ESSENTIALS OF DATA SCIENCE

Theory Activity No. 1

Formulate 20 problem statements for a given dataset using Numpy and Pandas and Apply Numpy and pandas methods to find the solution for the formulated problem statements.

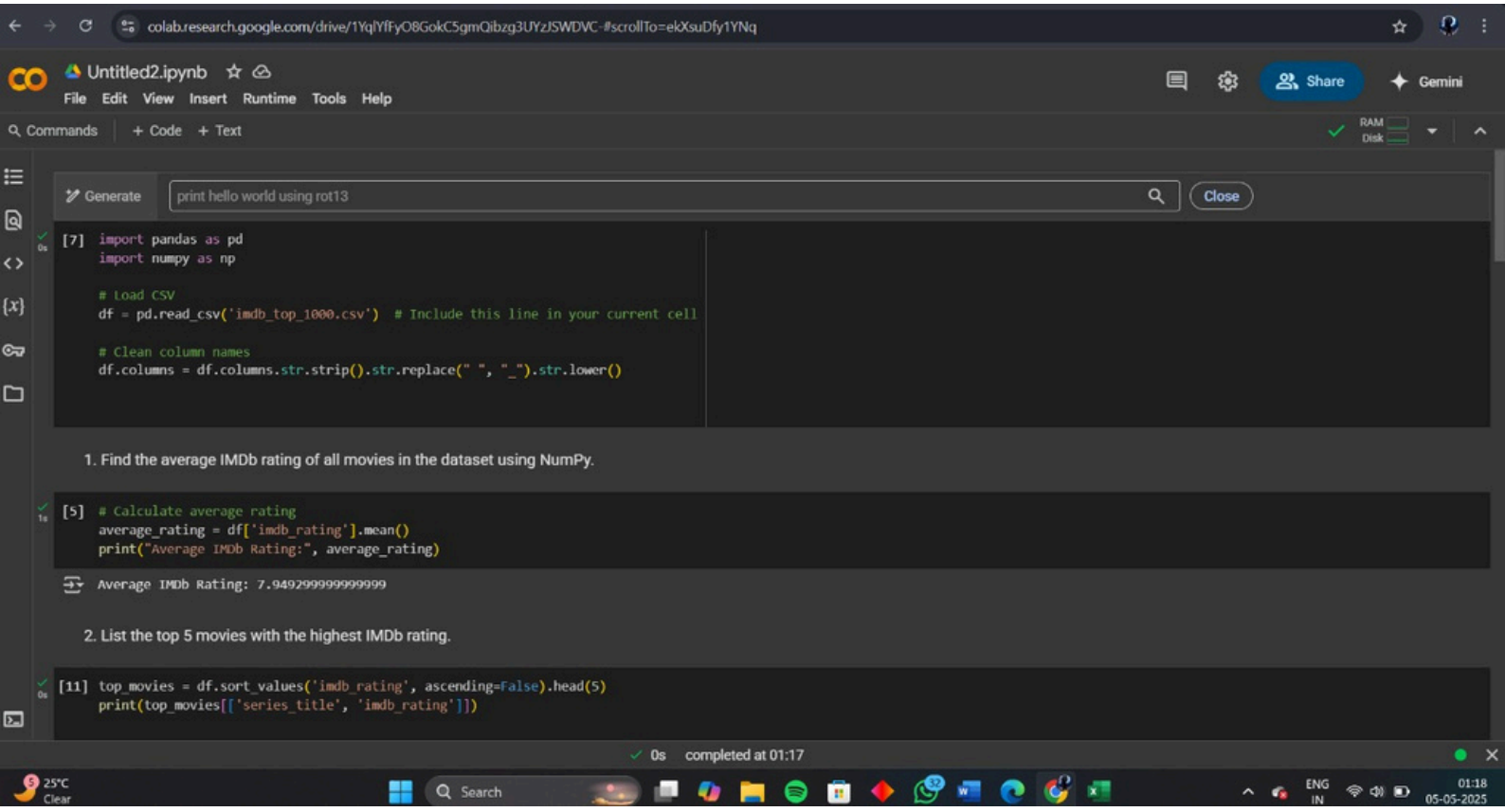
Dataset Name - IMDB Dataset

Name - Atharva Kadu

Division - CS2

Batch - C24 **Roll No** - 73

PRN - 202401040336



```
[7] df.columns = df.columns.str.strip().str.replace(" ", "_").str.lower()
```

1. Find the average IMDb rating of all movies in the dataset using NumPy.

```
# Calculate average rating
average_rating = df['imdb_rating'].mean()
print("Average IMDb Rating:", average_rating)
```

```
Average IMDb Rating: 7.949299999999999
```

