ALY 6110 – Module 5

Final Project

Draft Report

Prepared By: Mihir Dharaiya, Ayush Patel, Nithin Penta Reddy

Presented To: Hootan Kamran Habibkhani

Date: June 17th 2024

Introduction

This assignment involves analyzing a dataset of YouTube trending videos in the USA using Python for data processing and Tableau for visualizations. The goal is to uncover trends and patterns, focusing on factors that influence video popularity and categorization. By examining video titles, we aim to predict trending categories using a Naive Bayes classifier. These insights can help content creators, marketers, and platform developers refine their strategies and enhance user engagement.

Business Question: Can we accurately predict the category of a YouTube video based on its title using a Naive Bayes classifier?

Analysis

We began by importing libraries like NumPy and pandas for data operations in Python, as our dataset was in CSV format. Using `pd.read_csv()`, we loaded the dataset, which contains 14 features, such as ‘Channel_title’, ‘Category’, ‘ViewCount’, ‘Likes’, ‘Dislikes’, and ‘CommentCount’, with approximately 268,787 entries.

Data transformation involved dropping unnecessary columns like ‘video_id’ and ‘thumbnail_link’. The ‘Category’ column initially contained only Category IDs, which we mapped to their respective names using a JSON file. We also standardized the date columns, ‘PublishedAt’ and ‘trending_date’, to the yyyy-mm-dd format.

Next, we checked for null values with `isnull().sum()`, discovering 4,549 null values in the ‘description’ column, which we then removed using `dropna()`. We created a new column to calculate the age of each video by subtracting the published date from the trending date, helping us analyze how long it took for a video to trend.

Titles were cleaned by removing stop words and unnecessary characters like “*, #, |” to better understand how certain words influence a video's trending status. The `describe()` function provided descriptive statistics, such as mean, median, standard deviation, maximum, and minimum values.

Correlation coefficients between numeric variables were obtained using the `corr()` function, highlighting that ‘ViewCount’ had the highest correlation with ‘Likes’. Other variables also showed positive correlations, indicating their interdependence.

Finally, the modified dataset was exported for visualization in Tableau.

Data Visualization

View Count by Category [Figure 1]

We created a bar graph to show the average view counts of videos across different categories. ‘Entertainment’ led with the highest number of views, followed closely by ‘Music’. Categories like ‘Film & Animation’ and ‘Science & Technology’ had average view counts around 2,500K. Other categories had average view counts ranging from 1,000K to 1,500K.

View Count vs Likes [Figure 2]

The scatter plot analysis revealed a relationship between View Counts and Likes. As View Counts increase, there is a slight upward trend in Likes. However, this relationship is not strictly linear. Notably, when the View Count surpasses 10M, the number of Likes plateaus, even for higher View Counts between 2M and 3M. This suggests that videos reach a saturation point in terms of Likes, regardless of further increases in View Counts.

Top 10 Channels by Likes [Figure 3]

We produced another bar chart to identify the top 10 channels with the highest number of likes. ‘FFUNTV’ topped the list, primarily due to its trending videos. ‘Rockstar Games’ followed in second place, also with a substantial number of likes from trending content. The remaining channels in the top 10 represented various categories, including Music and Entertainment.

Number of Videos by Category [Figure 4]

A chart was created to quantify the number of videos within each category. It was clear that ‘Entertainment’ and ‘Gaming’ had the highest number of videos. This observation highlights the significant popularity of these categories among viewers in the US, with a substantial portion of trending videos falling into these categories. Following closely in third place was the ‘Music’ category.

Trend of View & Likes Overtime [Figure 5]

A line chart was created to examine the trend of view counts and likes over the years. It showed a decline from 2020 to 2022, followed by an increase leading up to 2024. This trend suggests a potential impact on likes and view counts during 2021 and 2022, possibly influenced by a lower number of videos compared to other years

Top Performing Channels **[Figure 6]**

A treemap visualization was created to identify top-performing channels based on view count, categorized by their respective categories. The analysis showed that channels primarily in the ‘Sports’ category were the top performers, closely followed by those in the ‘Entertainment’ category. This reaffirms earlier findings that the ‘Entertainment’ category is the most-watched category in the US.

Comment Count by Category **[Figure 7]**

We created a bubble chart where each bubble's size reflects the number of comments in each category. As observed in previous analyses, categories such as ‘Entertainment’, ‘Gaming’, ‘Music’, and ‘Sports’ emerged with the highest number of comments.

These visualizations indicate that content preferences in the US center around categories like ‘Entertainment’, ‘Gaming’, and ‘Music’. They also suggest that viewers engage more with creators in these categories, indicating a strong connection or interest in this content.

Word Cloud **[Figure 8]**

In our final step, we generated a word cloud from video titles to identify commonly used words by creators, aiming to enhance engagement and increase the likelihood of videos appearing in the trending section. Focusing on the top 50 frequently used words across the US, this analysis provides insight into key terms and themes driving viewer engagement and popularity.

To conclude our analysis, we developed a comprehensive dashboard featuring all visualizations. This dashboard provides a holistic view of the data, presenting insights, trends, and correlations derived from our analyses [Dashboard 1 & 2]. By consolidating visualizations into a single interface, users can efficiently explore and interpret data to gain valuable insights into viewer preferences, engagement patterns, and content trends across various categories and time periods.

Machine Learning using PySpark

Data Loading and Cleaning

Importing the dataset into a data processing environment (like PySpark in this case), ensuring data quality by filtering out empty or null titles.

Text Preprocessing

Tokenizing the titles into individual words and removing stop words (commonly used words that carry less meaning like "and", "the", etc.). This step prepares the textual data for further analysis and model training.

Exploratory Data Analysis (EDA)

Exploring the dataset to understand common words in titles, analyzing the length of titles, and examining the distribution of video categories. EDA provides insights into the dataset's characteristics and informs feature engineering decisions.

Machine Learning Model Development

Constructing a machine learning pipeline using techniques such as tokenization (splitting text into words), term frequency-inverse document frequency (TF-IDF) to weigh the importance of words, and a Naive Bayes classifier for predicting video categories based on titles.

Model Evaluation

Assessing the performance of the trained model using evaluation metrics such as accuracy. This step validates the model's ability to predict video categories accurately.

