Movie Analysis

by Domas Budrys

In [18]:

```
%matplotlib inline
import matplotlib
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import pandas as pd
import csv
```

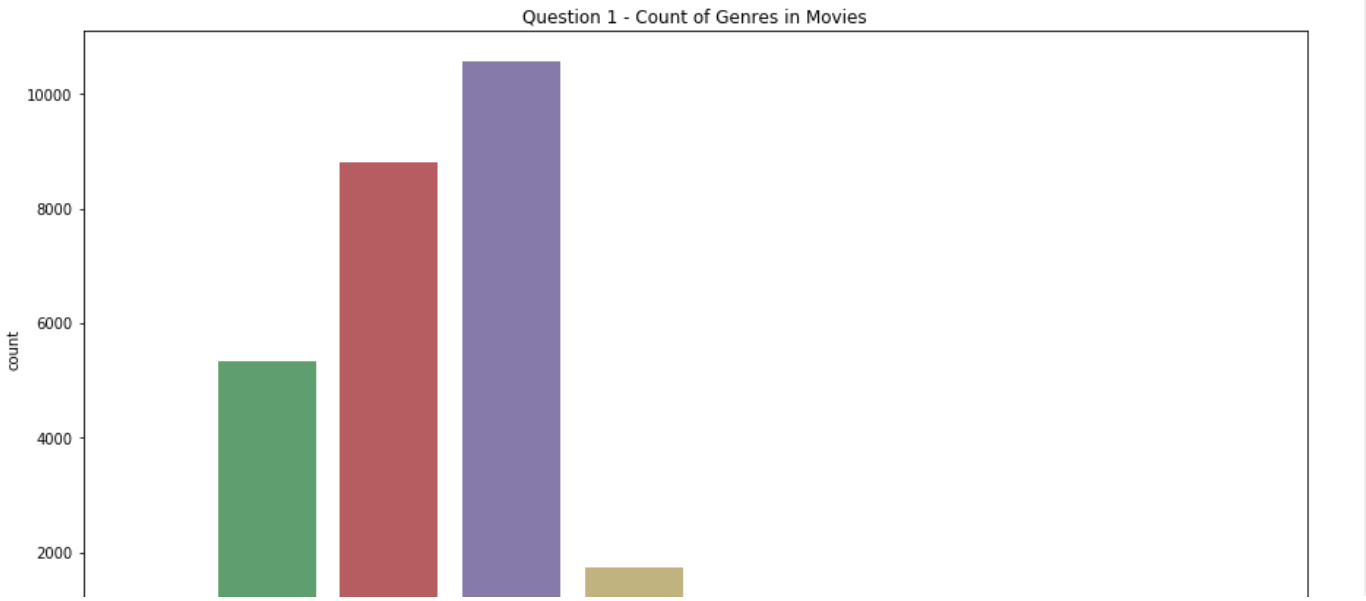
Importing all the necessary libraries

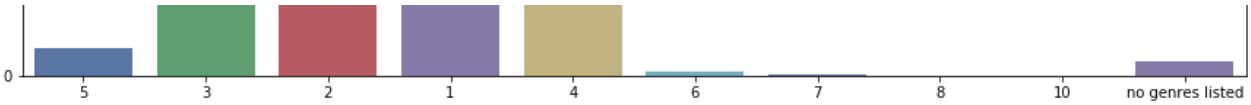
In [2]:

```
#Question 1
genre_column = []
genre_count = []
with open("movies.csv") as dataIn:
        reader = csv.DictReader(dataIn)

        for row in reader:
            genre_column.append(row['genres'])

for i in genre_column:

    if i == "(no genres listed)":
        i = "no genres listed"
        genre_count.append(i)

    else:
        i = i.count('|') +1
        genre_count.append(str(i))


#Set up and Display plot
plt.figure(figsize=(15,8))
plot1 = sns.countplot(genre_count, palette="deep")
plot1.set_title("Question 1 - Count of Genres in Movies")
```

Out[2]:

```
Text(0.5,1,'Question 1 - Count of Genres in Movies')
```





Question 1

Genres of each movie are appended to **genres_column**. Then, using *for loop* we check if an item in genres column is specified or not, and assign appropriate value to the new list **genre_count**. We are able to separate each genre by separetor "|":

```
for i in genre_column:

if i == "(no genres listed)":
    i = "no genres listed"
    genre_count.append(i)

else:
    i = i.count('|') +1
    genre_count.append(str(i))
```

