python™

2025

Top Lifetime Grosses Box-Office Collection

Presented by
Ketan Bhattacharjee

Next Slide

Data scraping

2025

https://www.boxofficemojo.com/chart/top_lifetime_gross/

library needs for this project:

```
import pandas as pd
import requests
from bs4 import BeautifulSoup
import time
import matplotlib.pyplot as plt
import seaborn as sns
```


@Ketan Bhattacharjee

Next Slide

Setting up the Url:

2025

```
base_url = "https://www.boxofficemojo.com/chart/top_lifetime_gross/?
area=XWW"
offset_url =
"https://www.boxofficemojo.com/chart/top_lifetime_gross/?
area=XWWa&offset={}"
headers = {
    "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.0.0
Safari/537.36"
}
```


Next Slide

@Ketan Bhattacharjee

Fetching data from website

@Ketan Bhattacharjee

2025

```
movies_data = []
page_urls = [base_url] + [offset_url.format(offset) for offset in range(200, 1000, 200)]
for page_num, url in enumerate(page_urls, start=1):
    #print(f"\nScraping Page {page_num}: {url}\n")
    response = requests.get(url, headers=headers)
    if response.status_code != 200:
        print(f"Failed to retrieve page {page_num}")
        continue
    soup = BeautifulSoup(response.text,"html.parser")
    rows = soup.select("tr")[1:]
    for row in rows:
        try:
            title = row.select_one("td a").text.strip()
            gross = row.select("td")[2].text.strip()
            release_year = row.select("td")[3].text.strip()

            # Append data to list
            movies_data.append([title, gross, release_year])
        except AttributeError:
            continue  # Skip invalid rows

    # Add a delay to avoid getting blocked
    time.sleep(2)
```


Next Slide

Creating DataFrame

2025

```
df = pd.DataFrame(movies_data, columns=["Title", "Gross Revenue", "Release Year"])
```

```
df.head()
```

	Title	Gross Revenue	Release Year
0	Avatar	$2,923,710,708	2009
1	Avengers: Endgame	$2,799,439,100	2019
2	Avatar: The Way of Water	$2,320,250,281	2022
3	Titanic	$2,264,812,968	1997
4	Star Wars: Episode VII - The Force Awakens	$2,071,310,218	2015

@Ketan Bhattacharjee

Data info & cleaning

2025

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 3 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Title           1000 non-null   object
 1   Gross Revenue   1000 non-null   object
 2   Release Year    1000 non-null   object
dtypes: object(3)
memory usage: 23.6+ KB
```

Checking for Null Values

```
df.isnull().sum()
```

```
Title             0
Gross Revenue     0
Release Year      0
dtype: int64
```

coverting data-types

```
df['Release Year'] = pd.to_datetime(df['Release Year']).dt.year
```

```
df['Gross Revenue'] = df['Gross Revenue'].replace('[/$,]', '', regex=True).astype(float)
```


Next Slide

@Ketan Bhattacharjee

Data Filtering Top 20 movies of All Time

2025

@Ketan Bhattacharjee

```
top_20 = df.nlargest(20, "Gross Revenue")
print(top_20)
```

	Title	Gross Revenue	Release Year
0	Avatar	2.923711e+09	2009
1	Avengers: Endgame	2.799439e+09	2019
2	Avatar: The Way of Water	2.320250e+09	2022
3	Titanic	2.264813e+09	1997
4	Star Wars: Episode VII - The Force Awakens	2.071310e+09	2015
5	Avengers: Infinity War	2.052415e+09	2018
6	Spider-Man: No Way Home	1.921408e+09	2021
7	Ne Zha 2	1.895832e+09	2025
8	Inside Out 2	1.698864e+09	2024
9	Jurassic World	1.671537e+09	2015
10	The Lion King	1.662021e+09	2019
11	The Avengers	1.520539e+09	2012
12	Furious 7	1.515342e+09	2015
13	Top Gun: Maverick	1.495696e+09	2022
14	Frozen II	1.453683e+09	2019
15	Barbie	1.447038e+09	2023
16	Avengers: Age of Ultron	1.405018e+09	2015
17	The Super Mario Bros. Movie	1.360848e+09	2023
18	Beauty and the Beast	1.356082e+09	2017
19	Black Panther	1.349926e+09	2018

Next Slide

Top movies Per Decade

2025

```
df['Decade'] = (df['Release Year']//10)*10
```

```
top_per_decade = df.loc[df.groupby("Decade")["Gross Revenue"].idxmax(), ["Decade", "Title", "Gross Revenue"]]
```

	Decade	Title	Gross Revenue
243	1930	Gone with the Wind	2.008822e+08
794	1940	Bambi	1.022472e+08
898	1950	Lady and the Tramp	9.360233e+07
388	1960	The Sound of Music	1.592875e+08
116	1970	Star Wars: Episode IV - A New Hope	7.753985e+08
107	1980	E.T. the Extra-Terrestrial	7.973074e+08
3	1990	Titanic	2.264813e+09
0	2000	Avatar	2.923711e+09
1	2010	Avengers: Endgame	2.799439e+09
2	2020	Avatar: The Way of Water	2.320250e+09

Next Slide

@Ketan Bhattacharjee

Creating Category based on Revenue

2025

```python
def category(revenue):
    if revenue >= 1_000_000_000:  # 1 billion
        return "Blockbuster"
    elif revenue >= 500_000_000:  # 500 million
        return "Super Hit"
    elif revenue >= 250_000_000:  # 250 million
        return "Hit"
    elif revenue >= 100_000_000:  # 100 million
        return "Average"
    else:
        return "Flop"
```

