MOVIE RECOMMENDATION SYSTEM

I'm building a baseline Movie Recommendation System using the TMDB 5000 Movie Dataset.

Importing Libraries

In [4]:
```
# Importing Libraries

import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
```

Loading the datasets

Here, let's load the TMDB 5000 Movie Dataset.

In [5]:
```
# Loading the datasets

df1=pd.read_csv('./tmdb_5000_credits.csv')
df2=pd.read_csv('./tmdb_5000_movies.csv')
```

In [6]:
```
df1.head(5)
```

Out[6]:

	movie_id	title	cast	crew
0	19995	Avatar	[{"cast_id": 242, "character": "Jake Sully", "...	[{"credit_id": "52fe48009251416c750aca23", "de...
1	285	Pirates of the Caribbean: At World's End	[{"cast_id": 4, "character": "Captain Jack Spa...	[{"credit_id": "52fe4232c3a36847f800b579", "de...
2	206647	Spectre	[{"cast_id": 1, "character": "James Bond", "cr...	[{"credit_id": "54805967c3a36829b5002c41", "de...
3	49026	The Dark Knight Rises	[{"cast_id": 2, "character": "Bruce Wayne / Ba...	[{"credit_id": "52fe4781c3a36847f81398c3", "de...
4	49529	John Carter	[{"cast_id": 5, "character": "John Carter", "c...	[{"credit_id": "52fe479ac3a36847f813eaa3", "de...

In [7]:

```
df2.head(3)
```

Out[7]:

	budget	genres	homepage	id	keywords	original_language	original_title	overview	popularity	produc
0	237000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://www.avatarmovie.com/	19995	[{"id": 1463, "name": "culture clash"}, {"id":...	en	Avatar	In the 22nd century, a paraplegic Marine is di...	150.437577	[{"n Fil
1	300000000	[{"id": 12, "name": "Adventure"}, {"id": 14, "...	http://disney.go.com/disneypictures/pirates/	285	[{"id": 270, "name": "ocean"}, {"id": 726, "na...	en	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	139.082615	[{"nar Pict
2	245000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://www.sonypictures.com/movies/spectre/	206647	[{"id": 470, "name": "spy"}, {"id": 818, "name...	en	Spectre	A cryptic message from Bond's past sends him o...	107.376788	[{"r

---

Merging the datasets into one

Now, I'll join the 2 datasets on the 'id' column.

In [8]:
```
df1.columns = ['id','ttitle','cast','crew']
df2 = df2.merge(df1, on='id')
```

In [9]:
```
df2.head(5)
```

Out[9]:

	budget	genres	homepage	id	keywords	original_language	original_title	overview	popularity	produc
0	237000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://www.avatarmovie.com/	19995	[{"id": 1463, "name": "culture clash"}, {"id":...	en	Avatar	In the 22nd century, a paraplegic Marine is di...	150.437577	[{"n Fil
1	300000000	[{"id": 12, "name": "Adventure"}, {"id": 14, "...	http://disney.go.com/disneypictures/pirates/	285	[{"id": 270, "name": "ocean"}, {"id": 726, "na...	en	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	139.082615	[{"nar Pict
2	245000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://www.sonypictures.com/movies/spectre/	206647	[{"id": 470, "name": "spy"}, {"id": 818, "name...	en	Spectre	A cryptic message from Bond's past sends him o...	107.376788	[{"r
3	250000000	[{"id": 28, "name": "Action"}, {"id": 80, "nam...	http://www.thedarkknightrises.com/	49026	[{"id": 849, "name": "dc comics"}, {"id": 853,...	en	The Dark Knight Rises	Following the death of District Attorney Harve...	112.312950	[{"na Pictu
4	260000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://movies.disney.com/john-carter	49529	[{"id": 818, "name": "based on novel"}, {"id":...	en	John Carter	John Carter is a war-weary, former military ca...	43.926995	[{"nar

5 rows × 23 columns

Demographic Filtering

Out of the three recommender systems, I'm implementing Demographic Filtering in this project.

So, I'll be using IMDB's Weighted Rating :-

formula pic

where

- **v** is the number of votes for the movie
- **m** is the minimum votes required to be listed in the chart
- **R** is the average rating of the movie
- **C** is the mean vote across the whole report

Since we already have v (vote_count) and R (vote_average), we'll calculate C and m.

In [10]:

```
C = df2['vote_average'].mean()
C
```

Out[10]:

```
6.092171559442011
```

So, the mean rating for all the movies is approximately 6 on a scale of 10. Now, I'll determine the value of **m**, the minimum votes required to be listed in the chart. Let's use 90th percentile as the cutoff (a movie must have more votes than at least 90% of the movies in the list to feature in the charts).

In [11]:

```
m = df2['vote_count'].quantile(0.9)
m
```

Out[11]:

```
1838.4000000000015
```

A movie must have more than 1838 votes to feature in the charts.

Now, we'll see how many movies qualify.

In [12]:
```
s_movies = df2.copy().loc[df2['vote_count'] >= m]
s_movies.shape
```

Out[12]:
```
(481, 23)
```

481 movies qualify to be in the list.

Weighted Rating function

Now, I'll create a function which will calculate the score based on the Weighted Rating formula that we are using.

In [13]:
```
def weighted_rating(x, m=m , C=C):
    v = x['vote_count']
    R = x['vote_average']

    # Calculation based on IMDB formula
    return (v/(v+m) * R) + (m/(m+v) * C)
```

Define a new feature 'calc_score' and calculate its value with the function 'weighted_rating()'.

In [14]:
```
s_movies['calc_score'] = s_movies.apply(weighted_rating, axis=1)
```

Now, we'll sort the DataFrame based on the new calculated score and output the title, vote count, vote average, and weighted rating (calc_score) of the movies.

