dataanalysttask9projectii-1

November 19, 2025

1 YouTube Channel Analytics Project

1.0.1 Data Analysis & Visualization using Python (Pandas, Matplotlib, Seaborn)

1.1 Project Objective

Analyze the Top YouTubers dataset to uncover: - The largest creators by subscribers, views, and uploads
- Category-wise performance
- Growth patterns based on channel start year
- View efficiency (views per video)
- Correlations between subscribers, views, and uploads

We will clean the data → perform EDA → visualize insights → export a processed dataset for dashboards.

1.2 Load Dataset

We load the CSV using Pandas and preview the structure.

[2]:
```python
import pandas as pd

file_path = r"D:\intern\project 2\Top Youtubers Dataset.csv"

df = pd.read_csv(file_path, encoding="cp1252")
df.head()
```

[2]:
```
   Rank                     Youtuber  Subscribers   Video Views  Video Count  \
0     1                      MrBeast    284000000  5.240290e+10          803
1     2                     T-Series    268000000  2.586240e+11        21237
2     3                YouTube Movies    181000000  0.000000e+00            0
3     4  Cocomelon - Nursery Rhymes    177000000  1.828810e+11         1188
4     5                    SET India    174000000  1.653950e+11       139720

          Category  Started
0    Entertainment     2012
1            Music     2006
2  Film & Animation     2015
3        Education     2006
4            Shows     2006
```

```
[3]: df.columns
```

```
[3]: Index(['Rank', 'Youtuber', 'Subscribers', 'Video Views', 'Video Count',
            'Category', 'Started'],
           dtype='object')
```

```
[6]: # Clean column names for easier access
     df.columns = df.columns.str.strip().str.lower().str.replace(" ", "_")

     df.head()
```

```
[6]:    rank                    youtuber  subscribers   video_views  video_count  \
     0     1                     MrBeast    284000000  5.240290e+10          803
     1     2                    T-Series    268000000  2.586240e+11        21237
     2     3              YouTube Movies    181000000  0.000000e+00            0
     3     4  Cocomelon - Nursery Rhymes    177000000  1.828810e+11         1188
     4     5                   SET India    174000000  1.653950e+11       139720

                category  started
     0     Entertainment     2012
     1             Music     2006
     2  Film & Animation     2015
     3         Education     2006
     4             Shows     2006
```

1.3 Data Cleaning Steps

- Standardize column names
- Convert numeric fields (Subscribers, Views, Video Count)
- Handle missing values
- Convert "Started" year into numeric

This ensures clean and consistent data for analysis.

```
[39]: numeric_cols = ['subscribers', 'video_views', 'video_count']
      df = df.drop_duplicates()
      for col in numeric_cols:
          df[col] = df[col].astype(str).str.replace(",", "").astype(float)
```

1.4 Feature Engineering

We create new useful metrics: - **views_per_video** = Total Views / Total Videos
- **subs_to_views_ratio** = Subscribers / Views
- **channel_age** = Current Year – Started Year

These KPIs help reveal deeper insights.

```
[40]: df['started'] = pd.to_numeric(df['started'], errors='coerce')
```

```
[41]: df['views_per_video'] = df['video_views'] / df['video_count'].replace(0,1)
```

```
[42]: df['subs_to_views_ratio'] = df['subscribers'] / df['video_views'].replace(0,1)
```

```
[43]: import datetime

      current_year = 2024
      df['channel_age'] = current_year - df['started']
```

```
[44]: df['subs_per_video'] = df['subscribers'] / df['video_count'].replace(0,1)
```

```
[45]: df['engagement_score'] = (df['video_views'] / df['subscribers']).
        ↪replace([float('inf'), -float('inf')], 0)
```