Analysis

Steps Taken

1. **Data Preparation**: Load and preprocess the dataset to extract relevant features (titles) and labels (categories).
2. **Text Preprocessing**: Tokenize titles into words and remove stop words to focus on meaningful content.
3. **Exploratory Data Analysis**: Analyze the frequency and distribution of words in titles, understand typical title lengths, and explore the distribution of video categories to inform model training.
4. **Machine Learning Model Training**: Develop a predictive model using a Naive Bayes classifier, which is suitable for text classification tasks due to its simplicity and effectiveness with sparse data.
5. **Model Evaluation**: Assess the model's accuracy in predicting video categories based on a separate test dataset. Evaluation metrics provide insights into the model's performance and its ability to generalize to new data.

Tools and Techniques Used

- **PySpark**: Used for scalable data processing and machine learning on large datasets.

- **Text Preprocessing**: Tokenization and stop words removal to prepare textual data for analysis.
- **Naive Bayes Classifier**: Employed for text classification, leveraging probabilistic methods to predict video categories based on title content.
- **Evaluation Metrics**: Multiclass Classification Evaluator to measure model accuracy and ensure robust performance.

Insights Beyond Basic Analysis

1. **Common Words in Titles**: Identification of frequent words provides insights into popular themes and topics across different video categories.
2. **Text Length Analysis**: Understanding typical title lengths informs feature engineering decisions and model optimization.
3. **Category Distribution**: Analysis of category distribution helps in understanding the balance of data and its impact on model training and prediction accuracy.

Results

Detailed Results

- **Model Accuracy**: The Naive Bayes classifier achieved an accuracy of 74% in predicting video categories based on titles. This demonstrates the model's ability to effectively classify videos into their respective categories using textual data.

Implications and Recommendations

- **Content Strategy**: Content creators can optimize video titles by including keywords associated with trending categories, enhancing visibility and engagement.
- **Marketing Strategies**: Marketers can target audiences more effectively by aligning video titles with popular trends and keywords.
- **Platform Development**: Enhancing platform features for automated categorization can improve user experience and content discoverability.

Future Work

- **Advanced Text Processing**: Implement advanced techniques like stemming, lemmatization, or sentiment analysis to further refine textual data preprocessing.
- **Model Optimization**: Explore alternative classification algorithms (e.g., Random Forest, SVM) to potentially improve prediction accuracy.
- **Feature Expansion**: Incorporate additional features such as video descriptions, tags, and viewer engagement metrics to enhance model performance and predictive capabilities.

Conclusion

This comprehensive analysis of YouTube trending videos in the USA has provided insights into the factors contributing to video popularity and categorization. Using Python for data processing and Tableau for visualizations, key trends and patterns across various categories were identified. PySpark enabled scalable machine learning, resulting in a Naive Bayes classifier with a 74% accuracy in predicting video categories based on titles. Findings highlight the dominance of 'Entertainment' and 'Music' categories, correlations between view counts and likes, and common words in trending video titles. These insights offer actionable recommendations for optimizing video titles, aligning marketing strategies with popular trends, and enhancing platform features for better user experience. Future work will focus on advanced text processing techniques, alternative classification algorithms, and incorporating additional features to improve model performance and predictive capabilities, supporting strategic decision-making in content creation, marketing, and platform development.

Python Code

```
import numpy as np
import pandas as pd
import json
import datetime

import re
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
import string


from collections import Counter

import seaborn as sns
import matplotlib.pyplot as plt
from matplotlib.pyplot import figure
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     /Users/mihirdharaiya/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

```
df = pd.read_csv('US_youtube_trending_data.csv')
```

```
df.head()
```

	video_id	title	publishedAt	channelId	channelTitle	categoryId	trending_date	tags
0	3C66w5Z0ixs	I ASKED HER TO BE MY GIRLFRIEND...	2020-08-11T19:20:14Z	UCvtRTOMP2TqYqu51xNrqAzg	Brawadis	22	2020-08-12T00:00:00Z	brawadis|prank|basketball|skits|ghost|funny vi...
1	M9Pmf9AB4Mo	Apex Legends | Stories from the Outlands – "Th...	2020-08-11T17:00:10Z	UC0ZV6M2THA81QT9hrVWJG3A	Apex Legends	20	2020-08-12T00:00:00Z	Apex Legends|Apex Legends characters|new Apex ...
2	J78aPJ3VyNs	I left youtube for a month and THIS is what ha...	2020-08-11T16:34:06Z	UCYzPXprvl5Y-Sf0g4vX-m6g	jacksepticeye	24	2020-08-12T00:00:00Z	jacksepticeye|funny|funny meme|memes|jacksepti...
3	kXLn3HkpjaA	XXL 2020 Freshman Class Revealed - Official An...	2020-08-11T16:38:55Z	UCbg_UMjlHJg_19SZckaKajg	XXL	10	2020-08-12T00:00:00Z	xxl freshman|xxl freshmen|2020 xxl freshman|20...
4	VIUo6yapDbc	Ultimate DIY Home Movie Theater for The LaBran...	2020-08-11T15:10:05Z	UCDVPcEbVLQgLZX0Rt6jo34A	Mr. Kate	26	2020-08-12T00:00:00Z	The LaBrant Family|DIY|Interior Design|Makeove...

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 268787 entries, 0 to 268786
Data columns (total 16 columns):
 #   Column             Non-Null Count   Dtype
---  ------             --------------   -----
 0   video_id           268787 non-null  object
 1   title              268787 non-null  object
 2   publishedAt        268787 non-null  object
 3   channelId          268787 non-null  object
 4   channelTitle       268787 non-null  object
 5   categoryId         268787 non-null  int64
 6   trending_date      268787 non-null  object
 7   tags               268787 non-null  object
 8   view_count         268787 non-null  int64
 9   likes              268787 non-null  int64
 10  dislikes           268787 non-null  int64
 11  comment_count      268787 non-null  int64
 12  thumbnail_link     268787 non-null  object
 13  comments_disabled  268787 non-null  bool
 14  ratings_disabled   268787 non-null  bool
 15  description        264238 non-null  object
dtypes: bool(2), int64(5), object(9)
memory usage: 29.2+ MB
```

```
df.drop(columns=['video_id','thumbnail_link'],inplace=True, errors='ignore')
```

```
with open("US_category_id.json") as f:
    categoryID = json.load(f)

categoryID = categoryID['items']
ID_to_Category = {int(item['id']): item['snippet']['title'] for item in categoryID}

df['category_title'] = df['categoryId'].map(ID_to_Category)
```

```
ID_to_Category
```

