

✓ Apply advanced statistical and analytical methods to solve complex problems

1. Implement time series analysis for forecasting trends and seasonality.
2. Perform sentiment analysis or text mining on unstructured data.
3. Explore clustering or classification techniques for segmentation and pattern recognition.

Sri Vidya Aiswarya

Loading and examining the dataset

```
import pandas as pd
```

```
# Load the dataset
data = pd.read_csv("/content/disney_plus_titles.csv")
```

```
# Display column names and data types
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1368 entries, 0 to 1367
Data columns (total 12 columns):
#   Column          Non-Null Count  Dtype
---  -
0   show_id         1368 non-null   object
1   type            1368 non-null   object
2   title           1368 non-null   object
3   director        928 non-null    object
4   cast            1194 non-null   object
5   country         1193 non-null   object
6   date_added      1365 non-null   object
7   release_year    1368 non-null   int64
8   rating          1366 non-null   object
9   duration        1368 non-null   object
10  listed_in       1368 non-null   object
11  description     1368 non-null   object
dtypes: int64(1), object(11)
memory usage: 128.4+ KB
```

```
# Display the first few rows of the dataset
data.head(3)
```

```
show_id  type  title  director  cast  country  date_added  release_year  rating
0        s1  Movie  A Spark  Jason  Apton  NaN        September  2021      TV-PG
           Story  Serman,  Corbin,
           Dare  Leanne  Louis
           Gonzales
1        s2  Movie  Spooky  Robert  Tucker  United  September  2011      G
           Buddies  Vince  Albrizzi,
           Bader,
           Ameko
           Eks
           Mas...
```

```
# Displaying the column names
data.columns.values
```

```
array(['show_id', 'type', 'title', 'director', 'cast', 'country',
       'date_added', 'release_year', 'rating', 'duration', 'listed_in',
       'description'], dtype=object)
```

```
# Checking for missing values
data.isnull().sum()
```

```
show_id      0
type         0
title        0
director     440
cast         174
```

```

country      175
date_added   3
release_year  0
rating       2
duration     0
listed_in    0
description  0
dtype: int64

```

```
#importing necessary libraries
```

```

import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
from textblob import TextBlob

```

Time Series Analysis

```
# Convert 'release_year' to datetime if it's not already
data['release_year'] = pd.to_datetime(data['release_year'], format='%Y', errors='coerce')
```

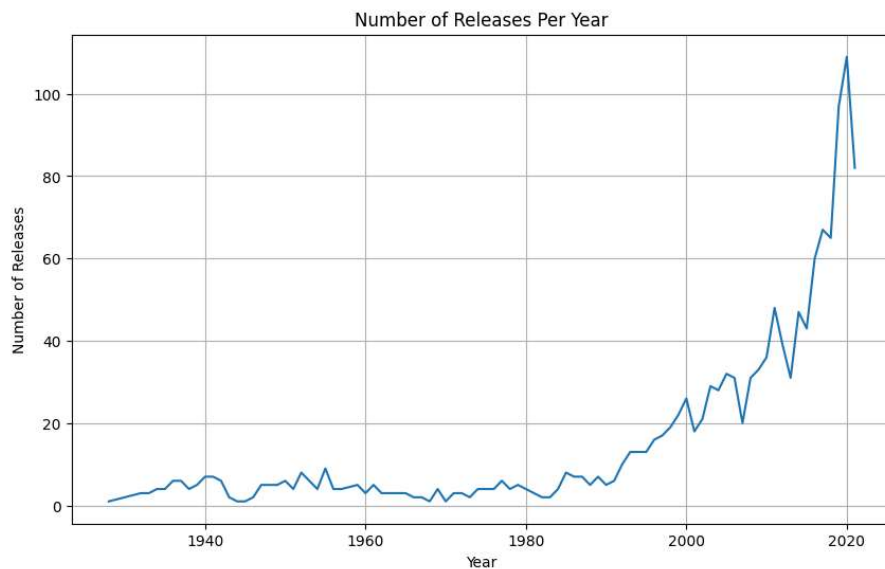
```
# Drop rows with missing release_year
data = data.dropna(subset=['release_year'])
```

```
# Count releases per year
releases_per_year = data['release_year'].dt.year.value_counts().sort_index()
```

```

# Plot the number of releases per year
plt.figure(figsize=(10, 6))
releases_per_year.plot(kind='line')
plt.title('Number of Releases Per Year')
plt.xlabel('Year')
plt.ylabel('Number of Releases')
plt.grid(True)
plt.show()

