ALL Necessary Libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import linear_kernel
from sklearn.model_selection import train_test_split
import nltk
import string
from wordcloud import WordCloud
```

```
nltk.download('stopwords')
from nltk.corpus import stopwords
stop_words = set(stopwords.words('english'))
```

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

loading the dataset

```
movies_df = pd.read_csv("movies.csv")
reviews_df = pd.read_csv("ratings.csv")
```

merging both the datasets

```
merged_df = pd.merge(movies_df, reviews_df, on="movieId", how="left")
avg_ratings = merged_df.groupby("movieId")["rating"].mean().reset_index()
avg_ratings.rename(columns={"rating": "avg_rating"}, inplace=True)
movies_df = pd.merge(movies_df, avg_ratings, on="movieId", how="left")
movies_df["avg_rating"].fillna(movies_df["avg_rating"].median(), inplace=True)
movies_df["combined_features"] = movies_df["title"] + " " + movies_df["genres"]
```

```
<ipython-input-5-401d5c9dbb9d>:5: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assi
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col

  movies_df["avg_rating"].fillna(movies_df["avg_rating"].median(), inplace=True)
```

Checking Basic Information

```
print("INFO:\n", merged_df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20825289 entries, 0 to 20825288
Data columns (total 6 columns):
 #   Column     Dtype
---  ------     -----
 0   movieId    int64
 1   title      object
 2   genres     object
 3   userId     float64
 4   rating     float64
 5   timestamp  float64
dtypes: float64(3), int64(1), object(2)
memory usage: 953.3+ MB
INFO:
 None
```

```
print("\nDESCRIBE:\n", merged_df.describe())
```

```

DESCRIBE:
             movieId        userId        rating     timestamp
count  2.082529e+07  2.082069e+07  2.082069e+07  2.082069e+07
mean   2.143557e+04  6.760068e+04  3.532698e+00  1.215605e+09
std    3.929864e+04  3.888665e+04  1.061995e+00  2.267272e+08
min    1.000000e+00  1.000000e+00  5.000000e-01  7.896520e+08
```

```
25%    1.196000e+03  3.382000e+04  3.000000e+00  1.012920e+09
50%    2.947000e+03  6.752300e+04  3.500000e+00  1.198221e+09
75%    8.633000e+03  1.010980e+05  4.000000e+00  1.447191e+09
max    2.091710e+05  1.354200e+05  5.000000e+00  1.574328e+09
```

```
print("\nHEAD:\n", merged_df.head())
```

```
HEAD:
    movieId             title                                       genres  \
0        1  Toy Story (1995)  Adventure|Animation|Children|Comedy|Fantasy
1        1  Toy Story (1995)  Adventure|Animation|Children|Comedy|Fantasy
2        1  Toy Story (1995)  Adventure|Animation|Children|Comedy|Fantasy
3        1  Toy Story (1995)  Adventure|Animation|Children|Comedy|Fantasy
4        1  Toy Story (1995)  Adventure|Animation|Children|Comedy|Fantasy

   userId  rating     timestamp
0     2.0     3.5  1.141416e+09
1     3.0     4.0  1.439472e+09
2     4.0     3.0  1.573944e+09
3     5.0     4.0  8.586259e+08
4     8.0     4.0  8.904925e+08
```

Checking Missing Values

