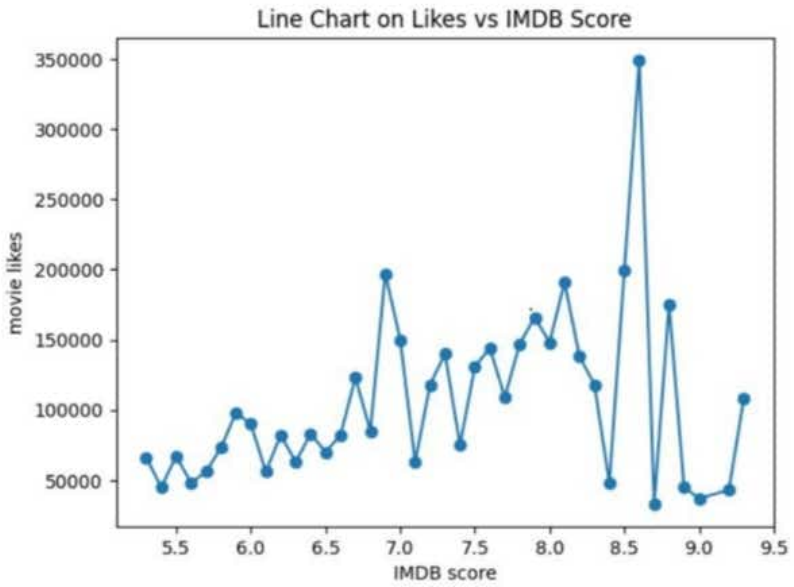
```
import pandas as pd
import matplotlib.pyplot as plt
import plotly.subplots as sp
import matplotlib.gridspec as gridspec
import plotly.graph_objects as go
import numpy as np

df =pd.read_csv("/content/movie_data.csv")
df.dropna(inplace=True)
df.drop_duplicates(inplace=True)
# print(df.isnull())
df.reset_index(drop=True, inplace=True)
```

1. Line Graph: Likes v/s IMDB Rating

Code:

```
df1=df.groupby('imdb_score').max()
df1 = df1.tail(40)

#plot the graph
plt.plot(df1.index,df1['movie_likes'], marker='o')

#customize the graph
plt.title("Line Chart on Likes vs IMDB Score")
plt.xlabel("IMDB score")
plt.ylabel("movie likes")

# Display the chart
plt.show()
```

Output:

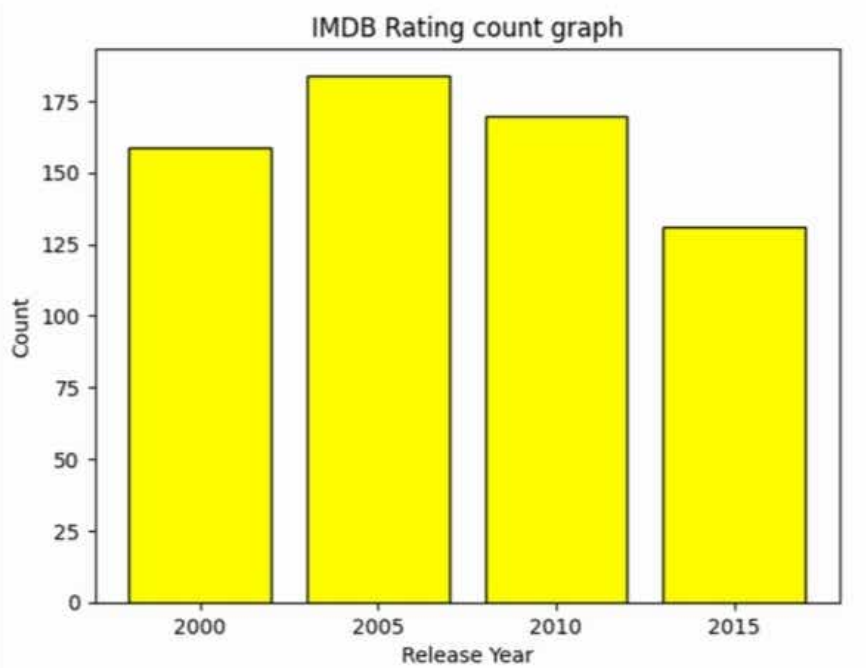


2. Vertical Bar Graph: IMDB Rating Bar Graph

Code:

```
year = list(df['title_year'].astype('int64'))
count_2000 = year.count(2000)
count_2005 = year.count(2005)
count_2010 = year.count(2010)
count_2015 = year.count(2015)

release_year = ['2000', '2005', '2010', '2015']
count = [count_2000, count_2005, count_2010, count_2015]

# Create a bar plot
plt.bar(release_year, count, color='yellow', edgecolor='black')

# Customize the plot
plt.title("IMDB Rating count graph")
plt.xlabel("Release Year")
plt.ylabel("Count")

# Display the plot
plt.show()
```

Output:

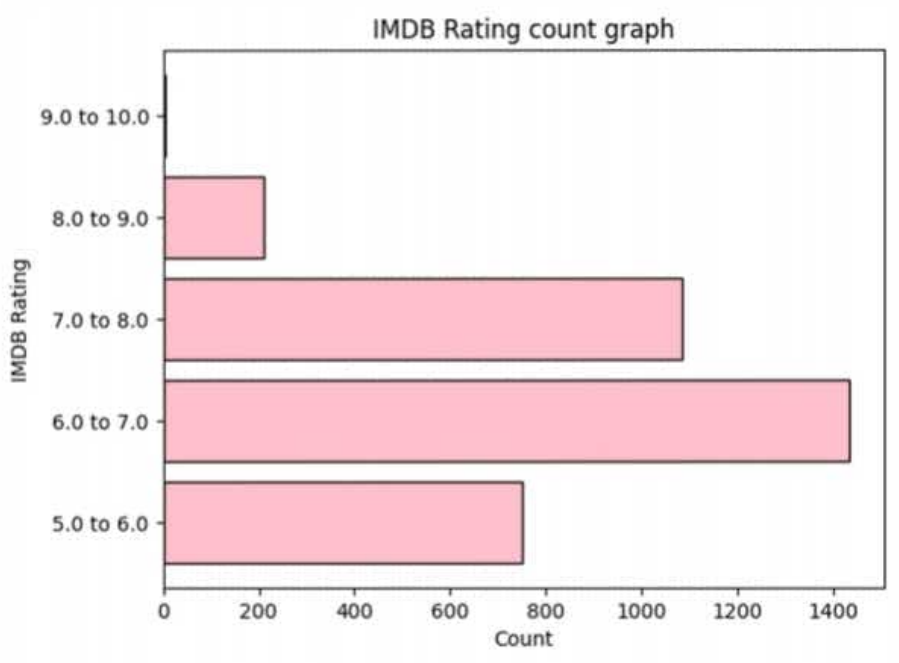


3. Horizontal Bar Graph: IMDB Rating Bar Graph

Code:

```
imdb = list(df['imdb_score'].astype('int64'))
count_5_6 = imdb.count(5)
count_6_7 = imdb.count(6)
count_7_8 = imdb.count(7)
count_8_9 = imdb.count(8)
count_9_10 = imdb.count(9)

imdb = ['5.0 to 6.0', '6.0 to 7.0', '7.0 to 8.0', '8.0 to 9.0', '9.0 to 10.0']
count = [count_5_6, count_6_7, count_7_8, count_8_9, count_9_10]

# Create a bar plot
plt.barh(imdb, count, color='pink', edgecolor='black')

# Customize the plot
plt.title("IMDB Rating count graph")
plt.xlabel("Count")
plt.ylabel("IMDB Rating")

# Display the plot
plt.show()
```

Output:

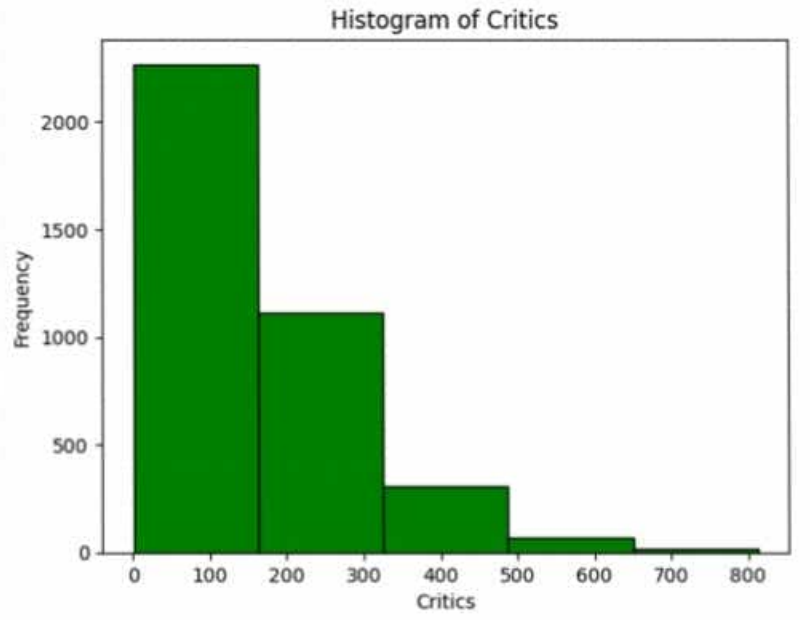


4. Histogram Graph: Critics histogram graph

Code:

```
num_critic = df['num_critic'].astype('int64')

# Plotting the histogram
plt.hist(num_critic, bins=5, edgecolor='black', color='green')

# Adding labels and title
plt.xlabel('Critics')
plt.ylabel('Frequency')
plt.title('Histogram of Critics')

# Displaying the histogram
plt.show()
```

Output:

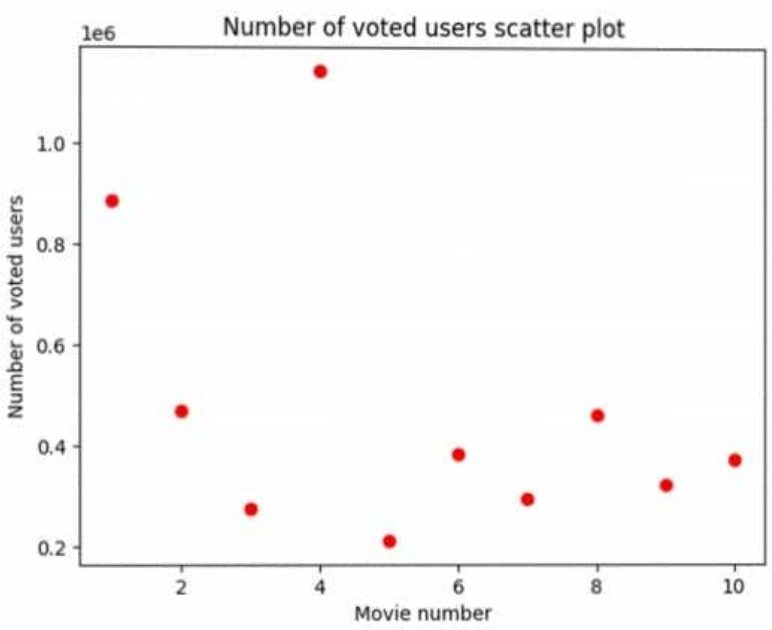


5. Scatter Graph: Number of voted user scatter graph

Code:

```
movie_number = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
num_voted_user = (df['num_voted_users'][0:10]).astype('int64')

# Plotting the scatter plot
plt.scatter(movie_number, num_voted_user, color='red')

# Adding labels and title
plt.xlabel('Movie number')
plt.ylabel('Number of voted users')
plt.title('Number of voted users scatter plot')

# Displaying the scatter plot
plt.show()
```

Output:



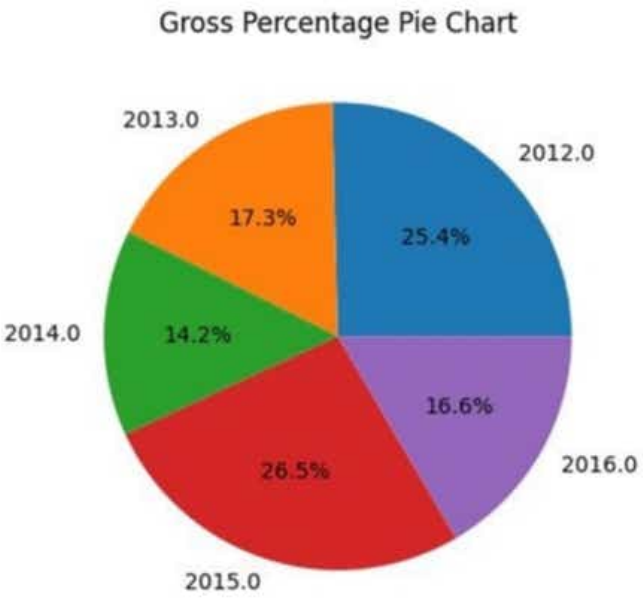
5. Pie Graph: Gross Percentage Pie Chart

Code:

```
df1 = df.groupby('title_year').max()
df1 = df1.tail(5)
# df1.first()

# Plotting the pie chart
plt.pie(df1['gross'], labels=df1.index, autopct='%1.1f%%')

# Adding a title
plt.title('Gross Percentage Pie Chart')
```

```
# Displaying the pie chart
plt.show()
```

Output:

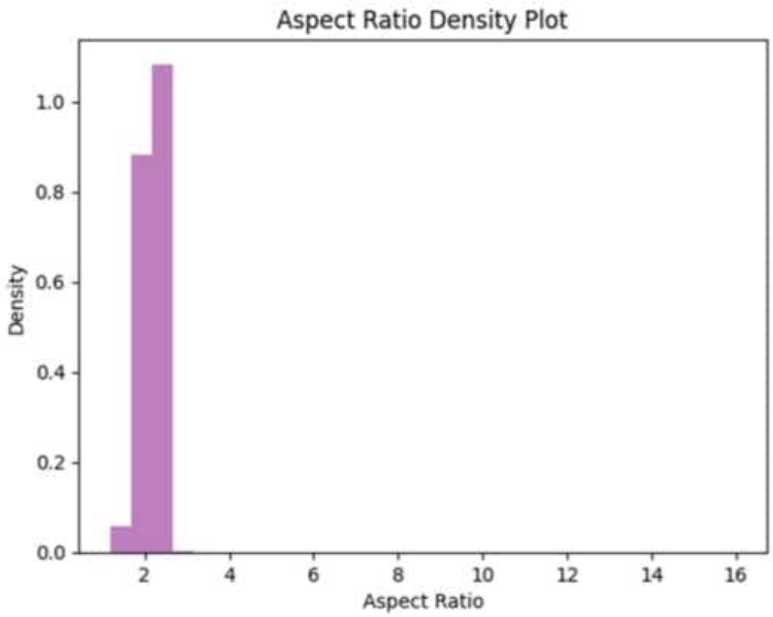


6. Density Graph: Aspect ratio density graph

Code:

```
aspect_ratio = df['aspect_ratio'].astype(float)

# Create a density plot
plt.hist(aspect_ratio, density=True, bins=30, alpha=0.5, color='purple')

# Add labels and title
plt.xlabel('Aspect Ratio')
plt.ylabel('Density')
plt.title('Aspect Ratio Density Plot')

# Show the plot
plt.show()
```

Output:



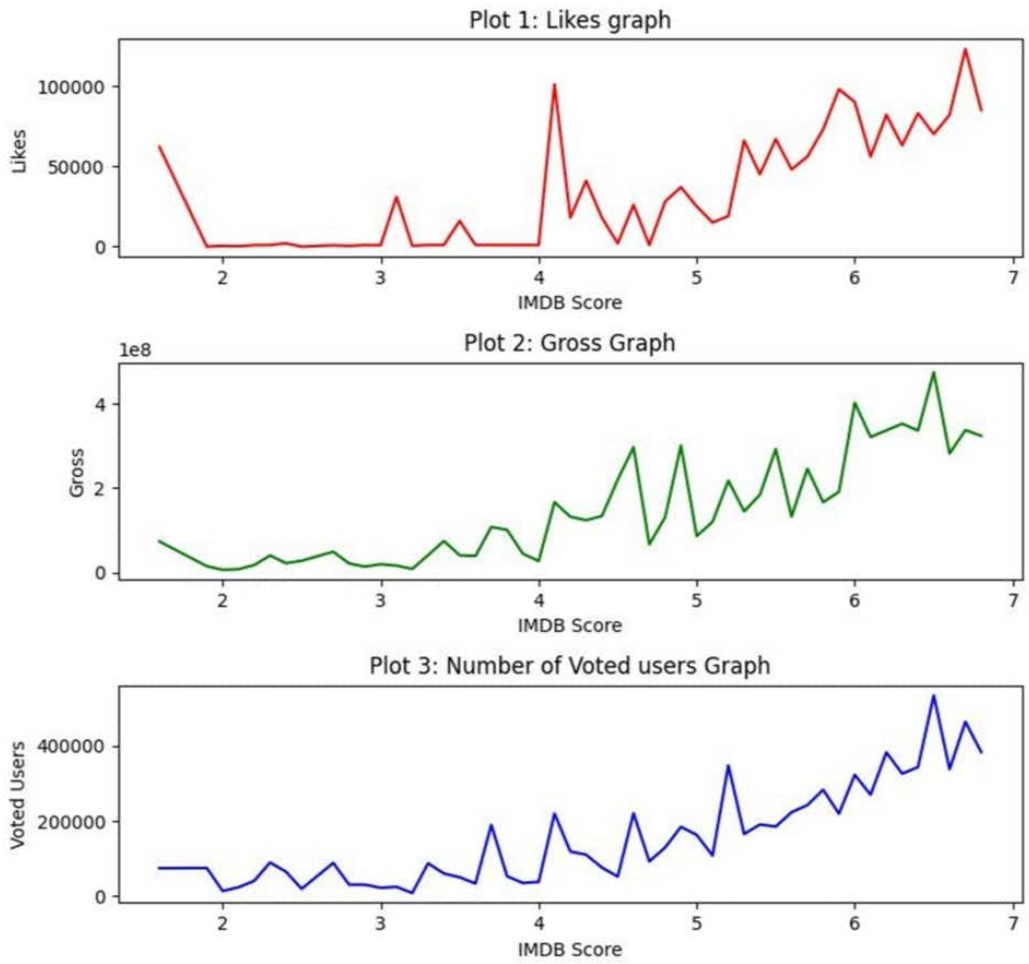
7. Group Plot: Likes graph, Gross graph, IMDB rating graph, Critic graph

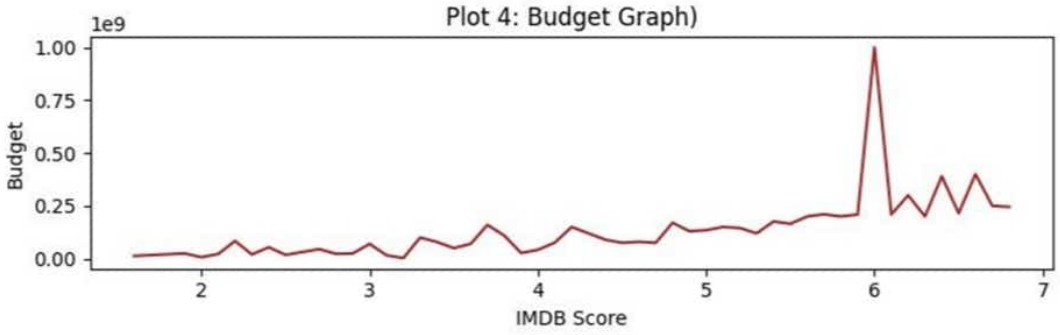
Code:

```
df1 = df.groupby('imdb_score').max()
df1 = df1.head(50)

# Create a figure with subplots
fig, axes = plt.subplots(4, 1, figsize=(8, 10))
# Plot data on each subplot

axes[0].plot(df1.index, df1['movie_likes'], color='red')
axes[0].set_title('Plot 1: Likes graph')
axes[0].set_xlabel('IMDB Score')
axes[0].set_ylabel('Likes')

axes[1].plot(df1.index, df1['gross'], color='green')
axes[1].set_title('Plot 2: Gross Graph')
axes[1].set_xlabel('IMDB Score')
axes[1].set_ylabel('Gross')

axes[2].plot(df1.index, df1['num_voted_users'], color='blue')
axes[2].set_title('Plot 3: Number of Voted users Graph')
axes[2].set_xlabel('IMDB Score')
axes[2].set_ylabel('Voted Users')

axes[3].plot(df1.index, df1['budget'], color='brown')
axes[3].set_title('Plot 4: Budget Graph)')
axes[3].set_xlabel('IMDB Score')
axes[3].set_ylabel('Budget')

# Adjust spacing between subplots
plt.tight_layout()
# Show the plot
```

```
plt.show()
```