```



```

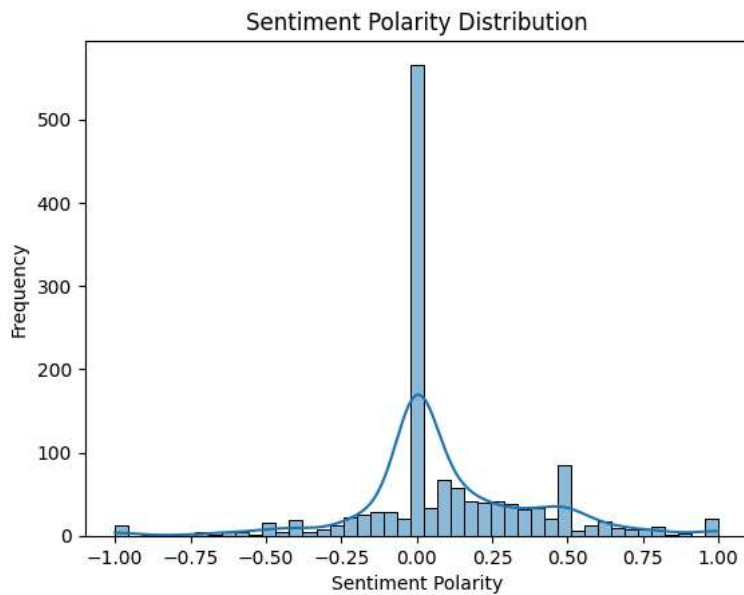
# Perform sentiment analysis on the 'description' column
data['description'] = data['description'].astype(str)

```

```
# Function to get sentiment
def get_sentiment(text):
    blob = TextBlob(text)
    return blob.sentiment.polarity, blob.sentiment.subjectivity

# Apply sentiment analysis
data['sentiment'] = data['description'].apply(lambda x: get_sentiment(x)[0])
data['subjectivity'] = data['description'].apply(lambda x: get_sentiment(x)[1])

# Plot sentiment distribution
sns.histplot(data['sentiment'], kde=True)
plt.title('Sentiment Polarity Distribution')
plt.xlabel('Sentiment Polarity')
plt.ylabel('Frequency')
plt.show()
```



Clustering

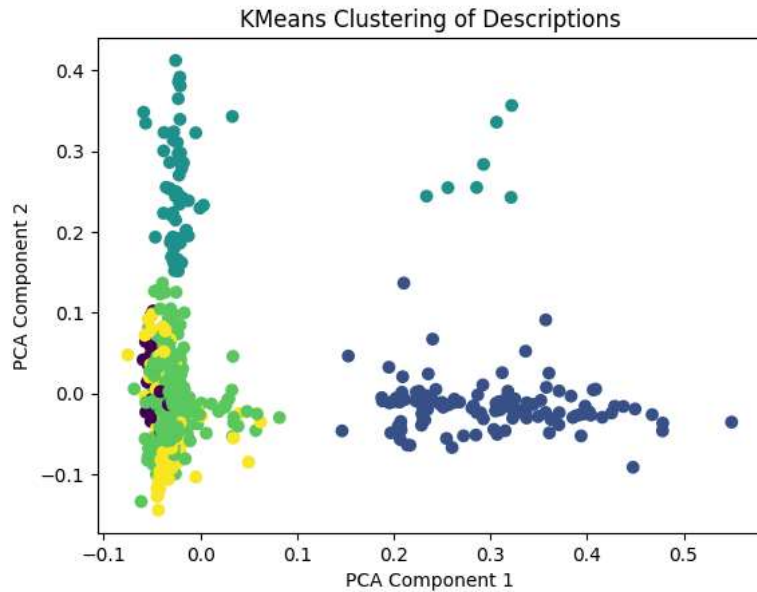
```
# Vectorize the 'description' column for clustering
vectorizer = TfidfVectorizer(stop_words='english')
X = vectorizer.fit_transform(data['description'])
```

```
# Apply KMeans clustering
kmeans = KMeans(n_clusters=5, random_state=42)
data['cluster'] = kmeans.fit_predict(X)
```

```
# Reduce dimensionality for visualization
pca = PCA(n_components=2, random_state=42)
X_pca = pca.fit_transform(X.toarray())
```

