Predictive Analytics for Trending Videos

The focus of this assessment is to explore what makes videos popular on various platforms, e.g., Netflix and YouTube. This notebook uses YouTube data for identification of patterns and predicting the number of views based on attributes in data

Required Libraries

```
import pandas as pd
import numpy as np
import plotly.express as px
import json
import matplotlib
import matplotlib.pyplot as plt
from matplotlib import cm
from datetime import datetime
import glob
import seaborn as sns
import re
import os
from matplotlib import pyplot
from sklearn.metrics import r2_score
import seaborn
import random
pd.options.mode.chained_assignment = None
```

Import your dataset using the following cells for the Youtube videos

Section 1. : Machine Learning with Sklearn

1.1.1 Data loading and Preprocessing

The dataset consists of a daily record of the top trending YouTube videos.

To determine the year's top-trending videos, YouTube uses a combination of factors including measuring users interactions, e.g., number of views, shares, comments and likes. "Note that they're not the most-viewed videos overall for the calendar year". Top performers on the YouTube trending list are music videos (such as the famously viral "Gangnam Style"), celebrity and/or reality TV performances, and the random dude-with-a-camera viral videos that YouTube is well-known for.

This dataset includes several months (and counting) of data on daily trending YouTube videos. Data is included for numerous countries, with up to 200 listed trending videos per day.

Each region's data is in a separate file. Data includes:

- Video Title
- Channel title
- Publish time
- Tags
- Views
- Likes
- Dislikes
- Description
- Comment count

The data also includes a category_id field, which varies between regions. To retrieve the categories for a specific video, find it in the associated JSON. One such file is included for each of the five regions in the dataset.

For more information on specific columns in the dataset refer to the column metadata.

1.1.1.1: Combining Multiple CSV's.

There are multiple csv files in the dataset, each corresponding to a specific country. As a first step, we need to read them and combine these csv files into a single dataframe. Use 'video_id' as your index.

While combining them, there is a need to create a column for "country" and fill it in the final dataframe. The country name can be extracted using the filename itself.

dataframe name: "combined_data".

```
files = [i for i in
glob.glob("/Users/goldyrana/work/ATU/Sem1/predictive/t_project_2/youtu
be_data/*.csv".format('csv'))]
files
sorted(files)

# Task: Merge all dataframes to single dataframe "combined_data" and
add a 'country' column.
all_dataframes = list()
for csv in files:
  frame = pd.read_csv(csv,index_col=0)
  frame['country'] = os.path.basename(csv)
  all_dataframes.append(frame)

combined_data = pd.concat(all_dataframes)
combined_data['country']=combined_data['country'].map(lambda x:
x.lstrip('+-').rstrip('videos.csv'))
combined_data
```

```
            trending_date
title  \
video_id
```

```
kzwfHumJyYc      17.14.11  Sharry Mann: Cute Munda ( Song Teaser) |
Parmi...
zUZ1z7FwLc8      17.14.11  पीरियड्स के समय, पेट पर पति करता ऐसा, देखकर दं...
10L1hZ9qa58      17.14.11  Stylish Star Allu Arjun @ ChaySam Wedding
Rece...
N1vE8iiEg64      17.14.11                      Eruma Saani | Tamil vs
English
kJzGH0PVQHQ      17.14.11  why Samantha became EMOTIONAL @ Samantha
naga ...
...                   ...
...
BZt0qjTWNhw      18.14.06                        The Cat Who Caught the
Laser
1h7KV2sjUWY      18.14.06                          True Facts : Ant
Mutualism
D6Oy4LfoqsU      18.14.06  I GAVE SAFIYA NYGAARD A PERFECT HAIR
MAKEOVER ...
oV0zkMe1K8s      18.14.06                   How Black Panther Should Have
Ended
ooyjaVdt-jA      18.14.06  Official Call of Duty®: Black Ops 4
— Multipla...

                        channel_title  category_id
publish_time  \
video_id

kzwfHumJyYc          Lokdhun Punjabi            1  2017-11-
12T12:20:39.000Z
zUZ1z7FwLc8                  HJ NEWS           25  2017-11-
13T05:43:56.000Z
10L1hZ9qa58                     TFPC           24  2017-11-
12T15:48:08.000Z
N1vE8iiEg64              Eruma Saani           23  2017-11-
12T07:08:48.000Z
kJzGH0PVQHQ               Filmylooks           24  2017-11-
13T01:14:16.000Z
...                              ...          ...
...
BZt0qjTWNhw            AaronsAnimals           15  2018-05-
18T13:00:04.000Z
1h7KV2sjUWY                 zefrank1           22  2018-05-
18T01:00:06.000Z
D6Oy4LfoqsU               Brad Mondo           24  2018-05-
18T17:34:22.000Z
oV0zkMe1K8s  How It Should Have Ended            1  2018-05-
17T17:00:04.000Z
ooyjaVdt-jA             Call of Duty           20  2018-05-
17T17:09:38.000Z
```

```
                                                          tags  
views  \
video_id                                                            

kzwfHumJyYc  sharry mann|"sharry mann new song"|"sharry man...  
1096327  
zUZ1z7FwLc8  पीरियड्स के समय|"पेट पर पति करता ऐसा"|"देखकर द...    590101
10L1hZ9qa58  Stylish Star Allu Arjun @ ChaySam Wedding Rece...  
473988  
N1vE8iiEg64  Eruma Saani|"Tamil Comedy Videos"|"Films"|"Mov...  
1242680  
kJzGH0PVQHQ  Filmylooks|"latest news"|"telugu movies"|"telu...  
464015  
...                                                        ...       .
..  
BZt0qjTWNhw  aarons animals|"aarons"|"animals"|"cat"|"cats"...  
1685609  
1h7KV2sjUWY                                             [none]  
1064798  
D6Oy4LfoqsU  I gave safiya nygaard a perfect hair makeover ...  
1066451  
oV0zkMe1K8s  Black Panther|"HISHE"|"Marvel"|"Infinity War"|...  
5660813  
ooyjaVdt-jA      call of duty|"cod"|"activision"|"Black Ops 4"  
10306119  

              likes  dislikes  comment_count  \
video_id                                       
kzwfHumJyYc   33966       798            882   
zUZ1z7FwLc8     735       904              0   
10L1hZ9qa58    2011       243            149   
N1vE8iiEg64   70353      1624           2684   
kJzGH0PVQHQ     492       293             66   
...             ...       ...            ...   
BZt0qjTWNhw   38160      1385           2657   
1h7KV2sjUWY   60008       382           3936   
D6Oy4LfoqsU   48068      1032           3992   
oV0zkMe1K8s  192957      2846          13088   
ooyjaVdt-jA  357079    212976         144795   

                                            thumbnail_link  \
video_id                                                      
kzwfHumJyYc  https://i.ytimg.com/vi/kzwfHumJyYc/default.jpg   
zUZ1z7FwLc8  https://i.ytimg.com/vi/zUZ1z7FwLc8/default.jpg   
10L1hZ9qa58  https://i.ytimg.com/vi/10L1hZ9qa58/default.jpg   
N1vE8iiEg64  https://i.ytimg.com/vi/N1vE8iiEg64/default.jpg   
kJzGH0PVQHQ  https://i.ytimg.com/vi/kJzGH0PVQHQ/default.jpg   
...                                                     ...   
BZt0qjTWNhw  https://i.ytimg.com/vi/BZt0qjTWNhw/default.jpg
```

```
1h7KV2sjUWY  https://i.ytimg.com/vi/1h7KV2sjUWY/default.jpg
D6Oy4LfoqsU  https://i.ytimg.com/vi/D6Oy4LfoqsU/default.jpg
oV0zkMe1K8s  https://i.ytimg.com/vi/oV0zkMe1K8s/default.jpg
ooyjaVdt-jA  https://i.ytimg.com/vi/ooyjaVdt-jA/default.jpg

             comments_disabled  ratings_disabled
video_error_or_removed  \
video_id

kzwfHumJyYc              False             False
False
zUZ1z7FwLc8               True             False
False
10L1hZ9qa58              False             False
False
N1vE8iiEg64              False             False
False
kJzGH0PVQHQ              False             False
False
...                        ...               ...                     .
..
BZt0qjTWNhw              False             False
False
1h7KV2sjUWY              False             False
False
D6Oy4LfoqsU              False             False
False
oV0zkMe1K8s              False             False
False
ooyjaVdt-jA              False             False
False

                                                   description country

video_id

kzwfHumJyYc  Presenting Sharry Mann latest Punjabi Song  Cu...      IN

zUZ1z7FwLc8  पीरियड्स के समय, पेट पर पति करता ऐसा, देखकर दं...      IN
10L1hZ9qa58  Watch Stylish Star Allu Arjun @ ChaySam Weddin...      IN

N1vE8iiEg64  This video showcases the difference between pe...      IN

kJzGH0PVQHQ  why Samantha became EMOTIONAL @ Samantha naga ...      IN

...                                                        ...     ...

BZt0qjTWNhw     The Cat Who Caught the Laser - Aaron's Animals      US

1h7KV2sjUWY                                                NaN      US
```

```
D6Oy4LfoqsU  I had so much fun transforming Safiyas hair in...      US
oV0zkMe1K8s  How Black Panther Should Have EndedWatch More ...      US
ooyjaVdt-jA  Call of Duty: Black Ops 4 Multiplayer raises t...      US

[159906 rows x 16 columns]
```

1.1.1.2: Map category Id's to categories

```
combined_data['category_id'] =
combined_data['category_id'].astype(str)
js_files = [i for i in
glob.glob('/Users/goldyrana/work/ATU/Sem1/predictive/t_project_2/youtu
be_data/*.json')]
sorted(js_files)

id_to_category = {}
for x in js_files:
  js = pd.read_json(x)
  for category in js ["items"]:
        id_to_category[category["id"]] = category["snippet"]["title"]
combined_data["category"] =
combined_data["category_id"].map(id_to_category)
#

combined_data.head(10)
```

```
            trending_date
title  \
video_id

kzwfHumJyYc      17.14.11  Sharry Mann: Cute Munda ( Song Teaser) |
Parmi...
zUZ1z7FwLc8      17.14.11  पीरियड्स के समय, पेट पर पति करता ऐसा, देखकर दं...
10L1hZ9qa58      17.14.11  Stylish Star Allu Arjun @ ChaySam Wedding
Rece...
N1vE8iiEg64      17.14.11                    Eruma Saani | Tamil vs
English
kJzGH0PVQHQ      17.14.11  why Samantha became EMOTIONAL @ Samantha
naga ...
il_pSa5l98w      17.14.11  MCA (Middle Class Abbayi) TEASER - Nani,Sai
Pa...
7MxiQ4v0EnE      17.14.11  Daang ( Full Video ) | Mankirt Aulakh |
Sukh S...
c64I9HNpiOY      17.14.11  Padmavati : Ek Dil Ek Jaan Video Song |
Deepik...
KObFEYCaRx8      17.14.11  Chiranjeevi in Naga Chaitanya - Samantha
```

```
Recep...
g8QsfJhFpjY      17.14.11                 New bike vs Old bike - the
reality

                channel_title category_id              publish_time  \
video_id
kzwfHumJyYc   Lokdhun Punjabi           1  2017-11-12T12:20:39.000Z
zUZ1z7FwLc8           HJ NEWS          25  2017-11-13T05:43:56.000Z
10L1hZ9qa58              TFPC          24  2017-11-12T15:48:08.000Z
N1vE8iiEg64       Eruma Saani          23  2017-11-12T07:08:48.000Z
kJzGH0PVQHQ        Filmylooks          24  2017-11-13T01:14:16.000Z
il_pSa5l98w          Dil Raju          24  2017-11-10T04:29:50.000Z
7MxiQ4v0EnE     Speed Records          10  2017-11-11T16:41:15.000Z
c64I9HNpiOY          T-Series          10  2017-11-11T06:14:19.000Z
KObFEYCaRx8  Top Telugu Media          24  2017-11-13T04:42:26.000Z
g8QsfJhFpjY         Jump Cuts          24  2017-11-12T04:30:01.000Z

                                                          tags
views  \
video_id

kzwfHumJyYc  sharry mann|"sharry mann new song"|"sharry man...
1096327
zUZ1z7FwLc8  पीरियड्स के समय|"पेट पर पति करता ऐसा"|"देखकर द...    590101
10L1hZ9qa58  Stylish Star Allu Arjun @ ChaySam Wedding Rece...
473988
N1vE8iiEg64  Eruma Saani|"Tamil Comedy Videos"|"Films"|"Mov...
1242680
kJzGH0PVQHQ  Filmylooks|"latest news"|"telugu movies"|"telu...
464015
il_pSa5l98w  Nenu Local|"Nenu Local Telugu Movie"|"Nani"|"S...
6106669
7MxiQ4v0EnE  punjabi songs|"punjabi bhangra"|"punjabi music...
5718766
c64I9HNpiOY  Ek Dil Ek Jaan Video Song|"'Ek Dil Ek Jaan'"|"...
10588371
KObFEYCaRx8  Chiranjeevi in Naga Chaitanya - Samantha Recep...
118223
g8QsfJhFpjY  Jump cuts|"Jumpcuts"|"Tamil comedy"|"Tamil Com...
969030

              likes  dislikes  comment_count  \
video_id
kzwfHumJyYc   33966       798            882
zUZ1z7FwLc8     735       904              0
10L1hZ9qa58    2011       243            149
N1vE8iiEg64   70353      1624           2684
kJzGH0PVQHQ     492       293             66
il_pSa5l98w   98612      4185           4763
7MxiQ4v0EnE  127477      7134           8063
```

```
c64I9HNpiOY  132738      8812          10847
KObFEYCaRx8     520        53             23
g8QsfJhFpjY   59798      1545           2404

                                             thumbnail_link  \
video_id
kzwfHumJyYc  https://i.ytimg.com/vi/kzwfHumJyYc/default.jpg
zUZ1z7FwLc8  https://i.ytimg.com/vi/zUZ1z7FwLc8/default.jpg
10L1hZ9qa58  https://i.ytimg.com/vi/10L1hZ9qa58/default.jpg
N1vE8iiEg64  https://i.ytimg.com/vi/N1vE8iiEg64/default.jpg
kJzGH0PVQHQ  https://i.ytimg.com/vi/kJzGH0PVQHQ/default.jpg
il_pSa5l98w  https://i.ytimg.com/vi/il_pSa5l98w/default.jpg
7MxiQ4v0EnE  https://i.ytimg.com/vi/7MxiQ4v0EnE/default.jpg
c64I9HNpiOY  https://i.ytimg.com/vi/c64I9HNpiOY/default.jpg
KObFEYCaRx8  https://i.ytimg.com/vi/KObFEYCaRx8/default.jpg
g8QsfJhFpjY  https://i.ytimg.com/vi/g8QsfJhFpjY/default.jpg

             comments_disabled  ratings_disabled
video_error_or_removed  \
video_id

kzwfHumJyYc              False             False
False
zUZ1z7FwLc8               True             False
False
10L1hZ9qa58              False             False
False
N1vE8iiEg64              False             False
False
kJzGH0PVQHQ              False             False
False
il_pSa5l98w              False             False
False
7MxiQ4v0EnE              False             False
False
c64I9HNpiOY              False             False
False
KObFEYCaRx8              False             False
False
g8QsfJhFpjY              False             False
False

                                                   description country
\
video_id

kzwfHumJyYc  Presenting Sharry Mann latest Punjabi Song  Cu...      IN

zUZ1z7FwLc8  पीरियड्स के समय, पेट पर पति करता ऐसा, देखकर दं...      IN
10L1hZ9qa58  Watch Stylish Star Allu Arjun @ ChaySam Weddin...      IN
```

```
N1vE8iiEg64  This video showcases the difference between pe...      IN
kJzGH0PVQHQ  why Samantha became EMOTIONAL @ Samantha naga ...      IN
il_pSa5l98w  Watch MCA- Middle Class Abbayi First Look Teas...      IN
7MxiQ4v0EnE  Song - Daang\nSinger - Mankirt Aulakh\nFaceboo...      IN
c64I9HNpiOY  Presenting the song 'Ek Dil Ek Jaan' from Padm...      IN
KObFEYCaRx8  Chiranjeevi in Naga Chaitanya - Samantha Recep...      IN
g8QsfJhFpjY  Jump Cuts is a Tamil entertaining group by Har...      IN
```

```
                     category
video_id
kzwfHumJyYc  Film & Animation
zUZ1z7FwLc8   News & Politics
10L1hZ9qa58     Entertainment
N1vE8iiEg64            Comedy
kJzGH0PVQHQ     Entertainment
il_pSa5l98w     Entertainment
7MxiQ4v0EnE             Music
c64I9HNpiOY             Music
KObFEYCaRx8     Entertainment
g8QsfJhFpjY     Entertainment
```

1.1.1.3: Fix datetime format and remove rows with NA's (1 pt)

The 'publish_time' and 'trending_date' features are not in a unix datetime format, so using pandas to_datetime() to convert it into the right format.