```
print("\nNULL VALUES:\n", merged_df.isnull().sum())
```

```
NULL VALUES:
 movieId         0
title           0
genres          0
userId       4599
rating       4599
timestamp    4599
dtype: int64
```

Cleaning stopwords

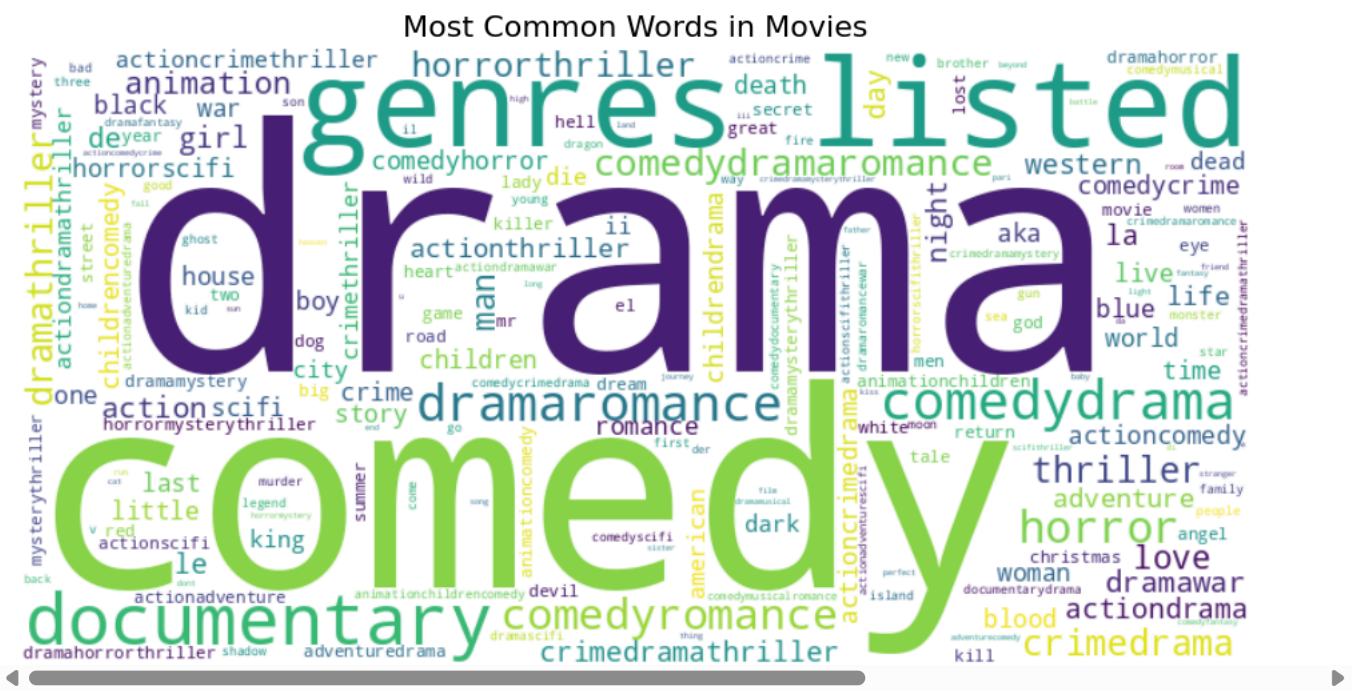
```
def clean_text(text):
    if isinstance(text, str):
        text = text.lower()
        text = text.translate(str.maketrans('', '', string.punctuation))
        words = text.split()
        words = [word for word in words if word not in stop_words]
        return ' '.join(words)
    return ""


movies_df["clean_features"] = movies_df["combined_features"].apply(clean_text)
```

Anaysis

```
movies_subset = movies_df.sample(frac=0.4, random_state=42)
tfidf = TfidfVectorizer(max_features=5000)
tfidf_matrix = tfidf.fit_transform(movies_subset["clean_features"])


