EDS Activity = 1

Name:Sanjana Santosh Patane

Div:ET1

Roll no:ET1-73

PRN:2024010700196

Load the IMDB dataset using Pandas And Numpy

Solution:

```
import pandas as pd
import numpy as np(for numpy only)
df = pd.read_csv('imdb_top_1000.csv')
# Display the first few rows
print(df.head())
print(df.columns)
```

Problem1

Find the top 5 highest-rated movies.

Solution:

```
top_5 = df.sort_values(by='rating',
ascending=False).head(5)
```

Problem 2:

Count the number of movies in each genre.

Solution:

```
genre_counts = df['genre'].value_counts()
```

Problem 3:

Find the average rating for each director.

Solution:

```
avg_rating_director =
df.groupby('director')['rating'].mean()
```

Problem 4:

List movies released after 2015 with a rating above 8.0.

Solution:

```
high_rated_recent = df[(df['year'] > 2015) &
(df['rating'] > 8.0)]
```

Problem 5:

Find the director who has made the most movies.

Solution:

```
top_director = df['director'].value_counts().idxmax()
```

Ploblem 6:

Compute the average duration of all movies.

Solution:

```
avg_duration = df['duration'].mean()
```

Problem 7:

Identify the movie with the maximum revenue.

Solution:

```
max_revenue_movie = df.loc[df['revenue'].idxmax()]
```

Problem 8:

Calculate the correlation between rating and revenue.

Solution:

```
correlation = df['rating'].corr(df['revenue'])
```

Problem 9:

List all movies with missing metascore.

Solution:

```
missing_metascore = df[df['metascore'].isnull()]
```

Problrm 10:

Replace missing metascore values with the average.

Solution:

```
df['metascore'].fillna(df['metascore'].mean(),
inplace=True)
```

Problem 11:

Determine the number of unique directors.

Solution:

```
unique_directors = df['director'].nunique()
```

Problem 12:

Find the median number of votes per movie.

Solution:

```
median_votes = df['votes'].median()
```

Problem 13:

Identify the genre with the highest average rating.

Solution:

```
genre_rating =
df.groupby('genre')['rating'].mean().idxmax()
```

Problem 14:

List the shortest and longest movies.

Solution:

```
shortest_movie = df.loc[df['duration'].idxmin()]
longest_movie = df.loc[df['duration'].idxmax()]
```

Problem 15:

Find the year with the most movie releases.

Solution:

```
most_releases_year =
df['year'].value_counts().idxmax()
```

Problem 16:

Filter movies released after 2010.

Solution:

```
recent_movies =
df.to_numpy()[df['year'].to_numpy() > 2010]
```

Problem 17:
Count how many movies are longer than 120 minutes.

Solution:

```
long_movies_count =
np.sum(df['duration'].to_numpy() > 120)
```

Problem 18:

Calculate the rating-to-votes ratio for each movie.

Solution:

```
ratings = df['rating'].to_numpy()
votes = df['votes'].to_numpy()
rating_per_vote = np.divide(ratings, votes,
out=np.zeros_like(ratings), where=votes!=0)
```

Problem 19:

Show movies that belong to multiple genres (e.g., 'Action, Comedy’)

Solution:

```
multi_genre_movies = df[df['genre'].str.contains(',')]
```

Problem 20:

Add a new column for revenue per vote.

Solution:

```
df['revenue_per_vote'] = df['revenue'] / df['votes']
```