Finally, after getting all the correct information and storying to the list **genre_count** we are able to display countplot:

```
plt.figure(figsize=(15,8))
plot1 = sns.countplot(genre_count, palette="deep")
plot1.set_title("Question 1 - Count of Genres in Movies")
```

In [3]:

```
#Question 2
genre_names =[]

for i in genre_column:
    i = i.split('|')

    for a in i:
        if a != "(no genres listed)":
            genre_names.append(a)
```

Question 2

For question 2 we are able to store all of the listed genre names who's values are not *(no genres listed)* to the list **genre_names**

In [4]:

```
from collections import Counter

#Set the list of ordered words
ordered_genres = [genre for genre, genre_count in Counter(genre_names).most_common()]
```

Question 2 (Continued)

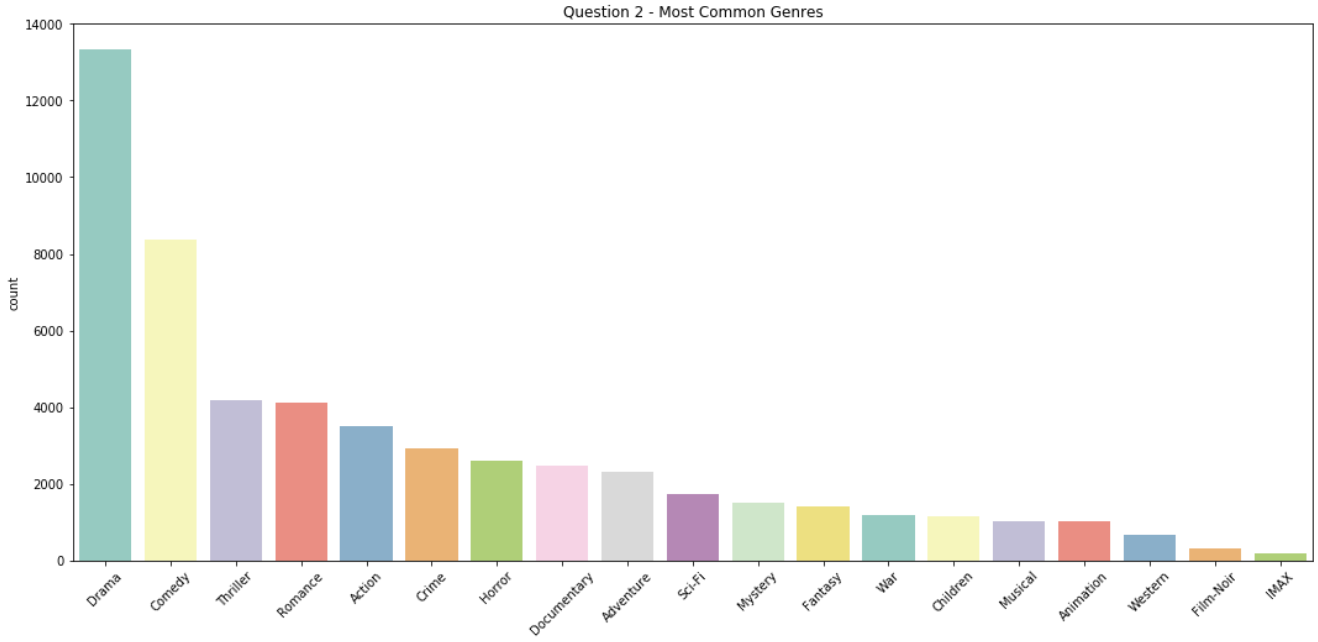
ordered_genres list is created to display only names of genres in sorted order. This list will be used to set the order of countplot

In [5]:

```
plt.figure(figsize=(18,8))
plt.xticks(rotation=45)

plot2 = sns.countplot(genre_names, palette="Set3", order = ordered_genres)
plot2.set_title("Question 2 - Most Common Genres")
```

Out[5]:

```
Text(0.5,1,'Question 2 - Most Common Genres')
```



Question 2 (Continued)