cosine_sim = linear_kernel(tfidf_matrix, tfidf_matrix)
```

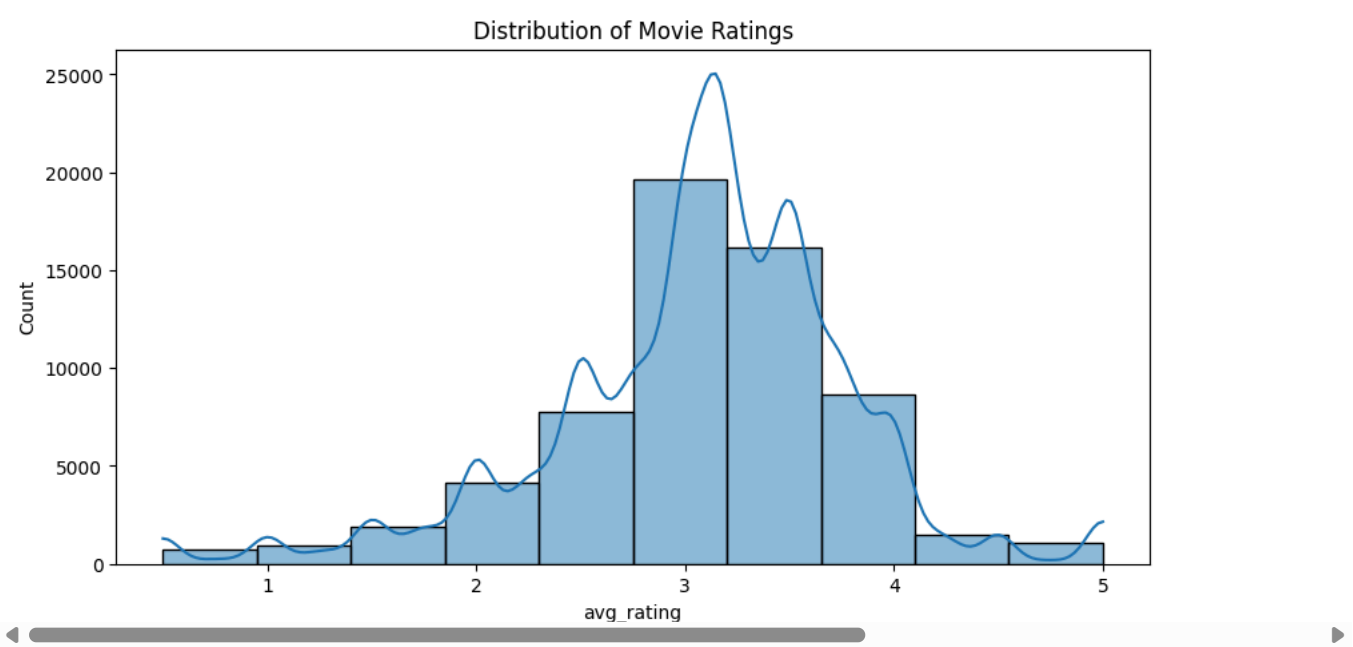
Word Cloud

```
all_words = ' '.join(movies_df["clean_features"].dropna())
wordcloud = WordCloud(width=800, height=400, background_color='white').generate(all_words)


plt.figure(figsize=(12, 6))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title("Most Common Words in Movies", fontsize=16)
plt.show()
```



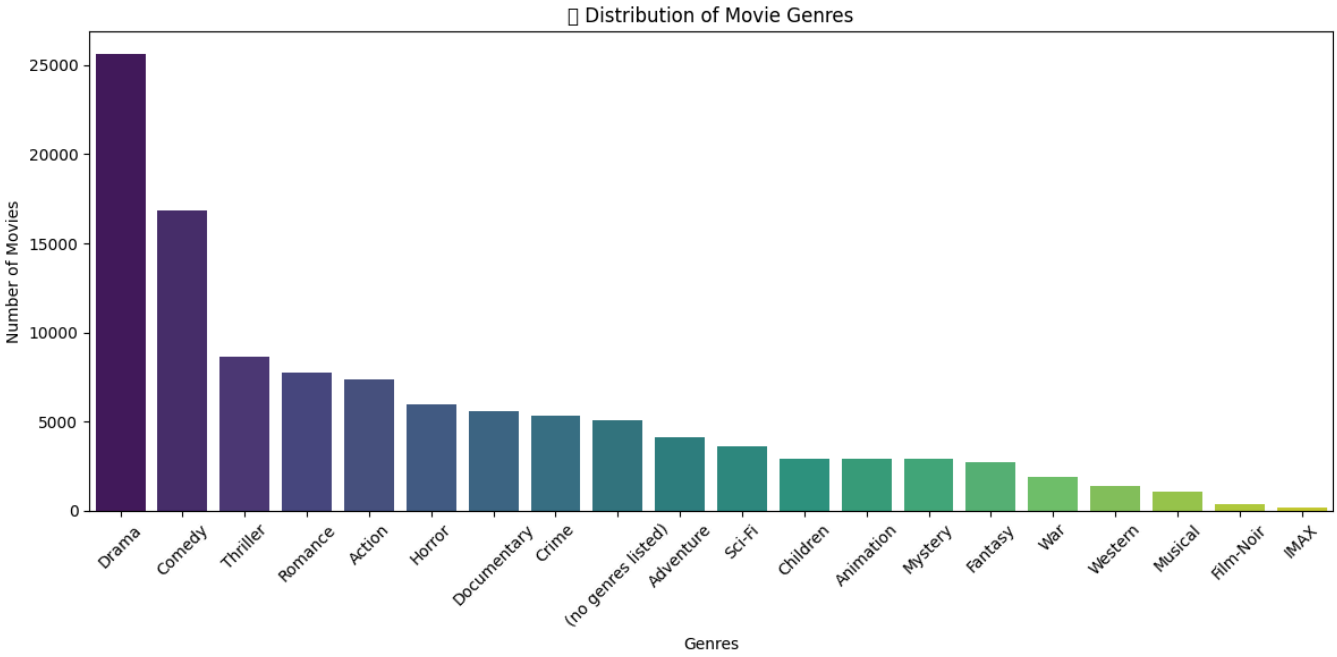
Visualization