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

```
ID_to_Category.keys()
```

```
dict_keys([1, 2, 10, 15, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 3
9, 40, 41, 42, 43, 44])
```

```
df.head()
```

	title	publishedAt	channelId	channelTitle	categoryId	trending_date	tags	view_count	
0	I ASKED HER TO BE MY GIRLFRIEND...	2020-08-11T19:20:14Z	UCvtRTOMP2TqYqu51xNrqAzg	Brawadis	22	2020-08-12T00:00:00Z	brawadis|prank|basketball|skits|ghost|funny vi...	1514614	15
1	Apex Legends | Stories from the Outlands – “Th...	2020-08-11T17:00:10Z	UC0ZV6M2THA81QT9hrVWJG3A	Apex Legends	20	2020-08-12T00:00:00Z	Apex Legends|Apex Legends characters|new Apex ...	2381688	14
2	I left youtube for a month and THIS is what ha...	2020-08-11T16:34:06Z	UCYzPXprvl5Y-Sf0g4vX-m6g	jacksepticeye	24	2020-08-12T00:00:00Z	jacksepticeye|funny|funny meme|memes|jacksepti...	2038853	35
3	XXL 2020 Freshman Class Revealed - Official An...	2020-08-11T16:38:55Z	UCbg_UMjlHJg_19SZckaKajg	XXL	10	2020-08-12T00:00:00Z	xxl freshman|xxl freshmen|2020 xxl freshman|20...	496771	2
4	Ultimate DIY Home Movie Theater for The LaBran...	2020-08-11T15:10:05Z	UCDVPcEbVLQgLZX0Rt6jo34A	Mr. Kate	26	2020-08-12T00:00:00Z	The LaBrant Family|DIY|Interior Design|Makeove...	1123889	4

```
def clean_trending_date(date):
    y,m,d = date.split('T')[0].split('-')
    return datetime.date(int(y), int(m), int(d))

def clean_publish_time(time):
    y,m,d = time.split('T')[0].split('-')
    return datetime.date(int(y), int(m), int(d))
```

```
df['trending_date'] = df['trending_date'].apply(clean_trending_date)
df['publishedAt'] = df['publishedAt'].apply(clean_publish_time)

df.head(n=2)
```

	title	publishedAt	channelId	channelTitle	categoryId	trending_date	tags	view_count	li
0	I ASKED HER TO BE MY GIRLFRIEND...	2020-08-11	UCvtRTOMP2TqYqu51xNrqAzg	Brawadis	22	2020-08-12	brawadis|prank|basketball|skits|ghost|funny vi...	1514614	156
1	Apex Legends | Stories from the Outlands – “Th...	2020-08-11	UC0ZV6M2THA81QT9hrVWJG3A	Apex Legends	20	2020-08-12	Apex Legends|Apex Legends characters|new Apex ...	2381688	146

```
df.isnull().sum()
```

```
title                   0
publishedAt             0
channelId               0
channelTitle            0
categoryId              0
trending_date           0
tags                    0
view_count              0
likes                   0
dislikes                0
comment_count           0
comments_disabled       0
ratings_disabled        0
description          4549
category_title          0
dtype: int64
```

```
df = df.dropna()
```

```
df['video_age'] = (df['trending_date'] - df['publishedAt']).dt.days
```

```
df.head()
```

	title	publishedAt	channelId	channelTitle	categoryId	trending_date	tags	view_count	li
0	I ASKED HER TO BE MY GIRLFRIEND...	2020-08-11	UCvtRTOMP2TqYqu51xNrqAzg	Brawadis	22	2020-08-12	brawadis|prank|basketball|skits|ghost|funny vi...	1514614	156
1	Apex Legends | Stories from the Outlands – “Th...	2020-08-11	UC0ZV6M2THA81QT9hrVWJG3A	Apex Legends	20	2020-08-12	Apex Legends|Apex Legends characters|new Apex ...	2381688	146
2	I left youtube for a month and THIS is what ha...	2020-08-11	UCYzPXprvl5Y-Sf0g4vX-m6g	jacksepticeye	24	2020-08-12	jacksepticeye|funny|funny meme|memes|jacksepti...	2038853	353
3	XXL 2020 Freshman Class Revealed - Official An...	2020-08-11	UCbg_UMjlHJg_19SZckaKajg	XXL	10	2020-08-12	xxl freshman|xxl freshmen|2020 xxl freshman|20...	496771	23
4	Ultimate DIY Home Movie Theater for The LaBran...	2020-08-11	UCDVPcEbVLQgLZX0Rt6jo34A	Mr. Kate	26	2020-08-12	The LaBrant Family|DIY|Interior Design|Makeove...	1123889	45

```
stop_words = set(stopwords.words('english'))

important_words = {'official', 'video', 'new', 'trailer', 'teaser', 'release', 'exclusive'}

def clean_titles(title):
    tokens = title.lower().split()
    cleaned = []
    for token in tokens:
        token = token.translate(str.maketrans('', '', string.punctuation))
        if (token in important_words or token not in stop_words) and token.isalnum():
            cleaned.append(token)
    return ' '.join(cleaned)

df['title_cl'] = df['title'].apply(clean_titles)

df[['title', 'title_cl']].head()
```

	title	title_cl
0	I ASKED HER TO BE MY GIRLFRIEND...	asked girlfriend
1	Apex Legends | Stories from the Outlands – "Th...	apex legends stories outlands
2	I left youtube for a month and THIS is what ha...	left youtube month happened
3	XXL 2020 Freshman Class Revealed - Official An...	xxl 2020 freshman class revealed official anno...
4	Ultimate DIY Home Movie Theater for The LaBran...	ultimate diy home movie theater labrant family

```
df['title_cl'][0]
```

'asked girlfriend'

```
df.head(1)
```

	title	publishedAt	channelId	channelTitle	categoryId	trending_date	tags	view_count	like
0	I ASKED HER TO BE MY GIRLFRIEND...	2020-08-11	UCvtRTOMP2TqYqu51xNrqAzg	Brawadis	22	2020-08-12	brawadis|prank|basketball|skits|ghost|funny vi...	1514614	15690

```
df.describe()
```

	categoryId	view_count	likes	dislikes	comment_count	video_age
count	264238.000000	2.642380e+05	2.642380e+05	264238.000000	2.642380e+05	264238.00000
mean	18.714061	2.722629e+06	1.306312e+05	1098.990637	1.024426e+04	4.16470
std	6.798227	9.794649e+06	4.539077e+05	7937.852758	7.320983e+04	2.56187
min	1.000000	0.000000e+00	0.000000e+00	0.000000	0.000000e+00	0.00000
25%	17.000000	4.726832e+05	1.789100e+04	0.000000	1.303000e+03	2.00000
50%	20.000000	9.358560e+05	3.993050e+04	0.000000	2.783000e+03	4.00000
75%	24.000000	2.102609e+06	9.775700e+04	463.000000	6.433000e+03	5.00000
max	29.000000	1.407644e+09	1.602153e+07	879354.000000	6.738537e+06	37.00000

```
df.corr()
```

