Final Project Submission

Please fill out:

- Student name: Ryan Keats
- Student pace: self paced / part time / full time
- Scheduled project review date/time: September 11, 11:59pm.
- Instructor name: Hardik Idnani

In [1]:
```
# Firstly, I have imported pandas and numpy to help me work on my d
import pandas as pd
import numpy as np

# I have imported matplotlib to be able to peform my graphs.
import matplotlib.pyplot as plt
```

In [2]:
```
# This is the first data set I have used and defined to start my pr
movie_info = pd.read_csv("zippedData/bom.movie_gross.csv.gz")
movie_info.head()
```

Out[2]:

	title	studio	domestic_gross	foreign_gross	year
0	Toy Story 3	BV	415000000.0	652000000	2010
1	Alice in Wonderland (2010)	BV	334200000.0	691300000	2010
2	Harry Potter and the Deathly Hallows Part 1	WB	296000000.0	664300000	2010
3	Inception	WB	292600000.0	535700000	2010
4	Shrek Forever After	P/DW	238700000.0	513900000	2010

In [3]:
```
# I have used .iloc to display another way I searched the data fram
movie_info.iloc[0:5,:]
```

Out[3]:

	title	studio	domestic_gross	foreign_gross	year
0	Toy Story 3	BV	415000000.0	652000000	2010
1	Alice in Wonderland (2010)	BV	334200000.0	691300000	2010
2	Harry Potter and the Deathly Hallows Part 1	WB	296000000.0	664300000	2010
3	Inception	WB	292600000.0	535700000	2010
4	Shrek Forever After	P/DW	238700000.0	513900000	2010

In [4]:
```
# I have used .columns to outline the columns of the data set.
movie_info.columns
```

Out[4]:
```
Index(['title', 'studio', 'domestic_gross', 'foreign_gross', 'year
'], dtype='object')
```

In [5]:
```
# I have chosen .shape to show the size of the first data set I am
movie_info.shape
```

Out[5]:
```
(3387, 5)
```

In [6]:
```
# I have sorted the coloumn domestic_gross here, displaying from hi
movie_info = movie_info.sort_values('domestic_gross', ascending = F
movie_info.head(10)
```

Out[6]:

	title	studio	domestic_gross	foreign_gross	year
1872	Star Wars: The Force Awakens	BV	936700000.0	1,131.6	2015
3080	Black Panther	BV	700100000.0	646900000	2018
3079	Avengers: Infinity War	BV	678800000.0	1,369.5	2018
1873	Jurassic World	Uni.	652300000.0	1,019.4	2015
727	Marvel's The Avengers	BV	623400000.0	895500000	2012
2758	Star Wars: The Last Jedi	BV	620200000.0	712400000	2017
3082	Incredibles 2	BV	608600000.0	634200000	2018
2323	Rogue One: A Star Wars Story	BV	532200000.0	523900000	2016
2759	Beauty and the Beast (2017)	BV	504000000.0	759500000	2017
2324	Finding Dory	BV	486300000.0	542300000	2016

In [7]:
```
# I wanted to show the annual domestic_gross average, by using .mea
movie_info.groupby('year').mean()
```

Out[7]:

year	domestic_gross
2010	3.144559e+07
2011	2.535052e+07
2012	2.767584e+07
2013	3.128212e+07
2014	2.643923e+07
2015	2.461338e+07
2016	2.598996e+07
2017	3.416646e+07
2018	3.601042e+07

In [8]:
```
# In addition to showing the average annual domestic_gross, I wante
movie_info.groupby('year').sum()
```

Out[8]:

year	domestic_gross
2010	1.015693e+10
2011	1.006416e+10
2012	1.087661e+10
2013	1.079233e+10
2014	1.033774e+10
2015	1.105141e+10
2016	1.125365e+10
2017	1.093327e+10
2018	1.109121e+10

In [9]:
```
# I wanted to get a description of the domestic_gross and the Inner
movie_info['domestic_gross'].describe()
```

Out[9]:
```
count    3.359000e+03
mean     2.874585e+07
std      6.698250e+07
min      1.000000e+02
25%      1.200000e+05
50%      1.400000e+06
75%      2.790000e+07
max      9.367000e+08
Name: domestic_gross, dtype: float64
```

In [10]:
```
# I have sorted the coloumns in order to create order from high to
movie_info = movie_info.sort_values('foreign_gross', ascending = Tr
# I have added .head() & .tail() throughout my code cells to help w
movie_info.head()
```