In [27]:
```
# Now, sort movies based on calculated score above

s_movies = s_movies.sort_values('calc_score',ascending=False)

s_movies[['title','vote_count','vote_average','calc_score']]
```

Out[27]:

	title	vote_count	vote_average	calc_score
1881	The Shawshank Redemption	8205	8.5	8.059258
662	Fight Club	9413	8.3	7.939256
65	The Dark Knight	12002	8.2	7.920020
3232	Pulp Fiction	8428	8.3	7.904645
96	Inception	13752	8.1	7.863239
...	...	...	...	...
41	Green Lantern	2487	5.1	5.521697
337	A Good Day to Die Hard	3493	5.2	5.507643
193	After Earth	2532	5.0	5.459420
91	Independence Day: Resurgence	2491	4.9	5.406234
242	Fantastic Four	2278	4.4	5.155730

481 rows × 4 columns

TOP 10 MOVIES

In [29]:

```
# The TOP 10 movies

s_movies[['title','vote_count','vote_average','calc_score']].head(10)
```

Out[29]:

	title	vote_count	vote_average	calc_score
1881	The Shawshank Redemption	8205	8.5	8.059258
662	Fight Club	9413	8.3	7.939256
65	The Dark Knight	12002	8.2	7.920020
3232	Pulp Fiction	8428	8.3	7.904645
96	Inception	13752	8.1	7.863239
3337	The Godfather	5893	8.4	7.851236
95	Interstellar	10867	8.1	7.809479
809	Forrest Gump	7927	8.2	7.803188
329	The Lord of the Rings: The Return of the King	8064	8.1	7.727243
1990	The Empire Strikes Back	5879	8.2	7.697884

MOST POPULAR MOVIES

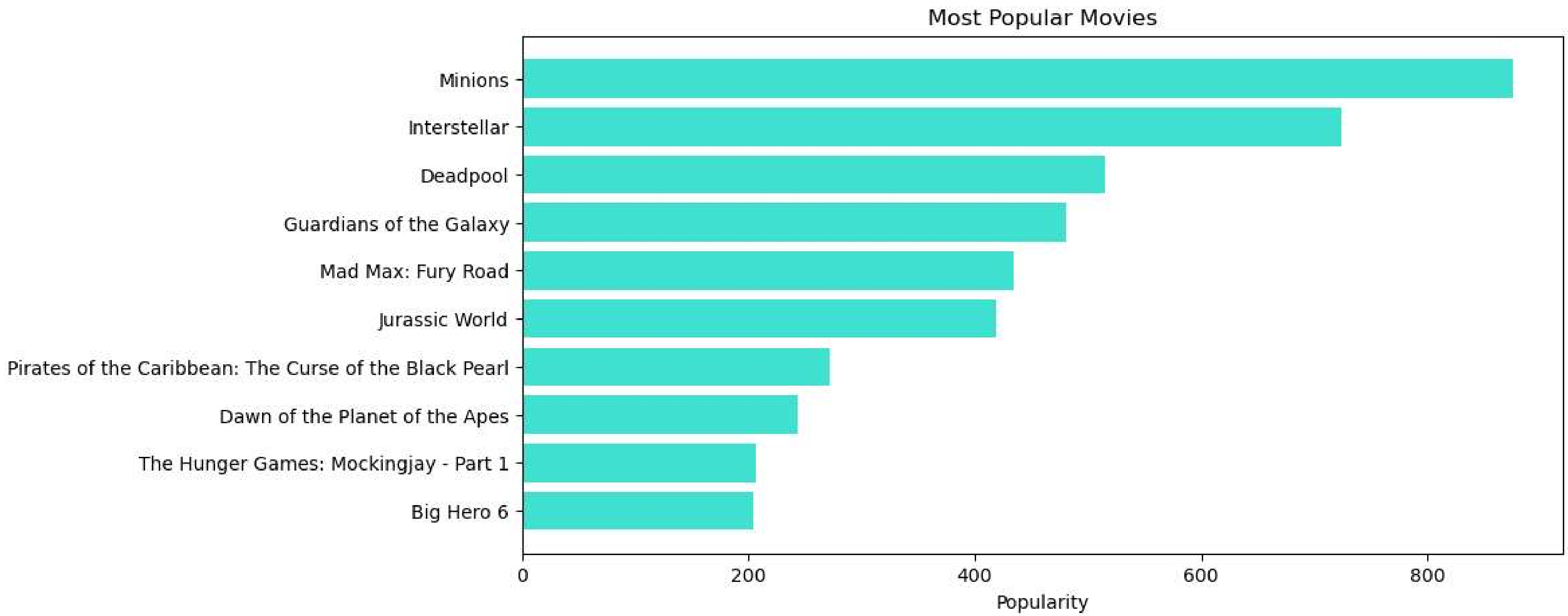
In [77]:

```
pop= df2.sort_values('popularity', ascending=False)
import matplotlib.pyplot as plt
plt.figure(figsize=(10,5))

plt.barh(pop['title'].head(10),pop['popularity'].head(10)    ,     color='turquoise')
plt.gca().invert_yaxis()
plt.xlabel("Popularity")
plt.title("Most Popular Movies")
```

Out[77]:

```
Text(0.5, 1.0, 'Most Popular Movies')
```


Most Popular Movies
Minions
Interstellar
Deadpool
Guardians of the Galaxy
Mad Max: Fury Road
Jurassic World
Pirates of the Caribbean: The Curse of the Black Pearl
Dawn of the Planet of the Apes
The Hunger Games: Mockingjay - Part 1
Big Hero 6
0
200
400
600
800
Popularity