	categoryId	view_count	likes	dislikes	comment_count	comments_disabled	ratings_disabled	video_age
categoryId	1.000000	-0.017938	-0.048274	-0.026545	-0.053809	0.073549	-0.002827	0.019401
view_count	-0.017938	1.000000	0.799887	0.305673	0.473070	0.000925	0.006150	0.336289
likes	-0.048274	0.799887	1.000000	0.382848	0.689861	-0.020519	-0.021614	0.256646
dislikes	-0.026545	0.305673	0.382848	1.000000	0.425958	0.012590	-0.010398	0.073162
comment_count	-0.053809	0.473070	0.689861	0.425958	1.000000	-0.016099	-0.004472	0.088397
comments_disabled	0.073549	0.000925	-0.020519	0.012590	-0.016099	1.000000	0.213166	0.008761
ratings_disabled	-0.002827	0.006150	-0.021614	-0.010398	-0.004472	0.213166	1.000000	-0.011751
video_age	0.019401	0.336289	0.256646	0.073162	0.088397	0.008761	-0.011751	1.000000

```python
# Export the processed dataframe to a CSV file
processed_file_path = 'processed_youtube_data.csv'
df.to_csv(processed_file_path, index=False)
```

```python
from pyspark.sql import SparkSession
from pyspark.sql.functions import explode, split, col
from pyspark.ml.feature import StopWordsRemover, Tokenizer, HashingTF, IDF, StringIndexer
from pyspark.ml.classification import NaiveBayes
from pyspark.ml.evaluation import MulticlassClassificationEvaluator
from pyspark.ml import Pipeline
from pyspark.sql.types import StringType
```

```python
# Initialize Spark session
spark = SparkSession.builder \
    .appName("YouTube Trending Videos Analysis") \
    .getOrCreate()
```

```
24/06/17 19:59:55 WARN Utils: Your hostname, Mihirs-MacBook-Air.local resolves to a loopback address: 127.0.0.1; us
ing 192.168.2.25 instead (on interface en0)
24/06/17 19:59:55 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another address
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
24/06/17 19:59:56 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-ja
va classes where applicable
24/06/17 20:00:08 WARN GarbageCollectionMetrics: To enable non-built-in garbage collector(s) List(G1 Concurrent G
C), users should configure it(them) to spark.eventLog.gcMetrics.youngGenerationGarbageCollectors or spark.eventLog.
gcMetrics.oldGenerationGarbageCollectors
```

```python
# Load cleaned dataset
data = spark.read.csv('processed_youtube_data.csv', header=True, inferSchema=True)
```

```python
data = data.filter(col("title_cl").isNotNull() & (col("title_cl") != ""))
```

```python
data.show(5)
```

```
+--------------------+-----------+--------------------+--------------+----------+-------------+--------------------
+----------+------+--------+-------------+-----------------+----------------+--------------------+--------------+--
-------+--------------------+
|               title|publishedAt|           channelId|  channelTitle|categoryId|trending_date|                tags
|view_count| likes|dislikes|comment_count|comments_disabled|ratings_disabled|         description|category_title|vi
deo_age|            title_cl|
+--------------------+-----------+--------------------+--------------+----------+-------------+--------------------
+----------+------+--------+-------------+-----------------+----------------+--------------------+--------------+--
-------+--------------------+
|Apex Legends | St...| 2020-08-11|UC0ZV6M2THA81QT9h...|  Apex Legends|        20|   2020-08-12|Apex Legends|Apex...
|   2381688|146739|    2794|        16549|            False|           False|While running her...|        Gaming|
1|apex legends stor...|
|I left youtube fo...| 2020-08-11|UCYzPXprvl5Y-Sf0g...| jacksepticeye|        24|   2020-08-12|jacksepticeye|fun...
|   2038853|353787|    2628|        40221|            False|           False|I left youtube fo...| Entertainment|
1|left youtube mont...|
|Ultimate DIY Home...| 2020-08-11|UCDVPcEbVLQgLZX0R...|      Mr. Kate|        26|   2020-08-12|The LaBrant Famil...
|   1123889| 45802|     964|         2196|            False|           False|Transforming The ...| Howto & Style|
1|ultimate diy home...|
|I Haven't Been Ho...| 2020-08-11|UC5zJwsFtEs9WYe3A...|Professor Live|        24|   2020-08-12|Professor injury|...
|    949491| 77487|     746|         7506|            False|           False|Subscribe To My C...| Entertainment|
1|havent honest inj...|
|OUR FIRST FAMILY ...| 2020-08-12|UCDSJCBYqL7VQrlXf...| Les Do Makeup|        26|   2020-08-12|              [None]
|    470446| 47990|     440|         4558|            False|           False|Hi babygirls!  Th...| Howto & Style|
0|  first family intro|
+--------------------+-----------+--------------------+--------------+----------+-------------+--------------------
+----------+------+--------+-------------+-----------------+----------------+--------------------+--------------+--
-------+--------------------+
only showing top 5 rows
```

```
words_df = data.select(split(col("title_cl"), " ").alias("words"))
```

```
words_df.show(5)
```

```
+--------------------+
|               words|
+--------------------+
|[apex, legends, s...|
|[left, youtube, m...|
|[ultimate, diy, h...|
|[havent, honest, ...|
|[first, family, i...|
+--------------------+
only showing top 5 rows
```

```
remover = StopWordsRemover(inputCol="words", outputCol="filtered_words")
pipeline = Pipeline(stages=[remover])
cleaned_words_df = pipeline.fit(words_df).transform(words_df)
```

```
cleaned_words_df.show(5)
```

```
+--------------------+--------------------+
|               words|      filtered_words|
+--------------------+--------------------+
|[apex, legends, s...|[apex, legends, s...|
|[left, youtube, m...|[left, youtube, m...|
|[ultimate, diy, h...|[ultimate, diy, h...|
|[havent, honest, ...|[havent, honest, ...|
|[first, family, i...|[first, family, i...|
+--------------------+--------------------+
only showing top 5 rows
```

```
exploded_df = cleaned_words_df.select(explode(col("filtered_words")).alias("word"))
```

```
exploded_df.show(20)
```

```
+--------+
|    word|
+--------+
|    apex|
| legends|
| stories|
|outlands|
|    left|
| youtube|
|   month|
|happened|
|ultimate|
|     diy|
|    home|
|   movie|
| theater|
| labrant|
|  family|
|  havent|
|  honest|
|  injury|
|   heres|
|   truth|
+--------+
only showing top 20 rows
```

```
top_words = exploded_df.limit(20).collect()
```

```python
# Print the collected words
for row in top_words:
    print(row['word'])
```

```
apex
legends
stories
outlands
left
youtube
month
happened
ultimate
diy
home
movie
theater
labrant
family
havent
honest
injury
heres
truth
```

```python
# Machine Learning Analysis for Category Prediction
# Select relevant columns and drop rows with null values in these columns
model_data = data.select("title_cl", "category_title").dropna()
```

```python
# Split the data into training and test sets
train_data, test_data = model_data.randomSplit([0.8, 0.2], seed=42)
```

```python
# Tokenize the titles
tokenizer = Tokenizer(inputCol="title_cl", outputCol="words")
```

```python
# Compute term frequencies
hashingTF = HashingTF(inputCol="words", outputCol="rawFeatures", numFeatures=10000)
```

```python
# Compute the IDF (Inverse Document Frequency)
idf = IDF(inputCol="rawFeatures", outputCol="features")
```

```python
# Index the labels (categories)
indexer = StringIndexer(inputCol="category_title", outputCol="label")
```

```python
# Define the Naive Bayes classifier
nb = NaiveBayes(modelType="multinomial")
```

```python
# Create a pipeline
pipeline = Pipeline(stages=[tokenizer, hashingTF, idf, indexer, nb])
```

```python
# Train the model
model = pipeline.fit(train_data)
```

```python
# Make predictions on the test data
predictions = model.transform(test_data)
```

```python
# Evaluate the model
evaluator = MulticlassClassificationEvaluator(predictionCol="prediction", labelCol="label", metricName="accuracy")
accuracy = evaluator.evaluate(predictions)

print(f"Test set accuracy = {accuracy:.2f}")
```

```
24/06/17 20:14:30 WARN DAGScheduler: Broadcasting large task binary with size 4.4 MiB
24/06/17 20:14:31 WARN InstanceBuilder: Failed to load implementation from:dev.ludovic.netlib.blas.JNIBLAS
24/06/17 20:14:31 WARN InstanceBuilder: Failed to load implementation from:dev.ludovic.netlib.blas.VectorBLAS
```

```
Test set accuracy = 0.74
```

```python
# Select specific columns to print (adjust column names as needed)
predictions.select("features", "label", "prediction").show()

# Alternatively, collect specific columns and print them
predictions_list = predictions.select("features", "label", "prediction").collect()
for row in predictions_list:
    print(f"Features: {row['features']}, Label: {row['label']}, Prediction: {row['prediction']}")
```

```
24/06/17 20:16:27 WARN DAGScheduler: Broadcasting large task binary with size 4.3 MiB
```

```
+--------------------+-----+----------+
|            features|label|prediction|
+--------------------+-----+----------+
|(10000,[3372,4703...| 22.0|      22.0|
|(10000,[3372,4703...| 22.0|      22.0|
|(10000,[387,1553,...| 44.0|      44.0|
|(10000,[387,1553,...| 44.0|      44.0|
|(10000,[3372,3560...| 16.0|      16.0|
|(10000,[3372,3560...| 16.0|      16.0|
|(10000,[2789,3372...| 15.0|      15.0|
|(10000,[2789,3372...| 15.0|      15.0|
|(10000,[488,1990,...| 50.0|      50.0|
|(10000,[488,1990,...| 50.0|      50.0|
|(10000,[488,1990,...| 50.0|      50.0|
|(10000,[488,1990,...| 50.0|      50.0|
|(10000,[793,2891,...| 49.0|      17.0|
|(10000,[1085,1585...| 53.0|      53.0|
|(10000,[2369,3216...| 18.0|      18.0|
```