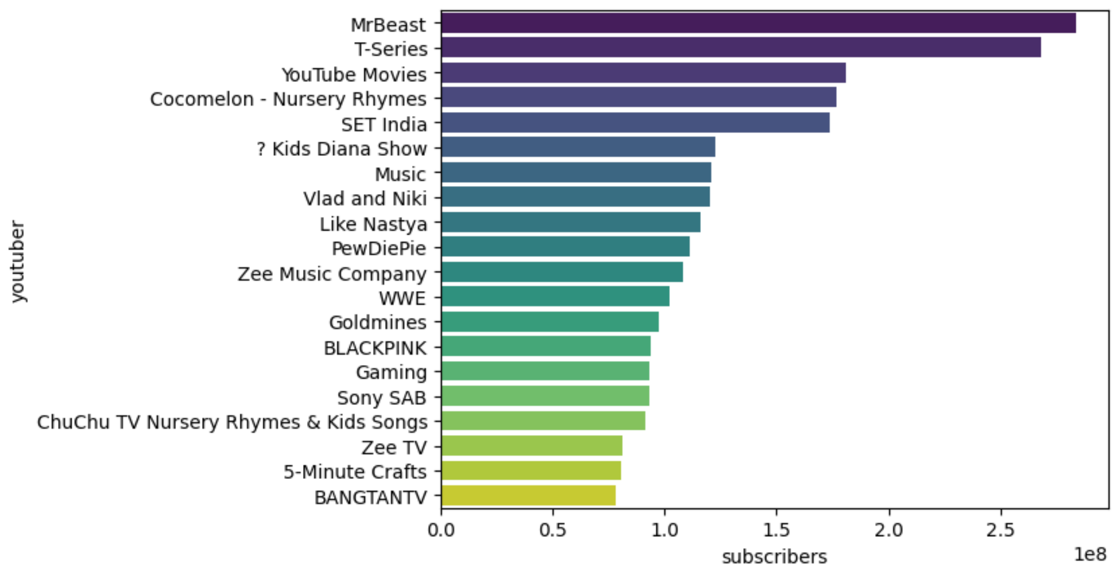
1.5 Data Visualizations

We create 10+ charts to understand: - Top YouTubers by Subscribers & Views
- Category distribution
- Growth by Start Year
- Upload count analysis
- Correlation heatmap
- Relationship between Subscribers, Views, and Video Count

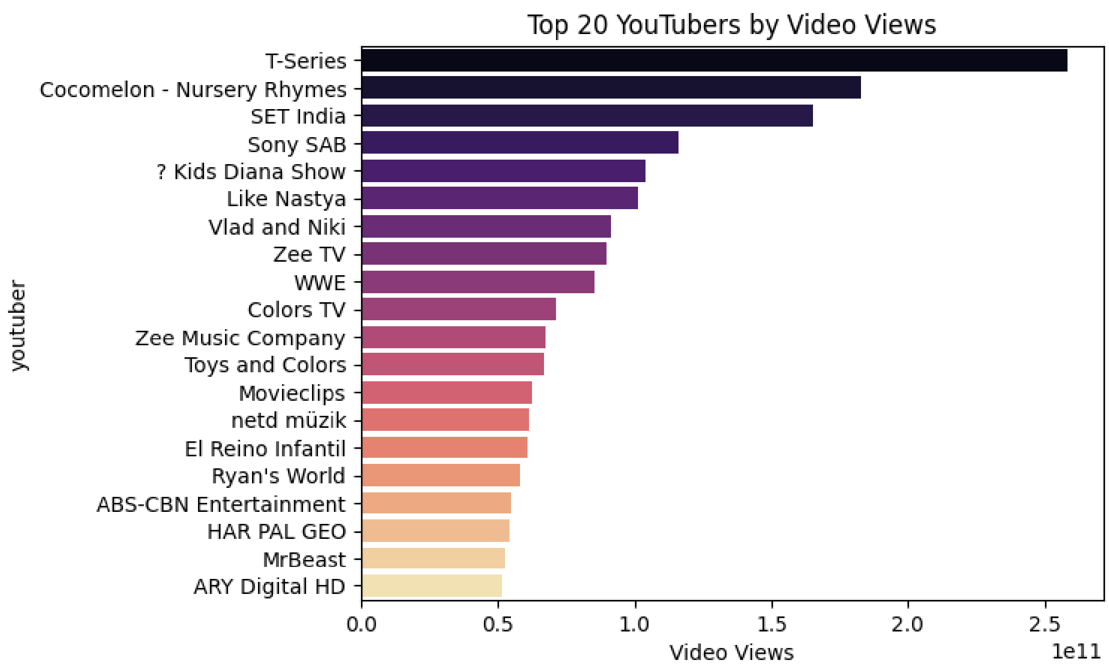
Charts are created using matplotlib & seaborn.

```
[46]: sns.barplot(
          data=top20,
          x='subscribers',
          y='youtuber',
          hue='youtuber',
          palette='viridis',
          legend=False
      )
```