After that is done removing all the rows which have NA's in them.

```
combined_data.info()
combined_data['trending_date'] = 
pd.to_datetime(combined_data["trending_date"],format ="%y.%d.%m")
combined_data['publish_time'] = 
pd.to_datetime(combined_data["publish_time"],format = "%Y-%m-%dT%H:%M:
%S.%fZ")

combined_data = combined_data.dropna()
combined_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 159906 entries, kzwfHumJyYc to ooyjaVdt-jA
Data columns (total 17 columns):
 #   Column                  Non-Null Count   Dtype
```

```
---  ------                  --------------   -----
 0   trending_date           159906 non-null  object
 1   title                   159906 non-null  object
 2   channel_title           159906 non-null  object
 3   category_id             159906 non-null  object
 4   publish_time            159906 non-null  object
 5   tags                    159906 non-null  object
 6   views                   159906 non-null  int64
 7   likes                   159906 non-null  int64
 8   dislikes                159906 non-null  int64
 9   comment_count           159906 non-null  int64
 10  thumbnail_link          159906 non-null  object
 11  comments_disabled       159906 non-null  bool
 12  ratings_disabled        159906 non-null  bool
 13  video_error_or_removed  159906 non-null  bool
 14  description             154567 non-null  object
 15  country                 159906 non-null  object
 16  category                159906 non-null  object
dtypes: bool(3), int64(4), object(10)
memory usage: 18.8+ MB
<class 'pandas.core.frame.DataFrame'>
Index: 154567 entries, kzwfHumJyYc to ooyjaVdt-jA
Data columns (total 17 columns):
 #   Column                  Non-Null Count   Dtype
---  ------                  --------------   -----
 0   trending_date           154567 non-null  datetime64[ns]
 1   title                   154567 non-null  object
 2   channel_title           154567 non-null  object
 3   category_id             154567 non-null  object
 4   publish_time            154567 non-null  datetime64[ns]
 5   tags                    154567 non-null  object
 6   views                   154567 non-null  int64
 7   likes                   154567 non-null  int64
 8   dislikes                154567 non-null  int64
 9   comment_count           154567 non-null  int64
 10  thumbnail_link          154567 non-null  object
 11  comments_disabled       154567 non-null  bool
 12  ratings_disabled        154567 non-null  bool
 13  video_error_or_removed  154567 non-null  bool
 14  description             154567 non-null  object
 15  country                 154567 non-null  object
 16  category                154567 non-null  object
dtypes: bool(3), datetime64[ns](2), int64(4), object(8)
memory usage: 18.1+ MB

# print
print('validate_na',(combined_data.shape))

validate_na (154567, 17)
```

1.2 Exploratory Data Analysis & Feature Engineering

Exploratory Data Analysis: EDA aims to analyze data sets by summarizing its key characteristics assisted by visualizations. EDA communicates insights beyond formal modeling/hypothesis testing with or without statistical model.

Feature Engineering: The primary object of feature engineering is to extract features using domain knowledge. It aims to extract features from raw data using various data mining approaches.

These features are fed to various machine learning classifiers. These features are also called as covariates, predicators, or simply a new column in data frame.

1.2.1: Calculating Mean, standard deviation, min and max.

In this section, statistics for numberical features is measured and then store into lists i.e., [views, likes, dislikes, comment_count].

means = [views_mean, likes_mean, dislikes_mean, comment_count_mean] and similarly for mins, maxs and stds.

```
combined_data.describe()
```

	trending_date	publish_time
count	154567	154567
mean	2018-02-27 02:23:21.598012416	2018-02-21 11:58:53.405118976
min	2017-11-14 00:00:00	2006-07-23 08:24:11
25%	2018-01-03 00:00:00	2018-01-01 02:28:48.500000
50%	2018-02-26 00:00:00	2018-02-23 17:00:02
75%	2018-04-24 00:00:00	2018-04-21 06:22:47
max	2018-06-14 00:00:00	2018-06-14 02:25:38
std	NaN	NaN

\

	views	likes	dislikes	comment_count
count	1.545670e+05	1.545670e+05	1.545670e+05	1.545670e+05
mean	1.281578e+06	4.096105e+04	2.056138e+03	4.606594e+03
min	2.230000e+02	0.000000e+00	0.000000e+00	0.000000e+00
25%	9.574900e+04	1.321000e+03	8.100000e+01	1.720000e+02
50%	3.134280e+05	6.336000e+03	2.980000e+02	7.650000e+02
75%	9.473390e+05	2.594050e+04	1.024000e+03	2.726000e+03
max	2.252119e+08	5.613827e+06	1.643059e+06	1.228655e+06
std	4.605292e+06	1.521490e+05	1.825854e+04	2.327823e+04

```
maxs = combined_data.describe().iloc[7].values.tolist()
mins = combined_data.describe().iloc[3].values.tolist()
stds = combined_data.describe().iloc[2].values.tolist()
means = combined_data.describe().iloc[1].values.tolist()
```

```
# print here
print('check_min_max_mean_std',([maxs, mins, stds, means]))
```

```
check_min_max_mean_std [[nan, nan, 4605292.478385794,
152148.95485475138, 18258.541379660703, 23278.225620617904],
[Timestamp('2018-01-03 00:00:00'), Timestamp('2018-01-01
02:28:48.500000'), 95749.0, 1321.0, 81.0, 172.0], [Timestamp('2017-11-
14 00:00:00'), Timestamp('2006-07-23 08:24:11'), 223.0, 0.0, 0.0,
0.0], [Timestamp('2018-02-27 02:23:21.598012416'), Timestamp('2018-02-
21 11:58:53.405118976'), 1281578.03423758, 40961.05191276275,
2056.138490104615, 4606.593742519425]]
```

1.2.2: Rescaling the features

From the above section, it is clear that the numerical values range is really high. we can use rescaling to avoid numerical instability problems. We can rescale likes, views, dislikes, and comment_count using log scale (base e). Let us store rescaled features in dataframe as likes_log, views_log, dislikes_log and comment_log.

NOTE- log 0 is not defined, therefore, 1 is added to each value prior to taking the log.

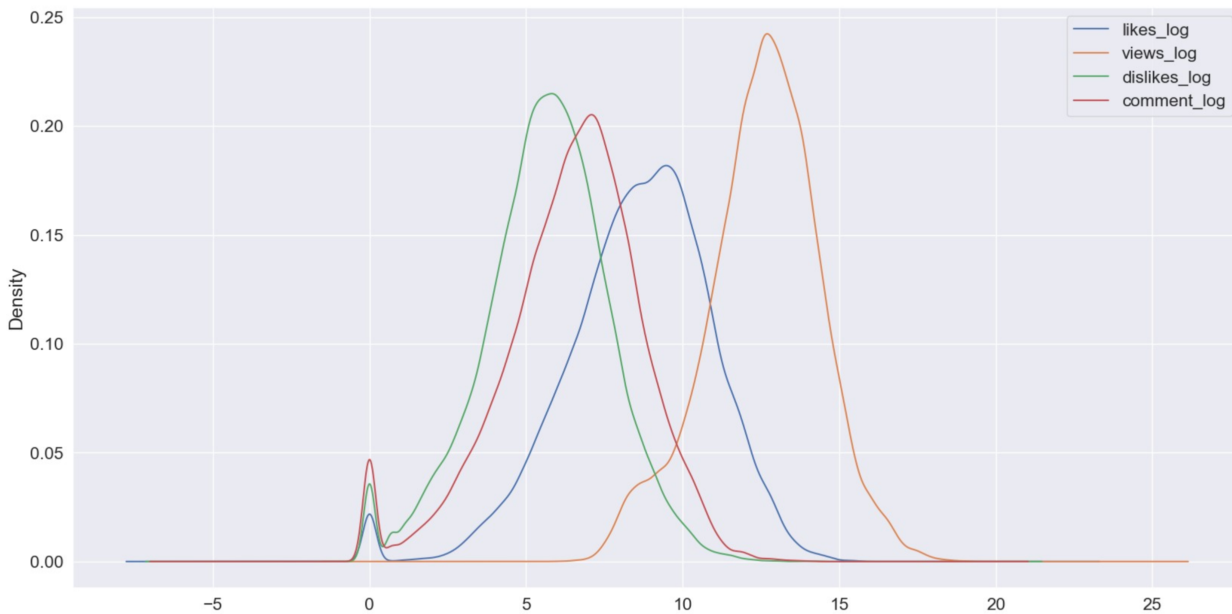
```
combined_data['likes_log'] = np.log(1 + combined_data['likes'])
combined_data['views_log'] = np.log(1 + combined_data['views'])
combined_data['dislikes_log'] = np.log(1 + combined_data['dislikes'])
combined_data['comment_log'] = np.log(1 +
combined_data['comment_count'])

# Print results
print('check_feature_rescaling',
([np.mean(combined_data['likes_log']),np.mean(combined_data['views_log
']),np.mean(combined_data['dislikes_log']),

np.mean(combined_data['comment_log'])]))
```

```
check_feature_rescaling [8.571590187188637, 12.552679805013259,
5.614484952080614, 6.424543615107988]
```

1.2.3: Plotting the distribution (2 pt)

Plotting the distribution for the newly created log features.

```
log_df=combined_data[['likes_log','views_log','dislikes_log','comment_
log']]
log_df.plot.kde(figsize=(20,10))
plt.show()
```



1.2.4: Comparing views, likes, dislikes against categories

Gaining isights into data using various catergories, views, likes and dislikes.

1.) How many videos are there for each category?

2.) What is the distribution of views against categories? (Use boxplot and views on log scale)

3.) What is the distribution of dislikes against categories? (Use boxplot and dislikes on log scale)

```
##1-Total videos for each category:
by_category =
(combined_data.groupby(["category"]).size().sort_values(ascending =
False)/len(combined_data)) * 100
print('Total videos for each category: \n', by_category)
```

```
Total videos for each category:
 category
Entertainment            31.678172
Music                    11.527687
Comedy                    9.568666
News & Politics           9.518849
People & Blogs            9.027800
Sports                    6.138438
Howto & Style             6.022631
Film & Animation          5.094231
Science & Technology      3.135210
Education                 2.939825
Gaming                    2.323911
Pets & Animals            0.984686
Autos & Vehicles          0.909638
Travel & Events           0.575155
```

```
Shows                          0.313780
Nonprofits & Activism          0.218675
Movies                         0.021350
Trailers                       0.001294
dtype: float64
```

```python
import plotly.express as px
df_cat=pd.DataFrame(by_category)
df_cat['category'] = df_cat.index
df_cat.columns=['a','b']
fig4 = px.bar(df_cat, x="b",
y="a",color='a',labels={'b':'Category','a':'Videos/Category'},
height=400)
fig4.update_layout({'plot_bgcolor': 'rgba(0, 0, 0,
0)','paper_bgcolor': 'rgba(0, 0, 0, 0)',})
```


```
{"config":{"plotlyServerURL":"https://plot.ly"},"data":
[{"alignmentgroup":"True","hovertemplate":"Category=%{x}<br>Videos/
Category=%{marker.color}<extra></extra>","legendgroup":"","marker":
{"color":
[31.678171925443333,11.527687022456282,9.568666015384913,9.51884943099
1091,9.027800241966268,6.1384383471245485,6.022630962624622,5.09423098
0739744,3.135209973668377,2.939825447864033,2.3239113135404064,0.98468
62525636132,0.9096378916586335,0.5751551107286808,0.31377978481823415,
0.21867539643002712,2.1349964740209747e-2,1.2939372569824089e-
3],"coloraxis":"coloraxis","pattern":
{"shape":""}},"name":"","offsetgroup":"","orientation":"v","showlegend
":false,"textposition":"auto","type":"bar","x":
["Entertainment","Music","Comedy","News & Politics","People &
Blogs","Sports","Howto & Style","Film & Animation","Science &
Technology","Education","Gaming","Pets & Animals","Autos &
Vehicles","Travel & Events","Shows","Nonprofits &
Activism","Movies","Trailers"],"xaxis":"x","y":
[31.678171925443333,11.527687022456282,9.568666015384913,9.51884943099
1091,9.027800241966268,6.1384383471245485,6.022630962624622,5.09423098
0739744,3.135209973668377,2.939825447864033,2.3239113135404064,0.98468
62525636132,0.9096378916586335,0.5751551107286808,0.31377978481823415,
0.21867539643002712,2.1349964740209747e-2,1.2939372569824089e-
3],"yaxis":"y"}],"layout":{"barmode":"relative","coloraxis":
{"colorbar":{"title":{"text":"Videos/Category"}},"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"]]},"height":400,"legend":
{"tracegroupgap":0},"margin":{"t":60},"paper_bgcolor":"rgba(0, 0, 0,
0)","plot_bgcolor":"rgba(0, 0, 0, 0)","template":{"data":{"bar":
[{"error_x":{"color":"#2a3f5f"},"error_y":
{"color":"#2a3f5f"},"marker":{"line":
{"color":"#E5ECF6","width":0.5},"pattern":
```

