YouTube Video Statistics

BY Aniruddha Sudhir Gujar
Under Guidance of Professor Dr.Riaz Sikora

Introduction

- YouTube is an American online video sharing and social media platform owned by google. So, for the current generation it has been a great platform to earn money as YouTube pays by a policy of pay per views. YouTube can be efficient platform for advertising of products through youtubers with great reach and engagement with their audience.

- Our aim is finding Youtubers with great number of views, category related to products to market the product to targeted audience to increase their sales.

- Our team motive is to find co-relation between variables of the dataset and apply advance techniques to find insights of the data which can be used for future YouTube channel growth.

- We are using techniques like regression, classification, clustering and NLP for analysis.

Data Description

- The dataset has been downloaded from Kaggle data source.

https://www.kaggle.com/datasets/datasnaek/youtube-new?select=USvideos.csv

- There are 16 columns and 48,000 rows in each of the 10 files segregated as per as country.

- Also each country has it’s .json file which can be mapped with the category id in the dataset to get the category of the video.

Research Questions

1

Besides the correlation between likes, views, dislikes, and comment counts, is there any other independent variable which can help to predict likes accurately?

2

Are there any patterns of similar words in the title of video?

3

Can we predict the category of the video using the title of the video?

4

Does having negative controversial description, title or tag give you more reach?

Data Preprocessing

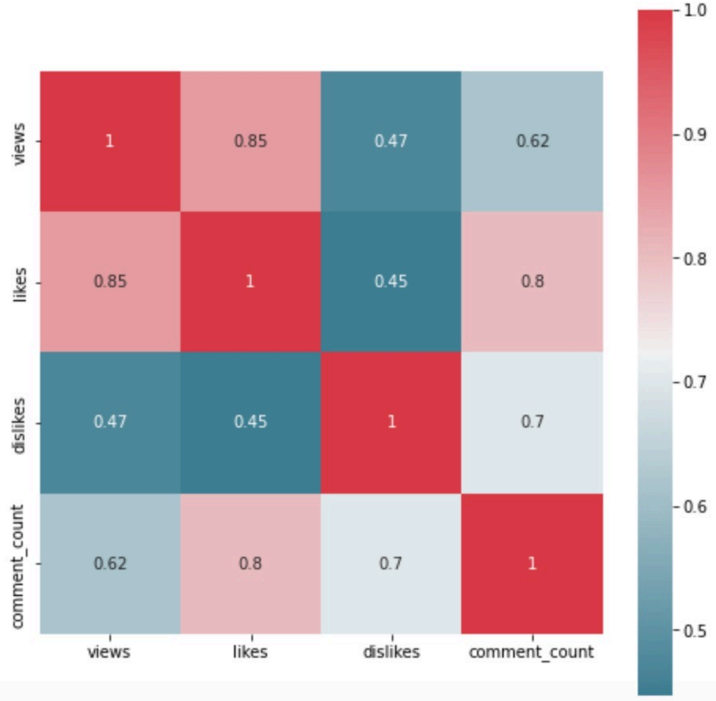
```
In [34]: print("Number of duplicate rows ",list(df.duplicated()).count(True))

Number of duplicate rows  48

In [35]: df=df.drop_duplicates()
         print(df.shape)

(40901, 16)
```

```
In [83]: def text_preprocess(data):
             #clean_data = []
             #print(data)
             data=re.sub('http[s]?://(?:[a-zA-Z]|[0-9]|[$-_@.&+]|[!*\(\), ]|(?:%[0-9a-fA-F][0-9a-fA-F]))+',"", str(data))
             data=re.sub(r'\\n+'," ",str(data))
             data=data.replace("|"," ")
             data=re.sub(r'[?|!|\'|"|#|,|-|@|(|)]', "", str(data))
             data=data.replace(": "," ").replace("-"," ").replace(" :"," ")
             data=re.sub(r' +',' ',str(data)).lower()
             #clean_data.append(data)
             return data
```

```
In [89]: text_preprocess(data)

Out[89]: '0 we want to talk about our marriage shantell ma...\n1 the trump presidency last week tonight with j...\n2 racist
superman rudy mancuso king bach & le...\n3 nickelback lyrics real or fake rhett and lin...\n4 i dare you going bald
ryan higa higatv ...\n ... \n40944 the cat who caught the laser aarons animals a...\n40945 true facts ant mutualism
[none] zefrank1 ...\n40946 i gave safiya nygaard a perfect hair makeover ...\n40947 how black panther should have e
nded black pant...\n40948 official call of duty® black ops 4 –\xa0multipla...\nname text_feature length 40901 dtype
object'
```

- There were no null values or empty rows.
- Removed the special charecters, numbers and spaces.
- Converted the whole data into lower case.
- Eliminated the duplicate rows.