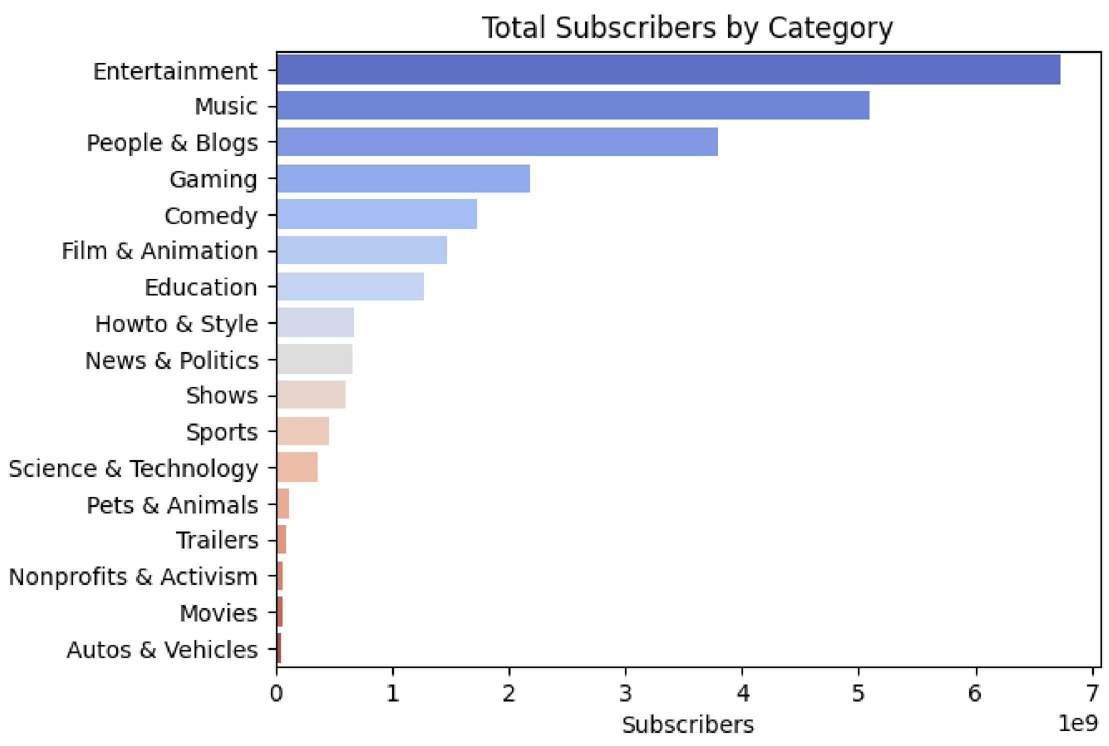
```
[46]: <Axes: xlabel='subscribers', ylabel='youtuber'>
```



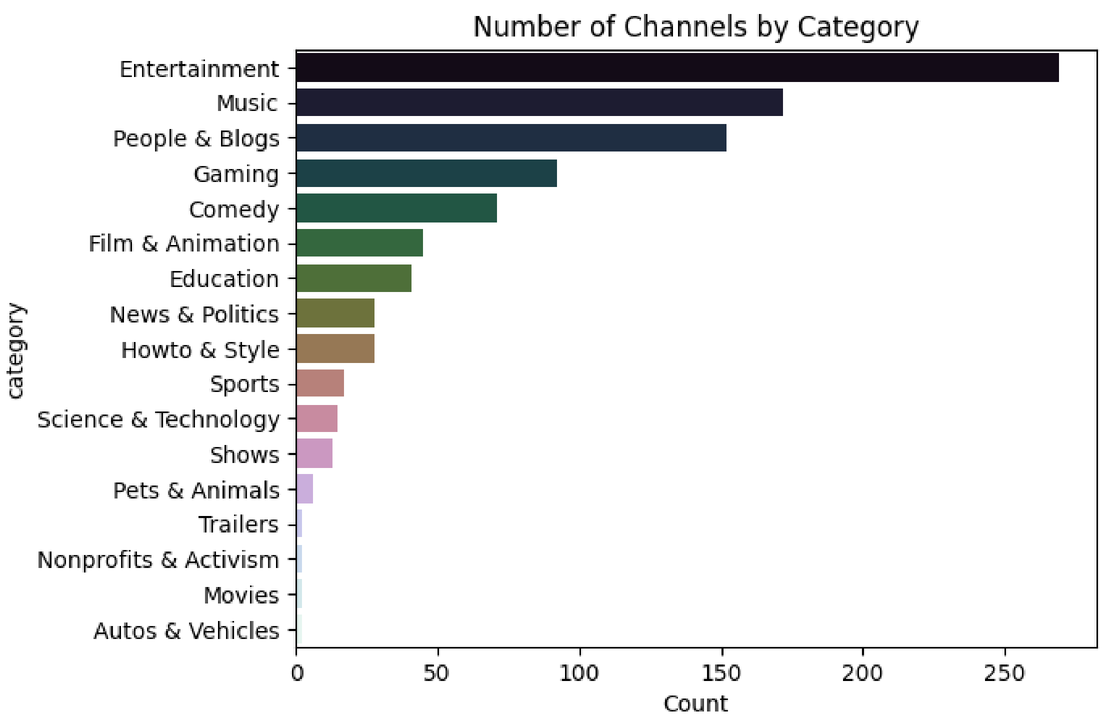
[47]:
```
top20_views = df.sort_values('video_views', ascending=False).head(20)

sns.barplot(
    data=top20_views,
    x='video_views',
    y='youtuber',
    hue='youtuber',
    palette='magma',
    legend=False
)
plt.title("Top 20 YouTubers by Video Views")
plt.xlabel("Video Views")
plt.show()
```