2. List the top 5 movies with the highest IMDb rating.

```
[11] top_movies = df.sort_values('imdb_rating', ascending=False).head(5)
     print(top_movies[['series_title', 'imdb_rating']])
```

```
              series_title  imdb_rating
0  The Shawshank Redemption          9.3
1             The Godfather          9.2
4              12 Angry Men          9.0
2           The Dark Knight          9.0
3    The Godfather: Part II          9.0
```

colab.research.google.com/drive/1YqlYfFyO8GokC5gmQibzg3UYzJSWDVC-#scrollTo=ekXsuDfy1YNq

3. Find the average IMDb rating grouped by film certificate (like A, U, UA).

```
avg_rating_by_cert = df.groupby('certificate')['imdb_rating'].mean()
print(avg_rating_by_cert)
```

```
certificate
16          8.100000
A           7.998985
Approved    7.945455
G           8.000000
GP          7.850000
PG          7.927027
PG-13       7.797674
Passed      8.020588
R           7.869863
TV-14       8.300000
TV-MA       8.100000
TV-PG       7.900000
U           7.976923
U/A         7.600000
UA          7.957143
Unrated     8.100000
Name: imdb_rating, dtype: float64
```

colab.research.google.com/drive/1YqlYfFyO8GokC5gmQibzg3UYzJSWDVC-#scrollTo=ekXsuDfy1YNq

Untitled2.ipynb

File Edit View Insert Runtime Tools Help

Share Gemini

Commands + Code + Text

RAM Disk

4. Count how many movies were directed by Christopher Nolan.

```
[13] nolan_count = df[df['director'] == 'christopher nolan'].shape[0]
     print("Christopher Nolan Movies:", nolan_count)
```

```
Christopher Nolan Movies: 0
```

5. Find the most frequent actor listed as the lead star (Star1)

```
[14] top_star = df['star1'].value_counts().idxmax()
     print("Most frequent lead actor:", top_star)
```

```
Most frequent lead actor: Tom Hanks
```

6. List all movies whose titles contain the word "Godfather".

```
[21] godfather_movies = df[df['series_title'].str.contains("Godfather", case=False)]
     print(godfather_movies[['series_title']])
```

```
                series_title
1             The Godfather
3     The Godfather: Part II
974  The Godfather: Part III
```

0s completed at 01:17

25°C Clear | Search | ENG IN | 01:19 05-05-2025

Untitled2.ipynb

File Edit View Insert Runtime Tools Help

Share Gemini

Commands + Code + Text

RAM Disk

7. Count how many movies are missing a meta score.

```
[22] missing_meta = df['meta_score'].isna().sum()
     print("Missing Meta Scores:", missing_meta)
```

```
Missing Meta Scores: 157
```

8. Identify the top 3 directors who have directed the most movies.

```
[23] top_directors = df['director'].value_counts().head(3)
     print("Top 3 Directors:\n", top_directors)
```

```
Top 3 Directors:
 director
Alfred Hitchcock    14
Steven Spielberg    13
Hayao Miyazaki      11
Name: count, dtype: int64
```

9. Find the number of movies directed by Francis Ford Coppola

```
coppola_count = df[df['director'] == 'Francis Ford Coppola'].shape[0]
print("Movies by Francis Ford Coppola:", coppola_count)
```

```
Movies by Francis Ford Coppola: 5
```

0s completed at 01:17

25°C Clear | Search | ENG IN | 01:20 05-05-2025

Untitled2.ipynb

File Edit View Insert Runtime Tools Help

Share Gemini

Commands + Code + Text

RAM Disk

```
Hayao Miyazaki      11
Name: count, dtype: int64
```

9. Find the number of movies directed by Francis Ford Coppola

```
[31] coppola_count = df[df['director'] == 'Francis Ford Coppola'].shape[0]
     print("Movies by Francis Ford Coppola:", coppola_count)
```

```
Movies by Francis Ford Coppola: 5
```

10. Find the top 5 actors (from Star1) with the most movie appearances.

```
top_actors = df['star1'].value_counts().head(5)
print("Top 5 Actors by Appearances:\n", top_actors)
```

```
Top 5 Actors by Appearances:
 star1
Tom Hanks           12
Robert De Niro      11
Al Pacino           10
Clint Eastwood      10
Humphrey Bogart      9
Name: count, dtype: int64
```