```python
df["Category"] = df["Gross Revenue"].apply(category)
```


Next Slide

@Ketan Bhattacharjee

Category Count

2025

```
category_count = df["Category"].value_counts()
```

```
category_count
```

```
Category
Average        646
Flop           154
Super Hit      143
Blockbuster     57
Name: count, dtype: int64
```


@Ketan Bhattacharjee

Next Slide

Most floped movies all time

2025

```
flop_movies = df[df["Category"] == "Flop"].sort_values(by = "Gross Revenue",ascending=True).head(10)
flop_movies[["Title", "Gross Revenue", "Release Year"]]
```

	Title	Gross Revenue	Release Year
999	The SpongeBob SquarePants Movie	85417988.0	2004
998	Real Steel	85468508.0	2011
997	Pokémon: The First Movie - Mewtwo Strikes Back	85744662.0	1999
996	Fury	85817906.0	2014
995	Alita: Battle Angel	85838210.0	2019
994	Spy Kids 2: Island of Lost Dreams	85846429.0	2002
993	Alvin and the Chipmunks: The Road Chip	85886987.0	2015
992	Die Hard	85892546.0	1988
991	Annie	85911262.0	2014
990	Mean Girls	86058055.0	2004

@Ketan Bhattacharjee

Next Slide

Blockbluster movies of all time

2025

```
block_blusters = df[df["Category"]=="Blockbuster"].sort_values(by = "Gross Revenue",ascending=False).head(10)
block_blusters[["Title", "Gross Revenue", "Release Year"]]
```

	Title	Gross Revenue	Release Year
0	Avatar	2.923711e+09	2009
1	Avengers: Endgame	2.799439e+09	2019
2	Avatar: The Way of Water	2.320250e+09	2022
3	Titanic	2.264813e+09	1997
4	Star Wars: Episode VII - The Force Awakens	2.071310e+09	2015
5	Avengers: Infinity War	2.052415e+09	2018
6	Spider-Man: No Way Home	1.921408e+09	2021
7	Ne Zha 2	1.895832e+09	2025
8	Inside Out 2	1.698864e+09	2024
9	Jurassic World	1.671537e+09	2015

@Ketan Bhattacharjee

Next Slide

Top 5 Super-Hitmovies of all time

2025

```
top_5_superhit_movies = df[df["Category"]=="Super Hit"].sort_values(by = "Gross Revenue",ascending=False).head(5)
top_5_superhit_movies[["Title","Gross Revenue","Release Year"]]
```

	Title	Gross Revenue	Release Year
57	The Lion King	979161373.0	1994
58	Oppenheimer	975811333.0	2023
59	Harry Potter and the Deathly Hallows: Part 1	974919063.0	2010
60	Despicable Me 2	970766005.0	2013
61	Despicable Me 4	969126452.0	2024

Top 5 Average movies

```
top5_avg_movies = df[df["Category"] =="Average"].sort_values(by = "Gross Revenue",ascending =False).head(5)
top5_avg_movies[["Title","Gross Revenue","Release Year"]]
```

	Title	Gross Revenue	Release Year
200	King Kong	218080025.0	2005
201	Ghost	217631306.0	1990
202	How to Train Your Dragon	217581231.0	2010
203	The Da Vinci Code	217536138.0	2006
204	Aladdin	217350219.0	1992

Next Slide

@Ketan Bhattacharjee

Top movie for each year

2025

```
top_per_year = df.loc[df.groupby("Release Year")["Gross Revenue"].idxmax()]
top_per_year[["Title","Release Year","Decade"]]
```

	Title	Release Year	Decade
279	Snow White and the Seven Dwarfs	1937	1930
243	Gone with the Wind	1939	1930
794	Bambi	1942	1940
907	Cinderella	1950	1950
972	Peter Pan	1953	1950
...	...	...	...
6	Spider-Man: No Way Home	2021	2020
2	Avatar: The Way of Water	2022	2020
15	Barbie	2023	2020
8	Inside Out 2	2024	2020
7	Ne Zha 2	2025	2020

@Ketan Bhattacharjee

Next Slide

Number of Movies Released Per Decade

2025

```
no_of_movie_releasd_per_decade = df['Decade'].value_counts().sort_index()
no_of_movie_releasd_per_decade
```

```
Decade
1930      2
1940      1
1950      3
1960      7
1970     21
1980     50
1990    145
2000    281
2010    384
2020    106
Name: count, dtype: int64
```


Next Slide

@Ketan Bhattacharjee

Revenue Trend Over Time