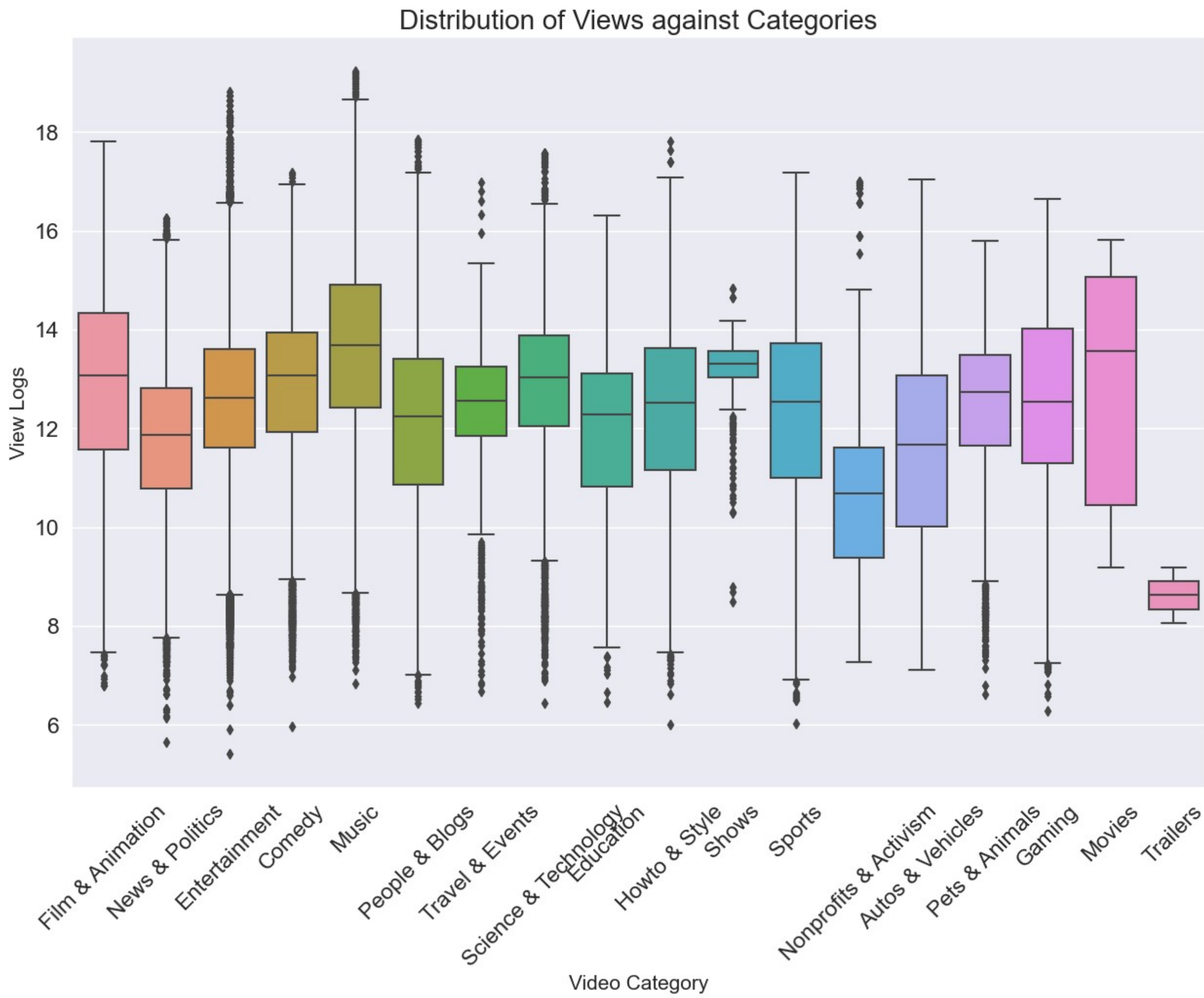
```
{"fillmode":"overlay","size":10,"solidity":0.2}},"type":"bar"}],"barpo
lar":[{"marker":{"line":{"color":"#E5ECF6","width":0.5},"pattern":
{"fillmode":"overlay","size":10,"solidity":0.2}},"type":"barpolar"}],"
carpet":[{"aaxis":
{"endlinecolor":"#2a3f5f","gridcolor":"white","linecolor":"white","min
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":"#E5ECF
6","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}}},"xaxis":
{"anchor":"y","domain":[0,1],"title":{"text":"Category"}},"yaxis":
{"anchor":"x","domain":[0,1],"title":{"text":"Videos/Category"}}}}
```

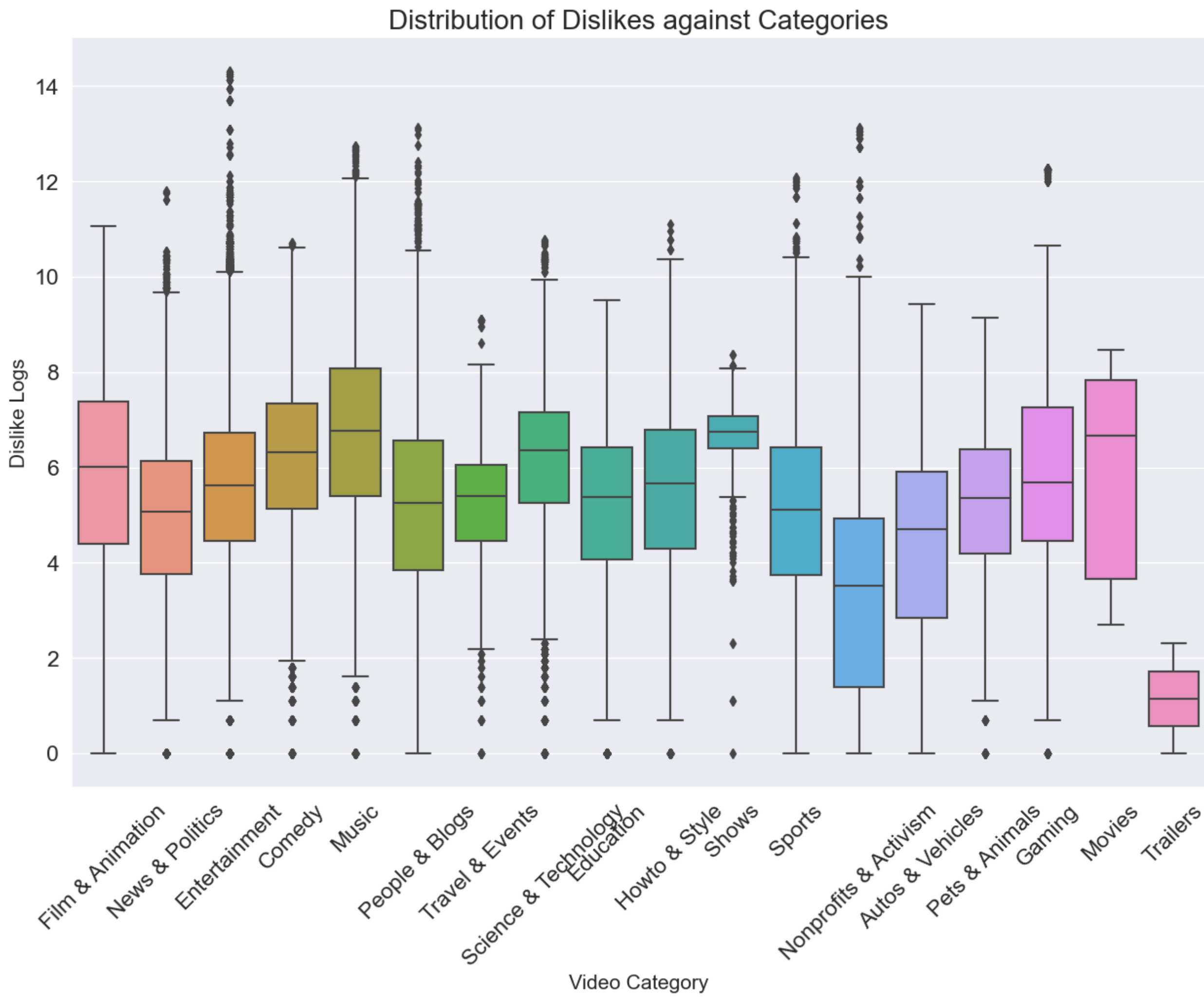

```python
##2-Distribution of views against categories, using boxplot and views
on log scale
fig, ax = pyplot.subplots(figsize=(15,10))
sns.boxplot(x="category", y="views_log",
data=combined_data[['category','views_log']]),plt.xticks(rotation =
45)
ax.set_title("Distribution of Views against
Categories",fontsize=20),ax.set_xlabel('Video
Category',fontsize=15),ax.set_ylabel(ylabel='View Logs',fontsize=15)
```

```
(Text(0.5, 1.0, 'Distribution of Views against Categories'),
 Text(0.5, 0, 'Video Category'),
 Text(0, 0.5, 'View Logs'))
```



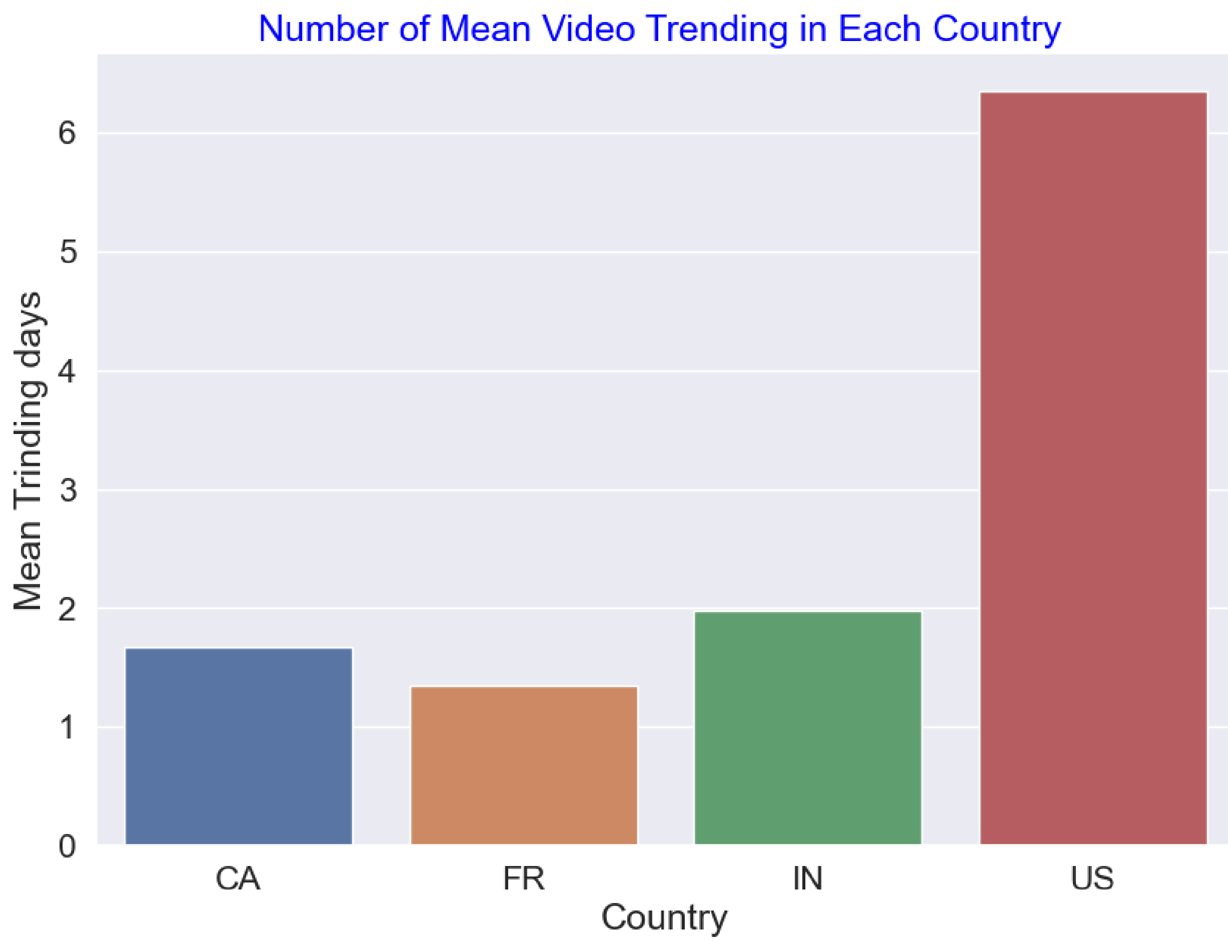

```
##3-Distribution of dislikes against categories, Using boxplot and
dislikes on log scale
fig, ax = pyplot.subplots(figsize=(15,10))
sns.boxplot(x="category", y="dislikes_log",
data=combined_data[['category','dislikes_log']]),plt.xticks(rotation =
45)
ax.set_title("Distribution of Dislikes against
Categories",fontsize=20),ax.set_xlabel('Video
Category',fontsize=15),ax.set_ylabel(ylabel='Dislike
Logs',fontsize=15)
```

```
(Text(0.5, 1.0, 'Distribution of Dislikes against Categories'),
 Text(0.5, 0, 'Video Category'),
 Text(0, 0.5, 'Dislike Logs'))
```



How long does a video trend in a country?

```
df=combined_data.drop_duplicates()
df1=df.groupby(['title','country']).size().reset_index(name='count')
trending=df1.groupby('country')
['count'].mean().to_frame().reset_index().rename(columns={"country":
"Country", "count": "Mean Trinding days"})

fig, ax = pyplot.subplots(figsize=(10, 7)),sns.set(font_scale=1.5)
sns.barplot(x="Country", y="Mean Trinding days",
data=trending,ax=ax),plt.title('Number of Mean Video Trending in Each
Country ',color='Blue')
plt.show()
```



What are some videos which got popular because they were disliked?

```
df2=df.groupby(['title','dislikes','likes']).size().reset_index(name='
Trend_Days')
trending_dislikes=df2.loc[(df2['dislikes'] > df2['likes']) &
(df2['Trend_Days'] >= 3)] #Videos which got trends and have dislikes
more than likes#
trending_dislikes=trending_dislikes.sort_values(['Trend_Days'],ascendi
ng=[ False])
print('Videos which got popular because they were disliked: \n \
n',trending_dislikes.title.to_string(index=False))
```

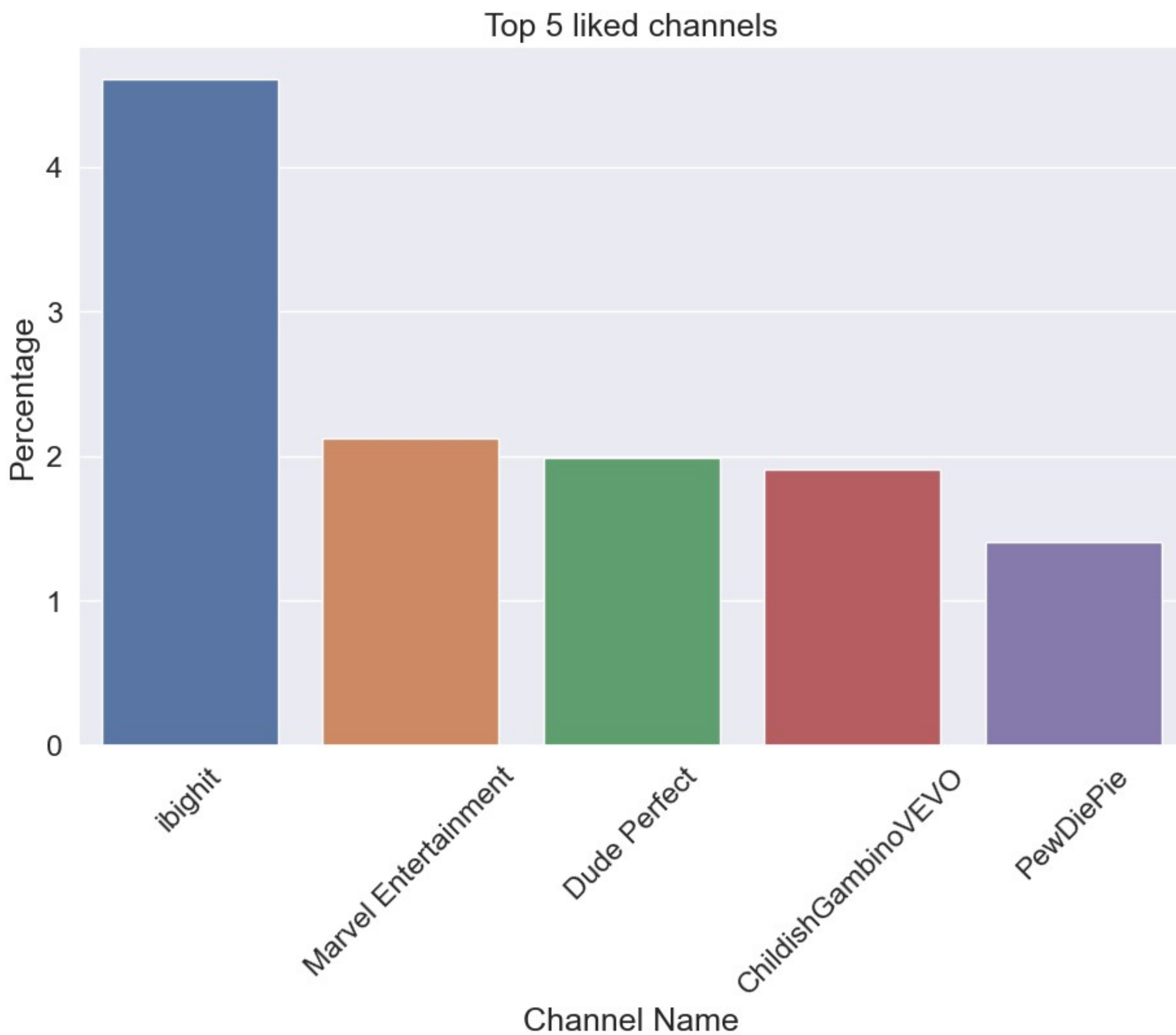
```
Videos which got popular because they were disliked:

 Jeffrey Tambor Fired From 'Transparent' Followi...
5 Things You Missed at the 2018 SAG Awards | E!...
Fergie Performs The U.S. National Anthem / 2018...
                        Staudt on Sports I 1-22-18
         WATCH: Sen. Mitch McConnell on tax reform
                               二贵摔跤 - tienghoa.net
```