Co-relation Between variables and prediction of likes



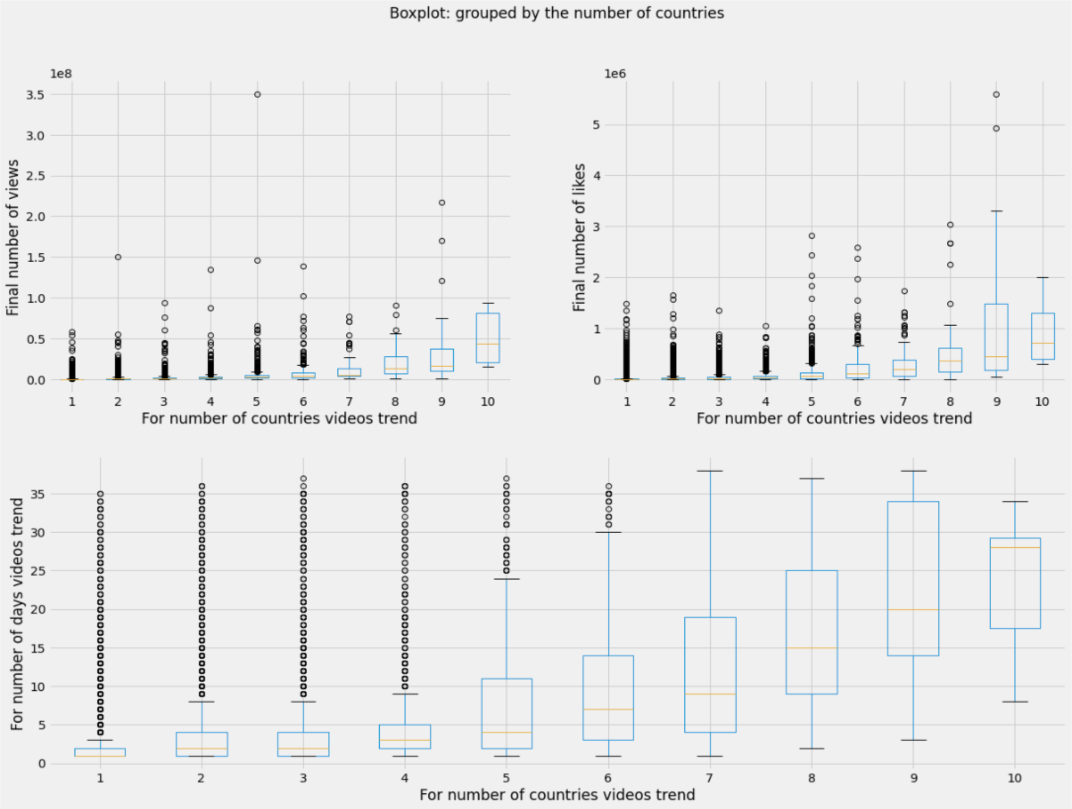
In [43]:
```
train_rows=X_train.shape[0]
data=pd.concat([X_train,X_test])

data=pd.get_dummies(data)

X_train=data[:train_rows].copy()
X_test=data[train_rows:].copy()

del data
gc.collect()

X_train.shape,X_test.shape
```

Out[43]: ((269055, 71), (75134, 71))