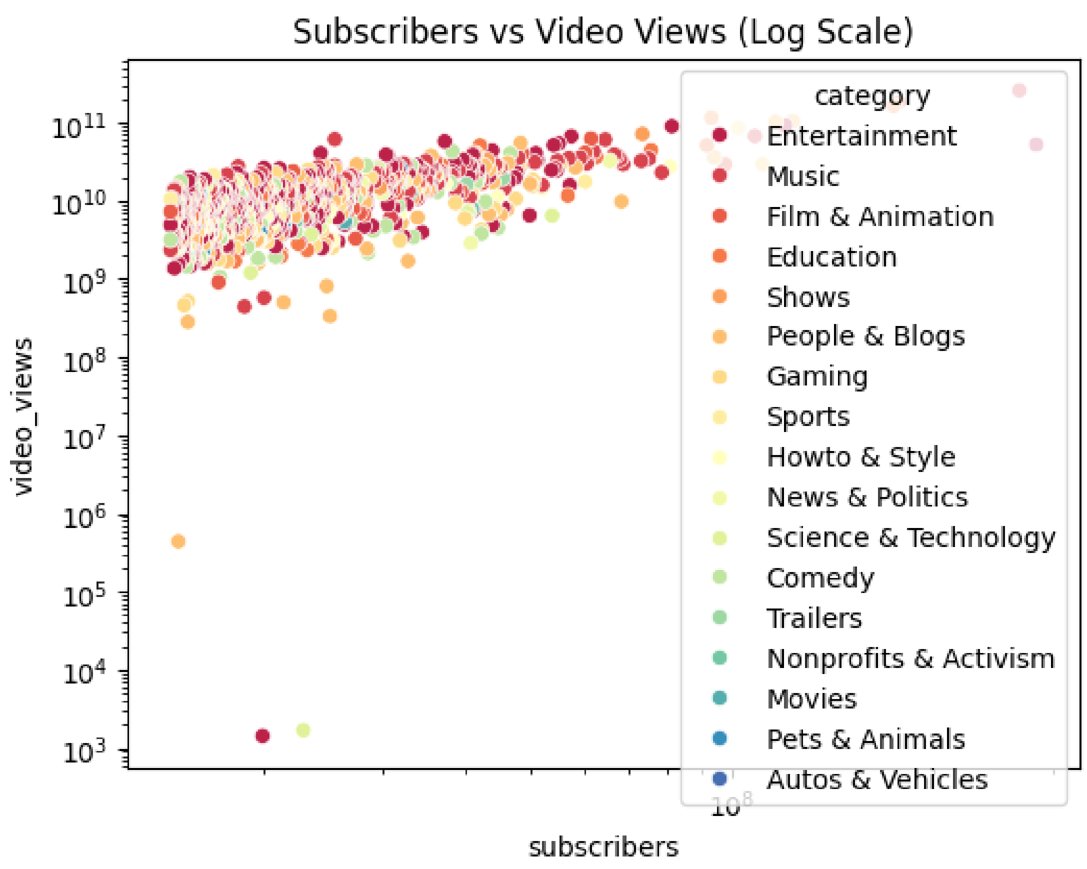
[48]:
```
cat_subs = df.groupby('category')['subscribers'].sum().
  ↪sort_values(ascending=False)

sns.barplot(
    x=cat_subs.values,
    y=cat_subs.index,
    hue=cat_subs.index,
    palette='coolwarm',
    legend=False
)
plt.title("Total Subscribers by Category")
plt.xlabel("Subscribers")
plt.ylabel("")
plt.show()
```