```
plt.figure(figsize=(10, 5))
sns.histplot(movies_df["avg_rating"], bins=10, kde=True)
plt.title("Distribution of Movie Ratings")
plt.show()
```



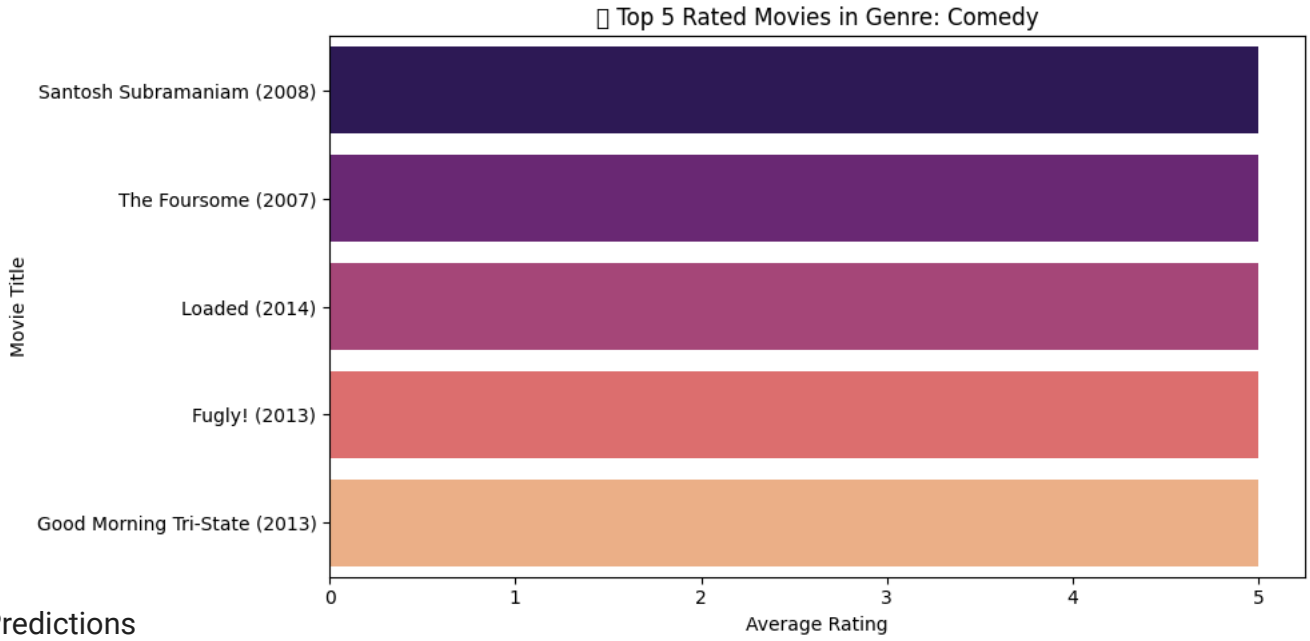