/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change from 10 to 1 in the future. Please set `n_init` to the desired value.

```
# Plot the clusters
plt.scatter(X_pca[:, 0], X_pca[:, 1], c=data['cluster'], cmap='viridis')
plt.title('KMeans Clustering of Descriptions')
plt.xlabel('PCA Component 1')
plt.ylabel('PCA Component 2')
plt.show()
```



```
# Display the first few rows and column names again to choose relevant features
print(data.head())
print(data.columns)
```



	show_id	type	title \
0	s1	Movie	A Spark Story
1	s2	Movie	Spooky Buddies
2	s3	Movie	The Fault in Our Stars
3	s4	TV Show	Dog: Impossible
4	s5	TV Show	Spidey And His Amazing Friends

	director \
0	Jason Sterman, Leanne Dare
1	Robert Vince
2	Josh Boone
3	NaN
4	NaN

	cast	country \
0	Aphthon Corbin, Louis Gonzales	NaN
1	Tucker Albrizzi, Diedrich Bader, Ameko Eks Mas...	United States, Canada
2	Shailene Woodley, Ansel Elgort, Laura Dern, Sa...	United States
3	Matt Beisner	United States
4	Benjamin Valic, Lily Sanfelippo, Jakari Fraser...	United States

	date_added	release_year	rating	duration \
0	September 24, 2021	2021-01-01	TV-PG	88 min
1	September 24, 2021	2011-01-01	G	93 min
2	September 24, 2021	2014-01-01	PG-13	127 min
3	September 22, 2021	2019-01-01	TV-PG	2 Seasons
4	September 22, 2021	2021-01-01	TV-Y	1 Season

	listed_in \
0	Documentary
1	Comedy, Fantasy, Kids
2	Coming of Age, Drama, Romance
3	Animals & Nature, Docuseries, Family
4	Action-Adventure, Animation, Kids

	description	sentiment	subjectivity \
0	Two Pixar filmmakers strive to bring their uni...	0.000	0.3
1	The puppies go on a spooky adventure through a...	0.000	1.0
2	Hazel and Gus share a love that sweeps them on...	0.650	0.8
3	Matt Beisner uses unique approaches to modifyi...	0.375	1.0
4	Spidey teams up with pals to become The Spidey...	0.000	0.0

	cluster
0	3
1	3
2	3
3	3
4	3

```
Index(['show_id', 'type', 'title', 'director', 'cast', 'country', 'date_added',
      'release_year', 'rating', 'duration', 'listed_in', 'description',
      'sentiment', 'subjectivity', 'cluster'],
      dtype='object', length=13)
```

```

dtype='object')

# Select relevant features for the pair plot
# Ensure 'release_year' is a numeric type for plotting
data['release_year'] = data['release_year'].dt.year

# Choose a subset of relevant columns for visualization
# Note: Modify column names based on actual dataset structure
selected_features = ['release_year', 'rating', 'cluster']

# Filter the DataFrame to include only selected features
data_selected = data[selected_features].dropna()

# Convert categorical data to numeric if necessary (e.g., rating)
# Assuming 'rating' is categorical, we can encode it numerically
data_selected['rating'] = data_selected['rating'].astype('category').cat.codes

# Create the pair plot
sns.pairplot(data_selected, hue='cluster', palette='viridis', diag_kind='kde')
plt.suptitle('Pair Plot of Selected Features Colored by Cluster', y=1.02)
plt.show()

```