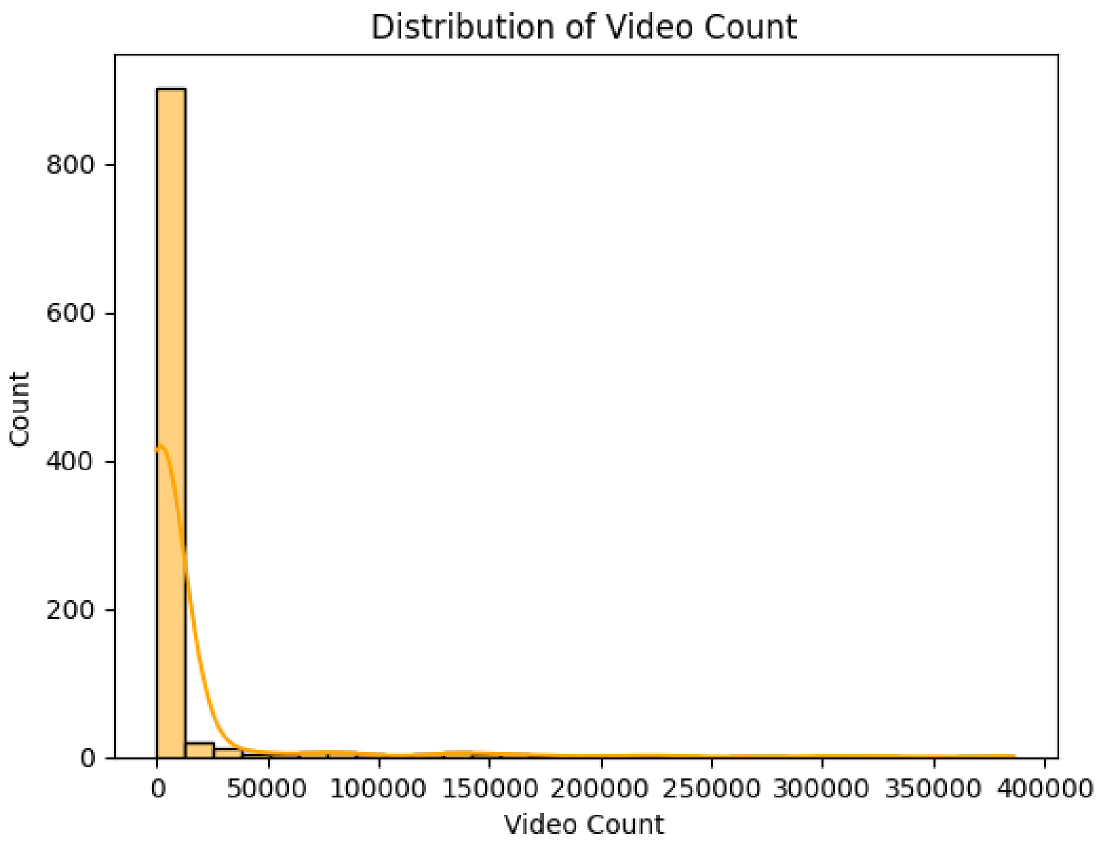
[49]:
```
cat_count = df['category'].value_counts()

sns.barplot(
    x=cat_count.values,
    y=cat_count.index,
    hue=cat_count.index,
    palette='cubehelix',
    legend=False
)
plt.title("Number of Channels by Category")
plt.xlabel("Count")
plt.show()
```