Out[10]:

	title	studio	domestic_gross	foreign_gross	year
2760	The Fate of the Furious	Uni.	226000000.0	1,010.0	2017
1873	Jurassic World	Uni.	652300000.0	1,019.4	2015
1872	Star Wars: The Force Awakens	BV	936700000.0	1,131.6	2015
1874	Furious 7	Uni.	353000000.0	1,163.0	2015
3079	Avengers: Infinity War	BV	678800000.0	1,369.5	2018

In [11]:
```
# I have grouped the movies by year, to illustrate their yearly pro
movie_info.groupby('year').count().describe()
```

Out[11]:

	title	studio	domestic_gross	foreign_gross
count	9.000000	9.000000	9.000000	9.000000
mean	376.333333	375.777778	373.222222	226.333333
std	51.441715	51.538281	51.226892	50.857644
min	308.000000	308.000000	308.000000	173.000000
25%	328.000000	327.000000	323.000000	191.000000
50%	395.000000	394.000000	391.000000	205.000000
75%	400.000000	399.000000	397.000000	250.000000
max	450.000000	450.000000	449.000000	314.000000

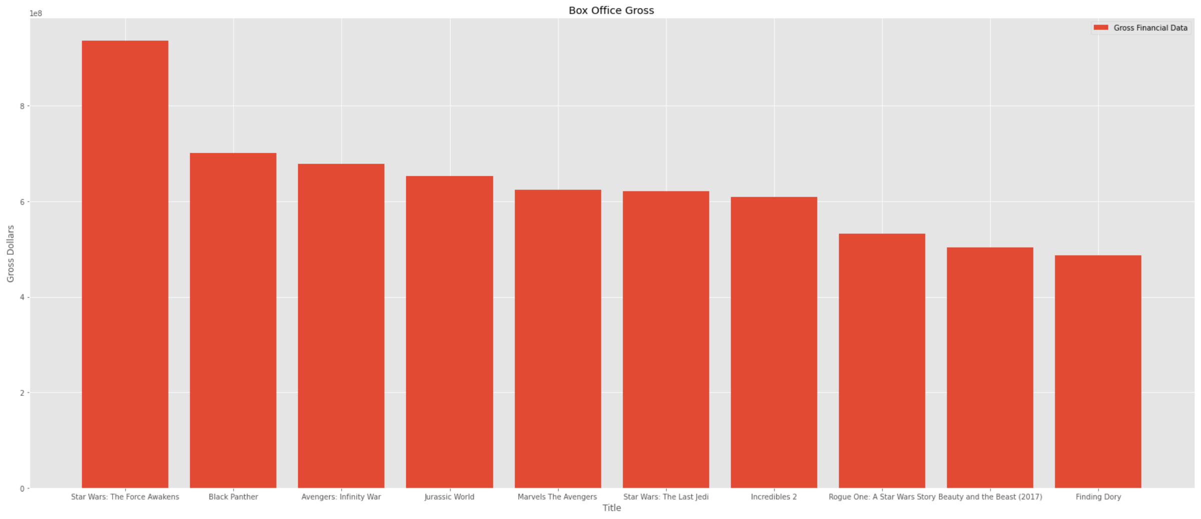
In [12]:
```
# I have added the value_counts() before .mean() to get the average
# Which is also displayed with a different code in the above cell.
movie_info['year'].value_counts().mean()
```