```
def plot_genre_distribution(movies_df):
    genre_list = movies_df['genres'].str.split('|').explode()
    genre_counts = genre_list.value_counts()

    plt.figure(figsize=(12, 6))
    sns.barplot(x=genre_counts.index, y=genre_counts.values, palette='viridis')
    plt.xticks(rotation=45)
    plt.title("📊 Distribution of Movie Genres")
    plt.xlabel("Genres")
    plt.ylabel("Number of Movies")
    plt.tight_layout()
    plt.show()

plot_genre_distribution(movies_df)
```

```
<ipython-input-17-d258d83cc909>:6: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le

  sns.barplot(x=genre_counts.index, y=genre_counts.values, palette='viridis')
<ipython-input-17-d258d83cc909>:11: UserWarning: Glyph 128202 (\N{BAR CHART}) missing from font(s) DejaVu Sans.
  plt.tight_layout()
/usr/local/lib/python3.11/dist-packages/IPython/core/pylabtools.py:151: UserWarning: Glyph 128202 (\N{BAR CHART}) missing from font
  fig.canvas.print_figure(bytes_io, **kw)
```




```
def plot_top_movies_in_genre(movies_df, genre, top_n=10):
    genre = genre.lower()
    genre_matches = movies_df[movies_df['genres'].str.lower().str.contains(genre)]
    if genre_matches.empty:
        print(f"No movies found for genre: {genre}")
        return
    top_movies = genre_matches.sort_values(by='avg_rating', ascending=False).head(top_n)

    plt.figure(figsize=(10, 5))
    sns.barplot(x='avg_rating', y='title', data=top_movies, palette='magma')
    plt.title(f"🎬 Top {top_n} Rated Movies in Genre: {genre.title()}")
    plt.xlabel("Average Rating")
    plt.ylabel("Movie Title")
    plt.tight_layout()
    plt.show()

genre = "comedy"
top_n = 5
plot_top_movies_in_genre(movies_df, genre, top_n=top_n)
```

```
<ipython-input-18-7df91d03321f>:10: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `le

  sns.barplot(x='avg_rating', y='title', data=top_movies, palette='magma')
<ipython-input-18-7df91d03321f>:14: UserWarning: Glyph 127916 (\N{CLAPPER BOARD}) missing from font(s) DejaVu Sans.
  plt.tight_layout()
/usr/local/lib/python3.11/dist-packages/IPython/core/pylabtools.py:151: UserWarning: Glyph 127916 (\N{CLAPPER BOARD}) missing from f
  fig.canvas.print_figure(bytes_io, **kw)
```



Predictions

```
def recommend_by_genre(genre_query, top_n=10):
    genre_query = genre_query.lower()
    genre_matches = movies_subset[movies_subset['genres'].str.lower().str.contains(genre_query)]
    if genre_matches.empty:
        print(f"No movies found for genre: {genre_query}")
        return []
    top_movies = genre_matches.sort_values(by='avg_rating', ascending=False).head(top_n)
    return top_movies[['title', 'genres', 'avg_rating']]

genre = "comedy"
top_n = 5
genre_results = recommend_by_genre(genre, top_n=top_n)

print(f"\n🎬 Top {top_n} Recommended Movies in Genre '{genre.title()}':\n")
print(genre_results)
```

```
🎬 Top 5 Recommended Movies in Genre 'Comedy':

                                  title        genres  avg_rating
26106         That’s What She Said (2012)        Comedy         5.0
38961         Dance With Me, Henry (1956)        Comedy         5.0
31717           The Movie Out Here (2012)        Comedy         5.0
55930                 High Fantasy (2017)  Comedy|Drama         5.0
44959  Brad Williams: Daddy Issues (2016)        Comedy         5.0
```