We create countplot:

```
plot2 = sns.countplot(genre_names, palette="Set3", order = ordered_genres)
plot2.set_title("Question 2 - Most Common Genres")
```

order = is used to set the order of bars in the countplot by specifying the name of the bar. However, we have already created the list **ordered_genres** which is used to provide values for ordering

In [6]:

```
#Question 3
tag_column = []
with open("tags.csv") as dataIn:
        reader = csv.DictReader(dataIn)

        for row in reader:

            tag_column.append(row['tag'])

# Very item is lower_case
tag_lower = [x.lower() for x in tag_column]
```

Question 3

We store each row of tag in to the list **tag_column** and then, using list comprehension we are able to store all the tag values coverted to lower case to the list **tag_lower**

In [7]:

```
ordered_tag = [word for word, word_count in Counter(tag_lower).most_common(10)]
ordered_tag.reverse()

#Append only 10 most common words from tag_lower to the new list
tag_top10 =[]

for i in tag_lower:

    if i in ordered_tag:
        tag_top10.append(i)
```

Quetion 3 (Continued)

ordered_tag list is created to display only tags of movies (not the count), in sorted order. Then, we create the new list **tag_top10** and append all of the values from **tag_lower**, which is the list of all the tags, that are included in **ordered_tag**:

```
    for i in tag_lower:

    if i in ordered_tag:
        tag_top10.append(i)
```

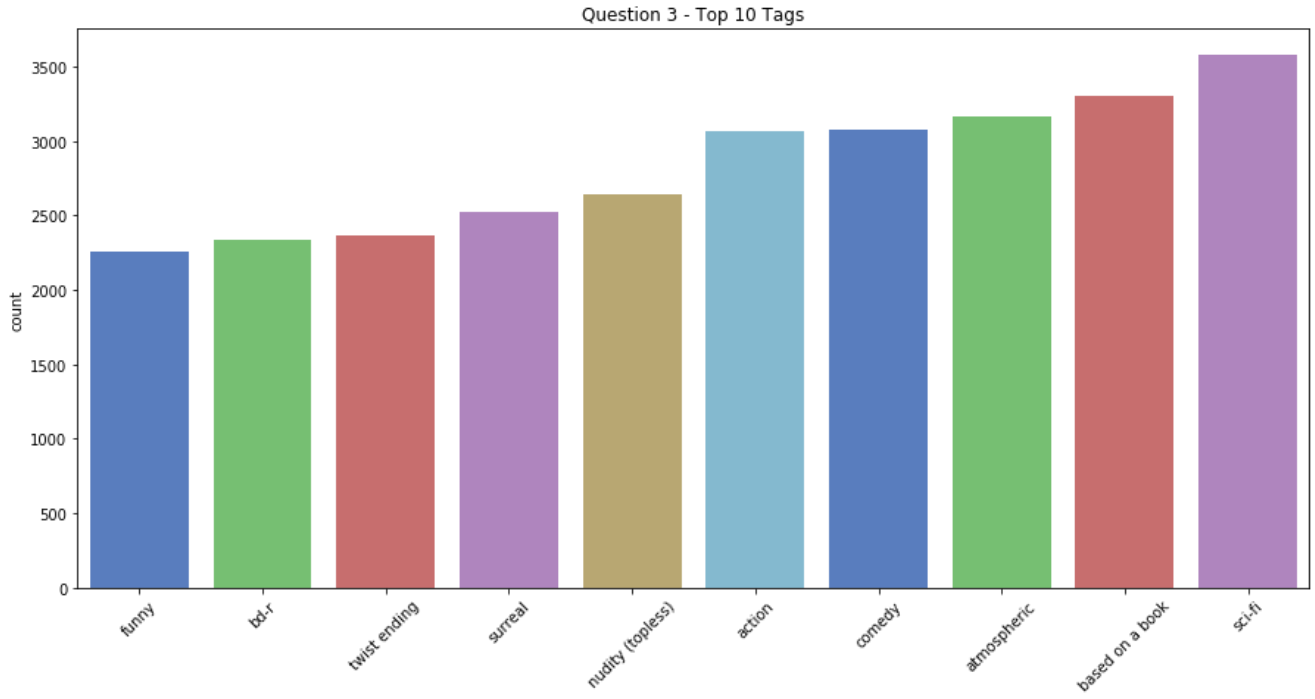
Also,we apply *reverse()* to **ordered_tag**