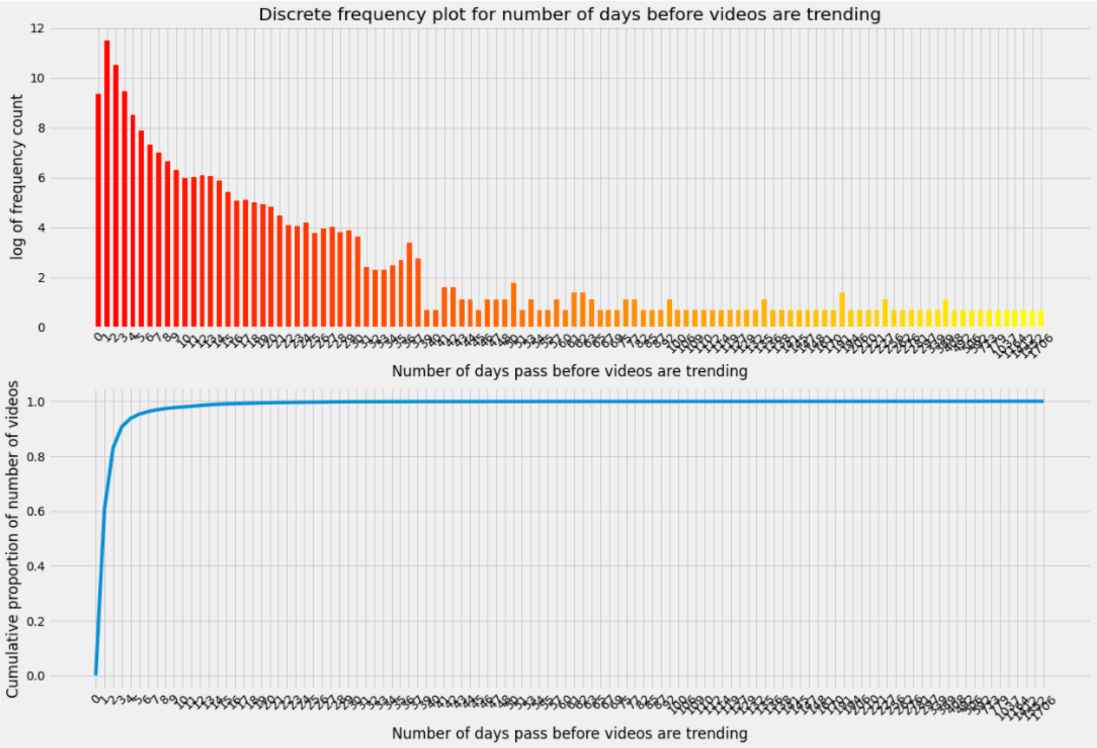
In [44]:
```
# Baseline linear regression model
from sklearn.linear_model import LinearRegression

lr=LinearRegression()
lr.fit(X_train,np.log(y_train+1))
lr.score(X_train,np.log(y_train+1))
```

Out[44]: 0.8690920374345559

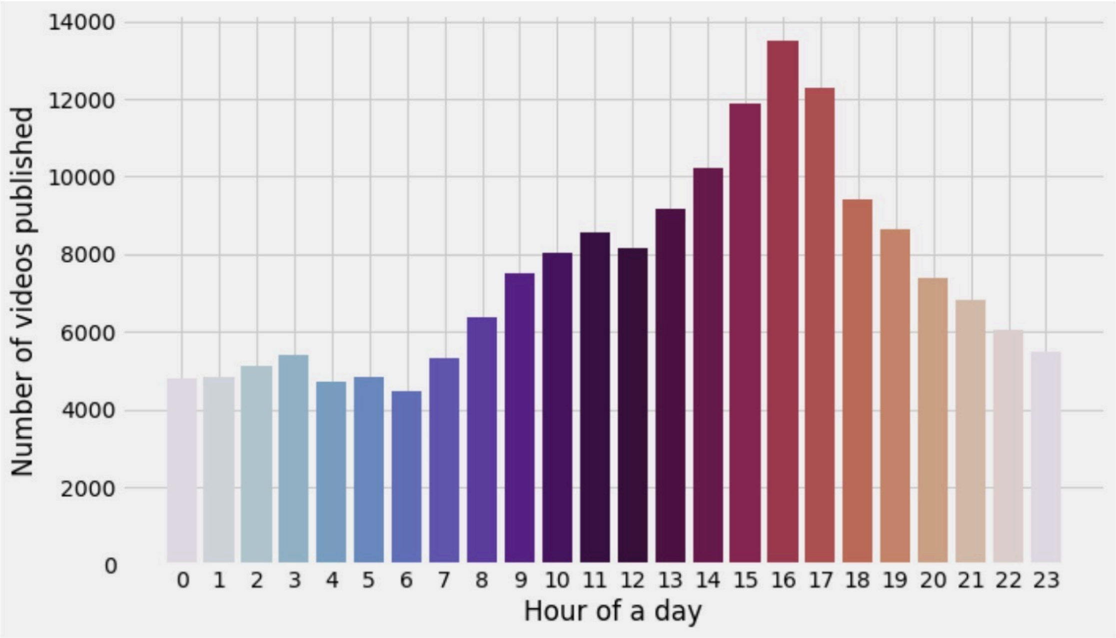
Does the trending in a greater number of countries get videos more number views and likes? Also does it help to stay in trend for a greater number of days?



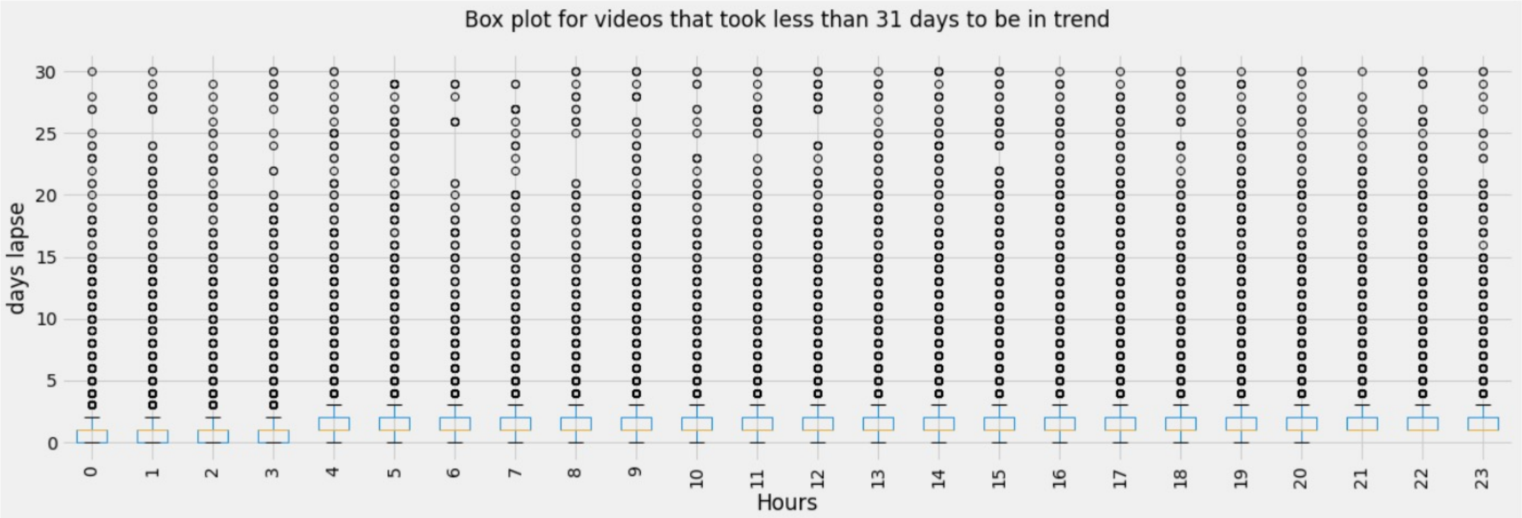
How long usually it takes for the videos to become trending?



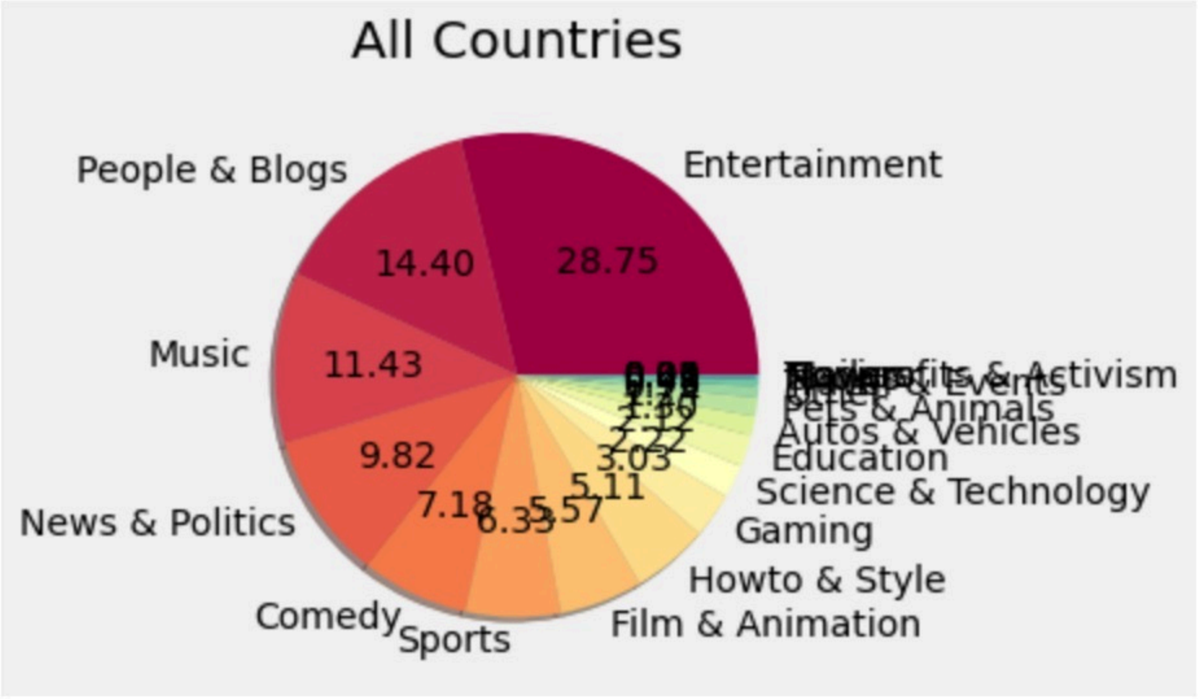
- **Do the trending videos from the data sets are published in specific time slot of 24 hrs day more than the other times?**



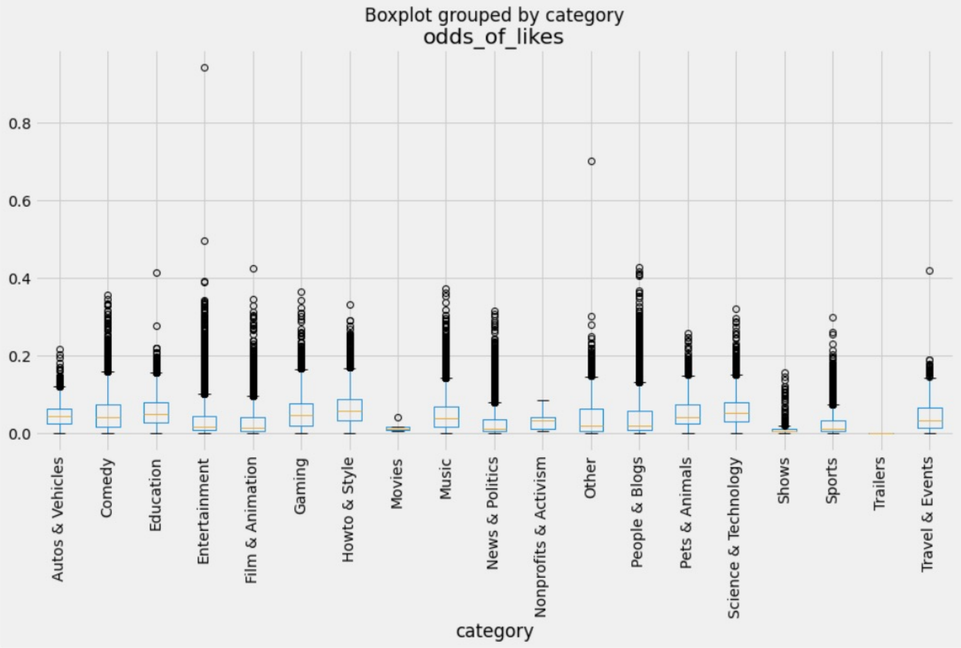
Resolving the misconception of the hours videos are published



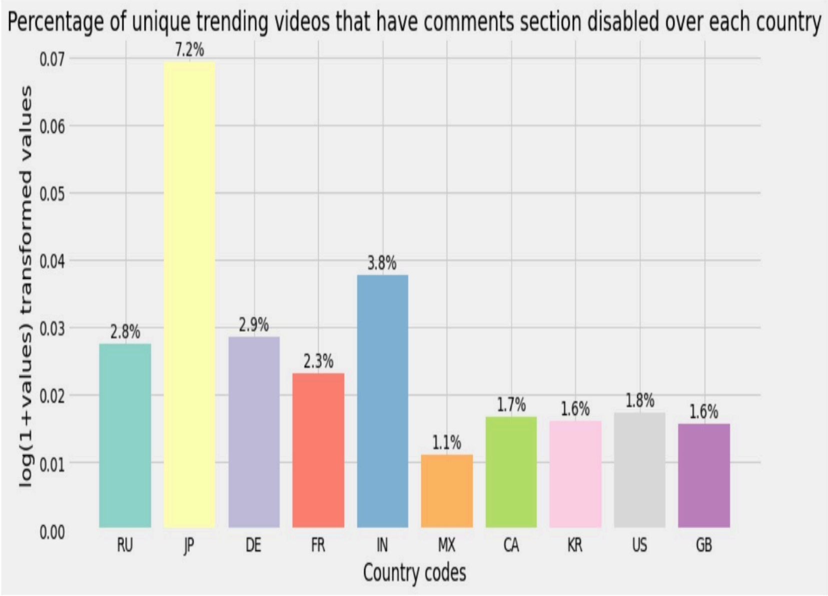
- **Proportion of videos categories trending in the countries over the entire period of given time**



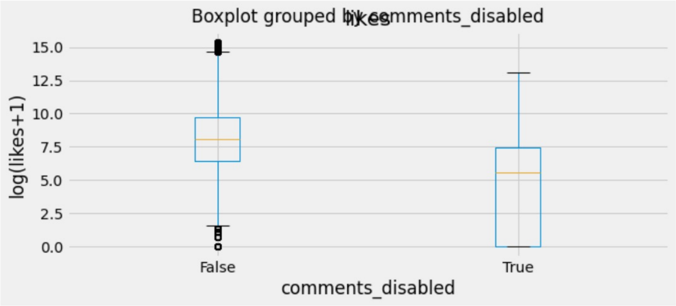
Distribution and box-plot for the proportion of likes per views for each category



Percentage of videos by the countries that have comments section disabled

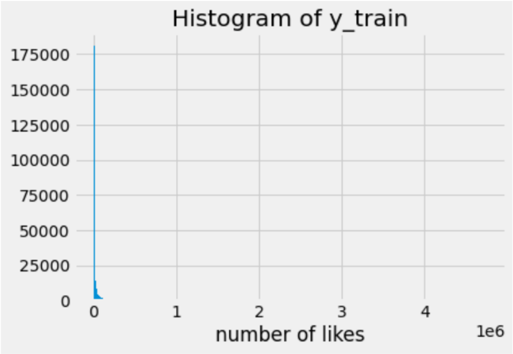


In [34]:
```python
fig,ax=plt.subplots(figsize=(10,4))
train.boxplot(column='likes',by='comments_disabled',ax=ax)
ax.set_ylabel('log(likes+1)');
```



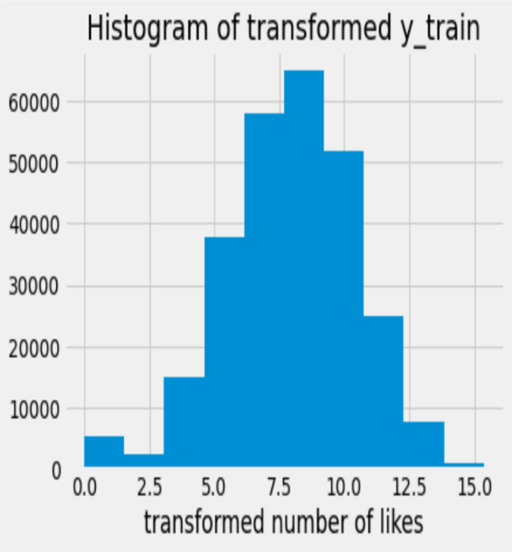
Log transforming

In [26]:
```python
fig,ax=plt.subplots()
ax.hist(y_train,bins=500)
ax.set_xlabel('number of likes')
ax.set_title('Histogram of y_train');
```



Rightly Skewed

In [27]:
```python
fig,ax=plt.subplots()
ax.hist(np.log(y_train+1))
ax.set_xlabel('transformed number of likes')
ax.set_title('Histogram of transformed y_train');
```



Linear Regression Model using entities based on exploratory analysis

```
In [42]: X_train=train[['num_countries','num_days','category','num_countries','comments_disabled','views_cat','comment_count
         X_train=train[['num_countries','num_days','category','num_countries','comments_disabled','views_cat','comment_count
```

```
In [80]: base_pred=np.repeat(np.mean(y_test),len(y_test))
         base_rms=np.sqrt(mean_squared_error(y_test,base_pred))
         rms=np.sqrt(mean_squared_error(y_test,prediction))
         print(base_rms)
         print(rms)

         1.928061562596312
         0.5873670180748854
```

```
In [81]: x1=data1.drop("likes",axis=1,inplace=False)
         x1=sm.add_constant(x1)
         y1=data1["likes"]

         model=sm.OLS(y,x).fit()
         print(model.summary())
```

```
                                 OLS Regression Results
==========================================================================================
=
Dep. Variable:                      y   R-squared (uncentered):                       0.92
7
Model:                            OLS   Adj. R-squared (uncentered):                  0.92
7
Method:                 Least Squares   F-statistic:                             4.726e+0
5
Date:                Tue, 29 Nov 2022   Prob (F-statistic):                            0.0
0
Time:                        15:33:32   Log-Likelihood:                             -35165
.
No. Observations:               39781   AIC:                                     7.038e+0
4
Df Residuals:                   39757   BIC:                                     7.058e+0
4
Df Model:                          24
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
x1             0.5490      0.004    133.339      0.000       0.541       0.557
x2            -0.1485      0.004    -38.249      0.000      -0.156      -0.141
x3             0.6401      0.004    173.808      0.000       0.633       0.647
const       2.426e-13   2.78e-13      0.872      0.383   -3.03e-13    7.88e-13
x4          9.011e-14   1.02e-13      0.884      0.377    -1.1e-13     2.9e-13
x5          -1.21e-13   1.38e-13     -0.876      0.381   -3.92e-13     1.5e-13
x6             0.0020      0.001      3.717      0.000       0.001       0.003
x7             0.0668      0.012      5.717      0.000       0.044       0.090
==============================================================================
Omnibus:                     5151.631   Durbin-Watson:                   1.981
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            34821.082
Skew:                          -0.432   Prob(JB):                         0.00
Kurtosis:                       7.501   Cond. No.                     1.01e+16
==============================================================================
```