```
revenue_trend = df.groupby("Release Year").agg({"Gross Revenue":"mean"})
revenue_trend
```

Release Year	Gross Revenue
1937	1.849255e+08
1939	2.008822e+08
1942	1.022472e+08
1950	9.314115e+07
1953	8.740465e+07
...	...
2021	3.763589e+08
2022	4.833987e+08
2023	3.475846e+08
2024	3.977797e+08
2025	7.286628e+08

2025

Next Slide

@Ketan Bhattacharjee

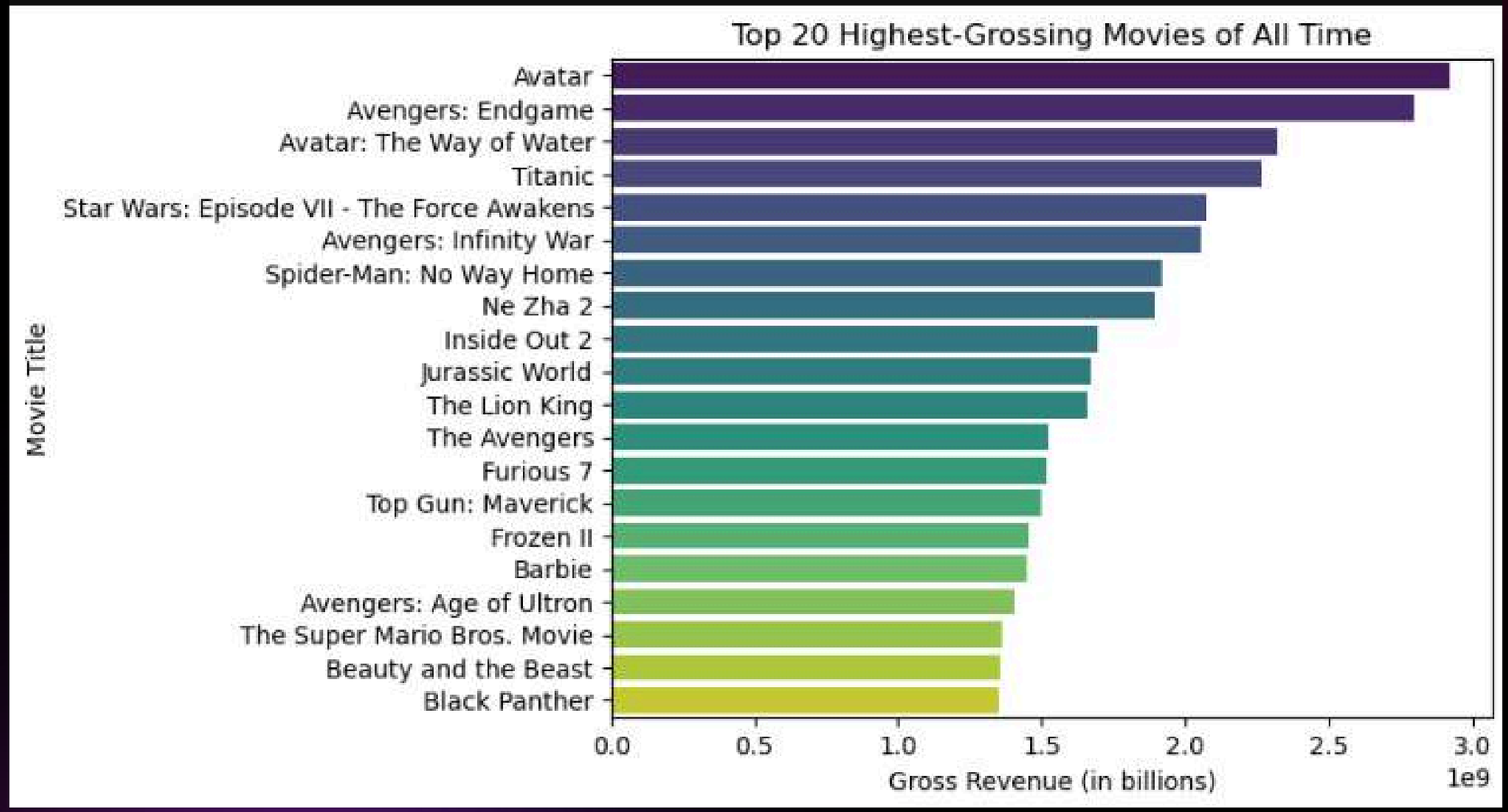
Visualization

@Ketan Bhattacharjee

2025

Top 20 Movies Based on Gross Revenue

```
sns.barplot(data = top_20,x="Gross Revenue", y="Title",palette="viridis",hue = "Title")
plt.xlabel("Gross Revenue (in billions)")
plt.ylabel("Movie Title")
plt.title("Top 20 Highest-Grossing Movies of All Time")
plt.show()
```



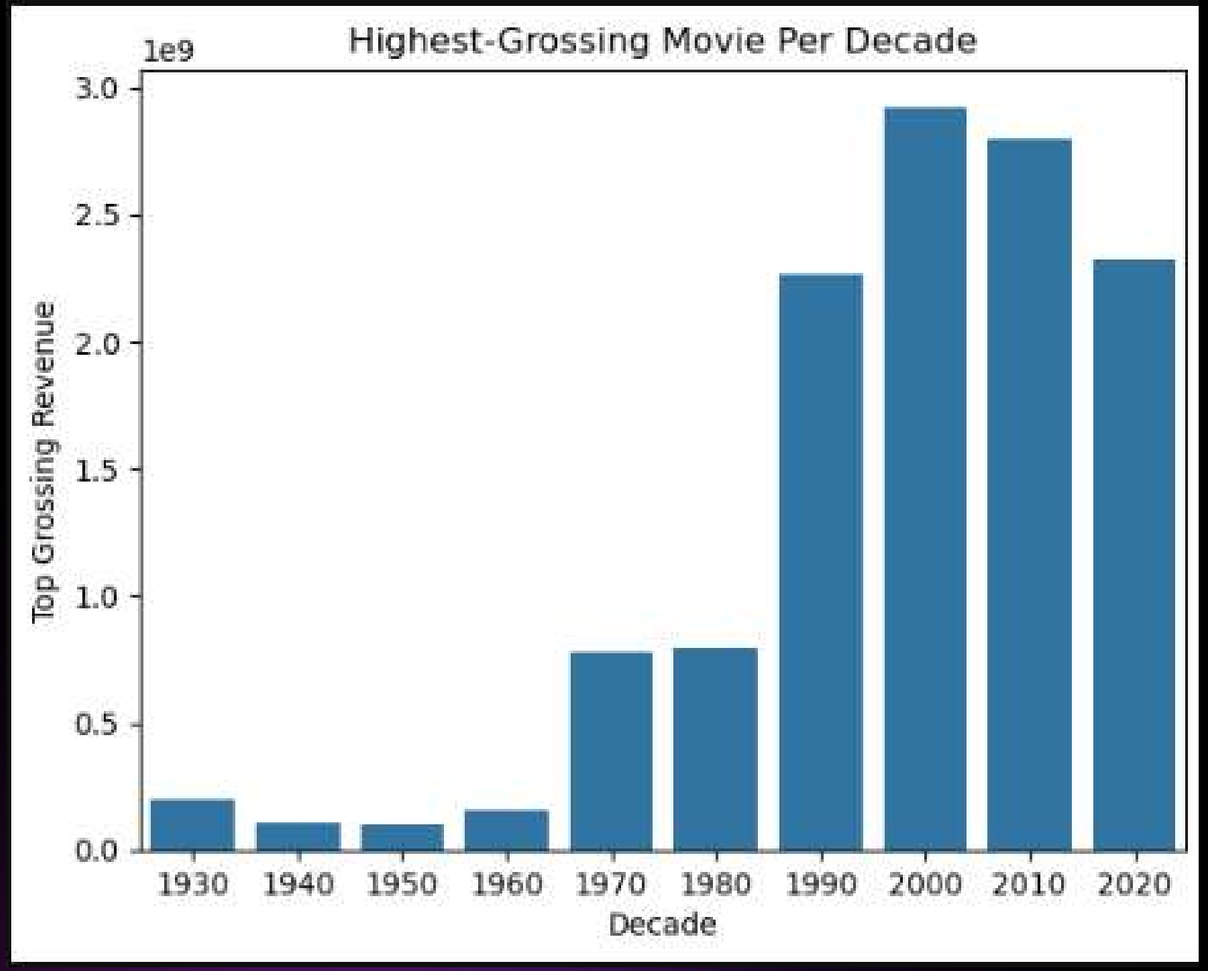
Next Slide

Visualization

Top Movie Per Decade (Revenue)

2025

```
sns.barplot(data =top_per_decade,x = "Decade",y ="Gross Revenue" )
plt.xlabel("Decade")
plt.ylabel("Top Grossing Revenue")
plt.title("Highest-Grossing Movie Per Decade")
plt.show()
```



Next Slide

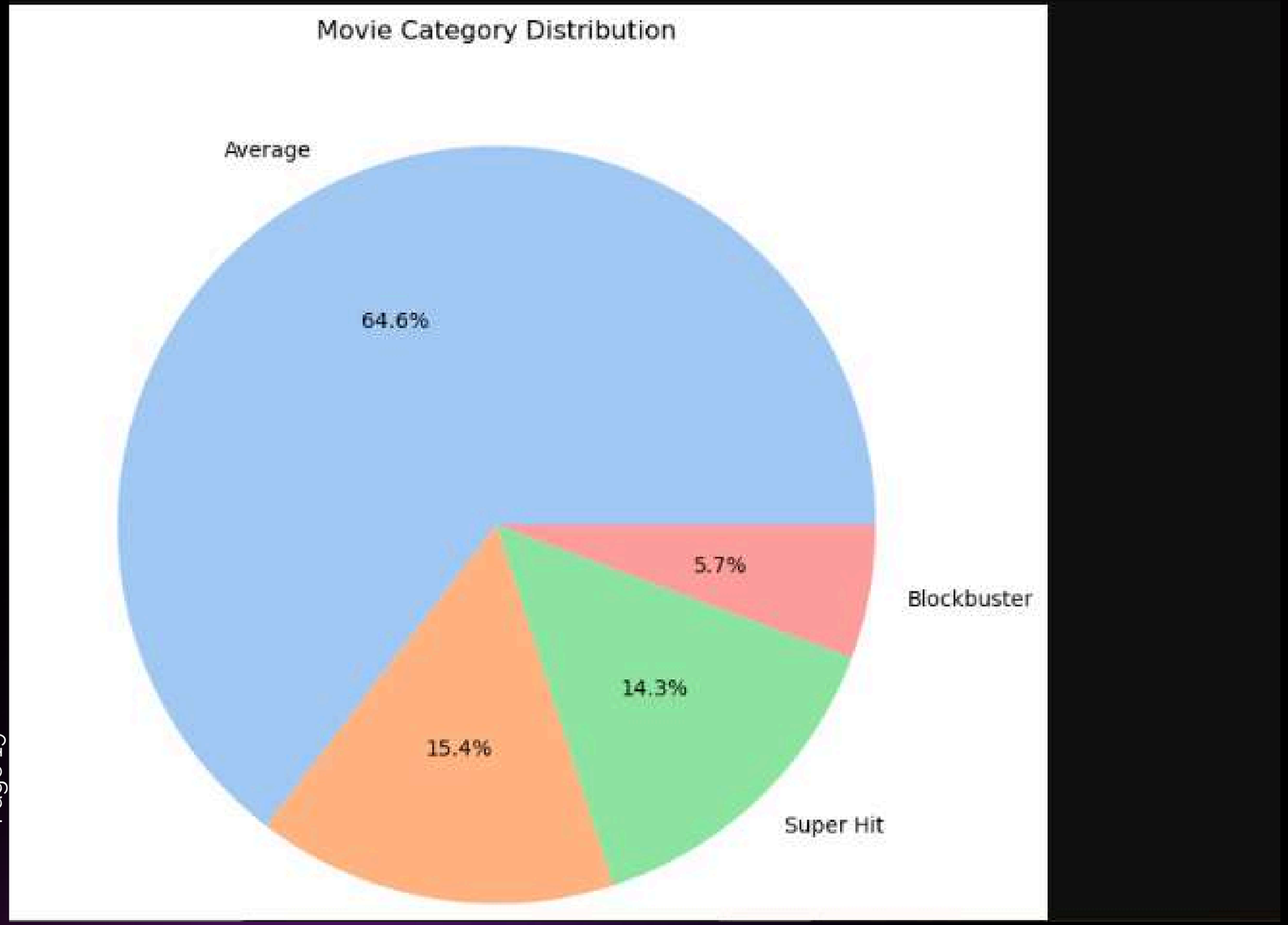
@Ketan Bhattacharjee

Visualization

Movie Category Distribution

2025

```
plt.figure(figsize=(8, 8))
plt.pie(category_count,labels=category_count.index, autopct="%1.1f%%", colors=sns.color_palette("pastel"))
plt.title("Movie Category Distribution")
plt.show()
```



Next Slide

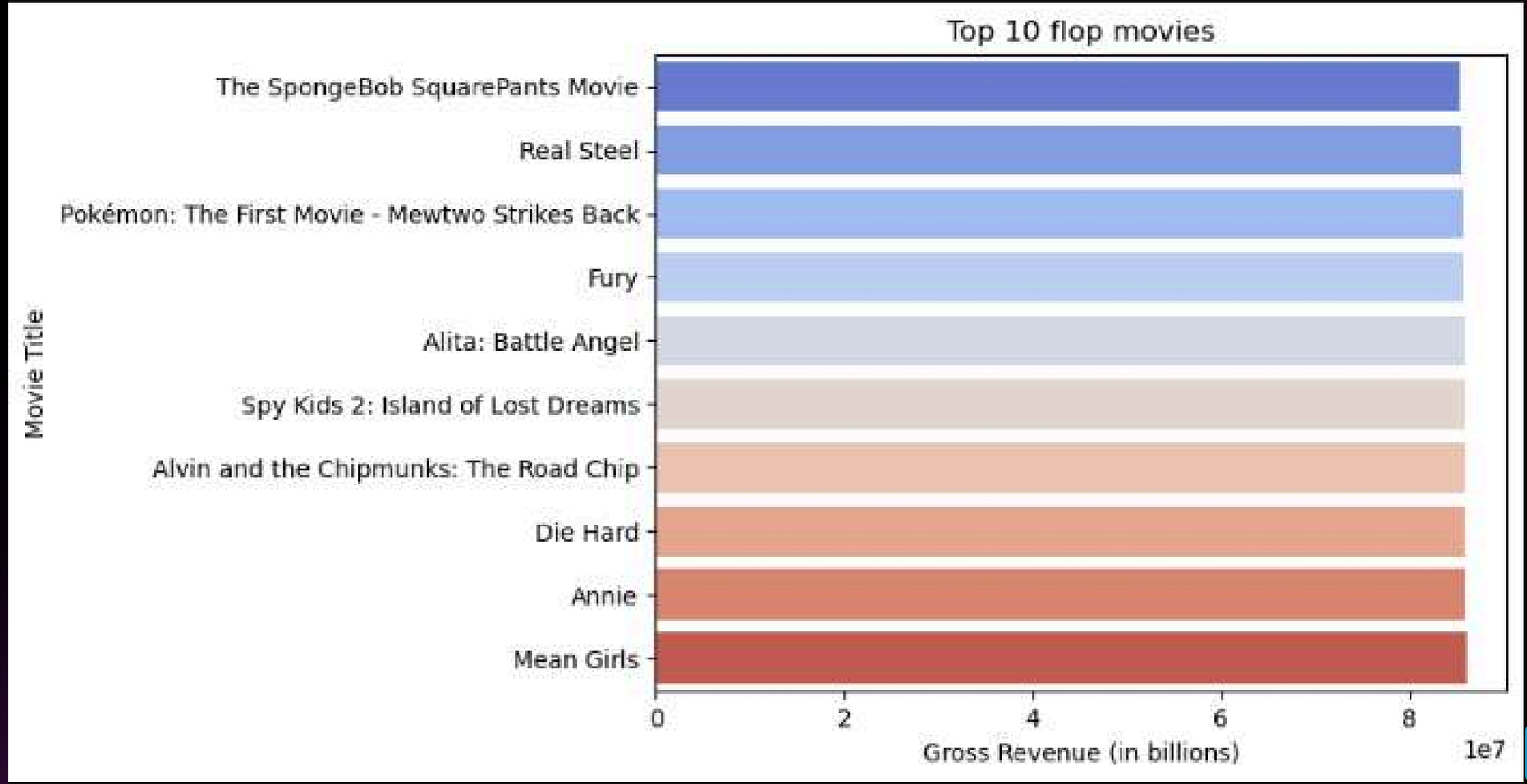
@Ketan Bhattacharjee

Visualization

Most Flopped Movies

2025

```
sns.barplot(data =flop_movies,y= "Title",x = "Gross Revenue", palette="coolwarm",hue ="Title")
plt.xlabel("Gross Revenue (in billions)")
plt.ylabel("Movie Title")
plt.title("Top 10 flop movies")
plt.show()
```



Next Slide

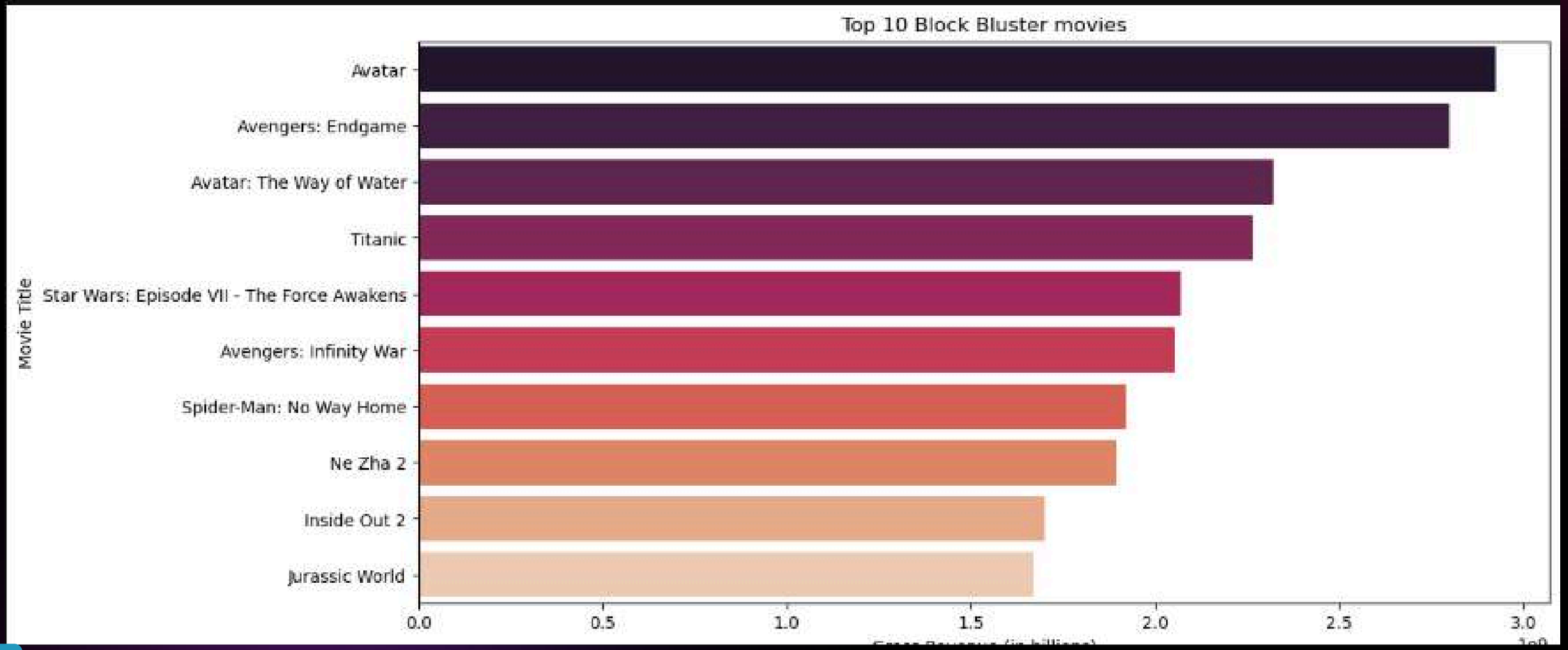
@Ketan Bhattacharjee

Visualization

2025

Top10 block bluester movies