Out[12]:
```
376.3333333333333
```

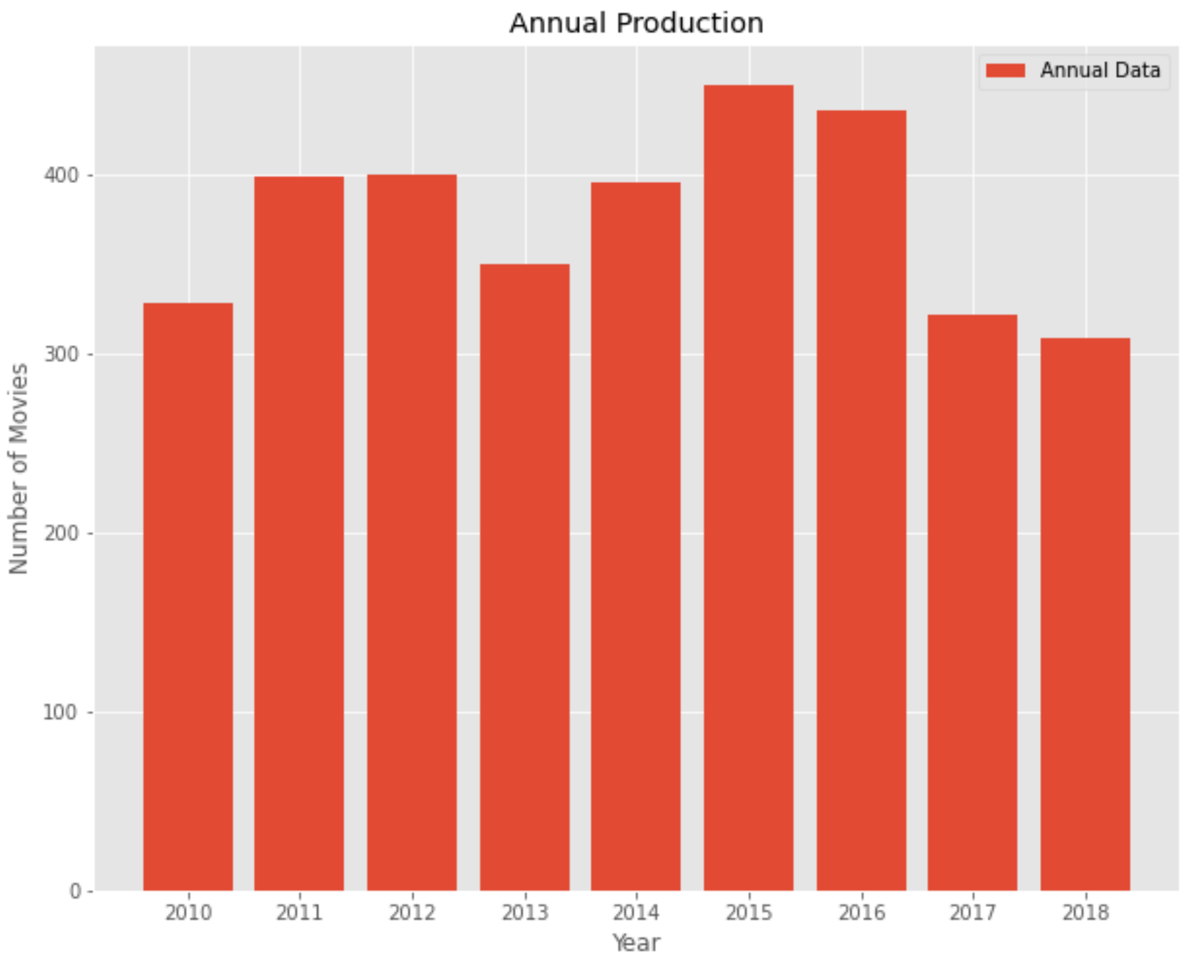
In [13]:
```
# I have shown how many movies are made each with .value_counts() a
movie_info['year'].value_counts().sort_values()
```

Out[13]:
```
2018    308
2017    321
2010    328
2013    350
2014    395
2011    399
2012    400
2016    436
2015    450
Name: year, dtype: int64
```