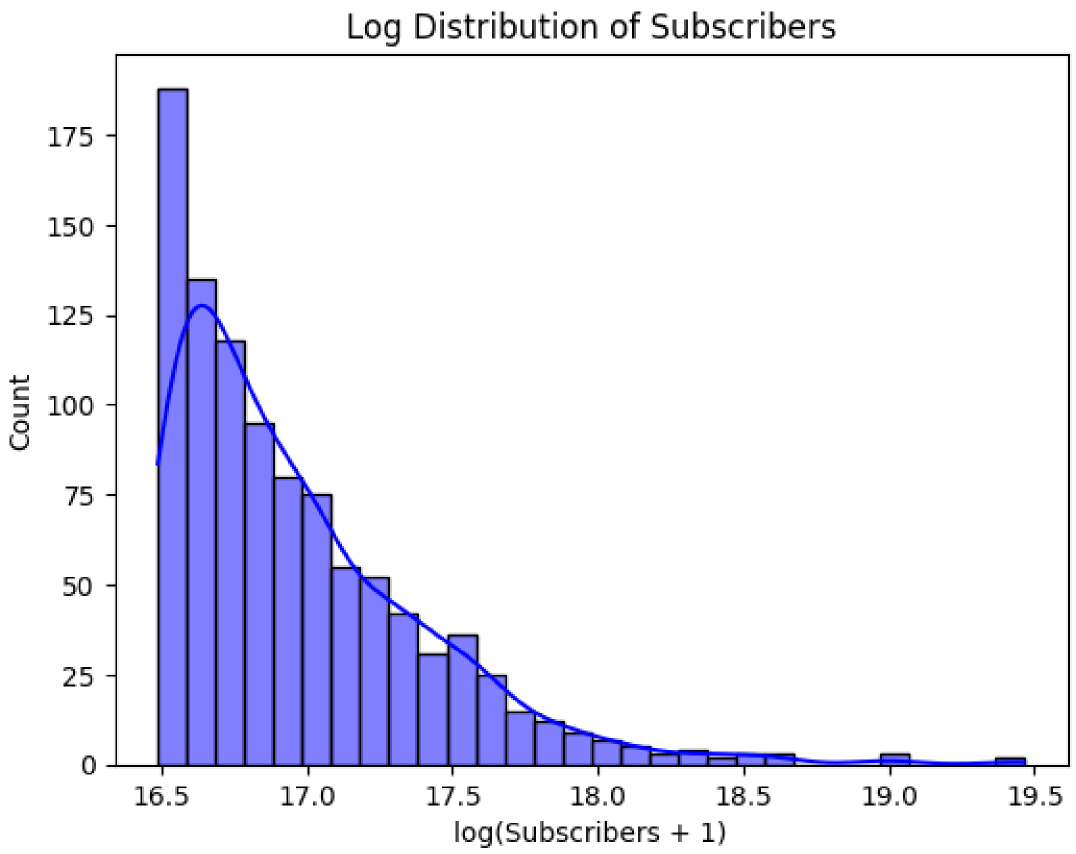
[50]:
```
sns.scatterplot(
    data=df,
    x='subscribers',
    y='video_views',
    hue='category',
    palette='Spectral'
)
plt.xscale('log')
plt.yscale('log')
plt.title("Subscribers vs Video Views (Log Scale)")
plt.show()
```



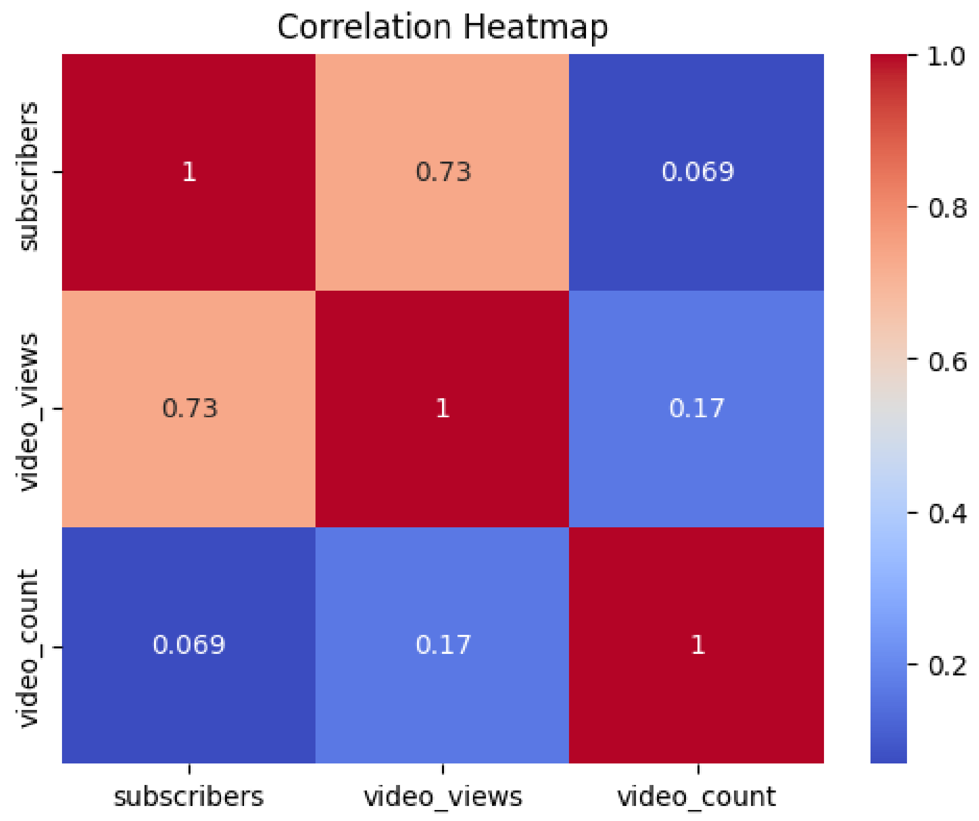
[51]:
```
sns.histplot(df['video_count'], bins=30, kde=True, color='orange')
plt.title("Distribution of Video Count")
plt.xlabel("Video Count")
plt.show()
```