In [8]:

```
#Setting up the Figure
plt.figure(figsize=(15,7))
plt.xticks(rotation=45)

plot3 = sns.countplot(tag_top10, palette="muted", order=ordered_tag)
plot3.set_title("Question 3 - Top 10 Tags")
```

Out[8]:

```
Text(0.5,1,'Question 3 - Top 10 Tags')
```



Quetion 3 (Continued)

Finally, we create countplot. Order is specified by assigning reversed **ordered_tag**

In [9]:

```
#Question 4
from datetime import datetime

datetime_column = []
rating_column = []

with open("ratings.csv") as dataIn:
        reader = csv.DictReader(dataIn)

        for row in reader:

            datetime_column.append(row['timestamp'])

            #Question 5
            rating_column.append(float(row['rating']))
```

```
convert_year= []
for i in datetime_column:
    dt = datetime.fromtimestamp(int(i))
    year = int(dt.strftime('%Y'))
    convert_year.append(year)
```

Question 4

For Question 4 we stored *timestamp* values from ratings.csv to the list **datetime_column**. Then, we import library `from datetime import datetime` and convert each *timestamp* to the year value

In [25]:

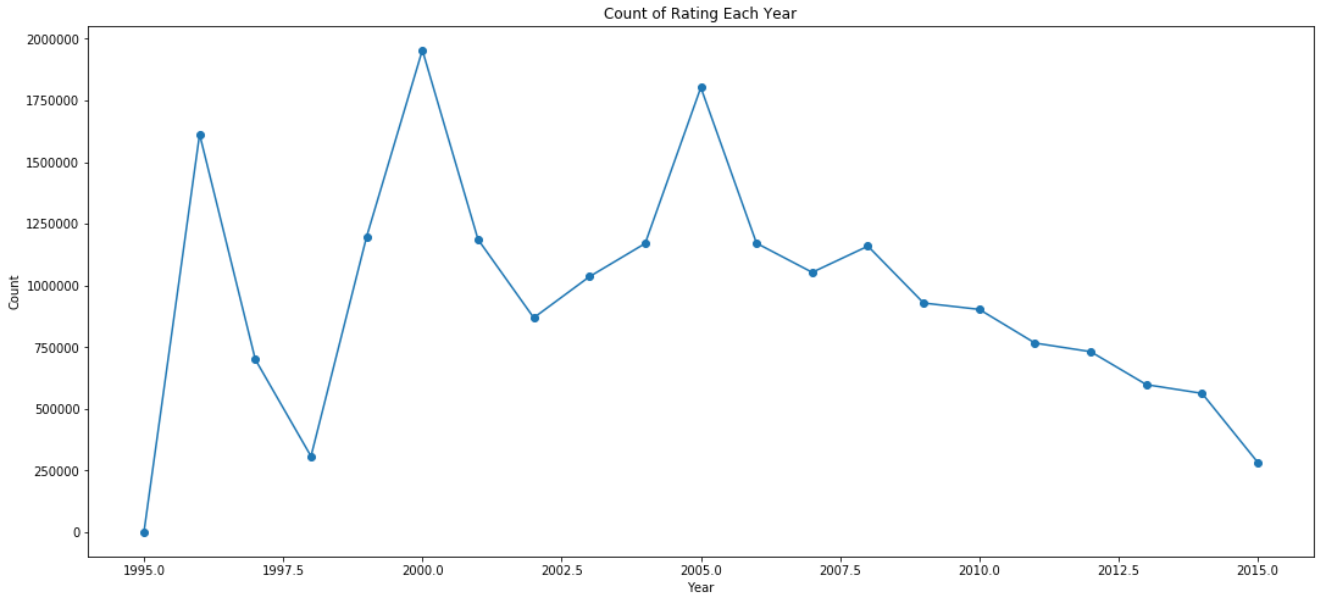
```
#Question 4
year_display = [year for year, year_count in Counter(convert_year).most_common()]
year_count = [year_count for year, year_count in Counter(convert_year).most_common()]

years_df = pd.DataFrame({
    "Year" : year_display,
    "Count" : year_count
})
years_df = years_df.sort_values(by=['Year'])
```