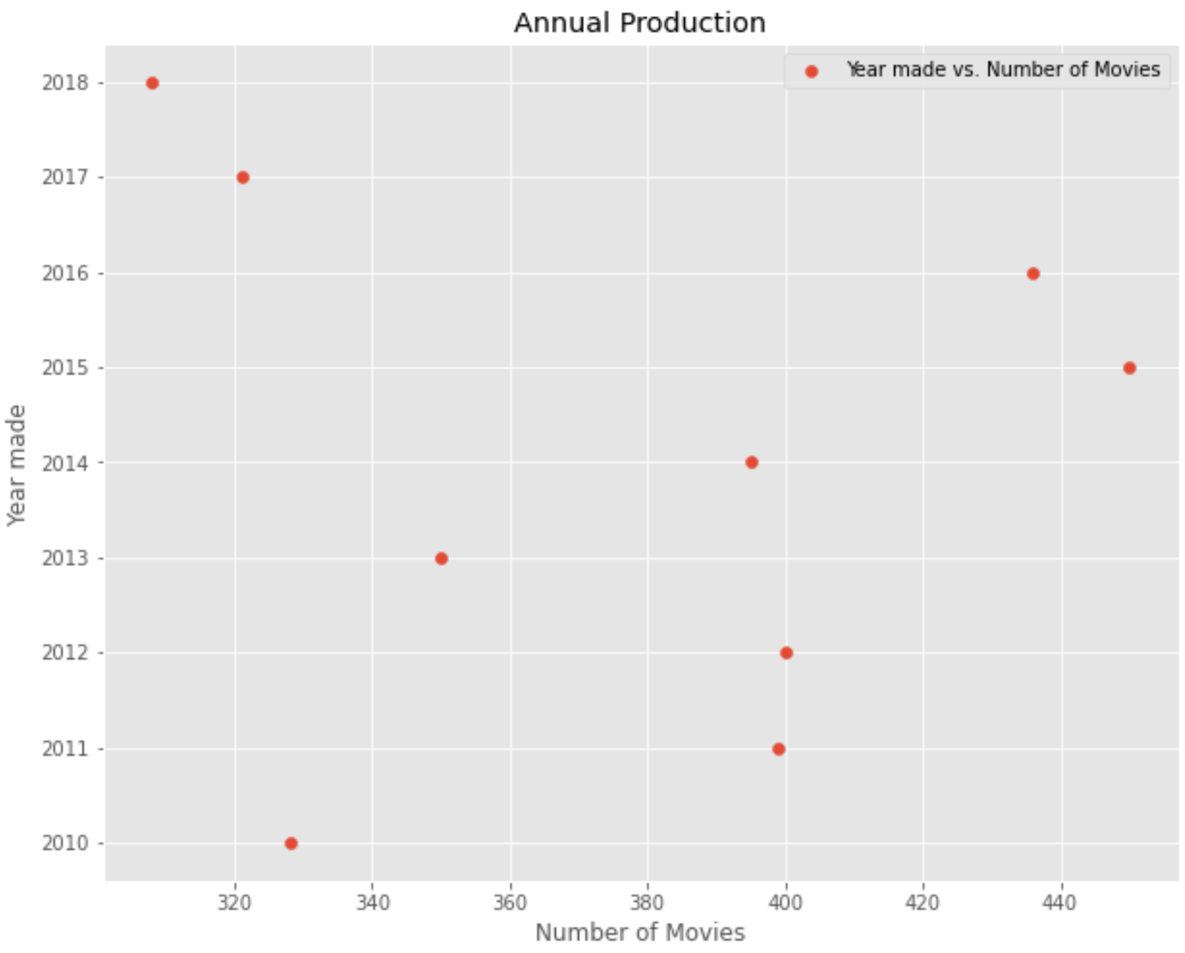
In [15]:
```
# I have chosen to use bar graphs as I feel they are the most simpl
y = [936700000.0, 700100000.0, 678800000.0, 652300000.0, 623400000.
    608600000.0, 532200000.0, 504000000.0, 486300000.0 ]
x = range(10)
labels = ['Star Wars: The Force Awakens', 'Black Panther', 'Avenger
          'Marvels The Avengers', 'Star Wars: The Last Jedi', 'Incr
        'Beauty and the Beast (2017)', 'Finding Dory' ]

# Create the plot
fig, ax = plt.subplots(figsize=(29,12))


ax.bar(x, y, tick_label = labels)

ax.set_title('Box Office Gross')
ax.set_ylabel('Gross Dollars')
ax.set_xlabel('Title');

ax.legend(['Gross Financial Data'], loc=1);

plt.style.use('ggplot')
```



In [16]:
```
# I have chosen to use bar graphs as I feel they are the most simpl
y = [328, 399, 400, 350, 395, 450, 436, 321, 308]
x = range(9)
labels = ['2010', '2011', '2012', '2013', '2014', '2015', '2016', '

# Create the plot
fig, ax = plt.subplots(figsize=(10, 8))


ax.bar(x, y, tick_label = labels)


ax.set_title('Annual Production')
ax.set_ylabel('Number of Movies')
ax.set_xlabel('Year');

ax.legend(['Annual Data'], loc=1);
```



In [17]:
```
# I have used a scatter plot for the same data as above, for a cont
# I have also showed this format to display that there is no correl
domestic_gross = [328, 399, 400, 350, 395, 450, 436, 321, 308]
year = [2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018]

# Create the plot
fig, ax = plt.subplots(figsize=(10,8))


ax.scatter(domestic_gross, year)

ax.set_xlabel('Number of Movies')
ax.set_ylabel('Year made')
ax.legend(['Year made' ' vs. ' 'Number of Movies'])
ax.set_title('Annual Production');

plt.style.use('ggplot')
```