```
plt.figure(figsize=(12, 6))
sns.barplot(data = block_blusters,x = "Gross Revenue", y = "Title", palette="rocket",hue = "Title")
plt.xlabel("Gross Revenue (in billions)")
plt.ylabel("Movie Title")
plt.title("Top 10 Block Bluster movies")
plt.show()
```



Next Slide

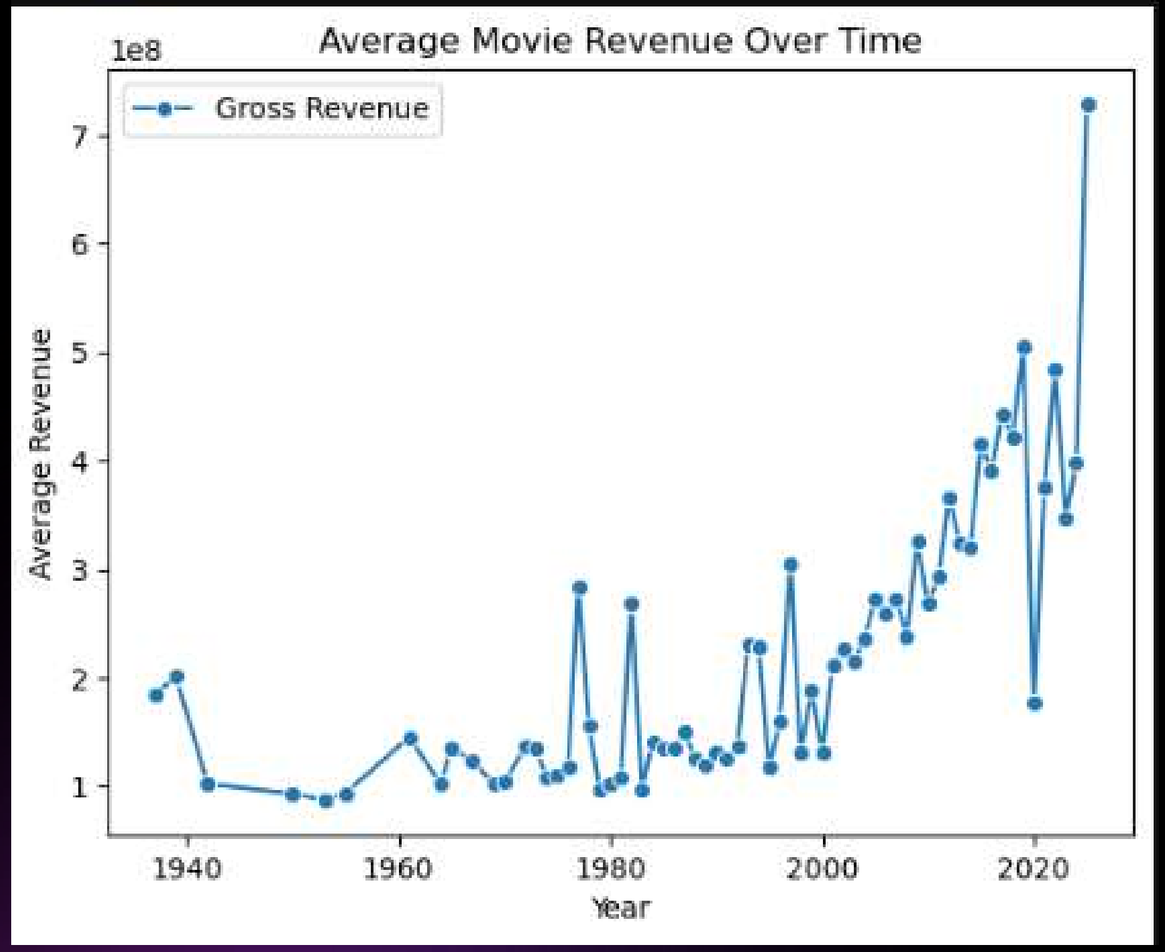
@Ketan Bhattacharjee

Visualization

Revenue Trend Over Years

2025

```
sns.lineplot(data  = revenue_trend,marker = "o")
plt.xlabel("Year")
plt.ylabel("Average Revenue")
plt.title("Average Movie Revenue Over Time")
plt.show()
```



Next Slide

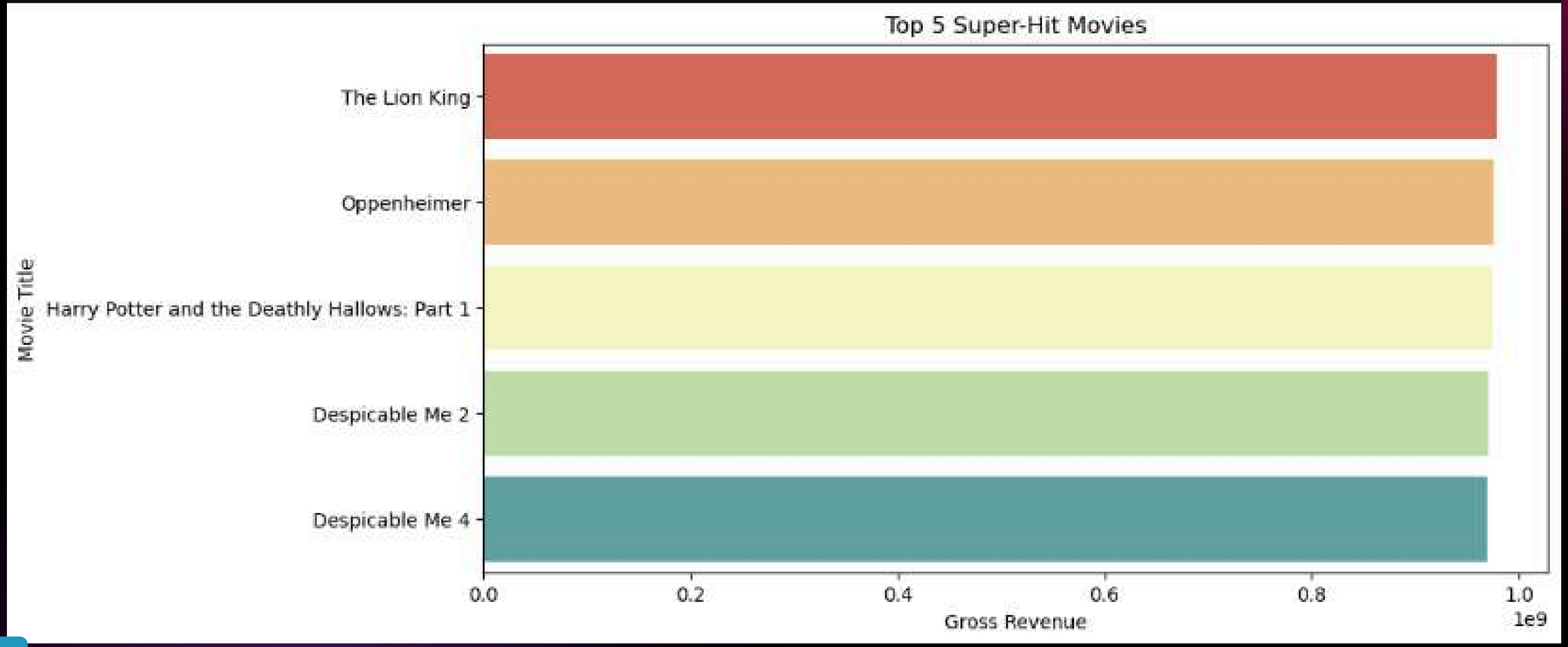
@Ketan Bhattacharjee

Top 5 Super-Hit Movies

Visualization

2025