Quetion 4 (Continued)

Then, we create to separate lists **year_display** and **year_count** to store values of the year and number of time when that year occured in the ratings.csv

In [28]:

```
plt.figure(figsize=(18,8))

plt.plot( years_df['Year'], years_df['Count'], 'o-')
plt.title('Count of Rating Each Year')
plt.xlabel('Year')
plt.ylabel('Count')
#plt.yticks(year_display)

plt.show()
```



Quetion 4 (Continued)

Finally, we create a plot using matplotlib:

```
plt.plot( year_count, year_display, 'o-')
```

```
    plt.title('Count of Rating Each Year')
    plt.xlabel('Count')
    plt.ylabel('Year')
    plt.yticks(year_display)

    plt.show()
```

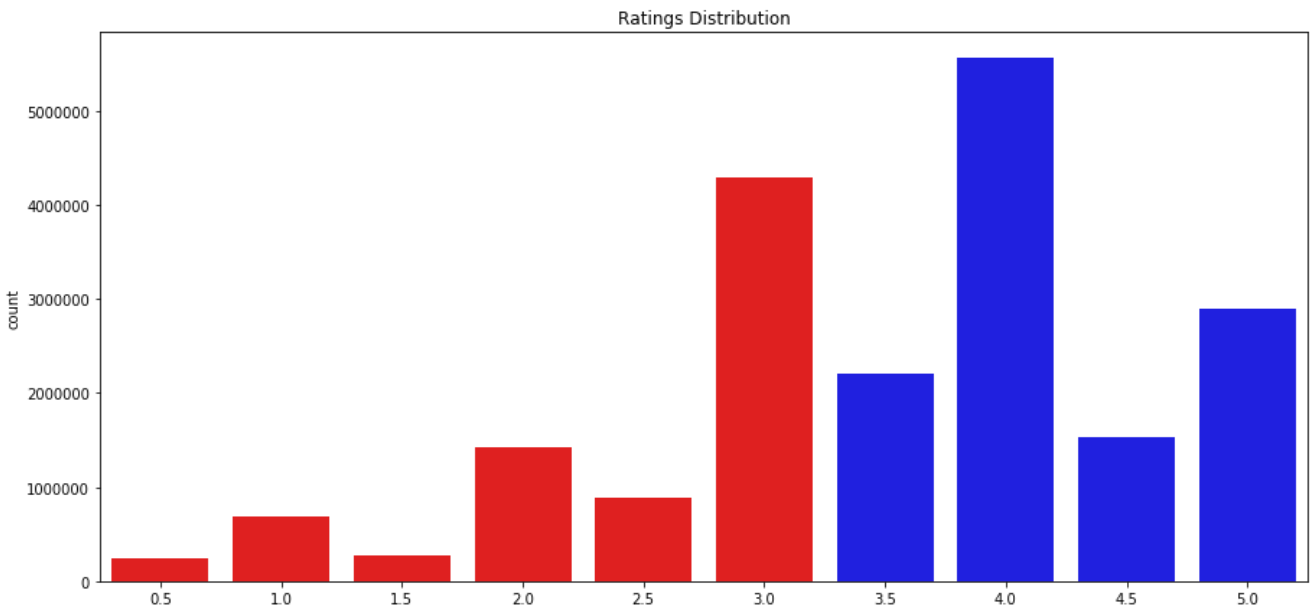
year_count is assigned to x-axis and **year_display** is assigned to y-axis. Then appropriate title and labels are given. `plt.yticks(year_display)` is used to specify that each value must be displayed on the y-axis

In [12]:

```
#Question 5

average = sum(rating_column) / len(rating_column)

plt.figure(figsize=(15,7))
color_choice = ['red' if (i < average) else 'blue' for i in rating_column]

plot4 = sns.countplot(rating_column, palette=color_choice)
plot4.set_title("Ratings Distribution")
```