In [18]:
```
# I have added the .value_counts() of the studio's production infor
movie_info['studio'].value_counts().head(10)
```

Out[18]:
```
IFC      166
Uni.     147
WB       140
Fox      136
Magn.    136
SPC      123
Sony     110
BV       106
LGF      103
Par.     101
Name: studio, dtype: int64
```

In [19]:
```
# I have also added .describe() to show general information of the
movie_info['studio'].describe()
```

Out[19]:
```
count     3382
unique     257
top        IFC
freq       166
Name: studio, dtype: object
```

In [20]:
```
# I wanted to add a command/comments cell gap between each of my da
# Also, for clarity and uniformity to display that I am moving onto
```

In [21]:
```
#I have imported pandas and numpy to help me work on my data sets.
import pandas as pd
import numpy as np

# I have imported matplotlib to be able to peform my graphs.
import matplotlib.pyplot as plt
```

In [22]:
```
#This is the second data set I have used to get my findings from.
movie_info2=pd.read_csv("zippedData/imdb.title.basics.csv.gz")
movie_info2.head()
```

Out[22]:

	tconst	primary_title	original_title	start_year	runtime_minutes	genres
0	tt0063540	Sunghursh	Sunghursh	2013	175.0	Action,Crime,Drama
1	tt0066787	One Day Before the Rainy Season	Ashad Ka Ek Din	2019	114.0	Biography,Drama
2	tt0069049	The Other Side of the Wind	The Other Side of the Wind	2018	122.0	Drama
3	tt0069204	Sabse Bada Sukh	Sabse Bada Sukh	2018	NaN	Comedy,Drama
4	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy

In [23]:
```
# I have used .columns to outline the columns of the data set.
movie_info2.columns
```

Out[23]:
```
Index(['tconst', 'primary_title', 'original_title', 'start_year',
       'runtime_minutes', 'genres'],
      dtype='object')
```

In [24]:
```
# I have chosen .shape to show the size of the first data set I am
movie_info2.shape
```

Out[24]:
```
(146144, 6)
```

In [25]:
```
#I wanted to use the .value_count() feature to display the top genr
movie_info2['genres'].value_counts().head(10)
```

Out[25]:
```
Documentary                 32185
Drama                       21486
Comedy                       9177
Horror                       4372
Comedy,Drama                 3519
Thriller                     3046
Action                       2219
Biography,Documentary        2115
Drama,Romance                2079
Comedy,Drama,Romance         1558
Name: genres, dtype: int64
```

In [26]:
```
#I have used .describe() on the runtime to outline the (IQR).
movie_info2['runtime_minutes'].describe()
```

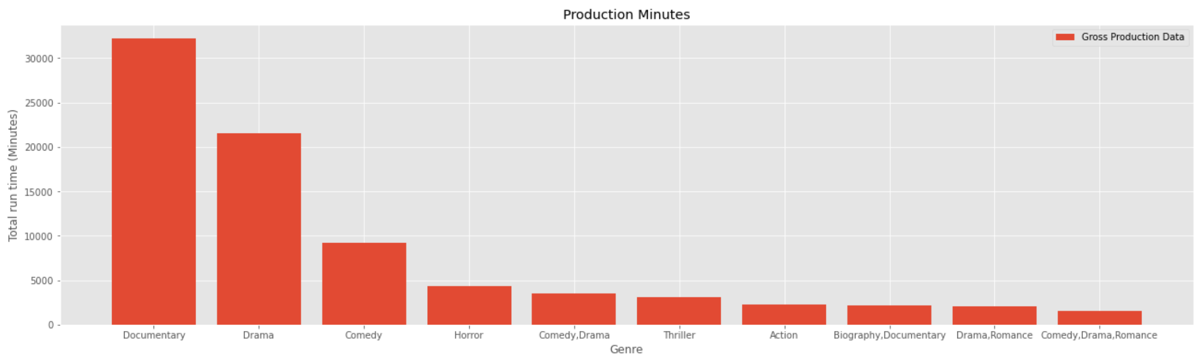
Out[26]:
```
count    114405.000000
mean         86.187247
std         166.360590
min           1.000000
25%          70.000000
50%          87.000000
75%          99.000000
max       51420.000000
Name: runtime_minutes, dtype: float64
```