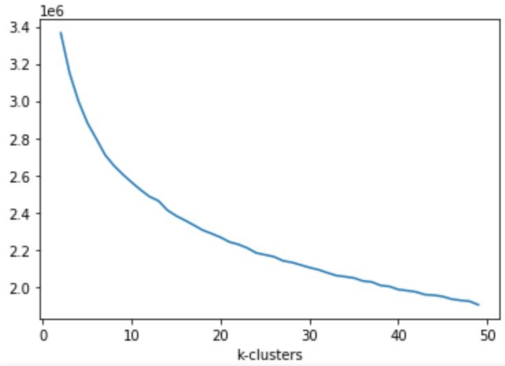
- **Using K-mean clustering for finding similar words into video title**

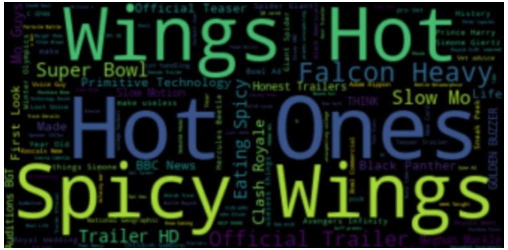
```python
def k_means(data,cluster_range):
    models=[]
    loss=[]
    for k in cluster_range:
        kmeans=KMeans(n_clusters=k,init='k-means++',n_jobs=-1).fit(data)
        models.append(kmeans)
        loss.append(kmeans.inertia_)
    plt.plot(cluster_range,loss)
    plt.xlabel('k-clusters')
    plt.ylabel('loss')
    plt.show()
    return models
```

```python
model_list=k_means(data_vect,range(2,50))
```

```python
def cluster_analysis(train_data,k):
  #For each cluster
  for i in range(0,k):
    #Extract cleaned text column
    data=train_data[train_data['labels']==i]
    list_of_words=[];
#     print("data: ",data)
    for sent in data['title']:
        #print("Title: ",sent)
        for word in sent.split():
            list_of_words.append(word)
    final_text=" ".join(list_of_words)
    #print("Cluster : ",i+1)
    #print("Number of reviews",len(data))
    #print("      Word Cloud ")
    wordcloud = WordCloud(collocations=True).generate(final_text)
    plt.figure()
    title="\nCluster : "+str(i+1)+"\n Number of Reviews: "+str(len(data))
    plt.title(title)
    plt.imshow(wordcloud, interpolation="bilinear")
    plt.axis("off")
    plt.show()
```

```python
# !pip install wordcloud
from wordcloud import WordCloud

df['labels']=model_list[18].labels_
cluster_analysis(df,20)
```





Using decision tree for classifying videos using titles

In [25]:
```
DTCtest = DecisionTreeClassifier().fit(X_train,Y_train)
dtc_predictions = DTCtest.predict(X_test)
acc_dtc = DTCtest.score(X_test, Y_test)
print('The Decision Tree Algorithm has an accuracy of', acc_dtc)
```

```
The Decision Tree Algorithm has an accuracy of 0.9877899877899878
```

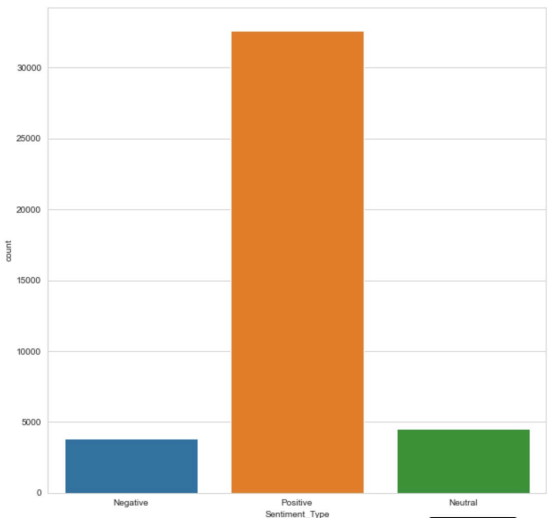
In [48]:
```
Titles = ["Hilarious cat plays with toy",
          "Best fashion looks for Spring 2018",
          "Olympics opening ceremony highlights",
          "Warriors basketball game versus the cavs",
          "CNN world news on donald trump",
          "Police Chase in Hollywood",
          "Ed Sheeran - Perfect (Official Music Video)",
          "how to do eyeshadow"
           ]
```

In [49]:
```
Titles_counts = vector.transform(Titles)
PredictDTC = DTC_Model.predict(Titles_counts)

CategoryNamesListDTC = []
for Category_ID in PredictDTC:
    MatchingCategoriesDTC = [x for x in category_dict if x["id"] == str(Category_ID)]
    if MatchingCategoriesDTC:
        CategoryNamesListDTC.append(MatchingCategoriesDTC[0]["title"])

TitleDataFrameDTC = []
for i in range(0, len(Titles)):
    TitleToCategoriesDTC = {'Title': Titles[i],  'Category': CategoryNamesListDTC[i]}
    TitleDataFrameDTC.append(TitleToCategoriesDTC)

PredictDFdtc = pd.DataFrame(PredictDTC)
TitleDFdtc = pd.DataFrame(TitleDataFrameDTC)
PreFinalDFdtc = pd.concat([PredictDFdtc, TitleDFdtc], axis=1)
PreFinalDFdtc.columns = (['Categ_ID', 'Predicted Category', 'Hypothetical Video Title'])
FinalDFdtc = PreFinalDFdtc.drop(['Categ_ID'],axis=1)
colsDTC = FinalDFdtc.columns.tolist()
colsDTC = colsDTC[-1:] + colsDTC[:-1]
FinalDFdtc= FinalDFdtc[colsDTC]
FinalDFdtc
```

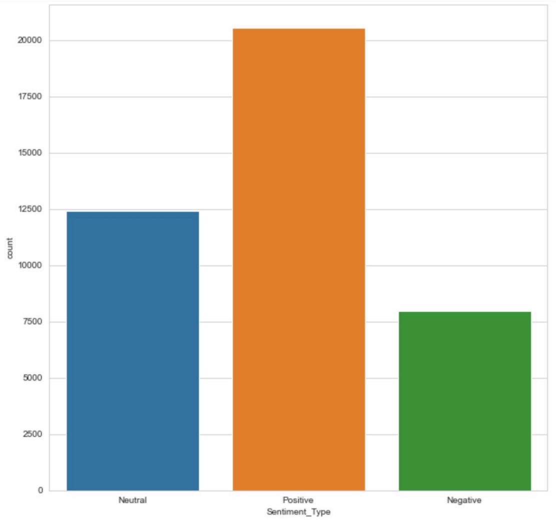
Out[49]:

	Hypothetical Video Title	Predicted Category
0	Pets & Animals	Hilarious cat plays with toy
1	People & Blogs	Best fashion looks for Spring 2018
2	Sports	Olympics opening ceremony highlights
3	Science & Technology	Warriors basketball game versus the cavs
4	Comedy	CNN world news on donald trump
5	Entertainment	Police Chase in Hollywood
6	Music	Ed Sheeran - Perfect (Official Music Video)
7	News & Politics	how to do eyeshadow

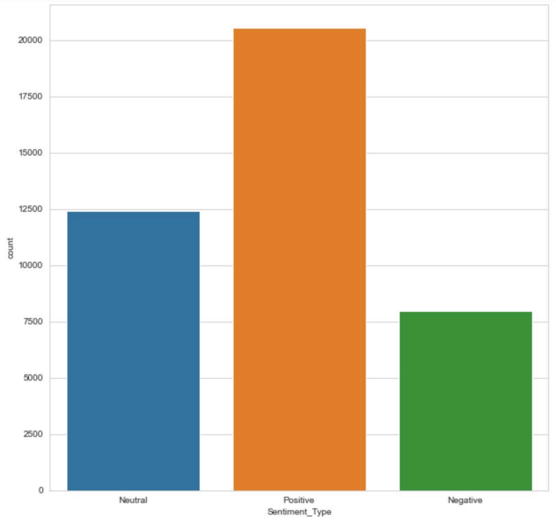
Doing sentimental analysis on description, title and tag using NLP.



For description



For tags



For title

Conclusion

- Views on the videos have a correlation with comments disabled, ratings disabled, video error or removed, category, country code ,num countries , num days , days lapse , likes, comment count, dislikes.

- Sometimes the video on YouTube is classified into different category than the content of the title. Classifying videos using the video title can be more significant.

- Taking the process further, by performing sentiment analysis, we can possibly suggest keeping the title, tags and description towards positive or neutral polarity can give more engagement to the videos which can help the video reach the trending page.

- By using k-mean clustering we found of similar word pattern in the titles of trending videos which can be further used to get the new videos on the search results.

Thank You