Apply advanced statistical and analytical methods to solve complex problems

```
import pandas as pd
```

```
data =  pd.read_csv("/content/disney_plus_titles.csv")
```

```
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
```

```
data.head()
```

	show_id	type	title	director	cast
0	s1	Movie	A Spark Story	Jason Sterman, Leanne Dare	Apthon Corbin, Louis Gonzales
1	s2	Movie	Spooky Buddies	Robert Vince	Tucker Albrizzi, Diedrich Bader, Ameko Eks Mas...
2	s3	Movie	The Fault in Our Stars	Josh Boone	Shailene Woodley, Ansel Elgort, Laura Dern, Sa...
3	s4	TV Show	Dog: Impossible	NaN	Matt Beisner
4	s5	TV Show	Spidey And His Amazing Friends	NaN	Benjamin Valic, Lily Sanfelippo, Jakari Fraser...

```
data.columns.values
```

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

```
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
```

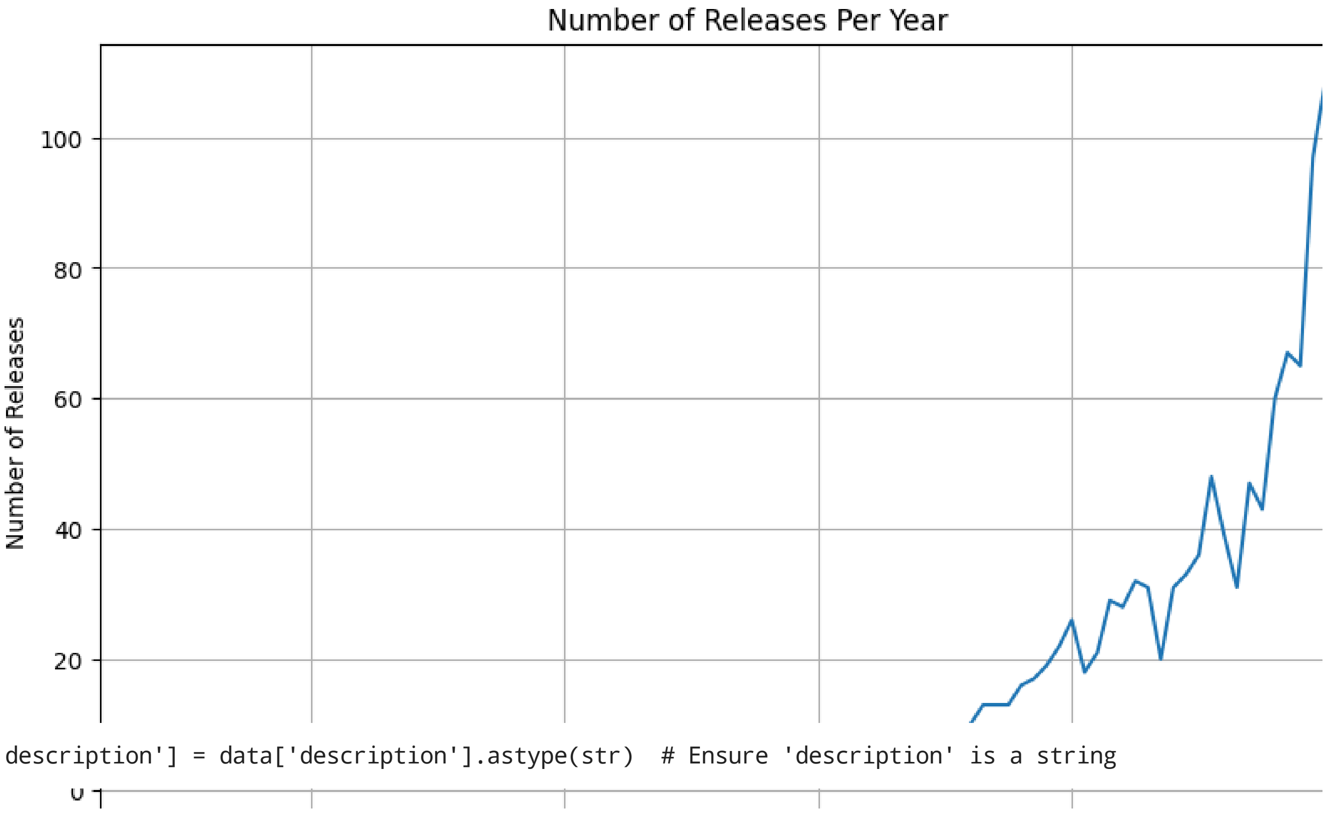
```
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
from textblob import TextBlob
```

```
data['release_year'] = pd.to_datetime(data['release_year'], format='%Y', errors='coerce')
```

```
data= data.dropna(subset=['release_year'])
```

```
releases_per_year = data['release_year'].dt.year.value_counts().sort_index()
```

```
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()
```