Appendix

Figure 1

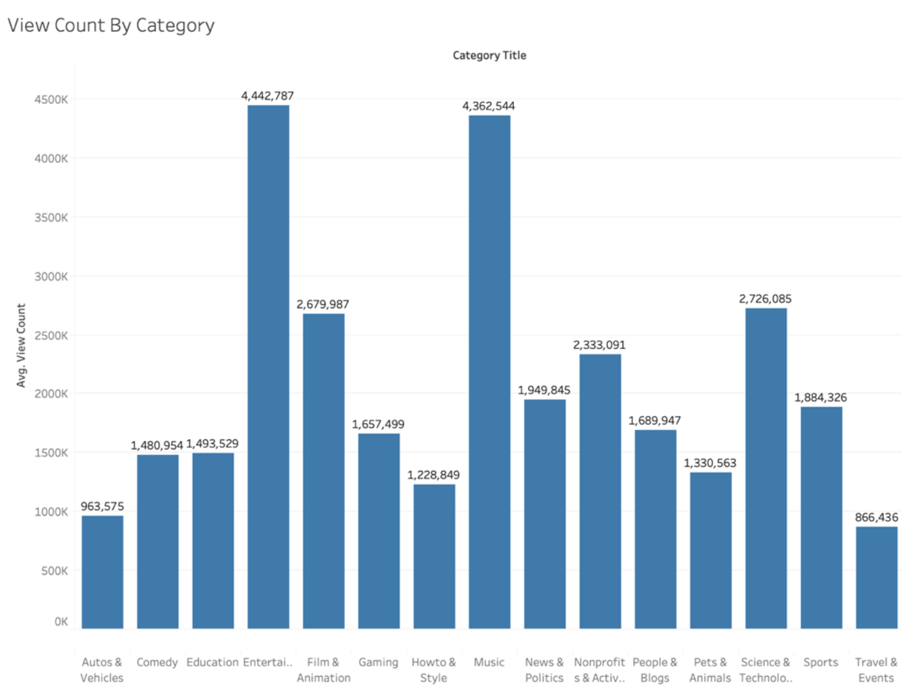


Figure 2

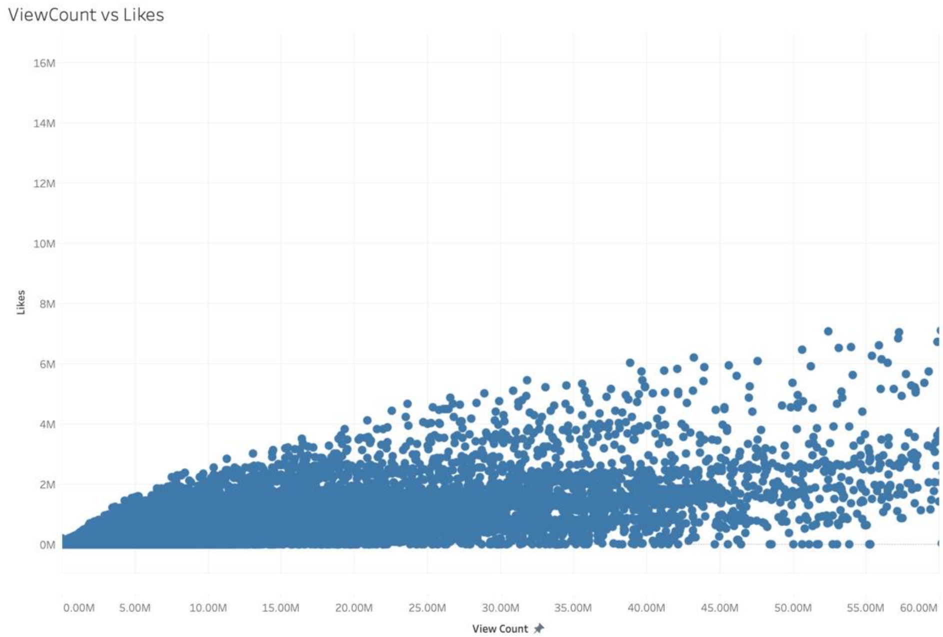


Figure 3

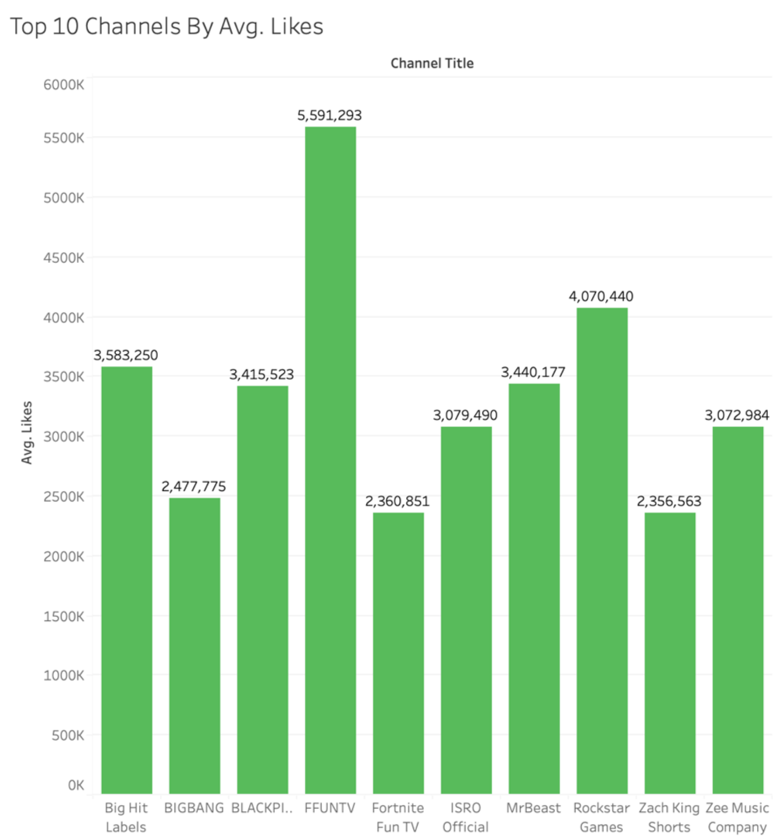


Figure 4

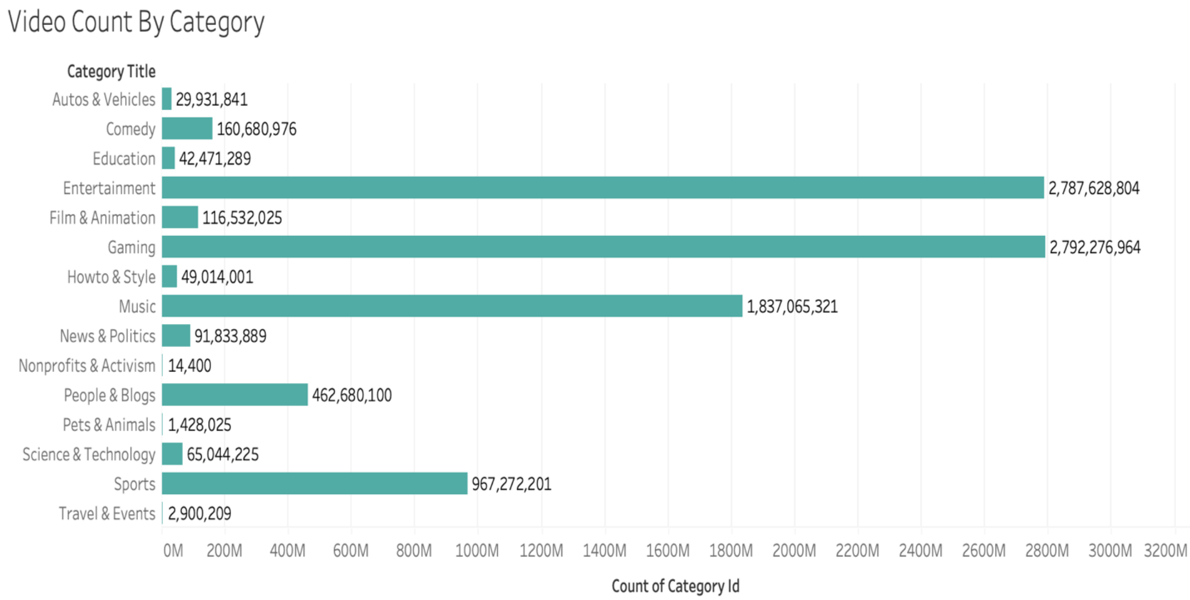


Figure 5

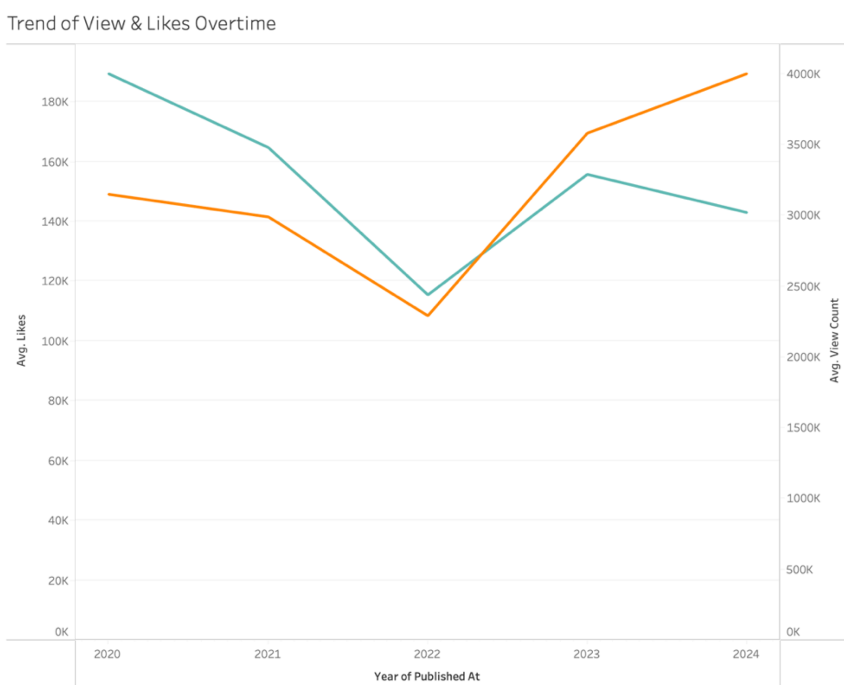


Figure 6