Which channel get the most likes?

```
channel_with_likes.head()
```

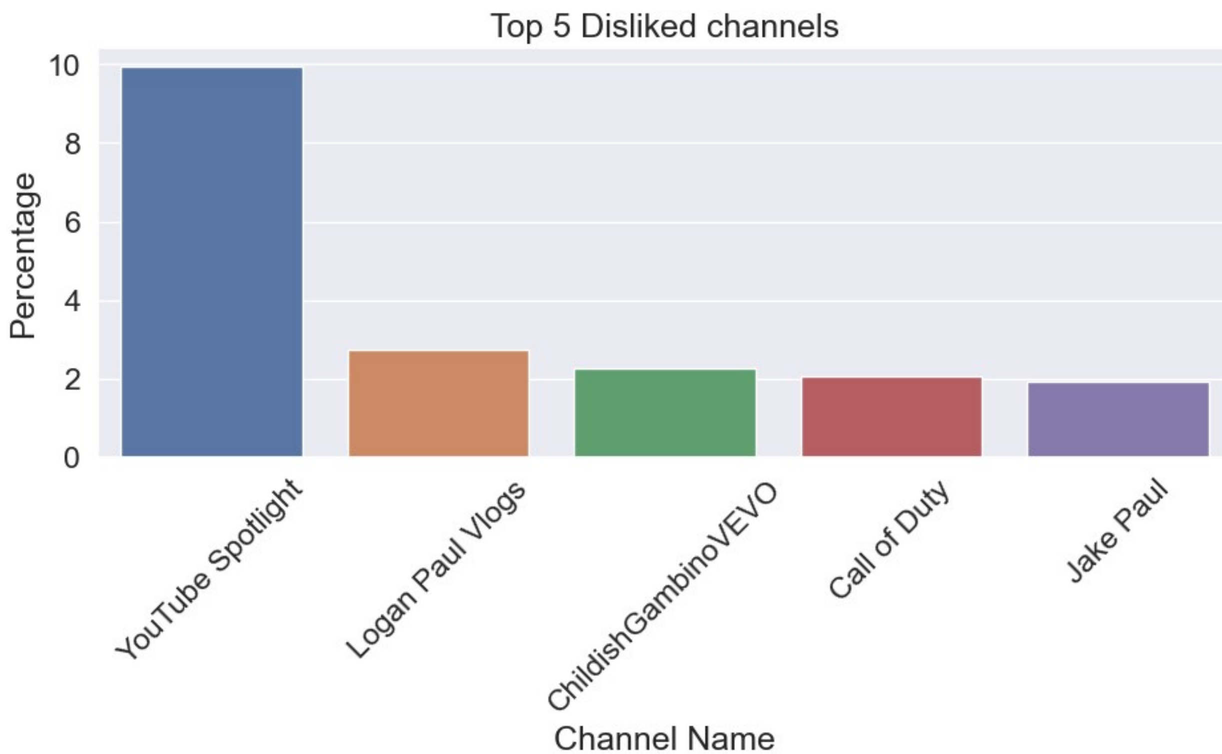
```
                              likes
channel_title
ibighit                    4.604578
Marvel Entertainment       2.120947
Dude Perfect               1.987921
ChildishGambinoVEVO        1.906940
PewDiePie                  1.408246
```

```
channel_with_likes = df[["channel_title",
"likes"]].groupby("channel_title").sum().sort_values("likes",
ascending=False)
channel_with_likes =
(channel_with_likes/channel_with_likes.likes.sum()) * 100
fig, ax = pyplot.subplots(figsize=(11, 7))
sns.barplot(x = channel_with_likes.head(5).index, y =
channel_with_likes.head(5).likes.values)
plt.xticks(rotation = 45)
plt.xlabel("Channel Name")
plt.ylabel("Percentage")
plt.title("Top 5 liked channels")
plt.show()
```

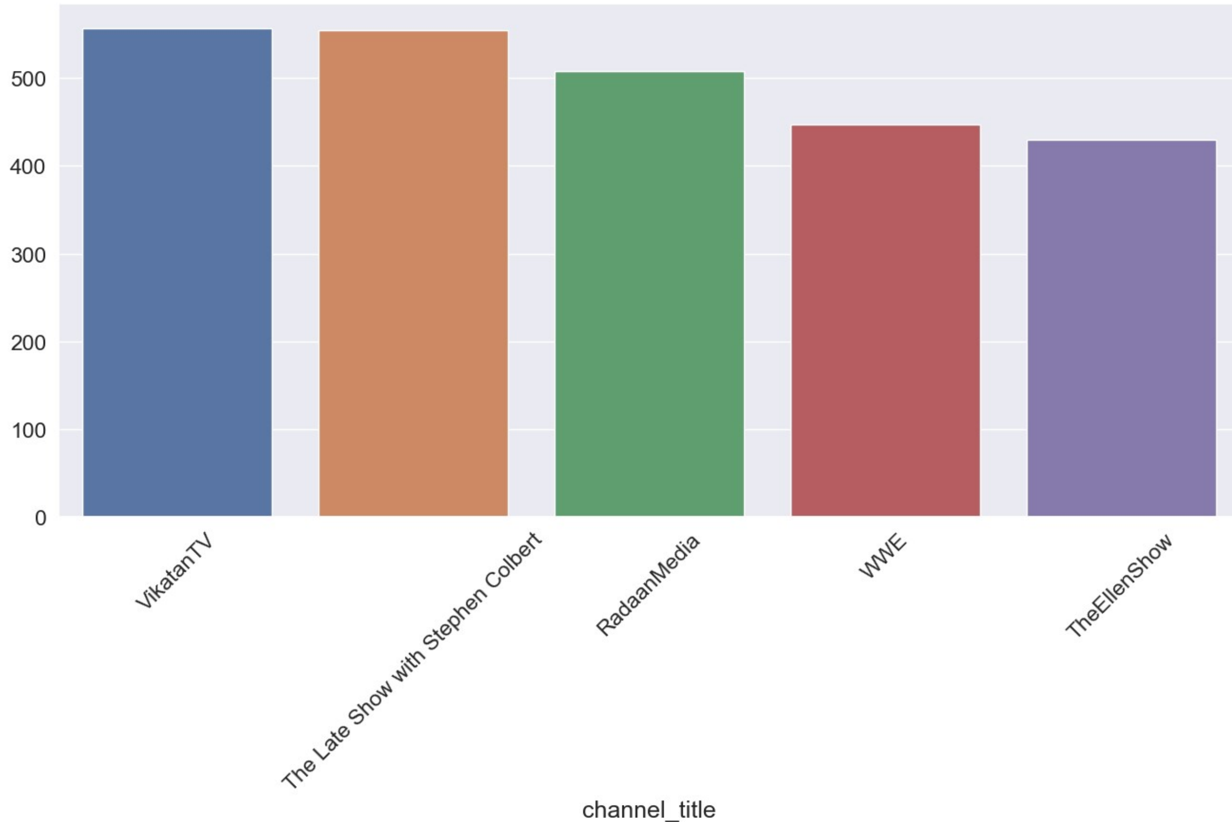


Which Channel get the most dislikes?

```
channel_with_dislikes = df[["channel_title",
"dislikes"]].groupby("channel_title").sum().sort_values("dislikes",
ascending=False)
channel_with_dislikes =
(channel_with_dislikes/channel_with_dislikes.dislikes.sum()) * 100
fig, ax = pyplot.subplots(figsize=(11, 4))
sns.barplot(x = channel_with_dislikes.head(5).index, y =
channel_with_dislikes.head(5).dislikes.values)
plt.xticks(rotation = 45)
plt.xlabel("Channel Name")
plt.ylabel("Percentage")
plt.title("Top 5 Disliked channels")
plt.show()
```




```
# Top contributing channels
fig, ax = pyplot.subplots(figsize=(16,7))
top_5_channels = df["channel_title"].value_counts().head(5)
sns.barplot(x = top_5_channels.index, y = top_5_channels.values, ax =
ax)
plt.xticks(rotation = 45)
plt.show()
```




```
top_5_channels
```

```
channel_title
VikatanTV                             557
The Late Show with Stephen Colbert    554
RadaanMedia                           508
WWE                                   447
TheEllenShow                          430
Name: count, dtype: int64
```

1.2.5: Feature Engineering

a. Processing tags

The feature tags in the dataset has a delimiter, use that delimiter to count the number of tags, create a feature called num_tags and add that to the dataset.

```
num_tags=[]
xdf=combined_data.reset_index(drop=True)
for i in range(len(combined_data)):
  if xdf.at[i,'tags']=='[none]': #some videos has no tags but instead
[none], so we are going to consider it as Zero tags.
    count=0
  else:
```

```
    count=(xdf.at[i,'tags']).count("|") + 1
  num_tags.append(count)
combined_data['num_tags']=num_tags
combined_data
```

```
            trending_date
title  \
video_id

kzwfHumJyYc    2017-11-14  Sharry Mann: Cute Munda ( Song Teaser) |
Parmi...
zUZ1z7FwLc8    2017-11-14  पीरियड्स के समय, पेट पर पति करता ऐसा, देखकर दं...
10L1hZ9qa58    2017-11-14  Stylish Star Allu Arjun @ ChaySam Wedding
Rece...
N1vE8iiEg64    2017-11-14                   Eruma Saani | Tamil vs
English
kJzGH0PVQHQ    2017-11-14  why Samantha became EMOTIONAL @ Samantha
naga ...
...                   ...
...
1PhPYr_9zRY    2018-06-14  BTS Plays With Puppies While Answering Fan
Que...
BZt0qjTWNhw    2018-06-14                     The Cat Who Caught the
Laser
D6Oy4LfoqsU    2018-06-14  I GAVE SAFIYA NYGAARD A PERFECT HAIR
MAKEOVER ...
oV0zkMe1K8s    2018-06-14                 How Black Panther Should Have
Ended
ooyjaVdt-jA    2018-06-14  Official Call of Duty®: Black Ops 4
— Multipla...

                        channel_title category_id        publish_time
\
video_id

kzwfHumJyYc           Lokdhun Punjabi           1 2017-11-12 12:20:39

zUZ1z7FwLc8                   HJ NEWS          25 2017-11-13 05:43:56

10L1hZ9qa58                      TFPC          24 2017-11-12 15:48:08

N1vE8iiEg64               Eruma Saani          23 2017-11-12 07:08:48

kJzGH0PVQHQ                Filmylooks          24 2017-11-13 01:14:16

...                               ...         ...                 ...

1PhPYr_9zRY            BuzzFeed Celeb          22 2018-05-18 16:39:29

BZt0qjTWNhw              AaronsAnimals          15 2018-05-18 13:00:04
```

```
D6Oy4LfoqsU                Brad Mondo          24 2018-05-18 17:34:22

oV0zkMe1K8s  How It Should Have Ended           1 2018-05-17 17:00:04

ooyjaVdt-jA              Call of Duty          20 2018-05-17 17:09:38


                                                          tags
views  \
video_id

kzwfHumJyYc  sharry mann|"sharry mann new song"|"sharry man...
1096327
zUZ1z7FwLc8  पीरियड्स के समय|"पेट पर पति करता ऐसा"|"देखकर द...    590101
10L1hZ9qa58  Stylish Star Allu Arjun @ ChaySam Wedding Rece...
473988
N1vE8iiEg64  Eruma Saani|"Tamil Comedy Videos"|"Films"|"Mov...
1242680
kJzGH0PVQHQ  Filmylooks|"latest news"|"telugu movies"|"telu...
464015
...                                                        ...        .
..
1PhPYr_9zRY  BuzzFeed|"BuzzFeedVideo"|"Puppy Interview"|"pu...
8259128
BZt0qjTWNhw  aarons animals|"aarons"|"animals"|"cat"|"cats"...
1685609
D6Oy4LfoqsU  I gave safiya nygaard a perfect hair makeover ...
1066451
oV0zkMe1K8s  Black Panther|"HISHE"|"Marvel"|"Infinity War"|...
5660813
ooyjaVdt-jA      call of duty|"cod"|"activision"|"Black Ops 4"
10306119

              likes  dislikes  comment_count  ... ratings_disabled  \
video_id                                      ...
kzwfHumJyYc   33966       798            882  ...             False
zUZ1z7FwLc8     735       904              0  ...             False
10L1hZ9qa58    2011       243            149  ...             False
N1vE8iiEg64   70353      1624           2684  ...             False
kJzGH0PVQHQ     492       293             66  ...             False
...             ...       ...            ...  ...               ...
1PhPYr_9zRY  645888      4052          62610  ...             False
BZt0qjTWNhw   38160      1385           2657  ...             False
D6Oy4LfoqsU   48068      1032           3992  ...             False
oV0zkMe1K8s  192957      2846          13088  ...             False
ooyjaVdt-jA  357079    212976         144795  ...             False

             video_error_or_removed  \
video_id
```

```
kzwfHumJyYc                      False
zUZ1z7FwLc8                      False
10L1hZ9qa58                      False
N1vE8iiEg64                      False
kJzGH0PVQHQ                      False
...                                ...
1PhPYr_9zRY                      False
BZt0qjTWNhw                      False
D6Oy4LfoqsU                      False
oV0zkMe1K8s                      False
ooyjaVdt-jA                      False

                                                     description
country  \
video_id

kzwfHumJyYc  Presenting Sharry Mann latest Punjabi Song  Cu...
IN
zUZ1z7FwLc8  पीरियड्स के समय, पेट पर पति करता ऐसा, देखकर दं...       IN
10L1hZ9qa58  Watch Stylish Star Allu Arjun @ ChaySam Weddin...
IN
N1vE8iiEg64  This video showcases the difference between pe...
IN
kJzGH0PVQHQ  why Samantha became EMOTIONAL @ Samantha naga ...
IN
...                                                        ...       ..
.
1PhPYr_9zRY  BTS with the PPS, the puppies. These adorable ...
US
BZt0qjTWNhw     The Cat Who Caught the Laser - Aaron's Animals
US
D6Oy4LfoqsU  I had so much fun transforming Safiyas hair in...
US
oV0zkMe1K8s  How Black Panther Should Have EndedWatch More ...
US
ooyjaVdt-jA  Call of Duty: Black Ops 4 Multiplayer raises t...
US

                     category  likes_log  views_log  dislikes_log  \
video_id
kzwfHumJyYc  Film & Animation  10.433145  13.907477      6.683361
zUZ1z7FwLc8   News & Politics   6.601230  13.288051      6.807935
10L1hZ9qa58     Entertainment   7.606885  13.068939      5.497168
N1vE8iiEg64            Comedy  11.161295  14.032782      7.393263
kJzGH0PVQHQ     Entertainment   6.200509  13.047674      5.683580
...                       ...        ...        ...           ...
1PhPYr_9zRY    People & Blogs  13.378383  15.926830      8.307213
BZt0qjTWNhw    Pets & Animals  10.549569  14.337638      7.234177
D6Oy4LfoqsU     Entertainment  10.780393  13.879848      6.940222
```

```
oV0zkMe1K8s  Film & Animation  12.170228  15.549078       7.954021
ooyjaVdt-jA            Gaming  12.785715  16.148248      12.268939

             comment_log  num_tags
video_id
kzwfHumJyYc     6.783325        15
zUZ1z7FwLc8     0.000000        19
10L1hZ9qa58     5.010635        14
N1vE8iiEg64     7.895436        20
kJzGH0PVQHQ     4.204693        11
...                  ...       ...
1PhPYr_9zRY    11.044696        27
BZt0qjTWNhw     7.885329        14
D6Oy4LfoqsU     8.292298        24
oV0zkMe1K8s     9.479527        22
ooyjaVdt-jA    11.883081         4

[154567 rows x 22 columns]
```