Output:



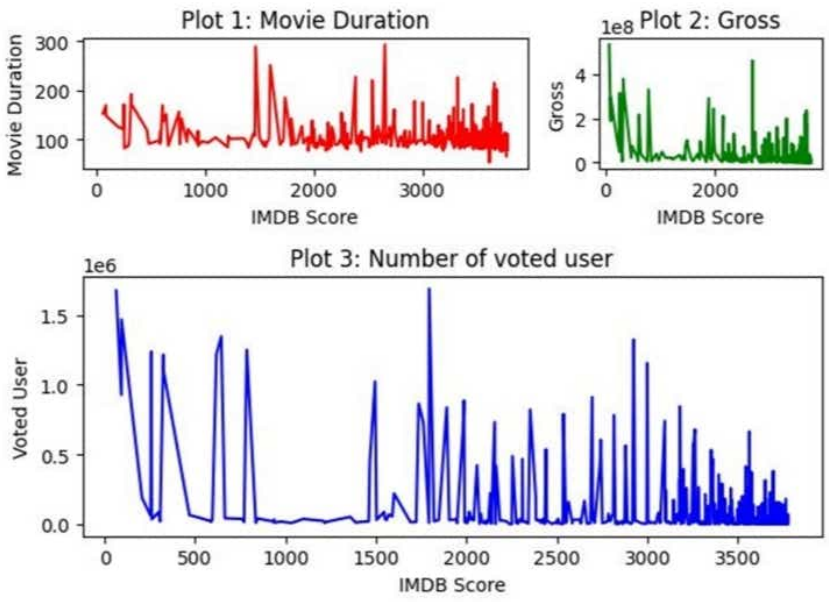


8. Subplot Graph: Movie duration, gross and number of voted user graphs

Code:

```
df1 = df.groupby('imdb_score')
df1 = df1.tail(10)

# Create a GridSpec with different-sized cells
gs = gridspec.GridSpec(2, 2, width_ratios=[2, 1], height_ratios=[1, 2])
# Create subplots within the GridSpec
ax1 = plt.subplot(gs[0, 0]) # Top-left subplot
ax2 = plt.subplot(gs[0, 1]) # Top-right subplot
ax3 = plt.subplot(gs[1, :]) # Bottom row, spanning both columns

# Plot data on each subplot
ax1.plot(df1.index, df1['duration'], color='red')
ax1.set_title('Plot 1: Movie Duration')
ax1.set_xlabel('IMDB Score')
ax1.set_ylabel('Movie Duration')
ax2.plot(df1.index, df1['gross'], color='green')
ax2.set_title('Plot 2: Gross')
ax2.set_xlabel('IMDB Score')
ax2.set_ylabel('Gross')
ax3.plot(df1.index, df1['num_voted_users'], color='blue')
ax3.set_title('Plot 3: Number of voted user')
ax3.set_xlabel('IMDB Score')
ax3.set_ylabel('Voted User')
plt.tight_layout() # Adjust spacing between subplots
plt.show() # Show the plot
```

Output:

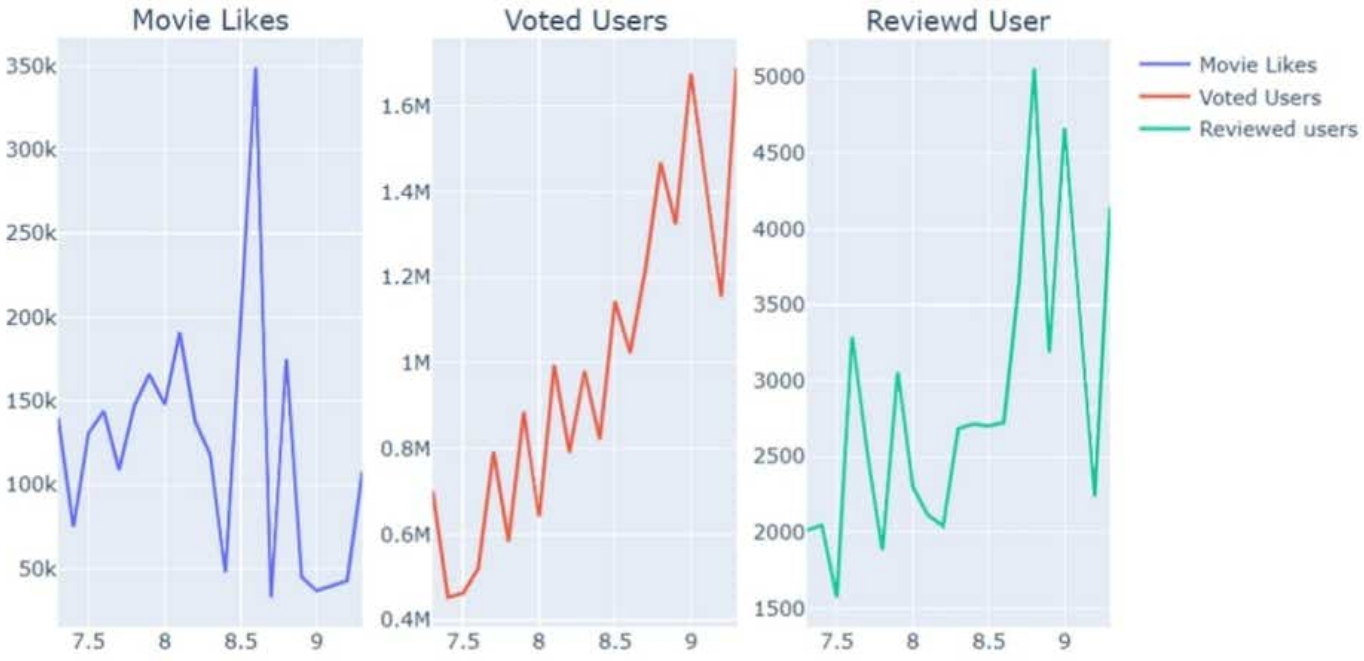


9. Panel Graph: Movie likes, voted users and reviewed users

Code:

```
# Create data for the panels
df1=df.groupby('imdb_score').max()
df1 = df1.tail(20)

# Create subplots with panels
fig = sp.make_subplots(rows=1, cols=3, subplot_titles=('Movie Likes', 'Voted
Users', 'Reviewd User'))

# Add traces to each panel
fig.add_trace(go.Scatter(x=df1.index, y=df1['movie_likes'], name='Movie Likes'),
row=1, col=1)
fig.add_trace(go.Scatter(x=df1.index, y=df1['num_voted_users'], name='Voted
Users'), row=1, col=2)
fig.add_trace(go.Scatter(x=df1.index, y=df1['num_user_for_reviews'], name='Reviewed
users'), row=1, col=3)

# Update layout and display the plot
fig.update_layout(showlegend=True)
fig.show()
```

Output:



10. Area Graph: Movie likes, voted users and reviewed users

Code:

```
df1 = df.groupby('imdb_score').max()
df1 = df1.head(50)
```

```
plt.stackplot(df1.index, df1['gross'], df1['movie_likes'], colors = ['c', 'g'])
plt.xlabel('IMDB Ratings')
plt.ylabel('Movie Likes')
plt.title('Movie Report Area Plot')
plt.legend()

# Print the chart
plt.show()
```

Output: