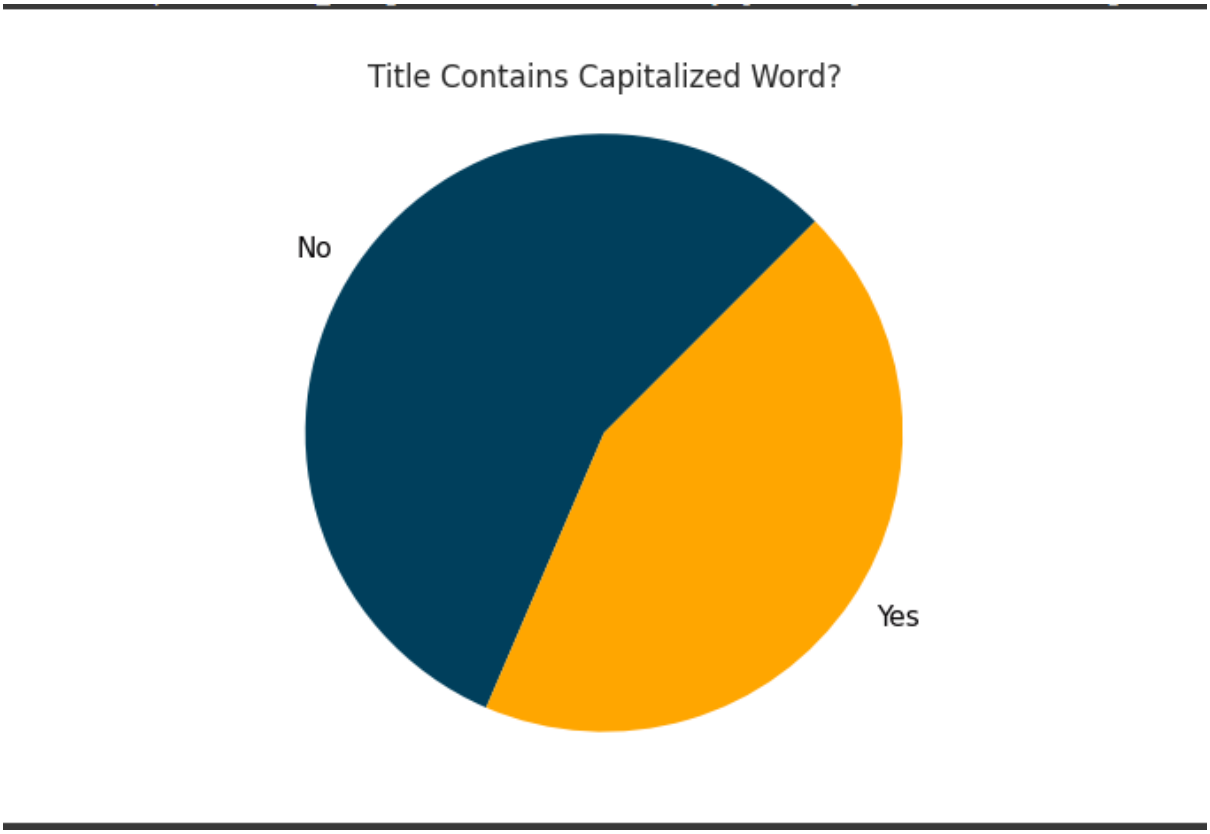
	category_id	views	likes	dislikes	comment_count
count	40949.00	40949.00	40949.00	40949.00	40949.00
mean	19.97	2360784.64	74266.70	3711.40	8446.80
std	7.57	7394113.76	228885.34	29029.71	37430.49
min	1.00	549.00	0.00	0.00	0.00
25%	17.00	242329.00	5424.00	202.00	614.00
50%	24.00	681861.00	18091.00	631.00	1856.00
75%	25.00	1823157.00	55417.00	1938.00	5755.00
max	43.00	225211923.00	5613827.00	1674420.00	1361580.00

VISUALIZING THE DATA:

PIE CHART:

```
def contains_capitalized_word(s):
    for w in s.split():
        if w.isupper():
            return True
    return False
df["contains_capitalized"] = df["title"].apply(contains_capitalized_wor
d)
value_counts = df["contains_capitalized"].value_counts().to_dict()
fig, ax = plt.subplots()
```