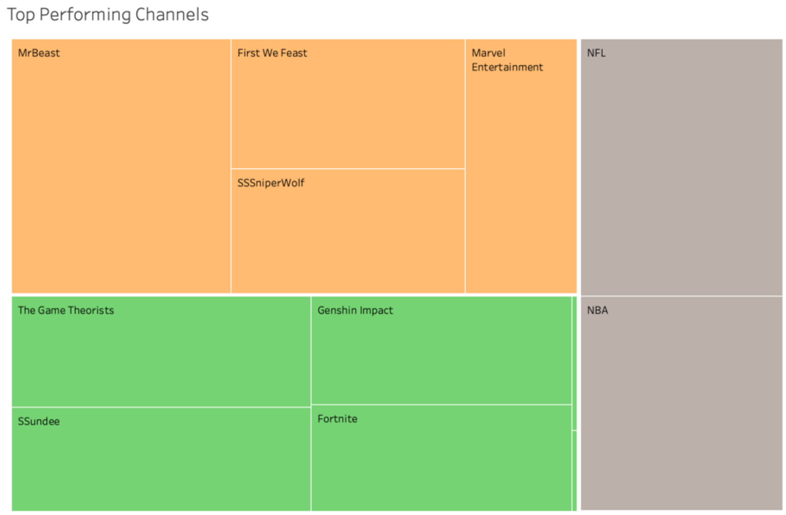


Figure 7

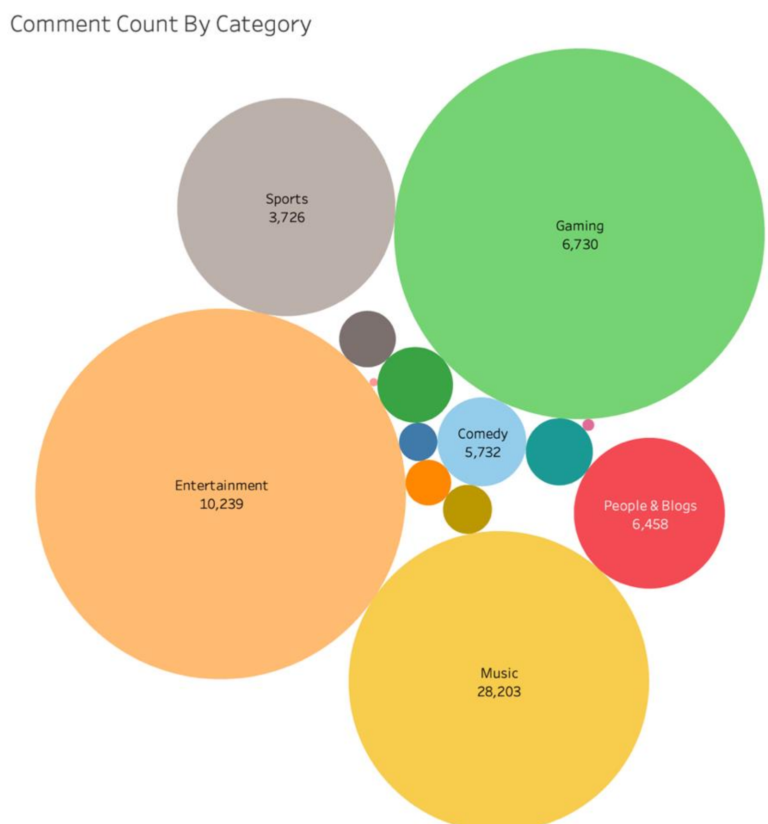
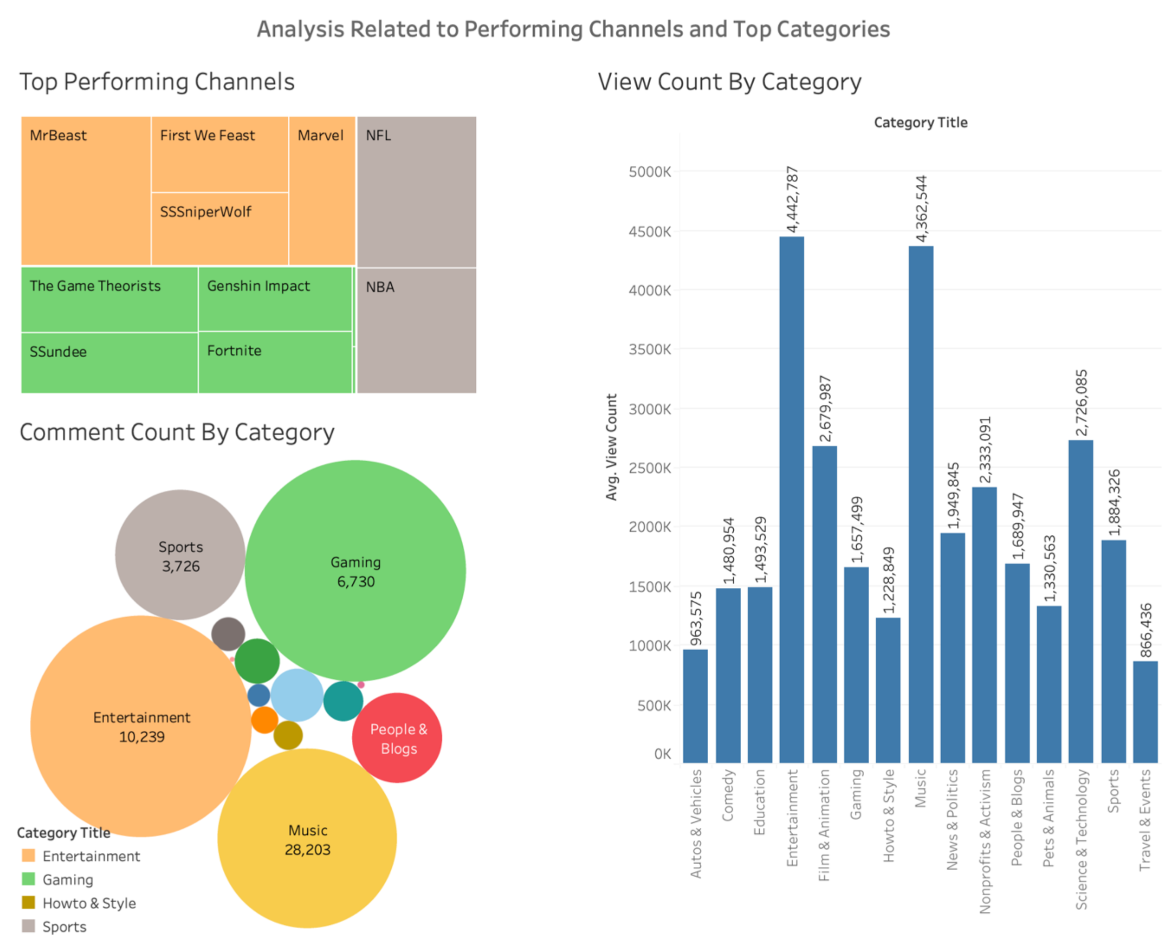