b. Processing description and title

Calculate the length of description and title and add them as features to the dataset

```
combined_data["desc_len"]=combined_data["description"].apply(lambda x:
len(x))

combined_data["len_title"]=combined_data["title"].apply(lambda x:
len(x))

combined_data["overall_len_title_desc"] = combined_data.desc_len +
combined_data.len_title

# Print cell
print('check_tags_title_description \n',
([combined_data['num_tags'].describe(),combined_data['desc_len'].descr
ibe(),combined_data['len_title'].describe()]))

check_tags_title_description
 [count    154567.000000
mean         18.580551
std          11.929906
min           0.000000
25%           9.000000
50%          17.000000
75%          26.000000
max         124.000000
Name: num_tags, dtype: float64, count    154567.000000
mean        959.949426
std         857.504028
min           1.000000
```

```
25%          363.000000
50%          717.000000
75%         1288.000000
max         5260.000000
Name: desc_len, dtype: float64, count    154567.000000
mean           56.408541
std            22.976198
min             2.000000
25%            38.000000
50%            54.000000
75%            74.000000
max           100.000000
Name: len_title, dtype: float64]
```

c. Processing publish_time.

Split 'publish_time' feature into three parts time, date, and weekday, where time will contain the time component of the original feature and date and weekday will store the corresponding date and weekday number respectively. Start with 1 for Monday and end with 7 for Sunday.

```
date_data=combined_data['publish_time']

combined_data['publish_time'] =date_data.apply(lambda x:
pd.to_datetime(x).time())
combined_data['publish_date'] =date_data.apply(lambda x:
pd.to_datetime(x).date())

#day on which video was published
combined_data['publish_weekday']=date_data.apply(lambda x:
x.dayofweek)+1

import random
random_index = random.randint(0,combined_data.shape[0]-1)

----------------------------------------------------------------------
-----
KeyboardInterrupt                         Traceback (most recent call
last)
Cell In[249], line 2
      1 combined_data['publish_time'] =date_data.apply(lambda x:
pd.to_datetime(x).time())
----> 2 combined_data['publish_date'] =date_data.apply(lambda x:
pd.to_datetime(x).date())
      4 #day on which video was published
      5 combined_data['publish_weekday']=date_data.apply(lambda x:
x.dayofweek)+1

File
~/Library/Python/3.9/lib/python/site-packages/pandas/core/series.py:46
30, in Series.apply(self, func, convert_dtype, args, **kwargs)
```

```
   4520 def apply(
   4521     self,
   4522     func: AggFuncType,
   (...)
   4525     **kwargs,
   4526 ) -> DataFrame | Series:
   4527     """
   4528     Invoke function on values of Series.
   4529 
   (...)
   4628     dtype: float64
   4629     """
-> 4630     return SeriesApply(self, func, convert_dtype, args,
kwargs).apply()

File
~/Library/Python/3.9/lib/python/site-packages/pandas/core/apply.py:102
5, in SeriesApply.apply(self)
   1022     return self.apply_str()
   1024 # self.f is Callable
-> 1025 return self.apply_standard()

File
~/Library/Python/3.9/lib/python/site-packages/pandas/core/apply.py:107
6, in SeriesApply.apply_standard(self)
   1074     else:
   1075         values = obj.astype(object)._values
-> 1076         mapped = lib.map_infer(
   1077             values,
   1078             f,
   1079             convert=self.convert_dtype,
   1080         )
   1082 if len(mapped) and isinstance(mapped[0], ABCSeries):
   1083     # GH#43986 Need to do list(mapped) in order to get treated
as nested
   1084     #  See also GH#25959 regarding EA support
   1085     return obj._constructor_expanddim(list(mapped),
index=obj.index)

File
~/Library/Python/3.9/lib/python/site-packages/pandas/_libs/lib.pyx:283
4, in pandas._libs.lib.map_infer()

Cell In[249], line 2, in <lambda>(x)
      1 combined_data['publish_time'] =date_data.apply(lambda x:
pd.to_datetime(x).time())
----> 2 combined_data['publish_date'] =date_data.apply(lambda x:
pd.to_datetime(x).date())
      4 #day on which video was published
      5 combined_data['publish_weekday']=date_data.apply(lambda x:
```

```
x.dayofweek)+1

File
~/Library/Python/3.9/lib/python/site-packages/pandas/core/tools/dateti
mes.py:1026, in to_datetime(arg, errors, dayfirst, yearfirst, utc,
format, exact, unit, infer_datetime_format, origin, cache)
   1023 if origin != "unix":
   1024     arg = _adjust_to_origin(arg, origin, unit)
-> 1026 convert_listlike = partial(
   1027     _convert_listlike_datetimes,
   1028     utc=utc,
   1029     unit=unit,
   1030     dayfirst=dayfirst,
   1031     yearfirst=yearfirst,
   1032     errors=errors,
   1033     exact=exact,
   1034 )
   1035 # pylint: disable-next=used-before-assignment
   1036 result: Timestamp | NaTType | Series | Index

KeyboardInterrupt:

# Print cell
print('check_date_time_processing',
([combined_data['publish_time'].iloc[random_index],combined_data['publ
ish_date'].iloc[random_index],sorted(list(combined_data["publish_weekd
ay"].value_counts()))]))

check_date_time_processing [datetime.time(18, 35), datetime.date(2018,
5, 9), [18641, 18931, 22146, 22501, 22523, 23573, 26252]]
```

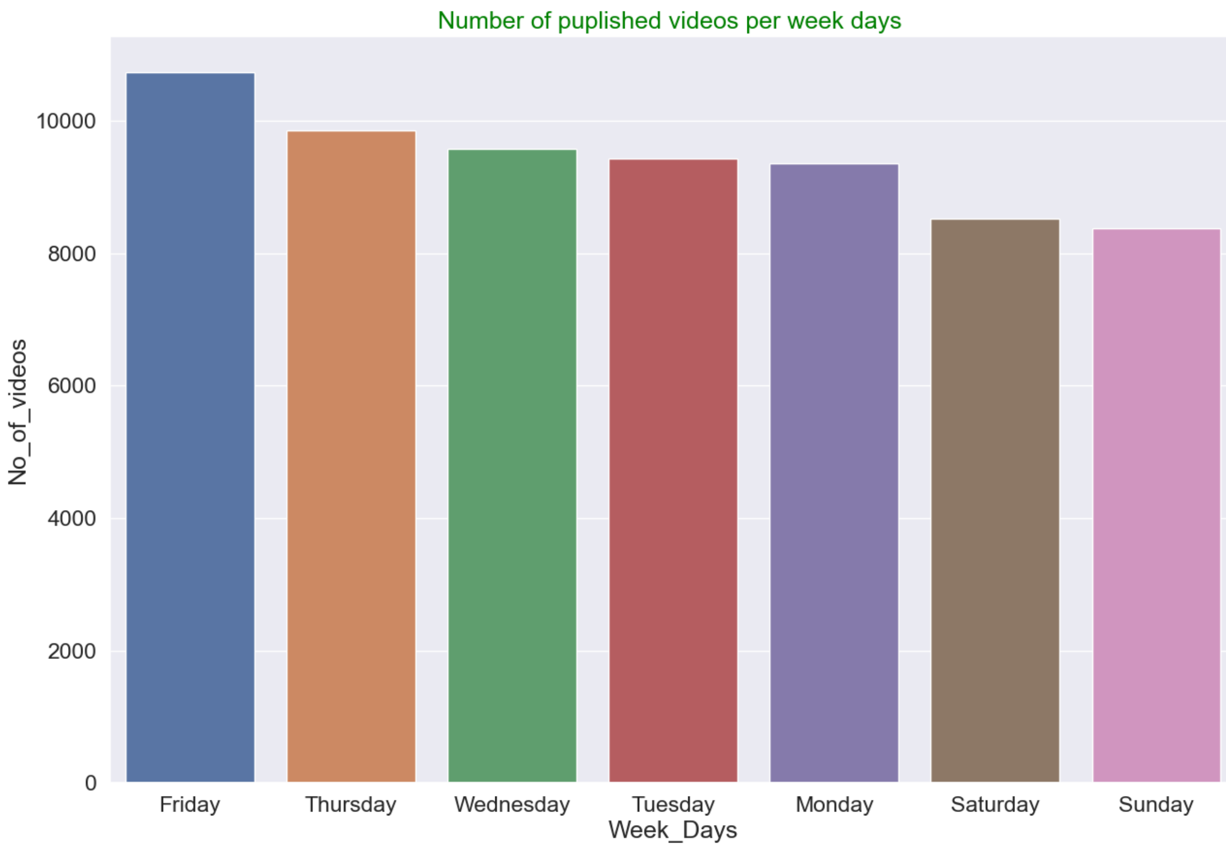
d. Number of videos per weekday

Calculate the number of videos published per day of the week. Which day of the week do people publish most videos? Make a visualization demonstrating the result.

```
# plot here

##Creating dataframe after deleting videos which stay trending for
more than one day according to the Video ID
dfx=combined_data.reset_index(level=0)
[['video_id','publish_weekday']].drop_duplicates(subset =
['video_id'], keep = 'last')

##Mapping the day number : day name
dayOfWeek={1:'Monday', 2:'Tuesday', 3:'Wednesday', 4:'Thursday',
5:'Friday', 6:'Saturday', 7:'Sunday'}
dfx['publish_weekday'] = dfx['publish_weekday'].map(dayOfWeek)

videos_weekday =
```

```python
dfx['publish_weekday'].value_counts().to_frame().reset_index().rename(
columns={"publish_weekday": "Week_Days", "count": "No_of_videos"})

##Calculating and plotting
fig, ax = pyplot.subplots(figsize=(15, 10)),sns.set(font_scale=1.5)
sns.barplot(x="Week_Days", y="No_of_videos",
data=videos_weekday,ax=ax),plt.title('Number of puplished videos per
week days ',color='Green')
# Plots will be manually graded
```

```
(<Axes: title={'center': 'Number of puplished videos per week days '},
xlabel='Week_Days', ylabel='No_of_videos'>,
 Text(0.5, 1.0, 'Number of puplished videos per week days '))
```



1.2.6: Dropping irrelevant non numeric columns

Drop all the columns that are non-numeric as we have processed them and stored the information captured in them in the dataset as numbers.

Note that a few key columns are non-numeric but should be kept in the dataframe:

comments_disabled, ratings_disabled, video_error_or_removed, country

Also drop original views, like, comments and dislikes as you have processed them as logs and stored them as separate feature.

```
combined_data.columns
```

```
Index(['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', 'country', 'category',
       'likes_log', 'views_log', 'dislikes_log', 'comment_log',
'num_tags',
       'desc_len', 'len_title', 'overall_len_title_desc',
'publish_date',
       'publish_weekday'],
      dtype='object')
```

```
combined_data.drop(['trending_date', 'title', 'channel_title',
'category_id',
                    'publish_time', 'tags', 'views', 'likes',
'dislikes', 'comment_count',
                    'thumbnail_link', 'description', 'publish_date'],
axis = 1,inplace = True)
```

```
combined_data
```

```
             comments_disabled  ratings_disabled
video_error_or_removed  \
video_id

kzwfHumJyYc              False             False
False
zUZ1z7FwLc8               True             False
False
10L1hZ9qa58              False             False
False
N1vE8iiEg64              False             False
False
kJzGH0PVQHQ              False             False
False
...                        ...               ...                      .
..
1PhPYr_9zRY              False             False
False
BZt0qjTWNhw              False             False
False
D6Oy4LfoqsU              False             False
False
oV0zkMe1K8s              False             False
False
ooyjaVdt-jA              False             False
False
```

```
            country           category  likes_log  views_log  
dislikes_log  \
video_id                                                       

kzwfHumJyYc      IN  Film & Animation  10.433145  13.907477   
6.683361   
zUZ1z7FwLc8      IN   News & Politics   6.601230  13.288051   
6.807935   
10L1hZ9qa58      IN     Entertainment   7.606885  13.068939   
5.497168   
N1vE8iiEg64      IN            Comedy  11.161295  14.032782   
7.393263   
kJzGH0PVQHQ      IN     Entertainment   6.200509  13.047674   
5.683580   
...             ...               ...        ...        ...   
...   
1PhPYr_9zRY      US    People & Blogs  13.378383  15.926830   
8.307213   
BZt0qjTWNhw      US    Pets & Animals  10.549569  14.337638   
7.234177   
D6Oy4LfoqsU      US     Entertainment  10.780393  13.879848   
6.940222   
oV0zkMe1K8s      US  Film & Animation  12.170228  15.549078   
7.954021   
ooyjaVdt-jA      US            Gaming  12.785715  16.148248   
12.268939   

             comment_log  num_tags  desc_len  len_title  \
video_id                                                  
kzwfHumJyYc     6.783325        15       920         81   
zUZ1z7FwLc8     0.000000        19      2232         58   
10L1hZ9qa58     5.010635        14       482         58   
N1vE8iiEg64     7.895436        20       263         30   
kJzGH0PVQHQ     4.204693        11       753         88   
...                  ...       ...       ...        ...   
1PhPYr_9zRY    11.044696        27       926         52   
BZt0qjTWNhw     7.885329        14        46         28   
D6Oy4LfoqsU     8.292298        24       775         84   
oV0zkMe1K8s     9.479527        22      3268         35   
ooyjaVdt-jA    11.883081         4       709         64   

             overall_len_title_desc  publish_weekday  
video_id                                              
kzwfHumJyYc                    1001                7  
zUZ1z7FwLc8                    2290                1  
10L1hZ9qa58                     540                7  
N1vE8iiEg64                     293                7  
kJzGH0PVQHQ                     841                1  
...                             ...              ...  
1PhPYr_9zRY                     978                5  
```

```
BZt0qjTWNhw                        74                 5
D6Oy4LfoqsU                       859                 5
oV0zkMe1K8s                      3303                 4
ooyjaVdt-jA                       773                 4

[154567 rows x 14 columns]
```

1.2.7: Convert categorical features in the dataset into one hot vectors.