In [27]:
```
# I have displayed the above information in a different format of c
# I prefer this outcome. But, thought it was good to display multip
movie_info2 = movie_info2.sort_values('runtime_minutes', ascending
movie_info2.describe()
```

Out[27]:

	start_year	runtime_minutes
count	146144.000000	114405.000000
mean	2014.621798	86.187247
std	2.733583	166.360590
min	2010.000000	1.000000
25%	2012.000000	70.000000
50%	2015.000000	87.000000
75%	2017.000000	99.000000
max	2115.000000	51420.000000

In [28]:
```
# I have chosen to use bar graphs as I feel they are the most simpl
y = [32185, 21486, 9177, 4372, 3519, 3046, 2219, 2115, 2079, 1558]
x = range(10)
labels = ['Documentary', 'Drama', 'Comedy', 'Horror', 'Comedy,Drama
          'Action', 'Biography,Documentary' ,  'Drama,Romance', 'Co

# Create the plot
fig, ax = plt.subplots(figsize=(22, 6))

ax.bar(x, y, tick_label = labels)

ax.set_title('Production Minutes')
ax.set_ylabel('Total run time (Minutes)')
ax.set_xlabel('Genre');

ax.legend(["Gross Production Data"], loc=1);

plt.style.use('ggplot')
```



In [29]:
```
# I wanted to add a command/comments cell gap between each of my da
# Also, for clarity and uniformity to display that I am moving onto
```

In [30]:
```
#I have imported pandas and numpy to help me work on my data sets.
import pandas as pd
import numpy as np

# I have imported matplotlib to be able to peform my graphs.
import matplotlib.pyplot as plt
```

In [31]:
```
#This is the third data set I have used to get my findings from.
movie_info3=pd.read_csv("zippedData/imdb.title.ratings.csv.gz")
movie_info3.head()
```

Out[31]:

	tconst	averagerating	numvotes
0	tt10356526	8.3	31
1	tt10384606	8.9	559
2	tt1042974	6.4	20
3	tt1043726	4.2	50352
4	tt1060240	6.5	21

In [32]:
```
# I have used .columns to outline the columns of the data set.
movie_info3.columns
```

Out[32]:
```
Index(['tconst', 'averagerating', 'numvotes'], dtype='object')
```

In [33]:
```
# I have chosen .shape to show the size of the third data set I am
movie_info3.shape
```