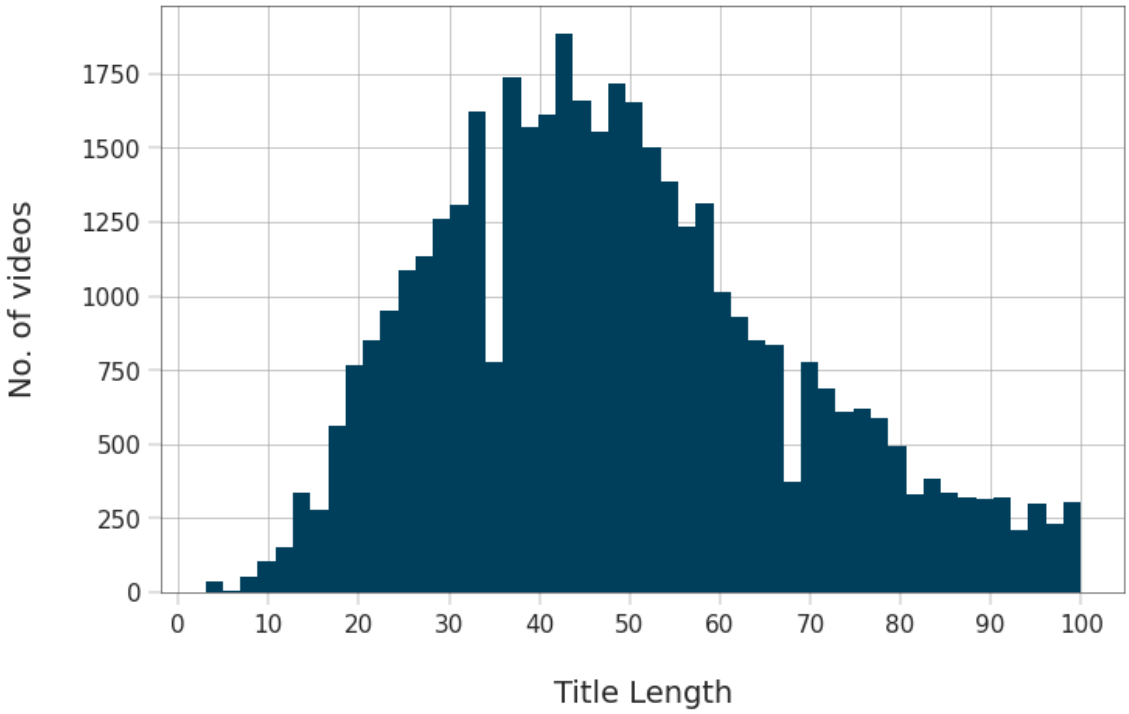
```
_ = ax.pie([value_counts[False], value_counts[True]], labels=['No', 'Ye
s'],
           colors=['#003f5c', '#ffa600'], textprops={'color': '#040204'
}, startangle=45)
_ = ax.axis('equal')
_ = ax.set_title('Title Contains Capitalized Word?')
```



We can see that 44% of trending video titles contain at least one word in all caps. We will use our added variable later to analyze the correlation between the variables.

HISTOGRAM:

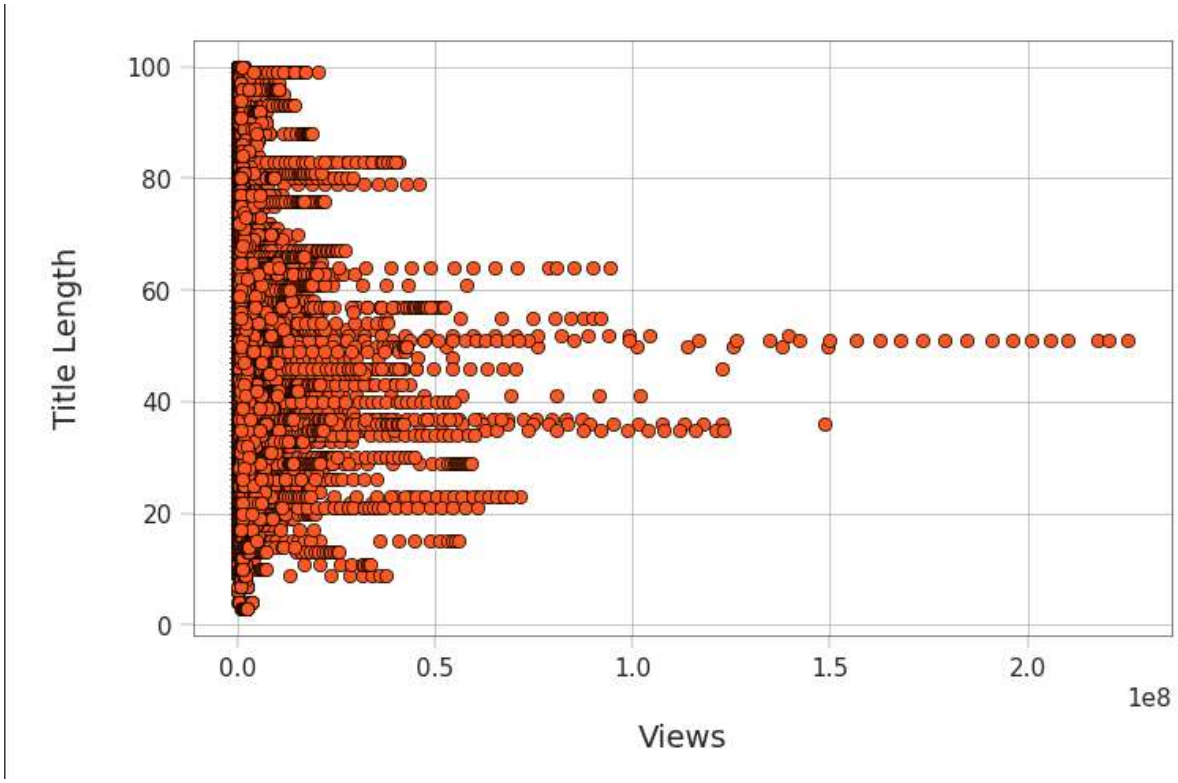
```
df["title_length"] = df["title"].apply(lambda x: len(x))
fig, ax = plt.subplots()
_ = sns.distplot(df["title_length"], kde=False, rug=False,
                 color=PLOT_COLORS[4], hist_kws={'alpha': 1}, ax=ax)
_ = ax.set(xlabel="Title Length", ylabel="No. of videos", xticks=range(
0, 110, 10))
```