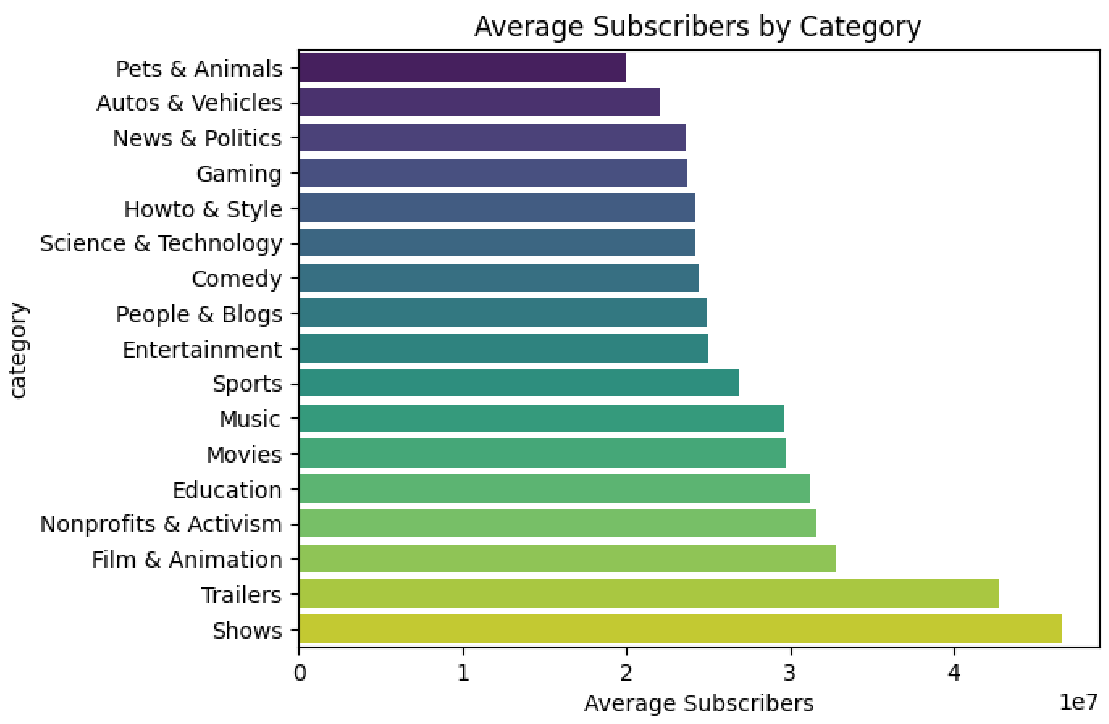
[52]:
```
sns.histplot(np.log1p(df['subscribers']), bins=30, kde=True, color='blue')
plt.title("Log Distribution of Subscribers")
plt.xlabel("log(Subscribers + 1)")
plt.show()
```