```
plt.figure(figsize=(10, 5))
sns.barplot(data=top_5_superhit_movies, x="Gross Revenue", y="Title", palette="Spectral",hue = "Title")
plt.xlabel("Gross Revenue")
plt.ylabel("Movie Title")
plt.title("Top 5 Super-Hit Movies")
plt.show()
```



Next Slide

@Ketan Bhattacharjee

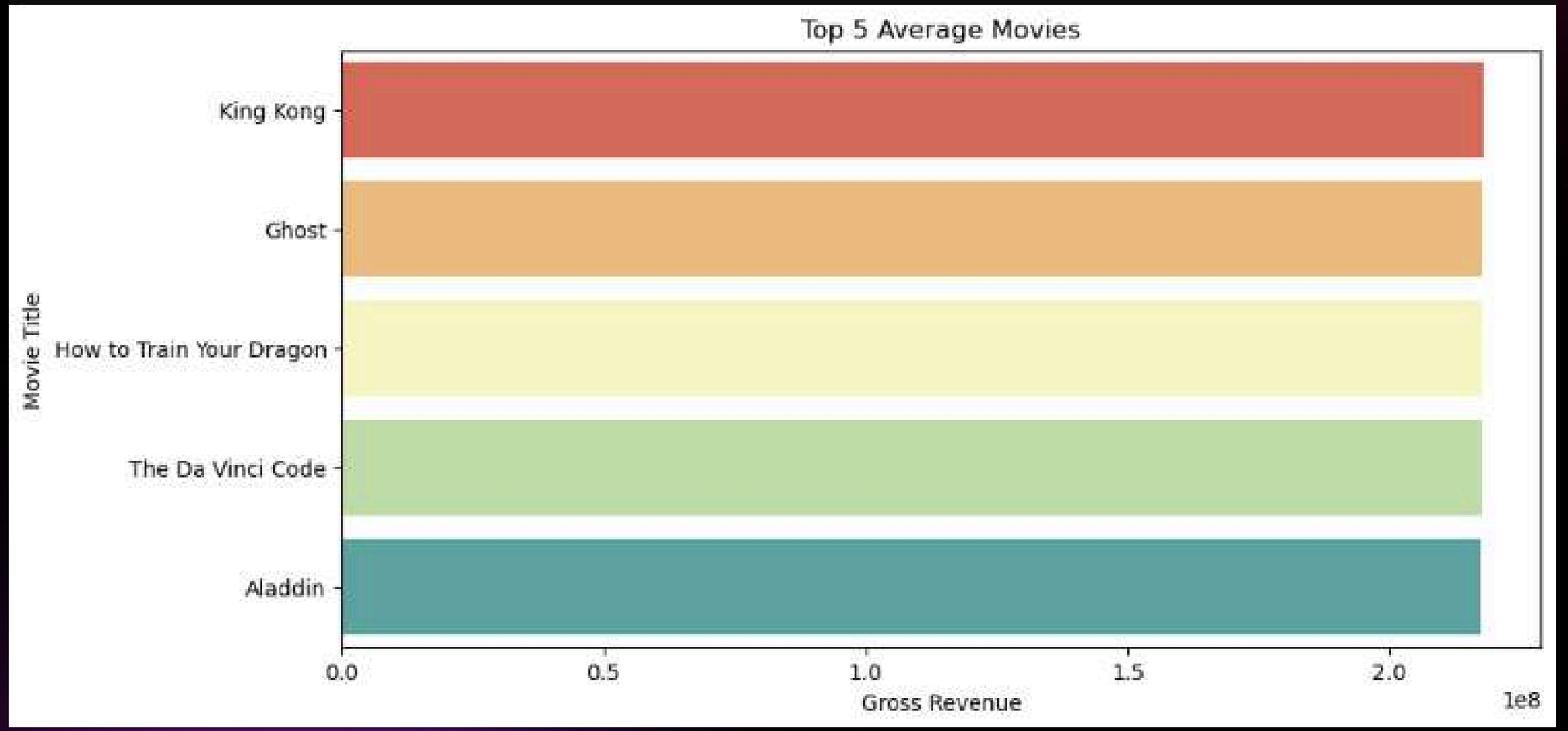
Visualization

@Ketan Bhattacharjee

2025

Top5 Average movie

```
plt.figure(figsize=(10, 5))
sns.barplot(data = top5_avg_movies,x = "Gross Revenue", y = "Title",palette="Spectral",hue = "Title")
plt.xlabel("Gross Revenue")
plt.ylabel("Movie Title")
plt.title("Top 5 Average Movies")
plt.show()
```



Next Slide

Conclusion

2025

"Avatar" stands as the highest-grossing movie of all time, with a massive box office collection of **$2,923,710,708.** In contrast, **"The SpongeBob SquarePants Movie"** had the lowest earnings, generating only **$85,417,988**.

When analyzing movie distribution, only **5.7%** (approximately 6%) of films achieved blockbuster status. The **2000s** emerged as the **decade** with the highest **total gross revenue**, while the average movie revenue saw a significant surge.

Finally, the year **2010** recorded the highest number of movie releases

Thank You

@Ketan Bhattacharjee