We can see that the videos title length distribution looks like a normal distribution, where most videos have a title length of around 30-60 characters.

SCATTER PLOT:

```
fig, ax = plt.subplots()
_ = ax.scatter(x=df['views'], y=df['title_length'], color=PLOT_COLORS[2
], edgecolors="#000000", linewidths=0.5)
_ = ax.set(xlabel="Views", ylabel="Title Length")
```



Looking at the scatter plot, we can tell that there is no relationship between the length of the title and the number of views. However, we do notice an interesting thing that the Videos having 100,000,000 and more views have a title length of between 33 and 55 characters or so.

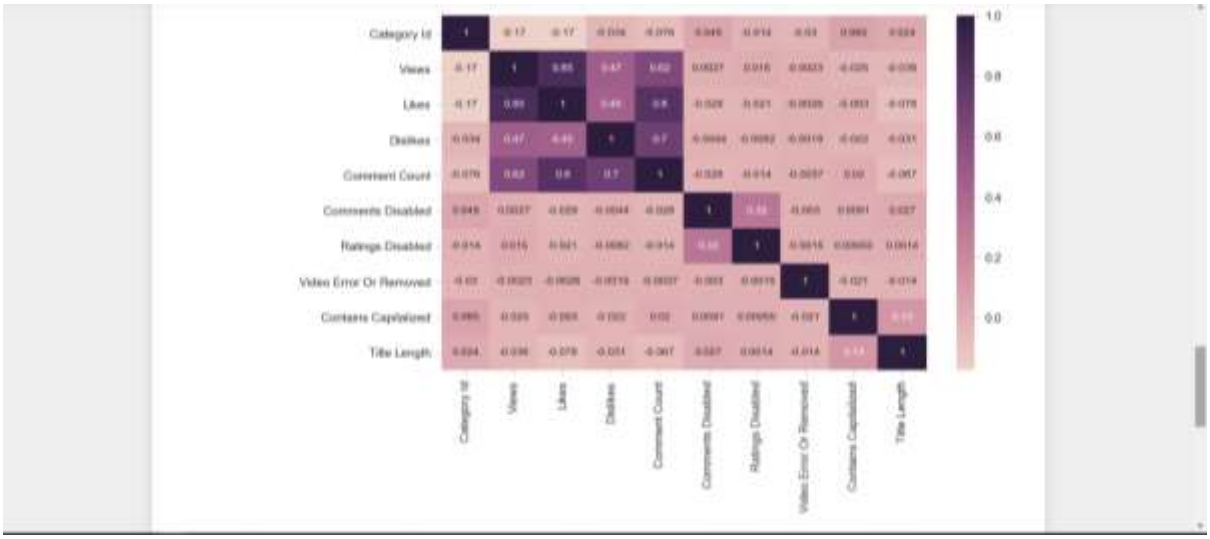
YOUTUBE TRENDING VIDEOS ANALYSIS : CORRELATION

We can see how views and likes correlate, meaning that views and likes increase and decrease together.

HEATMAP:

```
h_labels = [x.replace('_', ' ').title() for x in
            list(df.select_dtypes(include=['number', 'bool']).columns.v
alues)]

fig, ax = plt.subplots(figsize=(10,6))
_ = sns.heatmap(df.corr(), annot=True, xticklabels=h_labels, yticklabel
s=h_labels, cmap=sns.cubehelix_palette(as_cmap=True), ax=ax)
```



The correlation map and correlation table above indicate that views and likes are strongly positively correlated.

WORD CLOUD:

Word cloud for the titles of our trending videos, which is a way to visualize the most common words in the titles; the more common the word, the larger its font-size:

```
title_words = list(df["title"].apply(lambda x: x.split()))
title_words = [x for y in title_words for x in y]
wc = wordcloud.WordCloud(width=1200, height=500,
                         collocations=False, background_color="white",
                         colormap="tab20b").generate(" ".join(title_wor
ds))
plt.figure(figsize=(15,10))
plt.imshow(wc, interpolation='bilinear')
_ = plt.axis("off")
```


Official
Video
Trailer
New
Music
Challenge
Game
MAKEUP
Audio
Live
First
Day
Movie
Time
Black
Star
Show
World
Lyric
Make
Season
Full
War
One
Best
HD
Love
Year
Review
Test
Christmas
Teaser
Scene