Out[12]:

```
Text(0.5,1,'Ratings Distribution')
```



Question 5

All of the rating information is stored in the list **rating_column** which we have created in *Question 4*. Then to get the average number, we get the sum of the list and divide it by the number of values in **rating_column**. Then, we use list comprehension to compare each value of the rating to the average value and assign appropriate color name to the list **color_choice** : `color_choice = ['red' if (i < average) else 'blue' for i in rating_column]` Finally, this list of color values is assigned to the pallete

In [13]:

```
#Question 6

# 4 movies:
# 7451 "Mean girls"
# 8368 "Harry Potter and the Prisoner of Azkaban"
# 4306 "Shrek"
# 46062 "High School Musical"


movie1_column = []
```

```
movie2_column = []
movie3_column = []
movie4_column = []

with open("ratings.csv") as dataIn:
        reader = csv.DictReader(dataIn)

        for row in reader:

            if row['movieId'] == '7451':
                movie1_column.append(row['rating'])

            elif row['movieId'] == '8368':
                movie2_column.append(row['rating'])

            elif row['movieId'] == '4306':
                movie3_column.append(row['rating'])

            elif row['movieId'] == '46062':
                movie4_column.append(row['rating'])
```

Question 6

Movies: **Mean Girls Harry Potter and the Prisoner of Azkaban Shrek High School Musical**

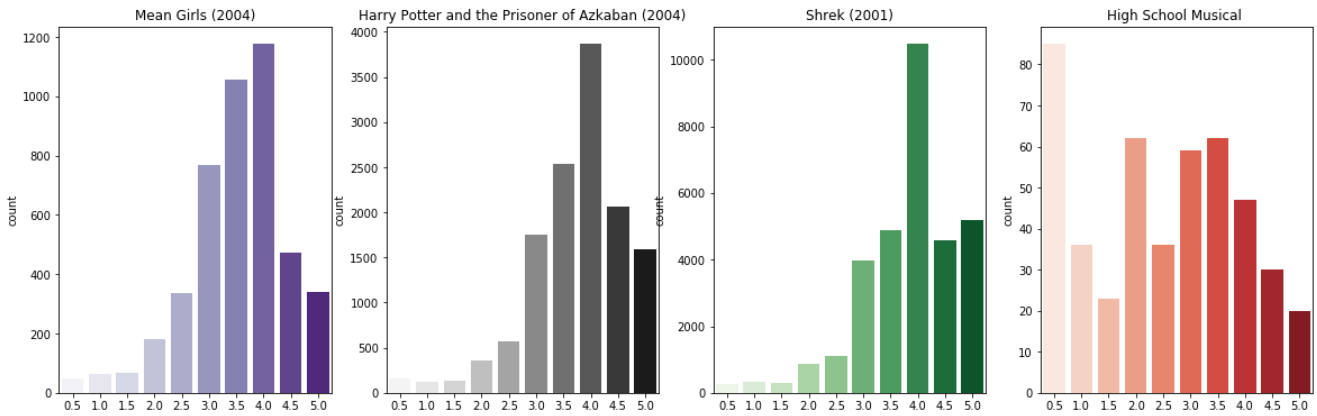
Each rating retrieved by *movieId* and is stored to the separate lists (**movie1_column, movie2_column...**).

In [14]:

```
f, (ax1, ax2, ax3, ax4) = plt.subplots(1, 4, figsize=(20, 6), sharex=True)

plot9 = sns.countplot(movie1_column, palette="Purples", ax=ax1)
plot9.set_title("Mean Girls (2004)")

plot6 = sns.countplot(movie2_column, palette="Greys", ax=ax2)
plot6.set_title("Harry Potter and the Prisoner of Azkaban (2004)")

plot7 = sns.countplot(movie3_column, palette="Greens", ax=ax3)
plot7.set_title("Shrek (2001)")

plot8 = sns.countplot(movie4_column, palette="Reds", ax=ax4)
plot8.set_title("High School Musical ")
```

Out[14]:

```
Text(0.5,1,'High School Musical ')
```



Question 6(Continued)

To create 4 subplots we use `f, (ax1, ax2, ax3, ax4) = plt.subplots(1, 4, figsize=(20, 6), sharex=True)`. This provide us with 4 subplots in one line with assigned names to each plot. Finally, we create each subplot by assigning appropriate movie list, different color palette, and different subplot(For example:`ax=ax1`)