There are three categorical features remaining in the dataset, identify them and convert them into one hot vectors. Be sure that when you one hot encode, the original column is replaced.

```
combined_data.publish_weekday =
combined_data.publish_weekday.astype('category')
combined_data.country = combined_data.country.astype('category')
combined_data.category = combined_data.category.astype('category')
combined_data= pd.get_dummies(combined_data)
# Hint: Use pd.get_dummies()range.

combined_data

             comments_disabled  ratings_disabled
video_error_or_removed  \
video_id

kzwfHumJyYc              False             False
False
zUZ1z7FwLc8               True             False
False
10L1hZ9qa58              False             False
False
N1vE8iiEg64              False             False
False
kJzGH0PVQHQ              False             False
False
...                        ...               ...                     .
..
1PhPYr_9zRY              False             False
False
BZt0qjTWNhw              False             False
False
D6Oy4LfoqsU              False             False
False
oV0zkMe1K8s              False             False
False
ooyjaVdt-jA              False             False
False

             likes_log  views_log  dislikes_log  comment_log  num_tags
\
```

```
video_id

kzwfHumJyYc  10.433145  13.907477      6.683361     6.783325         15

zUZ1z7FwLc8   6.601230  13.288051      6.807935     0.000000         19

10L1hZ9qa58   7.606885  13.068939      5.497168     5.010635         14

N1vE8iiEg64  11.161295  14.032782      7.393263     7.895436         20

kJzGH0PVQHQ   6.200509  13.047674      5.683580     4.204693         11

...                ...        ...           ...          ...        ...

1PhPYr_9zRY  13.378383  15.926830      8.307213    11.044696         27

BZt0qjTWNhw  10.549569  14.337638      7.234177     7.885329         14

D6Oy4LfoqsU  10.780393  13.879848      6.940222     8.292298         24

oV0zkMe1K8s  12.170228  15.549078      7.954021     9.479527         22

ooyjaVdt-jA  12.785715  16.148248     12.268939    11.883081          4


             desc_len  len_title  ...  category_Sports
category_Trailers  \
video_id                          ...
kzwfHumJyYc       920         81  ...            False
False
zUZ1z7FwLc8      2232         58  ...            False
False
10L1hZ9qa58       482         58  ...            False
False
N1vE8iiEg64       263         30  ...            False
False
kJzGH0PVQHQ       753         88  ...            False
False
...               ...        ...  ...              ...
...
1PhPYr_9zRY       926         52  ...            False
False
BZt0qjTWNhw        46         28  ...            False
False
D6Oy4LfoqsU       775         84  ...            False
False
oV0zkMe1K8s      3268         35  ...            False
False
ooyjaVdt-jA       709         64  ...            False
False
```

```
             category_Travel & Events  publish_weekday_1
publish_weekday_2  \
video_id

kzwfHumJyYc                     False              False
False
zUZ1z7FwLc8                     False               True
False
10L1hZ9qa58                     False              False
False
N1vE8iiEg64                     False              False
False
kJzGH0PVQHQ                     False               True
False
...                               ...                ...
...
1PhPYr_9zRY                     False              False
False
BZt0qjTWNhw                     False              False
False
D6Oy4LfoqsU                     False              False
False
oV0zkMe1K8s                     False              False
False
ooyjaVdt-jA                     False              False
False

             publish_weekday_3  publish_weekday_4
publish_weekday_5  \
video_id

kzwfHumJyYc              False              False              False

zUZ1z7FwLc8              False              False              False

10L1hZ9qa58              False              False              False

N1vE8iiEg64              False              False              False

kJzGH0PVQHQ              False              False              False

...                        ...                ...                ...

1PhPYr_9zRY              False              False               True

BZt0qjTWNhw              False              False               True

D6Oy4LfoqsU              False              False               True
```

```
oV0zkMe1K8s                False               True               False
ooyjaVdt-jA                False               True               False

             publish_weekday_6  publish_weekday_7
video_id
kzwfHumJyYc              False               True
zUZ1z7FwLc8              False              False
10L1hZ9qa58              False               True
N1vE8iiEg64              False               True
kJzGH0PVQHQ              False              False
...                        ...                ...
1PhPYr_9zRY              False              False
BZt0qjTWNhw              False              False
D6Oy4LfoqsU              False              False
oV0zkMe1K8s              False              False
ooyjaVdt-jA              False              False

[154567 rows x 40 columns]
```

```
# Print cell.
print('check_final_df',(combined_data.shape))
```

```
check_final_df (154567, 40)
```

Let's write out the modified data we created to a file so that we can reuse it in Section 2.

```
combined_data_sec_2 = combined_data.copy()
combined_data_sec_2.rename(columns = {'views_log':'label'}, inplace =
True)
combined_data_sec_2.to_csv('combined_data.csv')
```

1.2.8: Split into x and y

Split the data into features and label, in this case the features are anything but views_log and the label is views_log.

```
combined_data.columns
```

```
Index(['comments_disabled', 'ratings_disabled',
'video_error_or_removed',
       'likes_log', 'views_log', 'dislikes_log', 'comment_log',
'num_tags',
       'desc_len', 'len_title', 'overall_len_title_desc',
'country_CA',
       'country_FR', 'country_IN', 'country_US', 'category_Autos &
Vehicles',
       'category_Comedy', 'category_Education',
'category_Entertainment',
```

```
       'category_Film & Animation', 'category_Gaming',
       'category_Howto & Style', 'category_Movies', 'category_Music',
       'category_News & Politics', 'category_Nonprofits & Activism',
       'category_People & Blogs', 'category_Pets & Animals',
       'category_Science & Technology', 'category_Shows',
'category_Sports',
       'category_Trailers', 'category_Travel & Events',
'publish_weekday_1',
       'publish_weekday_2', 'publish_weekday_3', 'publish_weekday_4',
       'publish_weekday_5', 'publish_weekday_6', 'publish_weekday_7'],
      dtype='object')
```

```
combined_data=pd.read_csv('combined_data.csv').set_index('video_id')
label = combined_data['label']
features = combined_data.drop(['label'],axis=1)

combined_data
```

```
             comments_disabled  ratings_disabled
video_error_or_removed  \
video_id

kzwfHumJyYc              False             False
False
zUZ1z7FwLc8               True             False
False
10L1hZ9qa58              False             False
False
N1vE8iiEg64              False             False
False
kJzGH0PVQHQ              False             False
False
...                        ...               ...                    .
..
1PhPYr_9zRY              False             False
False
BZt0qjTWNhw              False             False
False
D6Oy4LfoqsU              False             False
False
oV0zkMe1K8s              False             False
False
ooyjaVdt-jA              False             False
False

             likes_log      label  dislikes_log  comment_log  num_tags
\
video_id

kzwfHumJyYc  10.433145  13.907477      6.683361     6.783325        15
```

```
zUZ1z7FwLc8   6.601230  13.288051      6.807935     0.000000        19
10L1hZ9qa58   7.606885  13.068939      5.497168     5.010635        14
N1vE8iiEg64  11.161295  14.032782      7.393263     7.895436        20
kJzGH0PVQHQ   6.200509  13.047674      5.683580     4.204693        11
...                ...        ...           ...          ...       ...
1PhPYr_9zRY  13.378383  15.926830      8.307213    11.044696        27
BZt0qjTWNhw  10.549569  14.337638      7.234177     7.885329        14
D6Oy4LfoqsU  10.780393  13.879848      6.940222     8.292298        24
oV0zkMe1K8s  12.170228  15.549078      7.954021     9.479527        22
ooyjaVdt-jA  12.785715  16.148248     12.268939    11.883081         4

             desc_len  len_title  ...  category_Sports  
category_Trailers  \
video_id                          ...                    
kzwfHumJyYc       920         81  ...            False  
False   
zUZ1z7FwLc8      2232         58  ...            False  
False   
10L1hZ9qa58       482         58  ...            False  
False   
N1vE8iiEg64       263         30  ...            False  
False   
kJzGH0PVQHQ       753         88  ...            False  
False   
...               ...        ...  ...              ...  
...   
1PhPYr_9zRY       926         52  ...            False  
False   
BZt0qjTWNhw        46         28  ...            False  
False   
D6Oy4LfoqsU       775         84  ...            False  
False   
oV0zkMe1K8s      3268         35  ...            False  
False   
ooyjaVdt-jA       709         64  ...            False  
False   

             category_Travel & Events  publish_weekday_1  
publish_weekday_2  \
```

```
video_id

kzwfHumJyYc                          False             False
False
zUZ1z7FwLc8                          False              True
False
10L1hZ9qa58                          False             False
False
N1vE8iiEg64                          False             False
False
kJzGH0PVQHQ                          False              True
False
...                                    ...               ...
...
1PhPYr_9zRY                          False             False
False
BZt0qjTWNhw                          False             False
False
D6Oy4LfoqsU                          False             False
False
oV0zkMe1K8s                          False             False
False
ooyjaVdt-jA                          False             False
False

             publish_weekday_3  publish_weekday_4
publish_weekday_5  \
video_id

kzwfHumJyYc              False              False              False

zUZ1z7FwLc8              False              False              False

10L1hZ9qa58              False              False              False

N1vE8iiEg64              False              False              False

kJzGH0PVQHQ              False              False              False

...                        ...                ...                ...

1PhPYr_9zRY              False              False               True

BZt0qjTWNhw              False              False               True

D6Oy4LfoqsU              False              False               True

oV0zkMe1K8s              False               True              False

ooyjaVdt-jA              False               True              False
```

```
              publish_weekday_6  publish_weekday_7
video_id
kzwfHumJyYc               False               True
zUZ1z7FwLc8               False              False
10L1hZ9qa58               False               True
N1vE8iiEg64               False               True
kJzGH0PVQHQ               False              False
...                         ...                ...
1PhPYr_9zRY               False              False
BZt0qjTWNhw               False              False
D6Oy4LfoqsU               False              False
oV0zkMe1K8s               False              False
ooyjaVdt-jA               False              False

[154567 rows x 40 columns]
```

```python
# print cell
print('check_x_y_split',([features.shape, label.describe()]))
```

```
check_x_y_split [(154567, 39), count    154567.000000
mean         12.552680
std           1.816821
min           5.411646
25%          11.469496
50%          12.655328
75%          13.761413
max          19.232552
Name: label, dtype: float64]
```

1.3 : Machine Learning using sklearn

Scikit-learn (formerly scikits.learn and also known as sklearn) is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

You can find the documentation here

Now we will train some machine learning models using sklearn to predict views, rather than predicting views directly we will predict views_log to avoid numerical instability issues

1.3.1 : Split data into train and test

Use sklearn's train_test_split library and split data into train and test sets, the split should be 80-20 meaning 80% for training and rest for testing.

```python
from sklearn.model_selection import train_test_split
# code here
```

```
x_train, x_test, y_train, y_test = train_test_split(features.values,
label.values, test_size=0.2, random_state=0)
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.transform(x_test)

# print cell.
print('check_data_split',
[x_train.shape,x_test.shape,y_train.shape,y_test.shape])
```

```
check_data_split [(123653, 39), (30914, 39), (123653,), (30914,)]
```

1.3.2: Train Machine Learning Models.

5.3.2.1 Linear Regression

In this step we will train a linear regression model using sklearn. Train using the training data and then make predictions of test, report the mean squared error obtained on both train and test sets.

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
# your code here
from sklearn.metrics import accuracy_score, mean_absolute_error,
mean_squared_error

# Create linear regression object
lin_R = LinearRegression()

# Train the model using the training sets
lin_R.fit(x_train, y_train)

# Make predictions

y_predict_test = lin_R.predict(x_test)
y_predict_train= lin_R.predict(x_train)

# The coefficients
#print('Coefficients: \n', lin_R.coef_)

print('r Squared: %.2f'
      % lin_R.score(x_test, y_test))

print('mse_value of Test=',mean_squared_error(y_test, y_predict_test))
print('mse_value of Train=',mean_squared_error(y_train,
y_predict_train))
mse_test= mean_squared_error(y_test, y_predict_test)
```

```
r Squared: 0.87
mse_value of Test= 0.4461092154382567
mse_value of Train= 0.4454481116256684
```

```
print('check_lr', (np.sqrt(mean_squared_error(y_test,
y_predict_test))))
```

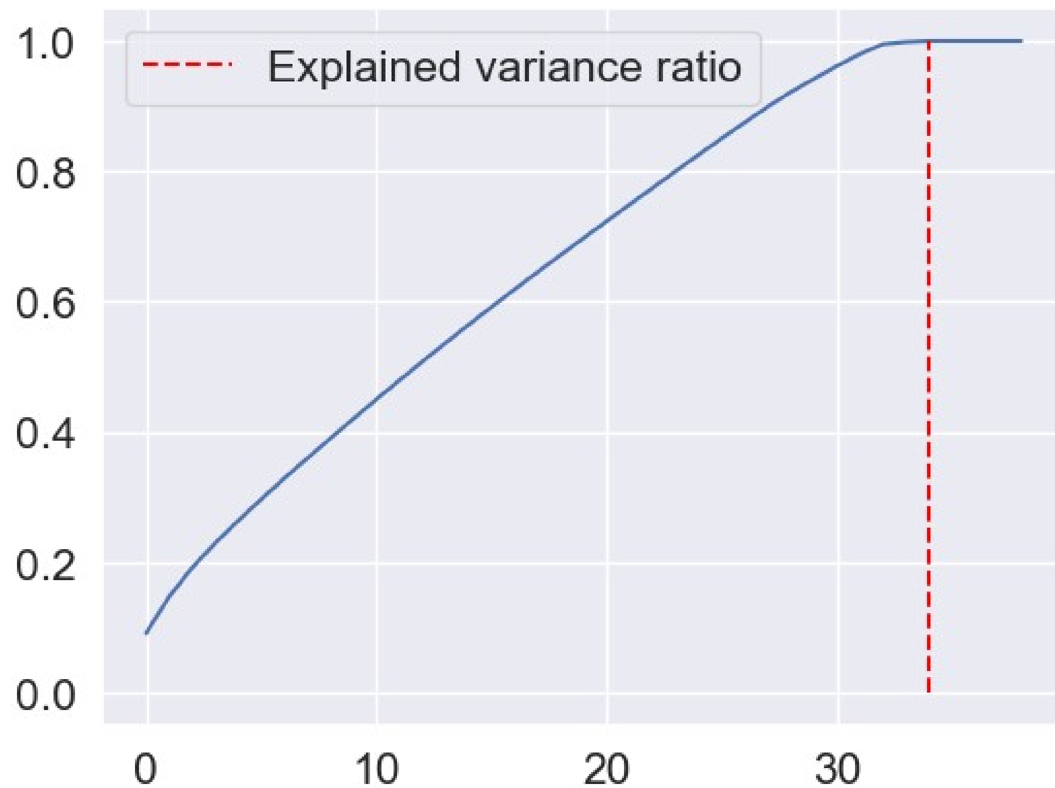
```
check_lr 0.6679140778859634
```

1.3.2.2 Dimensionality reduction with PCA

Step 1: Fitting PCA and explained_variance_ratiio

Use Principal component analysis to reduce number of dimensions of the dataset, as a first step fit a pca model on your train set and then plot the explained_variance_ratio against the number of components to decide the number of components you should keep.

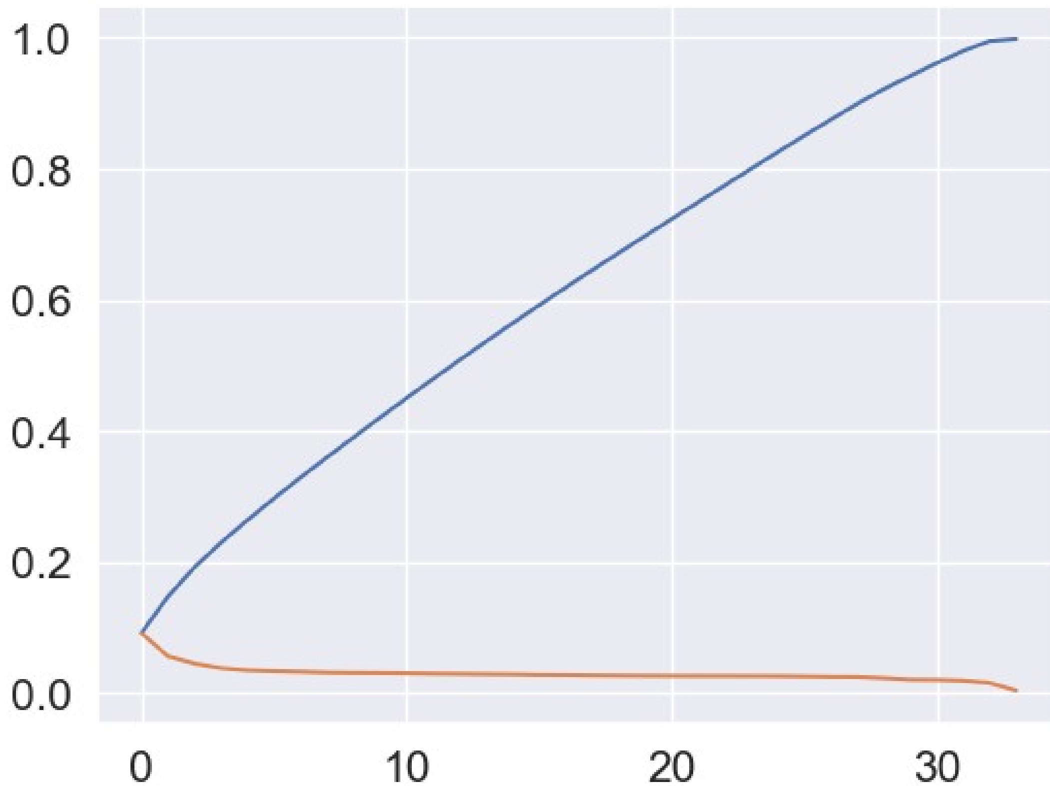
```
import numpy as np
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
# code here