0s completed at 01:17

25°C Clear | Search | ENG IN | 01:20 05-05-2025

Untitled2.ipynb

File Edit View Insert Runtime Tools Help

Share Gemini

Commands + Code + Text

RAM Disk

11. Find the difference between the highest and lowest Meta score.

```
[40] meta_diff = np.nanmax(df['meta_score'].values) - np.nanmin(df['meta_score'].values)
     print("Meta Score Range:", meta_diff)
```

```
Meta Score Range: 72.0
```

12. Calculate the total number of votes received by movies released in or after 2010.

```
[46] votes_2010_onward = np.sum(df[df['released_year'] >= 2010]['no_of_votes'].values)
     print("Total Votes (>= 2010):", votes_2010_onward)
```

```
Total Votes (>= 2010): 76885377
```

13. Find the percentage of movies with IMDb rating below the dataset's median rating.

```
[47] median_rating = np.median(df['imdb_rating'].values)
     below_median = np.sum(df['imdb_rating'].values < median_rating)
     percentage = (below_median / len(df)) * 100
     print("Movies Below Median Rating: {:.2f}%".format(percentage))
```

```
Movies Below Median Rating: 43.10%
```

0s completed at 01:39

25°C Clear | Search | ENG IN | 01:40 05-05-2025

Untitled2.ipynb

File Edit View Insert Runtime Tools Help

Share Gemini

Commands + Code + Text

RAM Disk

14. Calculate the average number of votes for movies with a Meta score above 75.

```
[48] votes_above_75 = np.mean(df[df['meta_score'] > 75]['no_of_votes'].values)
     print("Avg Votes (Meta Score > 75):", votes_above_75)
```

```
Avg Votes (Meta Score > 75): 301528.29766536964
```

15. Find the index of the movie with the lowest IMDb rating.

```
min_rating_idx = np.argmin(df['imdb_rating'].values)
print("Movie with Lowest Rating:", df.iloc[min_rating_idx]['series_title'])
```

```
Movie with Lowest Rating: Dark Waters
```

16. Check how many movies have both rating > 8.0 and Meta score > 80.

```
[52] condition = (df['imdb_rating'].values > 8.0) & (df['meta_score'].values > 80)
     count = np.sum(condition)
     print("Movies with Rating > 8.0 & Meta Score > 80:", count)
```

```
Movies with Rating > 8.0 & Meta Score > 80: 134
```

17. Identify the IMDb rating of the oldest movie in the dataset.

0s completed at 01:39

25°C Clear Search ENG IN 01:41 05-05-2025

Untitled2.ipynb

File Edit View Insert Runtime Tools Help

Share Gemini

Commands + Code + Text

RAM Disk

17. Identify the IMDb rating of the oldest movie in the dataset.

```
[56] oldest_index = np.argmin(df['released_year'].values)
     oldest_rating = df.iloc[oldest_index]['imdb_rating']
     print("Rating of Oldest Movie:", oldest_rating)
```

```
Rating of Oldest Movie: 7.6
```

18. Count movies that have both Gross and Meta score missing

```
[58] missing_both = np.sum(df['gross'].isna().values & df['meta_score'].isna().values)
     print("Movies Missing Both Gross and Meta Score:", missing_both)
```

```
Movies Missing Both Gross and Meta Score: 76
```

19. Find the proportion of movies released after 2015 with an IMDb rating above 8.0.

```
[59] recent = df['released_year'].values > 2015
     high_rated_recent = np.sum((df['imdb_rating'].values > 8.0) & recent)
     proportion = high_rated_recent / np.sum(recent)
     print("Proportion of High Rated Movies After 2015:", proportion)
```

```
Proportion of High Rated Movies After 2015: 0.336734693877551
```

0s completed at 01:39

24°C Clear Search ENG IN 01:41 05-05-2025

Untitled2.ipynb

File Edit View Insert Runtime Tools Help

Share Gemini

Commands + Code + Text

RAM Disk

```
Movies Missing Both Gross and Meta Score: 76
```

19. Find the proportion of movies released after 2015 with an IMDb rating above 8.0.

```
[59] recent = df['released_year'].values > 2015
     high_rated_recent = np.sum((df['imdb_rating'].values > 8.0) & recent)
     proportion = high_rated_recent / np.sum(recent)
     print("Proportion of High Rated Movies After 2015:", proportion)
```

```
Proportion of High Rated Movies After 2015: 0.336734693877551
```

20. Determine the maximum number of votes received by any movie in the dataset.

```
[60] max_votes = np.max(df['no_of_votes'].values)
     print("Maximum Number of Votes:", max_votes)
```

```
Maximum Number of Votes: 2343110
```

0s completed at 01:39

24°C Clear

Search

ENG IN

01:42 05-05-2025