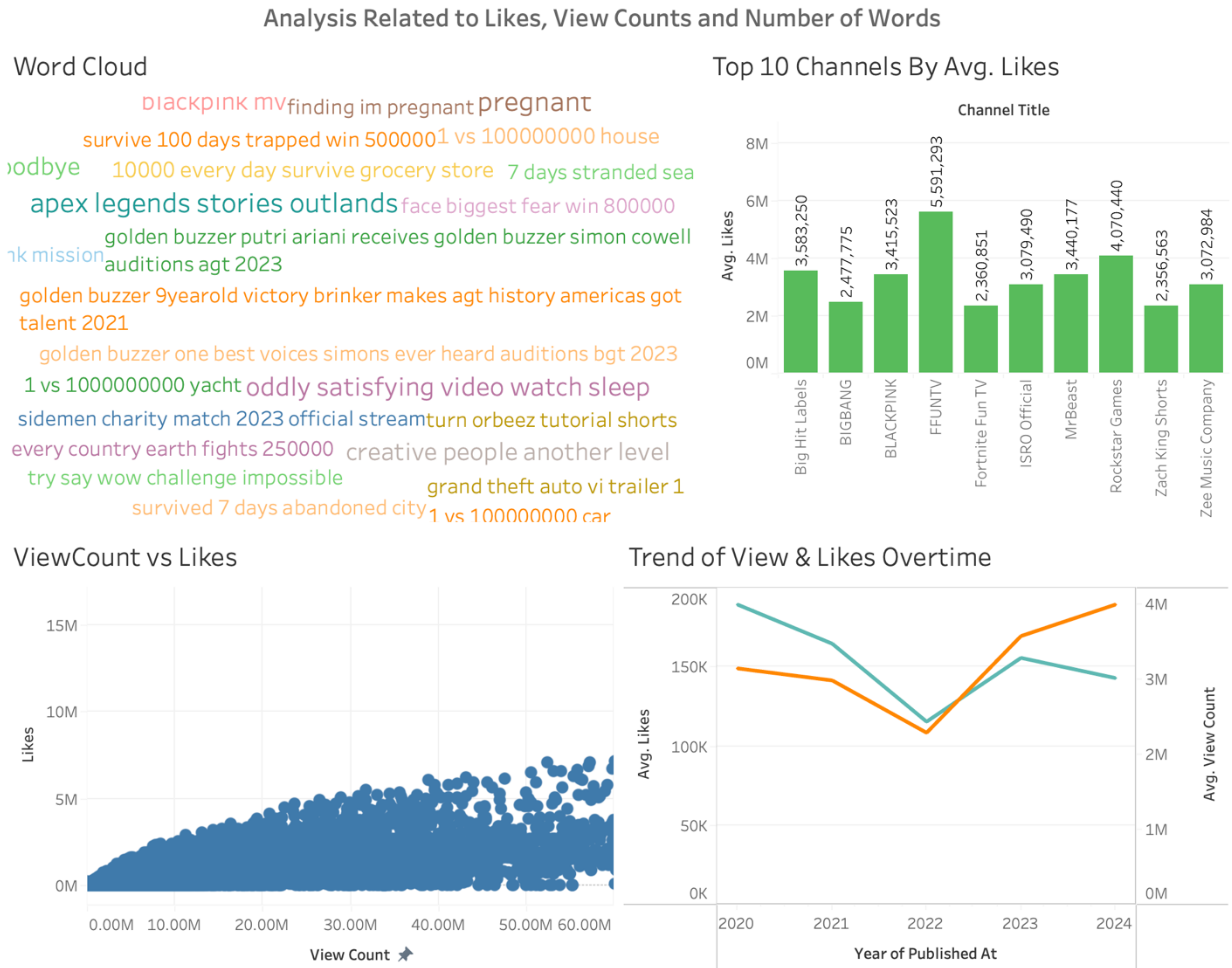
Figure 8

Word Cloud

Dashboard 1



Dashboard 2



References

1. Welcome to Spark Python API Docs! — PySpark 2.4.5 documentation. (n.d.). Spark.apache.org. https://spark.apache.org/docs/latest/api/python/index.html

2. NaiveBayes — PySpark master documentation. (n.d.). Spark.apache.org. Retrieved June 18, 2024, from https://spark.apache.org/docs/latest/api/python/reference/api/pyspark.ml.classification.NaiveBayes.html

3. Build a Treemap. (n.d.). Help.tableau.com. https://help.tableau.com/current/pro/desktop/en-us/buildexamples_treemap.htm

4. Python | Pandas dataframe.corr(). (2018, November 20). GeeksforGeeks.

https://www.geeksforgeeks.org/python-pandas-dataframe-corr/