pca = PCA()
sc = StandardScaler()
x_train_std=pca.fit_transform(x_train)
np.set_printoptions(suppress=True)
var_vs_pca = np.cumsum(pca.explained_variance_ratio_)
# print([(i,x) for i, x in enumerate(var_vs_pca)])
# plotting the explained_variance_ratio against the number of
components
plt.plot(var_vs_pca)
plt.vlines(34,0, 1, linestyles="dashed", label = "Explained variance
ratio", color = "red")
plt.legend()
plt.show()
```




```
#It is strating faltten with number of component = 34 (99% of
variance)
pca = PCA(34)
x_train_std=pca.fit_transform(x_train)
np.set_printoptions(suppress=True)
var_vs_pca = np.cumsum(pca.explained_variance_ratio_)
print([(i,x) for i, x in enumerate(var_vs_pca)])
# plotting the explained_variance_ratio against the number of
components
plt.plot(var_vs_pca)
plt.plot(pd.Series(pca.explained_variance_ratio_))
```

```
[(0, 0.09126409194749638), (1, 0.14699700996335624), (2,
0.1917013032821802), (3, 0.2293215438976535), (4,
0.26367107146149255), (5, 0.29668848837140577), (6,
0.3287273605024713), (7, 0.3595029717065902), (8,
0.38990749640497246), (9, 0.4201345651023148), (10,
0.44991901701329623), (11, 0.47910962569042215), (12,
0.5079265573924829), (13, 0.536242037635134), (14,
0.5642886550058703), (15, 0.591574771601573), (16,
0.6186132289523671), (17, 0.6451924007036212), (18,
0.6714127829279525), (19, 0.6973720528077321), (20,
0.7231677030753347), (21, 0.7488305581881112), (22,
0.7744831100241995), (23, 0.8000117398174672), (24,
0.8254065659256438), (25, 0.8503595619447274), (26,
```

```
0.8747817722557463), (27, 0.8990337947648571), (28,
0.9214836728973536), (29, 0.9415691727976258), (30,
0.9613202130779526), (31, 0.9798506602673468), (32,
0.9950221994162957), (33, 0.9983391613056003)]

[<matplotlib.lines.Line2D at 0x15055cf40>]
```



Step 2: Deciding number of components to keep

Use the plot to decide the number of components to keep, choose a number that explains atleast 95% of variance in the dataset. Then fit and transform your pca on training set using the number of components you decided.

Remember that your pca should be trained on the training set (and transformed here) but only transformed on the test set.

```
# code here
pca = PCA(n_components=34)
x_train_Trans=pca.fit_transform(x_train)
x_test_Trans=pca.transform(x_test)

# Print cell.
print('check_pca', (x_train_Trans[:50,:]))
```

```
check_pca [[ 1.23345231  1.34616323 -0.68053702 ...  0.1906452
0.33334627
   0.35944868]
 [ 1.71575638 -0.08051821 -1.91374054 ... -0.07776555 -0.70968815
  -0.77570453]
 [ 1.36113905  1.28973594 -1.81397901 ...  0.34260858 -0.17670947
  -0.25549642]
 ...
 [-0.41974953 -0.1341514   1.66671784 ...  1.39370159 -1.40067944
  -0.17567999]
 [ 1.52756597 -0.37853089 -0.97792695 ... -0.13571969  0.88779337
   0.66685668]
 [ 1.08848851  0.14520349  0.1638943  ...  0.41925738  0.31297909
  -0.00631595]]
```

To record the results

```
from tabulate import tabulate
from functools import partial
results = {"Algorithm": [], "Train R2 score": [], "Test R2 Score": [],
           "Train MSE": [], "Test MSE": [], "Comments": []}


def update_results(model, x_train = x_train_Trans, y_train = y_train,
x_test=  x_test_Trans, y_test = y_test, comment = None):
    global results
    try:

        y_train_pred = model.predict(x_train)
        y_test_pred = model.predict(x_test)

        train_mse = np.sqrt(mean_squared_error(y_train, y_train_pred))
        test_mse =  np.sqrt(mean_squared_error(y_test, y_test_pred))


        train_r2 = r2_score(y_train, y_train_pred)
        test_r2 = r2_score(y_test, y_test_pred)
    except Exception as err:
        print(err)


    results["Algorithm"].append(model.__class__.__name__)
    results["Train R2 score"].append(train_r2)
    results["Test R2 Score"].append(test_r2)
    results[ "Train MSE"].append(train_mse)
    results["Test MSE"].append(test_mse)
    results["Comments"].append(comment)
```

```
print(tabulate(results, headers = results.keys()))
```

1.3.2.3 Random Forest.

Step 1: Hyperparameter tuning.

Use grid search and train a random forest model on the transformed train dataset. Take a look at the sklearn `RandomForestRegressor` documentation and tune the max_depth hyperparameter using grid search. We have already tested the number of estimators hyperparameter for you. Note this section may take a while to run depending on how large your grid is.

(Hint: refer to the GridSearchCV documentation and do some reading on how the max_depth in a RF model affects the result - while theory may help guide a rough estimate of possible hyperparameters, we can cross validate values using tools like GridSearch.

Our autograder has tiered points for this question depending on your final MSE value but is fairly generous; we are not requiring that you find the **most** optimal value for this hyperparameter but rather demonstrate understanding of grid search optimization.

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import GridSearchCV
# Only tune the max depth of the trees in the RF hyperparameter.
grid = GridSearchCV(estimator=RandomForestRegressor(),
param_grid={'n_estimators':[140, 200],'max_depth':
[25,30,35,40,45]},cv=5, refit = True)
grid.fit(x_train_Trans, y_train)
grid.best_params_

# Best parameters for random forest
depth = [40]
nEstimator = [140]
```

Step 2: Fitting RF

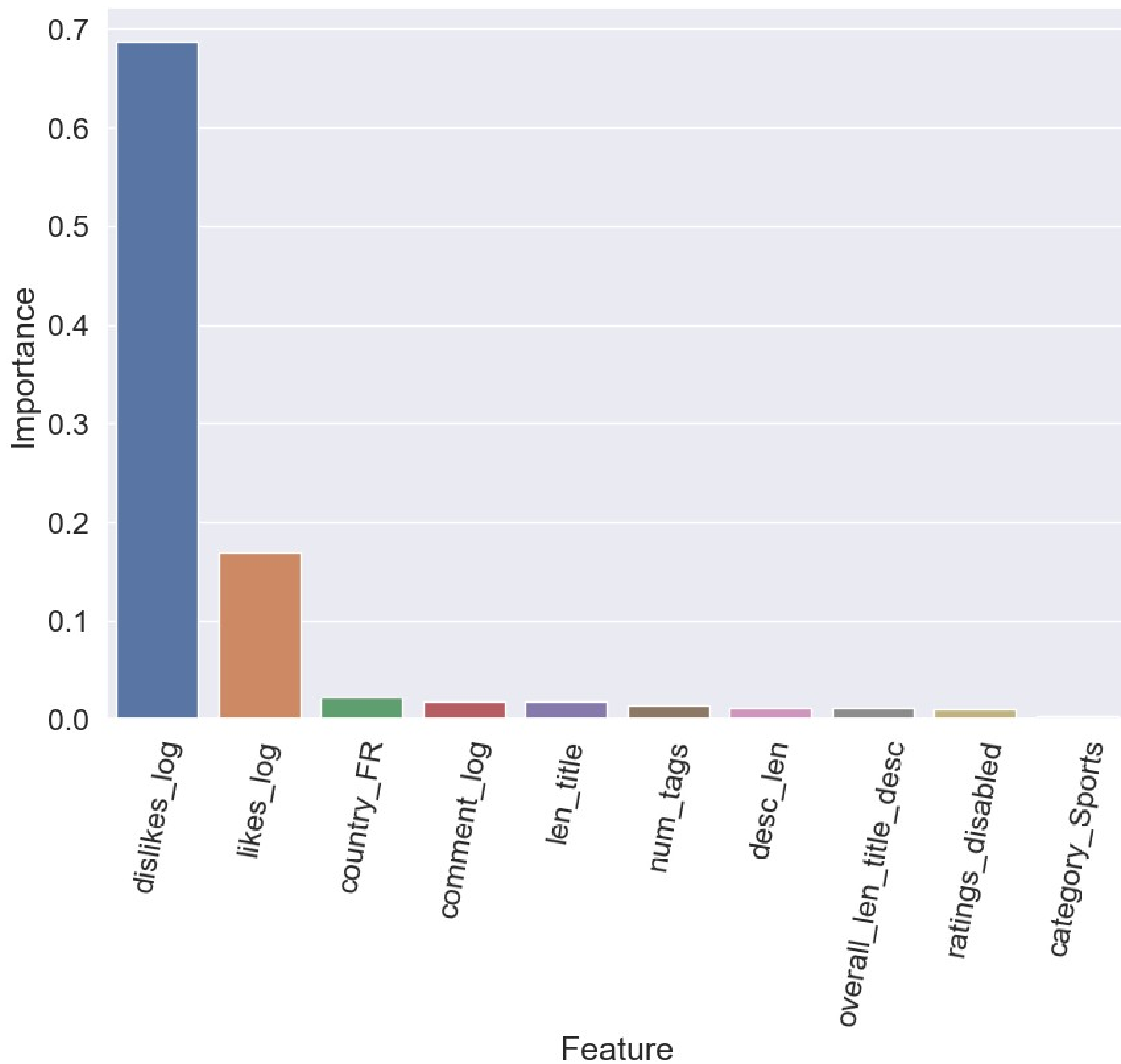
Fit the random forest on the training data using the parameters you computed above. Then make predictions on the test set, report the root mean squared error for the test set.

```
# Set n_estimators = 140
reg_RF = RandomForestRegressor(n_estimators=140, max_depth=40)
reg_RF.fit(x_train_Trans, y_train)
y_pred_RF = reg_RF.predict(x_test_Trans)

update_results(reg_RF, comment = "Trained on PCA dataset")
```

```
Algorithm                       Train R2 score    Test R2 Score    Train MSE
Test MSE  Comments
---------------------  ----------------  ---------------  -----------
----------  ----------------------
RandomForestRegressor          0.990549         0.932207     0.176539
0.473908  Trained on PCA dataset
```

Random Forest to pick the most important features

```
random_importance = RandomForestRegressor(n_estimators= 140 ,
max_depth= 40)
random_importance.fit(x_train, y_train)

RandomForestRegressor(max_depth=40, n_estimators=140)

important_features = pd.DataFrame({"Feature":
features.columns,"Importance":
random_importance.feature_importances_})
important_features = important_features.sort_values(by = "Importance",
ascending=False)

# picking top 10 variables
important_features_top_10 = important_features.head(10)
fig, ax = pyplot.subplots(figsize=(10, 7))
sns.barplot(x= important_features_top_10.Feature, y =
important_features_top_10.Importance)
plt.xticks(rotation = 80)

(array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]),
 [Text(0, 0, 'dislikes_log'),
  Text(1, 0, 'likes_log'),
  Text(2, 0, 'country_FR'),
  Text(3, 0, 'comment_log'),
  Text(4, 0, 'len_title'),
  Text(5, 0, 'num_tags'),
  Text(6, 0, 'desc_len'),
  Text(7, 0, 'overall_len_title_desc'),
  Text(8, 0, 'ratings_disabled'),
  Text(9, 0, 'category_Sports')])
```




```
# Training model on top 10 features
random_importance_top_f = RandomForestRegressor(n_estimators= 140 ,
max_depth= 40)
top_10_features =
features[important_features_top_10.Feature.to_list()]
top_f_x_train, top_f_x_test, top_f_y_train, top_f_y_test =
train_test_split(top_10_features, label.values, test_size=0.2,
random_state=0)

# top_f_train
# random_importance.fit(top_10_features, y_train)

top_f_x_test.head()
```

```
             dislikes_log  likes_log  country_FR  comment_log
len_title  \
video_id

xi8u9DcC0_Q      6.104793   9.372119       False     7.684324
68
VllA1V5M38E      7.481556   9.846123       False     8.395477
75
_Zow19iWudI      3.044522   8.207129        True     5.365976
31
48qS87hkgkQ      1.945910   6.980076        True     4.744932
77
FnAx2bLICOo      7.300473   9.409683       False     7.252054
21

             num_tags  desc_len  overall_len_title_desc
ratings_disabled  \
video_id

xi8u9DcC0_Q        32      1153                    1221
False
VllA1V5M38E        33      1264                    1339
False
_Zow19iWudI         0      2988                    3019
False
48qS87hkgkQ        21       543                     620
False
FnAx2bLICOo        11       140                     161
False

             category_Sports
video_id
xi8u9DcC0_Q            False
VllA1V5M38E            False
_Zow19iWudI            False
48qS87hkgkQ            False
FnAx2bLICOo             True

random_importance_top_f.fit(top_f_x_train, top_f_y_train)
update_results(random_importance_top_f, x_train = top_f_x_train,
y_train = top_f_y_train, x_test = top_f_x_test,
               y_test = top_f_y_test, comment = "Trained on top 10
features; extracted from random forest")

Algorithm                Train R2 score    Test R2 Score    Train MSE
Test MSE  Comments
---------------------  ---------------  ---------------  -----------
----------  ----------------------------------------------------
RandomForestRegressor         0.990549         0.932207     0.176539
0.473908  Trained on PCA dataset
```

```
RandomForestRegressor          0.990015         0.929391     0.181466
0.48365   Trained on top 10 features; extracted from random forest
```

XGBoost

```
import xgboost as xgb
from sklearn.metrics import r2_score
xgboost_regressor = xgb.XGBRegressor(n_estimators = 300)

params = {
    'n_estimators': 200,
    'objective': 'reg:squarederror',
    'max_depth': 3,
    'learning_rate': 0.1,
    'eval_metric': 'squareerror'
}


xgboost_regressor.fit(x_train_Trans, y_train)
# y_pred_xgb = xgboost_regressor.predict(x_test_Trans)
update_results(xgboost_regressor, comment="Trained on PCA dataset")

Algorithm                Train R2 score    Test R2 Score    Train MSE
Test MSE  Comments
---------------------  ----------------  ---------------  -----------
----------  -----------------------------------------------------
RandomForestRegressor          0.990549         0.932207     0.176539
0.473908  Trained on PCA dataset
RandomForestRegressor          0.990015         0.929391     0.181466
0.48365   Trained on top 10 features; extracted from random forest
XGBRegressor                   0.956509         0.925284     0.378713
0.497516  Trained on PCA dataset

xgboost_regressor = xgb.XGBRegressor(n_estimators = 300)
xgboost_regressor.fit(top_f_x_train, top_f_y_train)
update_results(xgboost_regressor, x_train = top_f_x_train, y_train =
top_f_y_train, x_test = top_f_x_test,
               y_test = top_f_y_test, comment = "XGBoost Trained on
top 10 features; extracted from random forest")

Algorithm                Train R2 score    Test R2 Score    Train MSE
Test MSE  Comments
---------------------  ----------------  ---------------  -----------
----------
----------------------------------------------------------------
RandomForestRegressor          0.990549         0.932207     0.176539
0.473908  Trained on PCA dataset
RandomForestRegressor          0.990015         0.929391     0.181466
```

```
0.48365   Trained on top 10 features; extracted from random forest
XGBRegressor                      0.956509         0.925284     0.378713
0.497516  Trained on PCA dataset
XGBRegressor                      0.939403         0.916315     0.447031
0.526532  XGBoost Trained on top 10 features; extracted from random
forest
```

```
from sklearn.ensemble import AdaBoostRegressor
ada_reg = AdaBoostRegressor(n_estimators=200)
ada_reg.fit(x_train_Trans, y_train)
update_results(ada_reg, comment="Trained on PCA dataset")
```

```
Algorithm                   Train R2 score    Test R2 Score    Train MSE
Test MSE  Comments
---------------------  ----------------  ---------------  -----------
----------
----------------------------------------------------------------
RandomForestRegressor           0.990549         0.932207     0.176539
0.473908  Trained on PCA dataset
RandomForestRegressor           0.990015         0.929391     0.181466
0.48365   Trained on top 10 features; extracted from random forest
XGBRegressor                      0.956509         0.925284     0.378713
0.497516  Trained on PCA dataset
XGBRegressor                      0.939403         0.916315     0.447031
0.526532  XGBoost Trained on top 10 features; extracted from random
forest
AdaBoostRegressor               0.752662         0.750876     0.903147
0.908465  Trained on PCA dataset
```

```
ada_reg_top_10 = AdaBoostRegressor(n_estimators=200)
ada_reg_top_10.fit(top_f_x_train, top_f_y_train)
update_results(ada_reg_top_10, x_train = top_f_x_train, y_train =
top_f_y_train, x_test = top_f_x_test,
               y_test = top_f_y_test, comment = "AdaBoost Trained on
top 10 features; extracted from random forest")
```

```
Algorithm                   Train R2 score    Test R2 Score    Train MSE
Test MSE  Comments
---------------------  ----------------  ---------------  -----------
----------
-----------------------------------------------------------------
RandomForestRegressor           0.990549         0.932207     0.176539
0.473908  Trained on PCA dataset
RandomForestRegressor           0.990015         0.929391     0.181466
0.48365   Trained on top 10 features; extracted from random forest
XGBRegressor                      0.956509         0.925284     0.378713
0.497516  Trained on PCA dataset
XGBRegressor                      0.939403         0.916315     0.447031
0.526532  XGBoost Trained on top 10 features; extracted from random
forest
```

```
AdaBoostRegressor                0.752662         0.750876     0.903147
0.908465  Trained on PCA dataset
AdaBoostRegressor                0.799606         0.799668     0.812933
0.814658  AdaBoost Trained on top 10 features; extracted from random
forest
AdaBoostRegressor
```

Grid Search CV

```python
from sklearn.tree import DecisionTreeRegressor
base_estimator = DecisionTreeRegressor()
adaboost_regressor = AdaBoostRegressor(base_estimator=base_estimator)

# GridSearchCV parameter
param_grid = {
    'n_estimators': [50, 100, 150],
    'learning_rate': [0.01, 0.1, 0.5, 1]
}

grid_search = GridSearchCV(adaboost_regressor, param_grid, cv=3,
scoring='neg_mean_squared_error', n_jobs=-1, refit = True )

# Fit the model
grid_search.fit(top_f_x_train, top_f_y_train)
update_results(grid_search, x_train = top_f_x_train, y_train =
top_f_y_train, x_test = top_f_x_test,
               y_test = top_f_y_test, comment = "Using Grid search
cv")
```

```
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
```

```
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
```

```
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
```

```
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning: `base_estimator` was renamed to
`estimator` in version 1.2 and will be removed in 1.4.
  warnings.warn(
/Users/goldyrana/Library/Python/3.9/lib/python/site-packages/sklearn/
ensemble/_base.py:156: FutureWarning:

`base_estimator` was renamed to `estimator` in version 1.2 and will be
removed in 1.4.


Algorithm                Train R2 score    Test R2 Score    Train MSE
Test MSE  Comments
---------------------  ----------------  ---------------  -----------
----------
-----------------------------------------------------------------
RandomForestRegressor          0.990549         0.932207    0.176539
0.473908  Trained on PCA dataset
RandomForestRegressor          0.990015         0.929391    0.181466
0.48365   Trained on top 10 features; extracted from random forest
XGBRegressor                   0.956509         0.925284    0.378713
0.497516  Trained on PCA dataset
XGBRegressor                   0.939403         0.916315    0.447031
0.526532  XGBoost Trained on top 10 features; extracted from random
forest
AdaBoostRegressor              0.752662         0.750876    0.903147
0.908465  Trained on PCA dataset
```

```
AdaBoostRegressor                0.799606          0.799668    0.812933
0.814658  AdaBoost Trained on top 10 features; extracted from random
forest
AdaBoostRegressor                0.999881          0.932819    0.0197926
0.471763  Using Grid search cv
GridSearchCV

print(tabulate(results, headers= results.keys()))

Algorithm                Train R2 score    Test R2 Score    Train MSE
Test MSE  Comments
---------------------  ----------------  ---------------  -----------
----------
-----------------------------------------------------------------
RandomForestRegressor            0.990549          0.932207    0.176539
0.473908  Trained on PCA dataset
RandomForestRegressor            0.990015          0.929391    0.181466
0.48365   Trained on top 10 features; extracted from random forest
XGBRegressor                     0.956509          0.925284    0.378713
0.497516  Trained on PCA dataset
XGBRegressor                     0.939403          0.916315    0.447031
0.526532  XGBoost Trained on top 10 features; extracted from random
forest
AdaBoostRegressor                0.752662          0.750876    0.903147
0.908465  Trained on PCA dataset
AdaBoostRegressor                0.799606          0.799668    0.812933
0.814658  AdaBoost Trained on top 10 features; extracted from random
forest
GridSearchCV                     0.999881          0.932819    0.0197926
0.471763  Using Grid search cv

print(grid_search.best_estimator_)

AdaBoostRegressor(base_estimator=DecisionTreeRegressor(),
learning_rate=0.1,
                  n_estimators=150)

xgb_regressor = xgb.XGBRegressor()

# Define the parameter grid for GridSearchCV
param_grid = {
    'learning_rate': [0.01, 0.1, 0.5],
    'n_estimators': [50, 100, 150],
    'max_depth': [3, 5, 7]
}

# Create GridSearchCV instance
grid_search_xgboost = GridSearchCV(xgb_regressor, param_grid, cv=3,
scoring='neg_mean_squared_error', n_jobs=-1)

# Fit the model
```

```
grid_search_xgboost.fit(top_f_x_train, top_f_y_train)
update_results(grid_search_xgboost, x_train = top_f_x_train, y_train =
top_f_y_train, x_test = top_f_x_test,
               y_test = top_f_y_test, comment = "Using Grid search cv
base estimator xgboost")
```

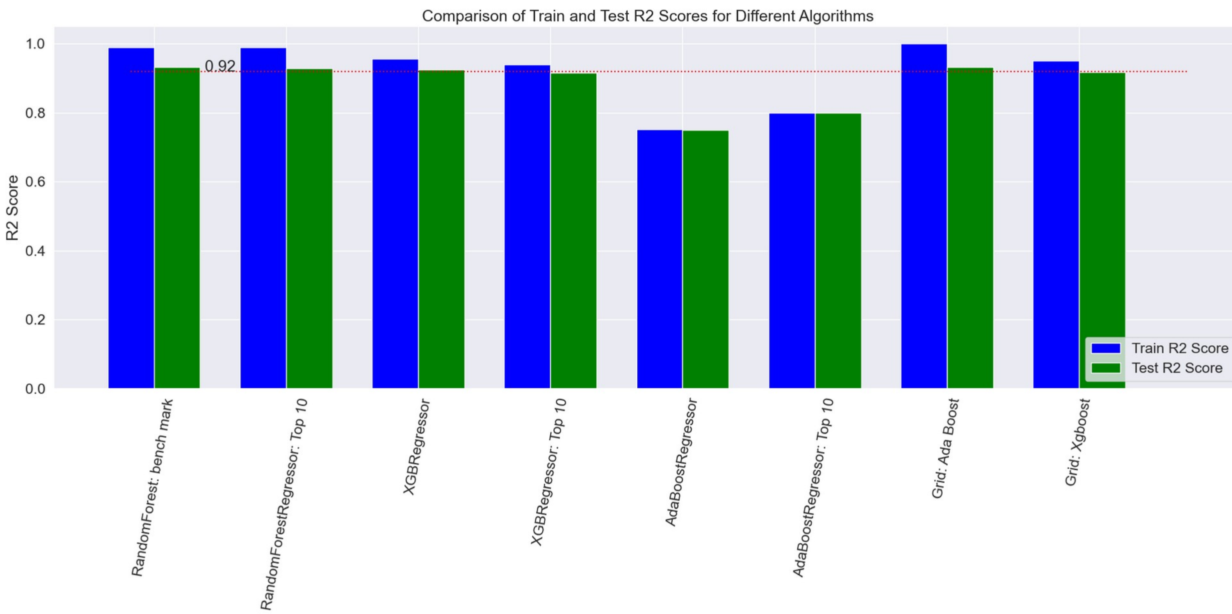
```
Algorithm                Train R2 score    Test R2 Score    Train MSE
Test MSE  Comments
---------------------  ---------------  ---------------  -----------
----------
-----------------------------------------------------------------
RandomForestRegressor          0.990549         0.932207    0.176539
0.473908  Trained on PCA dataset
RandomForestRegressor          0.990015         0.929391    0.181466
0.48365   Trained on top 10 features; extracted from random forest
XGBRegressor                   0.956509         0.925284    0.378713
0.497516  Trained on PCA dataset
XGBRegressor                   0.939403         0.916315    0.447031
0.526532  XGBoost Trained on top 10 features; extracted from random
forest
AdaBoostRegressor              0.752662         0.750876    0.903147
0.908465  Trained on PCA dataset
AdaBoostRegressor              0.799606         0.799668    0.812933
0.814658  AdaBoost Trained on top 10 features; extracted from random
forest
GridSearchCV                   0.999881         0.932819    0.0197926
0.471763  Using Grid search cv
GridSearchCV                   0.950449         0.918595    0.404238
0.519311  Using Grid search cv base estimator xgboost
```

```
results
```

```
{'Algorithm': ['RandomForestRegressor',
  'RandomForestRegressor',
  'XGBRegressor',
  'XGBRegressor',
  'AdaBoostRegressor',
  'AdaBoostRegressor',
  'GridSearchCV',
  'GridSearchCV'],
 'Train R2 score': [0.9905494448759586,
  0.990014629945052,
  0.9565094326751161,
  0.9394030928671782,
  0.7526616674455536,
  0.7996061340703311,
  0.9998812094115259,
  0.9504493990623508],
 'Test R2 Score': [0.9322067718205513,
  0.9293909339188394,
```

```
  0.9252840214094074,
  0.9163149817649782,
  0.7508760087083606,
  0.7996682814260334,
  0.9328191009498839,
  0.9185945975637091],
 'Train MSE': [0.17653930008344196,
  0.18146581467210393,
  0.3787130556223173,
  0.4470313384733453,
  0.9031470234287254,
  0.8129333722371799,
  0.019792641456668433,
  0.4042380623759653],
 'Test MSE': [0.4739077363397034,
  0.4836496465724773,
  0.4975163820266425,
  0.5265316703148832,
  0.9084654552171509,
  0.8146583241065508,
  0.4717626419031121,
  0.5193106807749848],
 'Comments': ['Trained on PCA dataset',
  'Trained on top 10 features; extracted from random forest',
  'Trained on PCA dataset',
  'XGBoost Trained on top 10 features; extracted from random forest',
  'Trained on PCA dataset',
  'AdaBoost Trained on top 10 features; extracted from random forest',
  'Using Grid search cv',
  'Using Grid search cv base estimator xgboost']}

df.to_csv("results.csv")

results_frame= pd.DataFrame(results)

# Plotting

fig, ax = plt.subplots(figsize=(20, 10))

# Grouped bar chart for Train and Test R2 scores
bar_width = 0.35
bar_positions_train = range(len(results_frame['Algorithm']))
bar_positions_test = [pos + bar_width for pos in bar_positions_train]

ax.bar(bar_positions_train, results_frame['Train R2 score'],
width=bar_width, label='Train R2 Score', color='blue')
ax.bar(bar_positions_test, results_frame['Test R2 Score'],
width=bar_width, label='Test R2 Score', color='green')

ax.set_xticks([pos + bar_width / 2 for pos in bar_positions_train])
```

```python
ax.set_xticklabels(results_frame['Algorithm'])
ax.set_ylabel('R2 Score')
ax.set_title('Comparison of Train and Test R2 Scores for Different 
Algorithms')
ax.legend(loc = "lower right")
plt.xticks(rotation = 80)
plt.hlines(y = 0.92, xmin = 0, xmax = 8, color = "red", 
linestyles="dotted")
plt.annotate("0.92", [0.56, 0.92])
plt.tight_layout()
plt.show()
```




```python
plt.figure(figsize=(10, 6))

# Scatter plot for Linear Regression
plt.scatter(x_train_Trans, y_test, color='blue', label='True Values')
plt.scatter(x_test_Trans, y_pred_RF, color='red', label=f'Linear 
Regression Predictions (MSE: {mse_linear:.2f})')

# Scatter plot for Random Forest Regression
plt.scatter(X_test, y_pred_rf, color='green', label=f'Random Forest 
Predictions (MSE: {mse_rf:.2f})')

plt.xlabel('X')
plt.ylabel('y')
plt.title('Comparison of Regression Model Predictions')
plt.legend()
plt.show()
```

Almost halfway there :)

Well done! Almost halfway there :)

Submission

Submission on the blackboard. ** PDF submission for the simalarity check and .ipynb for original submission **

Go to the "File" tab at the top left, and click "Download .ipynb". Submit under 'scalableMachinelearning.ipynb'.

You must submit your notebook to blackboard for the grading.

```
#TO PDF
%%capture
!wget -nc https://raw.githubusercontent.com/brpy/colab-pdf/master/colab_pdf.py
from colab_pdf import colab_pdf
colab_pdf('Yourname_UID.ipynb')
```