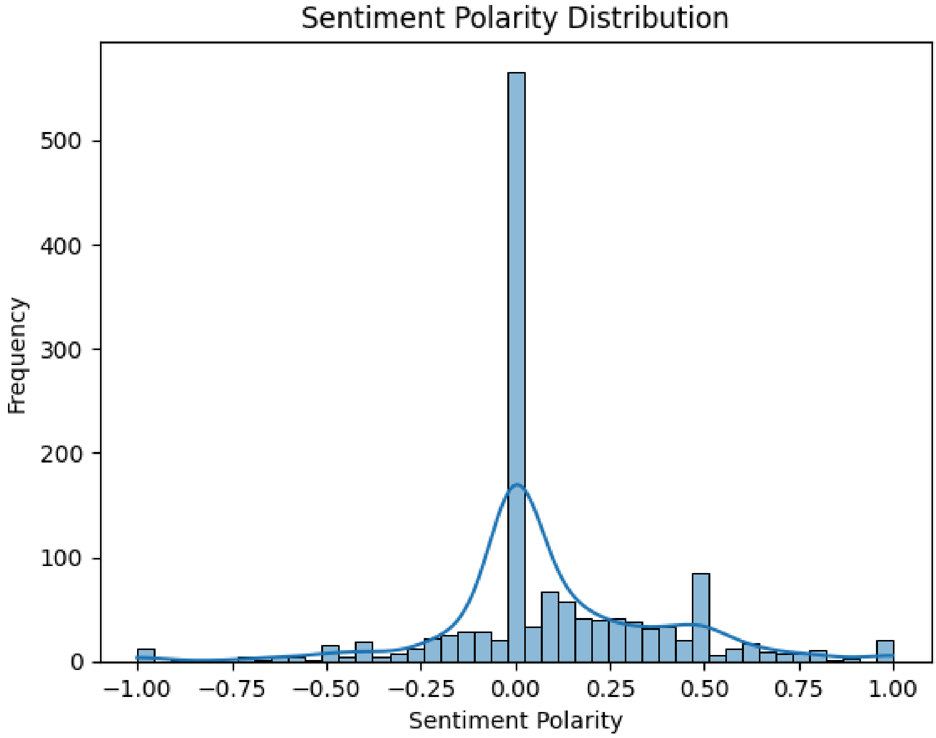


```
data['description'] = data['description'].astype(str)  # Ensure 'description' is a string

def get_sentiment(text):
    blob = TextBlob(text)
    return blob.sentiment.polarity, blob.sentiment.subjectivity


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

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



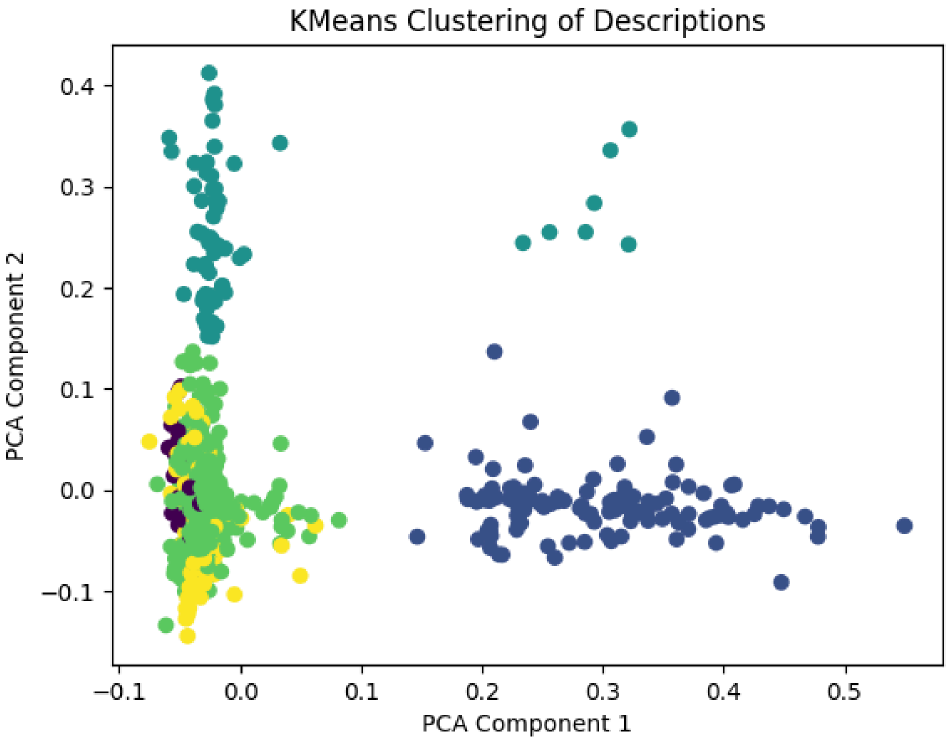
```
vectorizer = TfidfVectorizer(stop_words='english')
X = vectorizer.fit_transform(data['description'])


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

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
  warnings.warn(
```

```
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()
```


KMeans Clustering of Descriptions
PCA Component 2
PCA Component 1