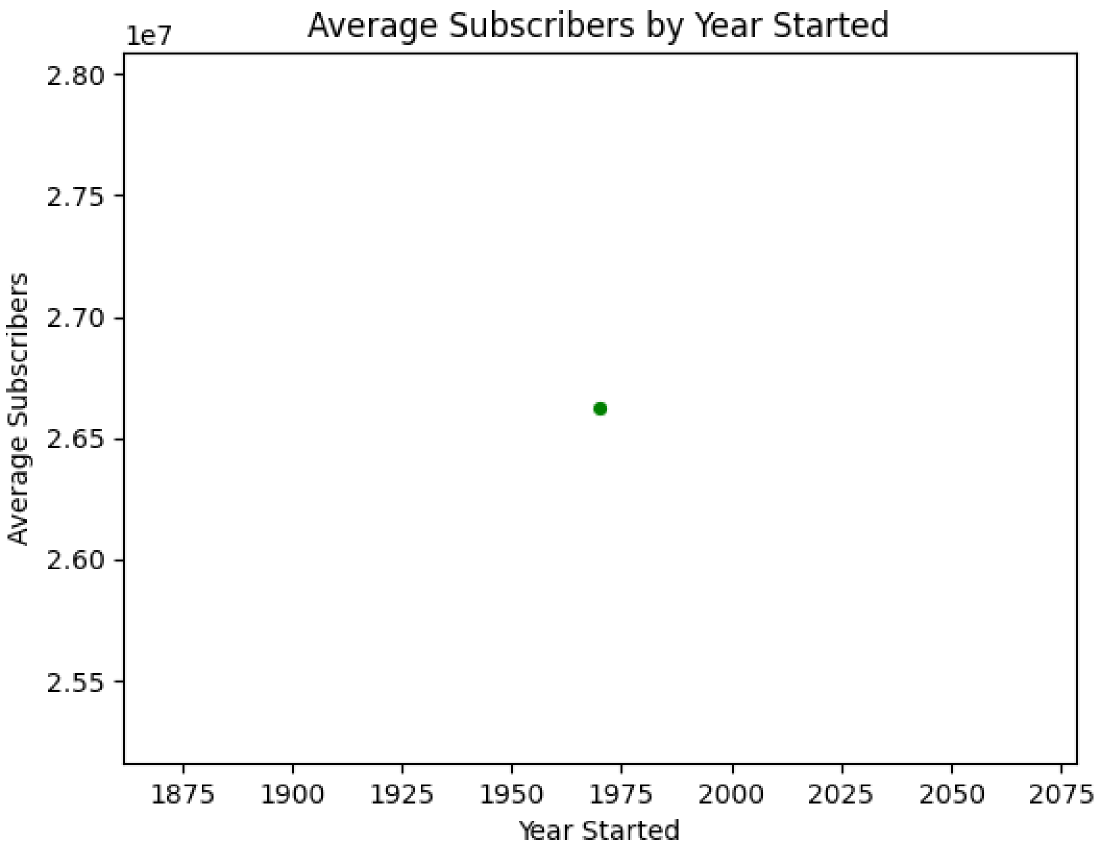
[53]:
```
corr = df[['subscribers','video_views','video_count']].corr()

sns.heatmap(corr, annot=True, cmap='coolwarm')
plt.title("Correlation Heatmap")
plt.show()
```



[54]:
```
cat_avg = df.groupby('category')['subscribers'].mean().sort_values()

sns.barplot(
    x=cat_avg.values,
    y=cat_avg.index,
    hue=cat_avg.index,
    palette='viridis',
    legend=False
)
plt.title("Average Subscribers by Category")
plt.xlabel("Average Subscribers")
plt.show()
```



[55]:
```
df['started'] = pd.to_datetime(df['started'], errors='coerce')
df['year'] = df['started'].dt.year

yr_avg = df.groupby('year')['subscribers'].mean().dropna()

sns.lineplot(
    x=yr_avg.index,
    y=yr_avg.values,
    marker='o',
    color='green'
)
plt.title("Average Subscribers by Year Started")
plt.xlabel("Year Started")
plt.ylabel("Average Subscribers")
plt.show()
```



1.6 Save Processed Dataset

We export the cleaned + feature-engineered dataset as:

[56]:
```
# Save cleaned dataset
df.to_csv("Processed_youtube_dataset.csv", index=False)

print("Saved as Processed_youtube_dataset.csv")
```

```
Saved as Processed_youtube_dataset.csv
```