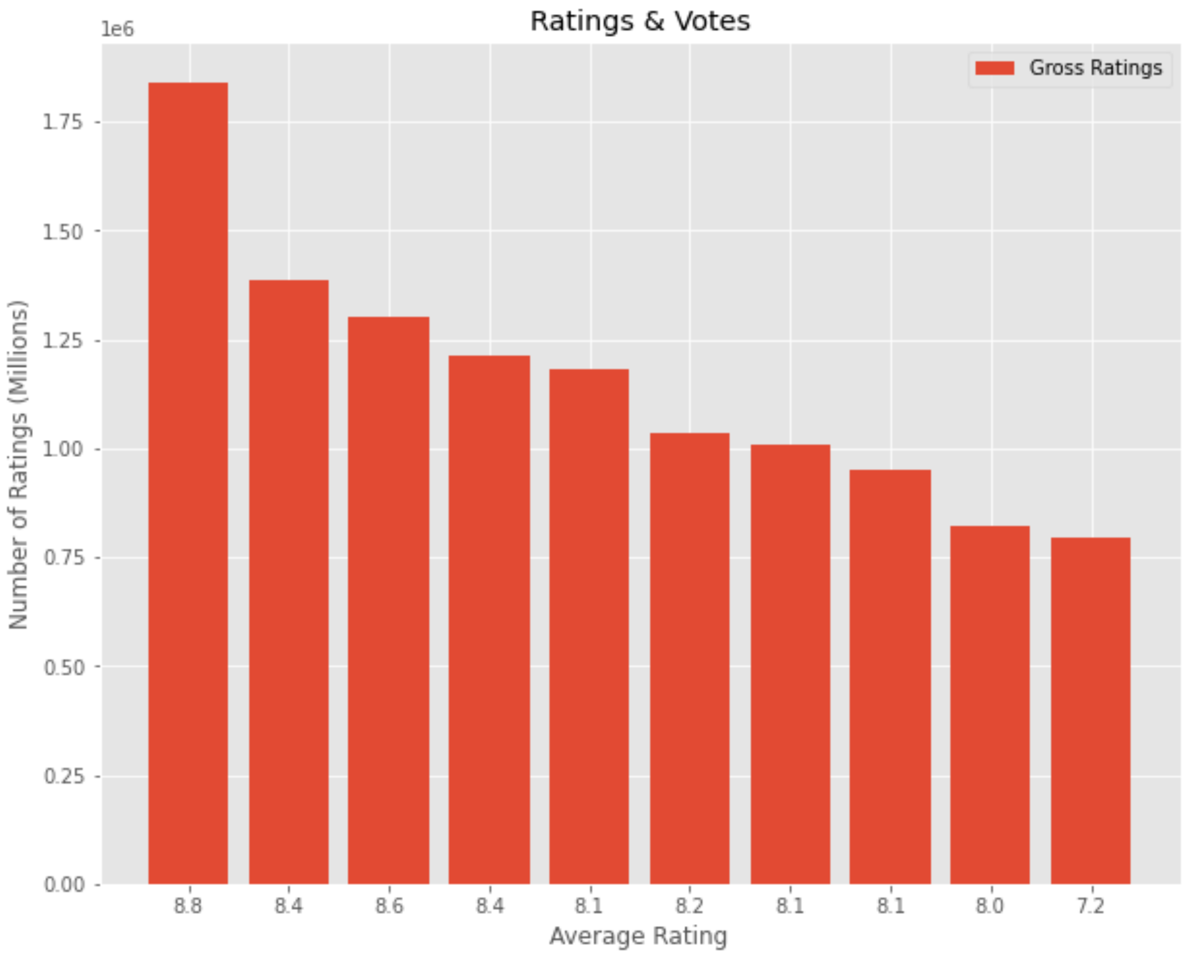
Out[33]:
```
(73856, 3)
```

In [34]:
```
# I have sorted the coloumn numvotes here, displaying from high to
movie_info3 = movie_info3.sort_values('numvotes', ascending = False
movie_info3.head(10)
```

Out[34]:

	tconst	averagerating	numvotes
63498	tt1375666	8.8	1841066
8738	tt1345836	8.4	1387769
24920	tt0816692	8.6	1299334
38058	tt1853728	8.4	1211405
48221	tt0848228	8.1	1183655
39356	tt0993846	8.2	1035358
3140	tt1130884	8.1	1005960
25777	tt2015381	8.1	948394
60518	tt1431045	8.0	820847
63506	tt1392170	7.2	795227

In [35]:
```
# I have chosen to use bar graphs as I feel they are the most simpl
y = [1841066, 1387769, 1299334, 1211405, 1183655, 1035358, 1005960,
x = range(10)
labels = ['8.8', '8.4', '8.6', '8.4', '8.1', '8.2', '8.1', '8.1', '

# Create the plot
fig, ax = plt.subplots(figsize=(10, 8))

ax.bar(x, y, tick_label = labels)

ax.set_title('Ratings & Votes')
ax.set_ylabel('Number of Ratings (Millions)')
ax.set_xlabel('Average Rating');

ax.legend(['Gross Ratings'], loc=1);

plt.style.use('ggplot')
```



In [36]:
```
# I have displayed a different way to produce the above information
movie_info3 = movie_info3.sort_values(by=['numvotes','averagerating
movie_info3.tail(10)
```

Out[36]:

	tconst	averagerating	numvotes
63506	tt1392170	7.2	795227
60518	tt1431045	8.0	820847
25777	tt2015381	8.1	948394
3140	tt1130884	8.1	1005960
39356	tt0993846	8.2	1035358
48221	tt0848228	8.1	1183655
38058	tt1853728	8.4	1211405
24920	tt0816692	8.6	1299334
8738	tt1345836	8.4	1387769
63498	tt1375666	8.8	1841066

In [37]:
```
# I have displayed the IQR with this code.
movie_info3 = movie_info3.sort_values('averagerating', ascending =
movie_info3.describe()
```

Out[37]:

	averagerating	numvotes
count	73856.000000	7.385600e+04
mean	6.332729	3.523662e+03
std	1.474978	3.029402e+04
min	1.000000	5.000000e+00
25%	5.500000	1.400000e+01
50%	6.500000	4.900000e+01
75%	7.400000	2.820000e+02
max	10.000000	1.841066e+06