INVESTIGATE AND ANALYZE MOVIE DATASET FROM THE MOVIE DATABASE (TMDB)

September 25, 2019

BY:

OLUREMI OMOWALE OSIBANJO

1 Project: Movies Data Analysis

1.1 Table of Contents

Introduction
Data Wrangling
Exploratory Data Analysis
Conclusions
Introduction

We will be investigating a movie dataset from TMDB (the movie database). Our analysis will focus on why a specific genre is popular among viewers over the years and which movie genres viewers likes the most in terms of voting. Furthermore, we will look at the properties associated with movies with high revenues.

```
In [2]: import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        import seaborn as sns

        % matplotlib inline
```

Data Wrangling

In this section of the report, we will load in the movie data from the database, this dataset will be cleaned, and then trimmed and prepared for analysis. This will give us a better exploration of our data for vitualization purposes.

1.1.1 General Properties

```
In [3]: # Load your data and print out a few lines. Perform operations to inspect data
        df = pd.read_csv('tmdb.csv')
        df.head()
```

Out[3]:

	id	imdb_id	popularity	budget	revenue	original_title	cast	homepage	director	tagline	...	overview	runtime	genres
0	135397	tt0369610	32.985763	150000000	1513528810	Jurassic World	Chris Pratt|Bryce Dallas Howard|Irrfan Khan|Vi...	http://www.jurassicworld.com/	Colin Trevorrow	The park is open.	...	Twenty-two years after the events of Jurassic ...	124	Action|Adventure|Science Fiction|Thriller
1	76341	tt1392190	28.419936	150000000	378436354	Mad Max: Fury Road	Tom Hardy|Charlize Theron|Hugh Keays-Byrne|Nic...	http://www.madmaxmovie.com/	George Miller	What a Lovely Day.	...	An apocalyptic story set in the furthest reach...	120	Action|Adventure|Science Fiction|Thriller
2	262500	tt2908446	13.112507	110000000	295238201	Insurgent	Shailene Woodley|Theo James|Kate Winslet|Ansel...	http://www.thedivergentseries.movie/#insurgent	Robert Schwentke	One Choice Can Destroy You	...	Beatrice Prior must confront her inner demons ...	119	Adventure|Science Fiction|Thriller
3	140607	tt2488496	11.173104	200000000	2068178225	Star Wars: The Force Awakens	Harrison Ford|Mark Hamill|Carrie Fisher|Adam D...	http://www.starwars.com/films/star-wars-episod...	J.J. Abrams	Every generation has a story.	...	Thirty years after defeating the Galactic Empi...	136	Action|Adventure|Science Fiction|Fantasy
4	168259	tt2820852	9.335014	190000000	1506249360	Furious 7	Vin Diesel|Paul Walker|Jason Statham|Michelle ...	http://www.furious7.com/	James Wan	Vengeance Hits Home	...	Deckard Shaw seeks revenge against Dominic Tor...	137	Action|Crime|Thriller

```
                                    production_companies release_date vote_count  \
0  Universal Studios|Amblin Entertainment|Legenda...       6/9/15       5562
1  Village Roadshow Pictures|Kennedy Miller Produ...      5/13/15       6185
2  Summit Entertainment|Mandeville Films|Red Wago...      3/18/15       2480
3           Lucasfilm|Truenorth Productions|Bad Robot     12/15/15       5292
4  Universal Pictures|Original Film|Media Rights ...       4/1/15       2947

   vote_average  release_year    budget_adj   revenue_adj
0           6.5          2015  1.379999e+08  1.392446e+09
1           7.1          2015  1.379999e+08  3.481613e+08
2           6.3          2015  1.012000e+08  2.716190e+08
3           7.5          2015  1.839999e+08  1.902723e+09
4           7.3          2015  1.747999e+08  1.385749e+09

[5 rows x 21 columns]
```

```
In [4]: df.shape
```

```
Out[4]: (10866, 21)
```

```
In [5]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10866 entries, 0 to 10865
Data columns (total 21 columns):
id                      10866 non-null int64
imdb_id                 10856 non-null object
popularity              10866 non-null float64
budget                  10866 non-null int64
revenue                 10866 non-null int64
original_title          10866 non-null object
cast                    10790 non-null object
homepage                2936 non-null object
director                10822 non-null object
tagline                 8042 non-null object
keywords                9373 non-null object
overview                10862 non-null object
runtime                 10866 non-null int64
genres                  10843 non-null object
production_companies    9836 non-null object
release_date            10866 non-null object
vote_count              10866 non-null int64
vote_average            10866 non-null float64
release_year            10866 non-null int64
budget_adj              10866 non-null float64
revenue_adj             10866 non-null float64
dtypes: float64(4), int64(6), object(11)
```

```
memory usage: 1.7+ MB
```

```
In [6]: df.describe()
```

Out[6]:

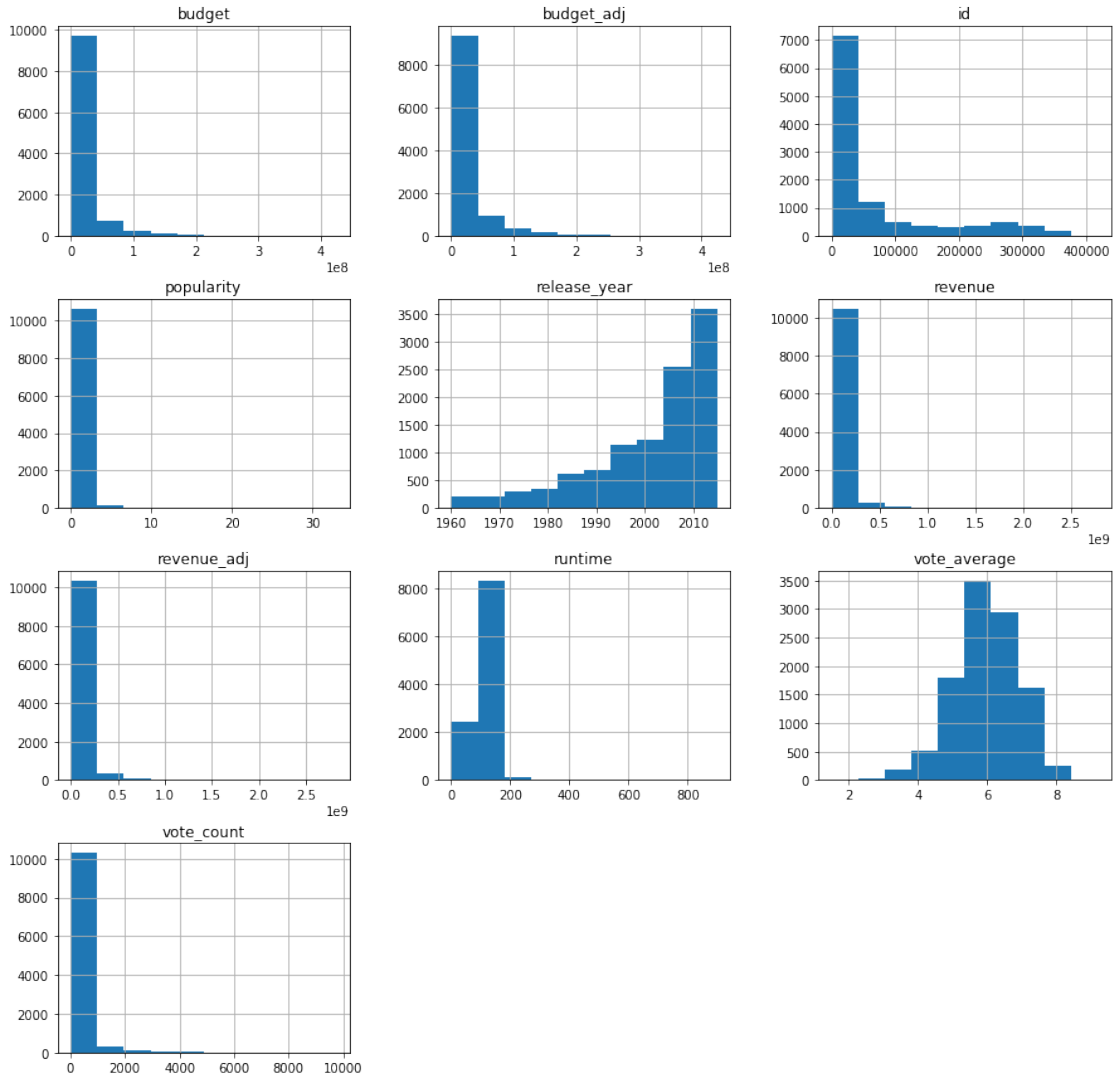
	id	popularity	budget	revenue	runtime
count	10866.000000	10866.000000	1.086600e+04	1.086600e+04	10866.000000
mean	66064.177434	0.646441	1.462570e+07	3.982332e+07	102.070863
std	92130.136561	1.000185	3.091321e+07	1.170035e+08	31.381405
min	5.000000	0.000065	0.000000e+00	0.000000e+00	0.000000
25%	10596.250000	0.207583	0.000000e+00	0.000000e+00	90.000000
50%	20669.000000	0.383856	0.000000e+00	0.000000e+00	99.000000
75%	75610.000000	0.713817	1.500000e+07	2.400000e+07	111.000000
max	417859.000000	32.985763	4.250000e+08	2.781506e+09	900.000000

	vote_count	vote_average	release_year	budget_adj	revenue_adj
count	10866.000000	10866.000000	10866.000000	1.086600e+04	1.086600e+04
mean	217.389748	5.974922	2001.322658	1.755104e+07	5.136436e+07
std	575.619058	0.935142	12.812941	3.430616e+07	1.446325e+08
min	10.000000	1.500000	1960.000000	0.000000e+00	0.000000e+00
25%	17.000000	5.400000	1995.000000	0.000000e+00	0.000000e+00
50%	38.000000	6.000000	2006.000000	0.000000e+00	0.000000e+00
75%	145.750000	6.600000	2011.000000	2.085325e+07	3.369710e+07
max	9767.000000	9.200000	2015.000000	4.250000e+08	2.827124e+09

```
In [7]: df.nunique()
```

```
Out[7]: id                      10865
        imdb_id                 10855
        popularity              10814
        budget                    557
        revenue                  4702
        original_title          10571
        cast                    10719
        homepage                 2896
        director                 5067
        tagline                  7997
        keywords                 8804
        overview                10847
        runtime                   247
        genres                   2039
        production_companies     7445
        release_date             5909
        vote_count               1289
        vote_average               72
        release_year               56
        budget_adj               2614
        revenue_adj              4840
        dtype: int64
```

```
In [8]: df.hist(figsize=(15, 15));
```



2 Data Cleaning : We will clean our data and remove unwanted/Null rows and columns, as well as separate rows with mutiple values.

3 Here we will see that there are some columns that we really don't need. we will remove those columns.

```
In [9]: df.head(2)
```

```
Out[9]:       id     imdb_id  popularity     budget     revenue      original_title  \
        0  135397  tt0369610   32.985763  150000000  1513528810      Jurassic World
        1   76341  tt1392190   28.419936  150000000   378436354  Mad Max: Fury Road
```

```
                                                cast  \
0  Chris Pratt|Bryce Dallas Howard|Irrfan Khan|Vi...
1  Tom Hardy|Charlize Theron|Hugh Keays-Byrne|Nic...

                        homepage         director             tagline  \
0  http://www.jurassicworld.com/  Colin Trevorrow   The park is open.
1    http://www.madmaxmovie.com/    George Miller  What a Lovely Day.

       ...                                                overview runtime  \
0      ...       Twenty-two years after the events of Jurassic ...     124
1      ...       An apocalyptic story set in the furthest reach...     120

                                      genres  \
0  Action|Adventure|Science Fiction|Thriller
1  Action|Adventure|Science Fiction|Thriller

                                production_companies release_date vote_count  \
0  Universal Studios|Amblin Entertainment|Legenda...       6/9/15       5562
1  Village Roadshow Pictures|Kennedy Miller Produ...      5/13/15       6185

   vote_average  release_year    budget_adj   revenue_adj
0           6.5          2015  1.379999e+08  1.392446e+09
1           7.1          2015  1.379999e+08  3.481613e+08

[2 rows x 21 columns]
```

4 Columns that will be removed are homepage', 'overview', 'tagline', 'imdb_id', 'keywords', 'production_companies', 'release_date

```
In [10]: df.drop(['homepage', 'overview', 'tagline', 'imdb_id', 'keywords', 'production_companie

In [11]: df.head(2)

Out[11]:        id  popularity     budget     revenue      original_title  \
         0  135397   32.985763  150000000  1513528810      Jurassic World
         1   76341   28.419936  150000000   378436354  Mad Max: Fury Road

                                                cast         director  \
         0  Chris Pratt|Bryce Dallas Howard|Irrfan Khan|Vi...  Colin Trevorrow
         1  Tom Hardy|Charlize Theron|Hugh Keays-Byrne|Nic...    George Miller

            runtime                                      genres  vote_count  \
         0      124  Action|Adventure|Science Fiction|Thriller        5562
         1      120  Action|Adventure|Science Fiction|Thriller        6185

            vote_average  release_year    budget_adj   revenue_adj
```

```
0           6.5          2015  1.379999e+08  1.392446e+09
1           7.1          2015  1.379999e+08  3.481613e+08
```

5 we check for duplicates and drop all duplicates

```
In [12]: sum(df.duplicated())
```

```
Out[12]: 1
```

```
In [13]: df.drop_duplicates(inplace=True)
```

```
In [14]: df.dropna(inplace=True)
```

```
In [15]: # checks if any of columns in 2008 have null values - should print False
         df.isnull().sum().any()
```

```
Out[15]: False
```

6 Furthermore, we need to separate columns that has more than two values as we can see that the genres, cast and director columns has inputs separated with the pipe character

```
In [16]: # Columns with pipes in movies
         df_pipe = df[df['genres'].str.contains('|')]
         df_pipe.head(2)
```

```
Out[16]:        id  popularity     budget     revenue      original_title  \
         0  135397   32.985763  150000000  1513528810      Jurassic World
         1   76341   28.419936  150000000   378436354  Mad Max: Fury Road

                                                 cast         director  \
         0  Chris Pratt|Bryce Dallas Howard|Irrfan Khan|Vi...  Colin Trevorrow
         1  Tom Hardy|Charlize Theron|Hugh Keays-Byrne|Nic...    George Miller

            runtime                                     genres  vote_count  \
         0      124  Action|Adventure|Science Fiction|Thriller        5562
         1      120  Action|Adventure|Science Fiction|Thriller        6185

            vote_average  release_year    budget_adj   revenue_adj
         0           6.5          2015  1.379999e+08  1.392446e+09
         1           7.1          2015  1.379999e+08  3.481613e+08
```

```
In [17]: # create four copies of the columns with pipes dataframe
         df1 = df_pipe.copy()
         df2 = df_pipe.copy()
         df3 = df_pipe.copy()
         df4 = df_pipe.copy()
```

```
# Each one should look like this
df1.head(1)
```

Out[17]:

```
        id  popularity     budget     revenue  original_title  \
0  135397   32.985763  150000000  1513528810  Jurassic World

                                                cast         director  \
0  Chris Pratt|Bryce Dallas Howard|Irrfan Khan|Vi...  Colin Trevorrow

   runtime                                     genres  vote_count  \
0      124  Action|Adventure|Science Fiction|Thriller        5562

   vote_average  release_year    budget_adj   revenue_adj
0           6.5          2015  1.379999e+08  1.392446e+09
```

In [18]:

```
# columns to split by "|"
split_columns = ['genres', 'cast', 'director']

# apply split function to each column of each dataframe copy
for c in split_columns:
    df1[c] = df1[c].astype(str).str.split("|").str[0]
    df2[c] = df2[c].astype(str).str.split("|").str[1]
    df3[c] = df3[c].astype(str).str.split("|").str[2]
    df4[c] = df4[c].astype(str).str.split("|").str[3]
```

In [19]:

```
df4.head(2)
```

Out[19]:

```
        id  popularity     budget     revenue      original_title  \
0  135397   32.985763  150000000  1513528810      Jurassic World
1   76341   28.419936  150000000   378436354  Mad Max: Fury Road

                cast director  runtime    genres  vote_count  vote_average  \
0  Vincent D'Onofrio      NaN      124  Thriller        5562           6.5
1     Nicholas Hoult      NaN      120  Thriller        6185           7.1

   release_year    budget_adj   revenue_adj
0          2015  1.379999e+08  1.392446e+09
1          2015  1.379999e+08  3.481613e+08
```

7 Now we will combine each dataframes using the pandas 'append'

In [20]:

```
# combine dataframes to add to the original dataframe
new_df = df1.append(df2)
```

In [21]:

```
new_df2 = df3.append(df4)
```

In [22]:

```
new_rows = new_df.append(new_df2)
```

```
In [23]: new_rows.head(2)
```

```
Out[23]:       id  popularity     budget     revenue      original_title         cast  \
         0  135397   32.985763  150000000  1513528810      Jurassic World  Chris Pratt
         1   76341   28.419936  150000000   378436354  Mad Max: Fury Road    Tom Hardy

                director  runtime  genres  vote_count  vote_average  release_year  \
         0  Colin Trevorrow      124  Action        5562           6.5          2015
         1    George Miller      120  Action        6185           7.1          2015

             budget_adj   revenue_adj
         0  1.379999e+08  1.392446e+09
         1  1.379999e+08  3.481613e+08
```

```
In [24]: # drop the original pipe rows from the dataframe
         df.drop(df_pipe.index, inplace=True)

         # add in our newly separated rows
         df = df.append(new_rows, ignore_index=True)
```

```
In [25]: df.head()
```

```
Out[25]:       id  popularity     budget     revenue                original_title  \
         0  135397   32.985763  150000000  1513528810                Jurassic World
         1   76341   28.419936  150000000   378436354            Mad Max: Fury Road
         2  262500   13.112507  110000000   295238201                     Insurgent
         3  140607   11.173104  200000000  2068178225  Star Wars: The Force Awakens
         4  168259    9.335014  190000000  1506249360                     Furious 7

                       cast          director  runtime     genres  vote_count  \
         0       Chris Pratt   Colin Trevorrow      124     Action        5562
         1         Tom Hardy     George Miller      120     Action        6185
         2  Shailene Woodley  Robert Schwentke      119  Adventure        2480
         3     Harrison Ford       J.J. Abrams      136     Action        5292
         4        Vin Diesel         James Wan      137     Action        2947

            vote_average  release_year    budget_adj   revenue_adj
         0           6.5          2015  1.379999e+08  1.392446e+09
         1           7.1          2015  1.379999e+08  3.481613e+08
         2           6.3          2015  1.012000e+08  2.716190e+08
         3           7.5          2015  1.839999e+08  1.902723e+09
         4           7.3          2015  1.747999e+08  1.385749e+09
```

8 We will again check the new poperties of our dataset

```
In [26]: df.shape
```

```
Out[26]: (42924, 14)
```

```
In [27]: # checks if any of columns in 2008 have null values - should print False
         df.isnull().sum().any()
```

```
Out[27]: True
```

```
In [28]: df.dropna(inplace=True)
```

```
In [29]: # checks if any of columns in 2008 have null values - should print False
         df.isnull().sum().any()
```

```
Out[29]: False
```

```
In [30]: df.shape
```

```
Out[30]: (11353, 14)
```

Exploratory Data Analysis

8.0.1 Research Question: Which movie genres was voted for the most among viewers

```
In [31]: df.vote_count.min()
```

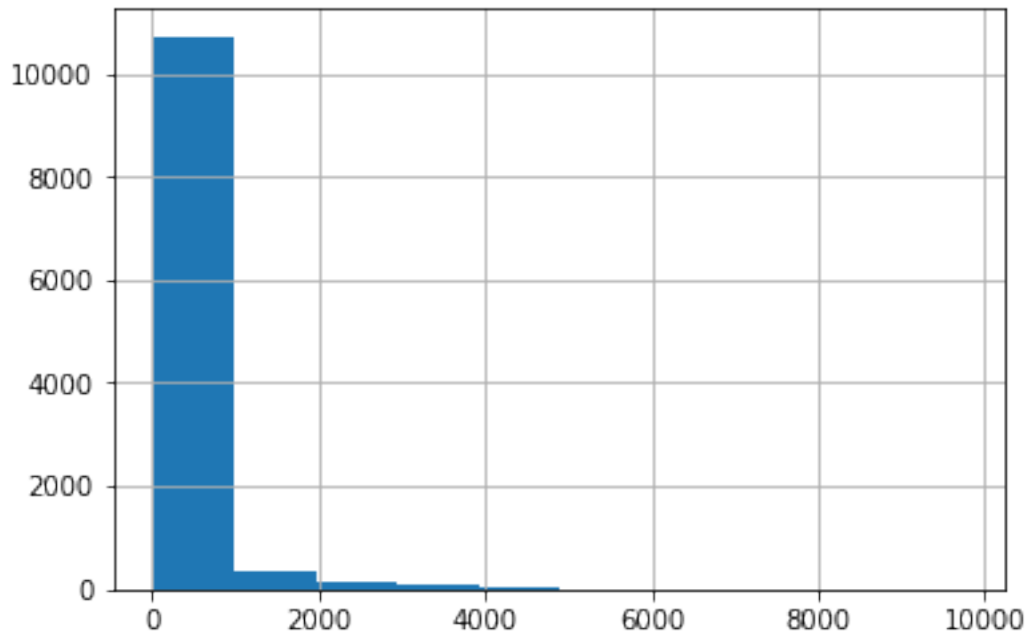
```
Out[31]: 10
```

```
In [32]: df.vote_count.mean()
```

```
Out[32]: 226.56998150268652
```

9 Distribution of votes among viewers

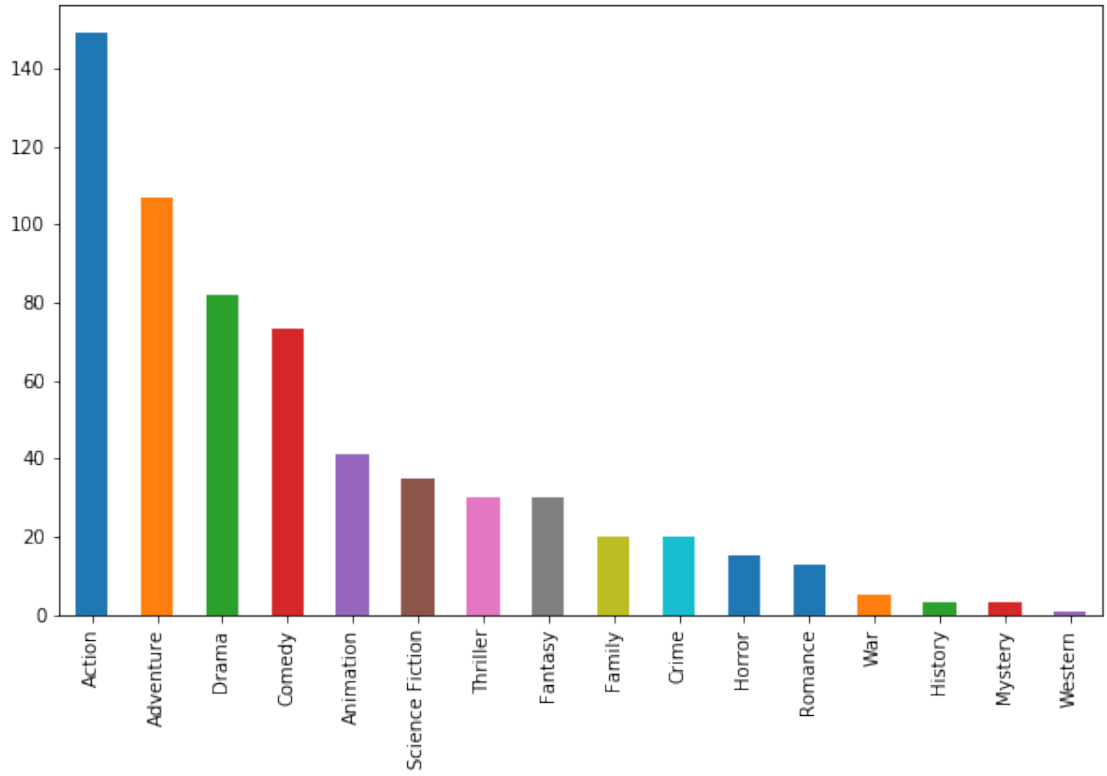
```
In [33]: df.vote_count.hist();
```



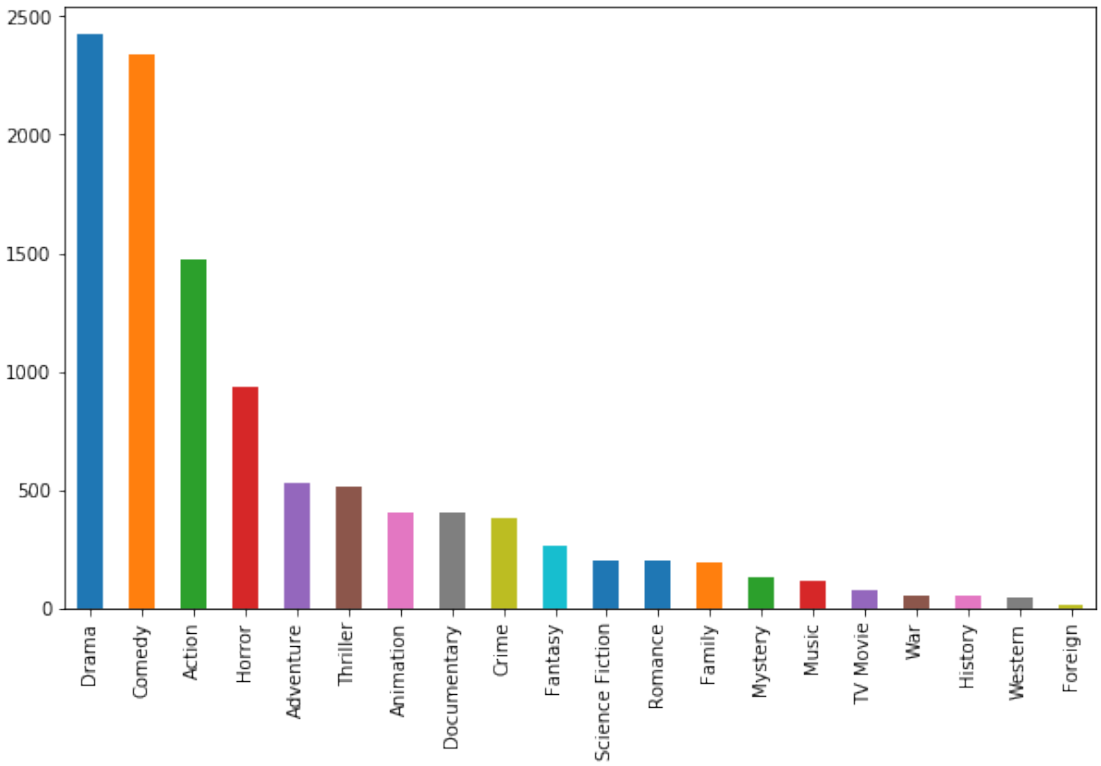
10 Let us explore the most voted for genres among viewers

```
In [34]: df_va = df[df['vote_count']  > 1000]
         df_vb = df[df['vote_count'] <= 1000]
```

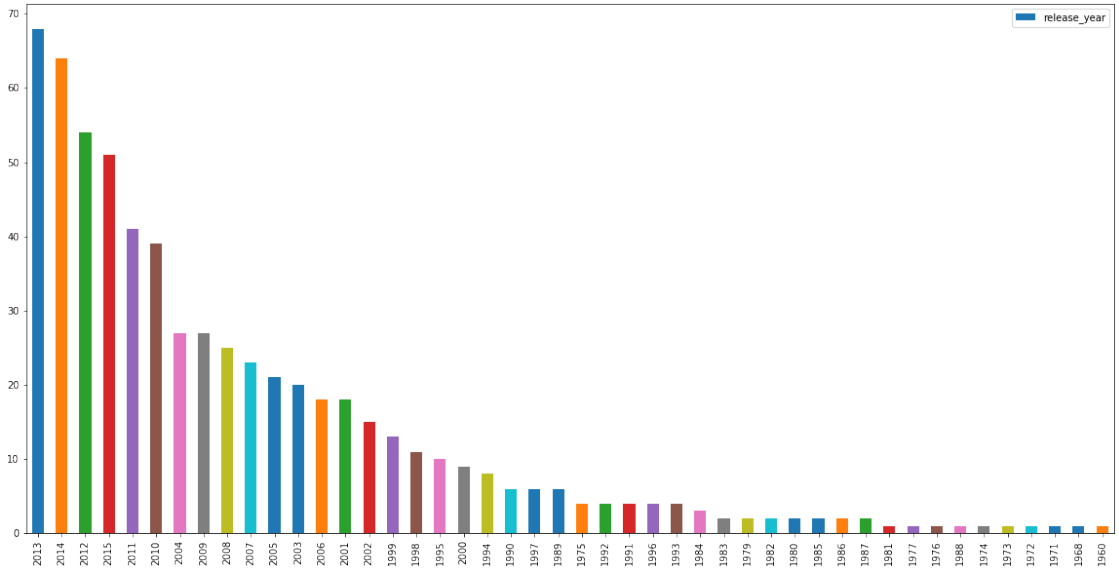
```
In [69]: df_va['genres'].value_counts().plot(kind = 'bar', figsize = (10, 6));
```




```
In [68]: df_vb['genres'].value_counts().plot(kind = 'bar',figsize = (10, 6));
```




```
In [88]: df_va['release_year'].value_counts().plot(kind = 'bar', figsize= (20,10))
         plt.legend();
```



11 Most popular movie among viewers

```
In [38]: df.popularity.max()
```

```
Out[38]: 32.985762999999999
```

```
In [39]: df[df.popularity == 32.985762999999999]
```

```
Out[39]:        id  popularity     budget     revenue  original_title         cast  \
         0  135397   32.985763  150000000  1513528810  Jurassic World  Chris Pratt

                director  runtime  genres  vote_count  vote_average  release_year  \
         0  Colin Trevorrow      124  Action        5562           6.5          2015

              budget_adj   revenue_adj
         0  1.379999e+08  1.392446e+09
```

11.0.1 Research Question 2 : Most popular genres among viewers!

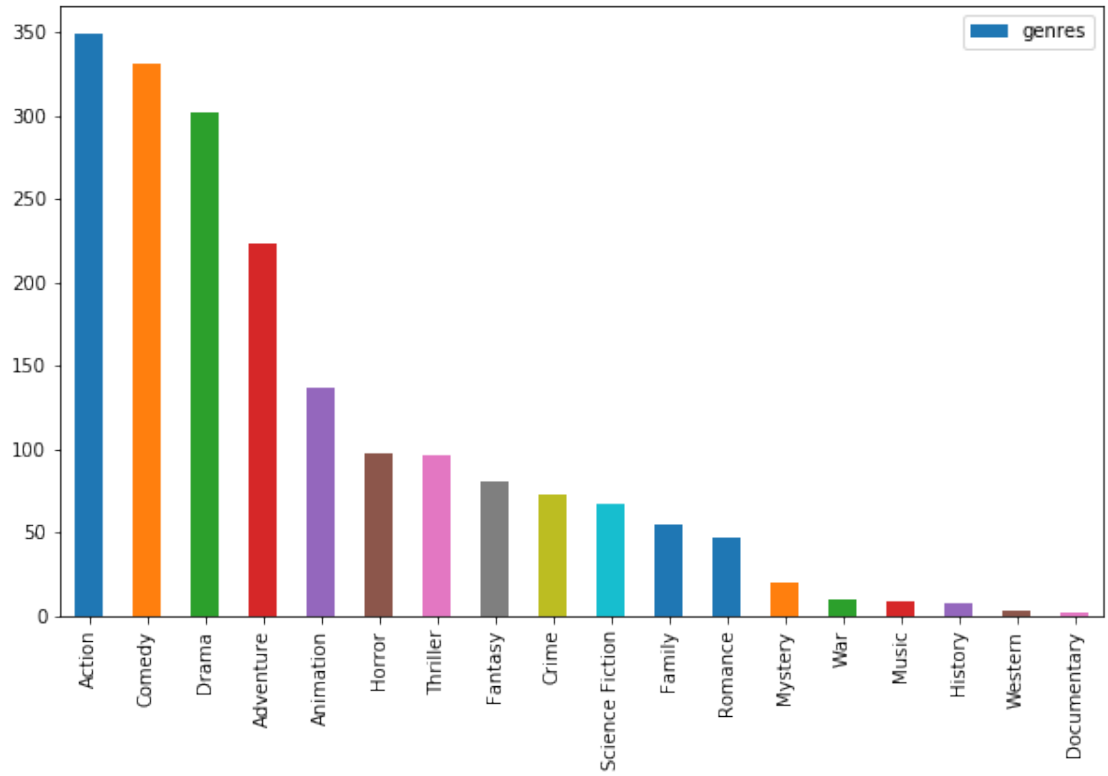
```
In [40]: # we create a mask
         df_pa = df[df['popularity']  > 1]
         df_pb = df[df['popularity'] <= 1]
```

```
In [41]: df.popularity.max()
```

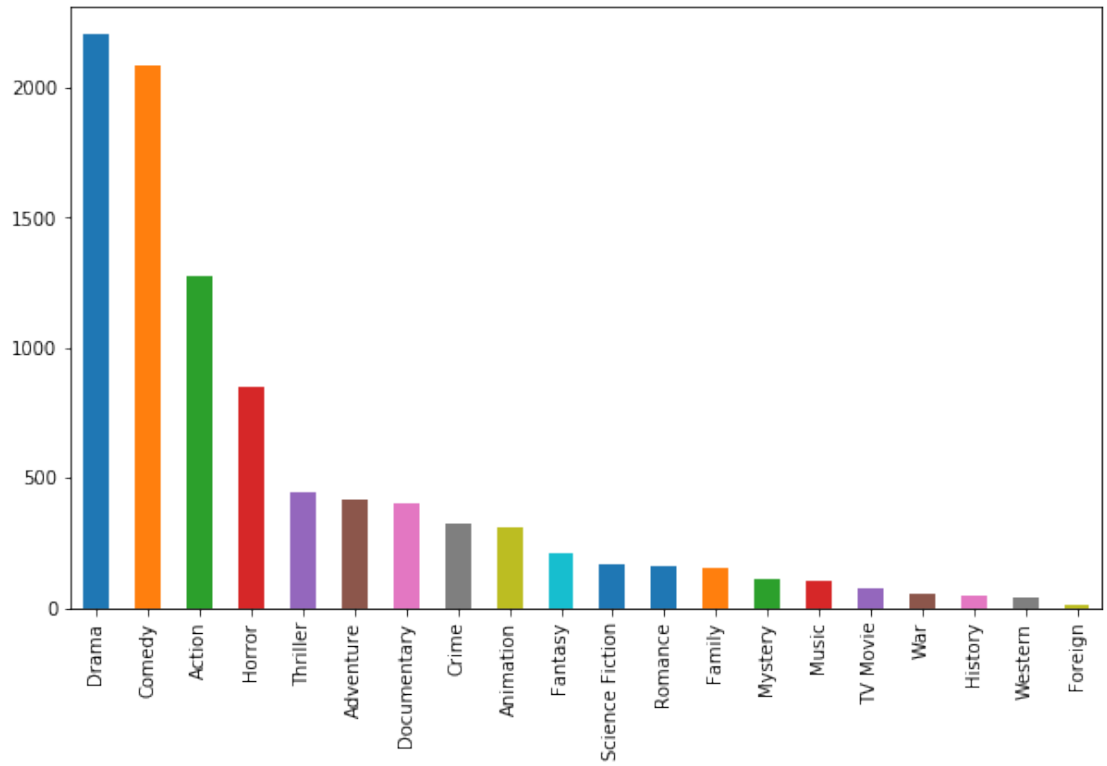
```
Out[41]: 32.985762999999999
```

12 Action movies is the most popular genre among viewers

```
In [89]: df_pa['genres'].value_counts().plot(kind = 'bar', figsize = (10, 6))
         plt.legend();
```

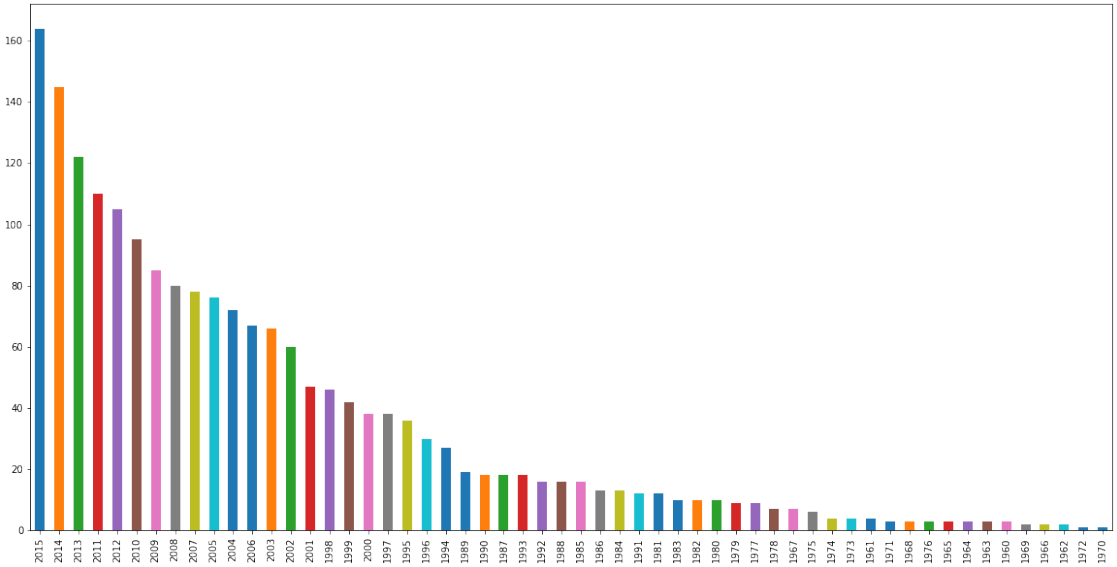



```
In [66]: df_pb['genres'].value_counts().plot(kind = 'bar',figsize = (10, 6));
```



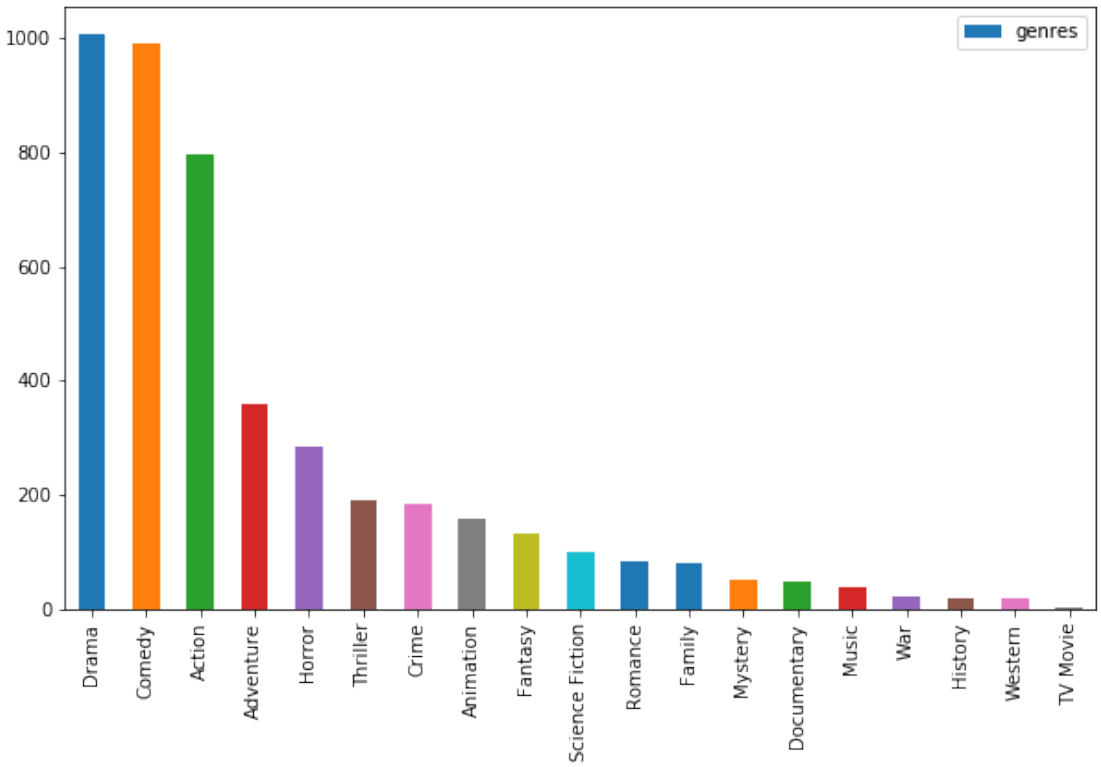
13 Movies produced in 2015 are the most popular movies

```
In [44]: df_pa['release_year'].value_counts().plot(kind = 'bar', figsize= (20,10));
```

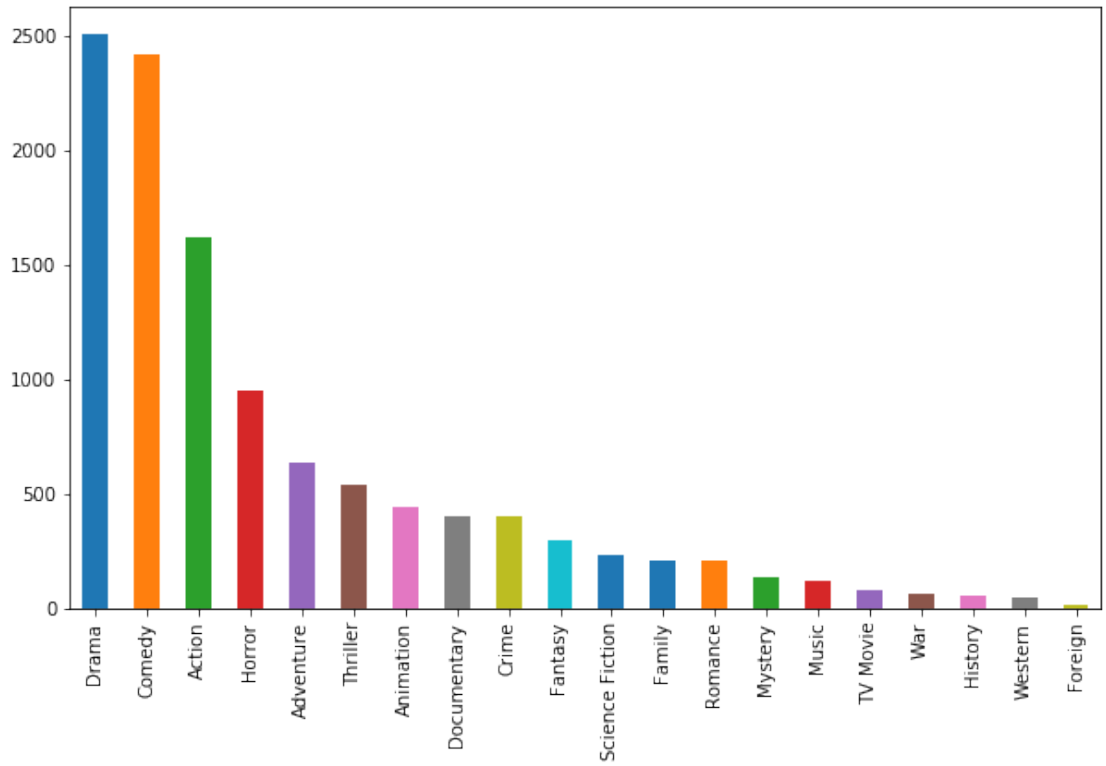


14 The movie that generated the highest revenue

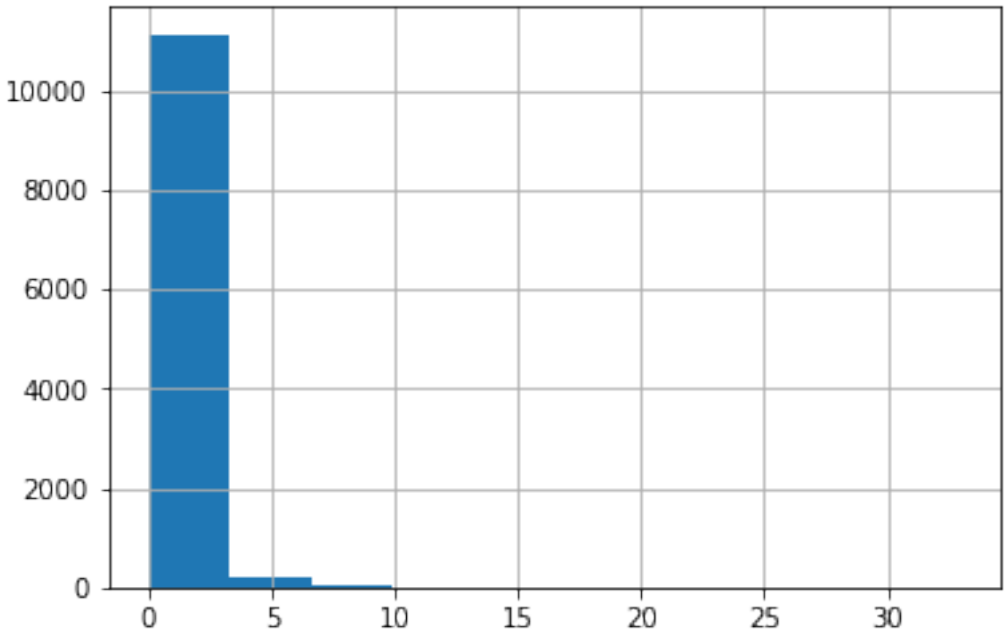
```
In [45]: #the movie with the highest revenue
         df.revenue.mean()

Out[45]: 41867167.889985025

In [46]: # we create a mask for revenue

         df_ra = df[df['revenue']  > 1000000]
         df_rb = df[df['vote_count'] <= 1000000]

In [63]: df_ra['genres'].value_counts().plot(kind = 'bar', figsize = (10, 6))
         plt.legend();
```



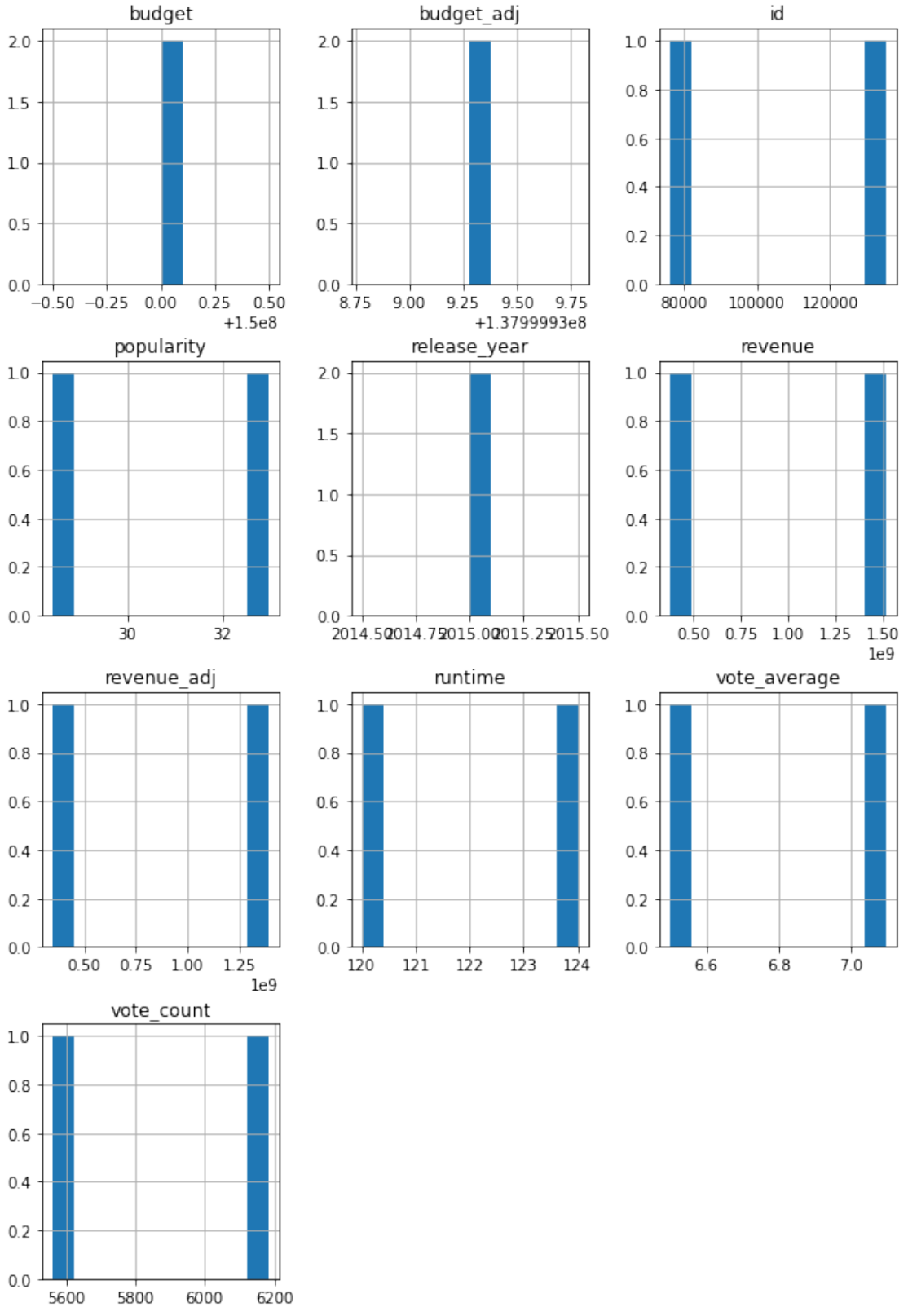

```
In [65]: df_rb['genres'].value_counts().plot(kind = 'bar', figsize = (10, 6));
```



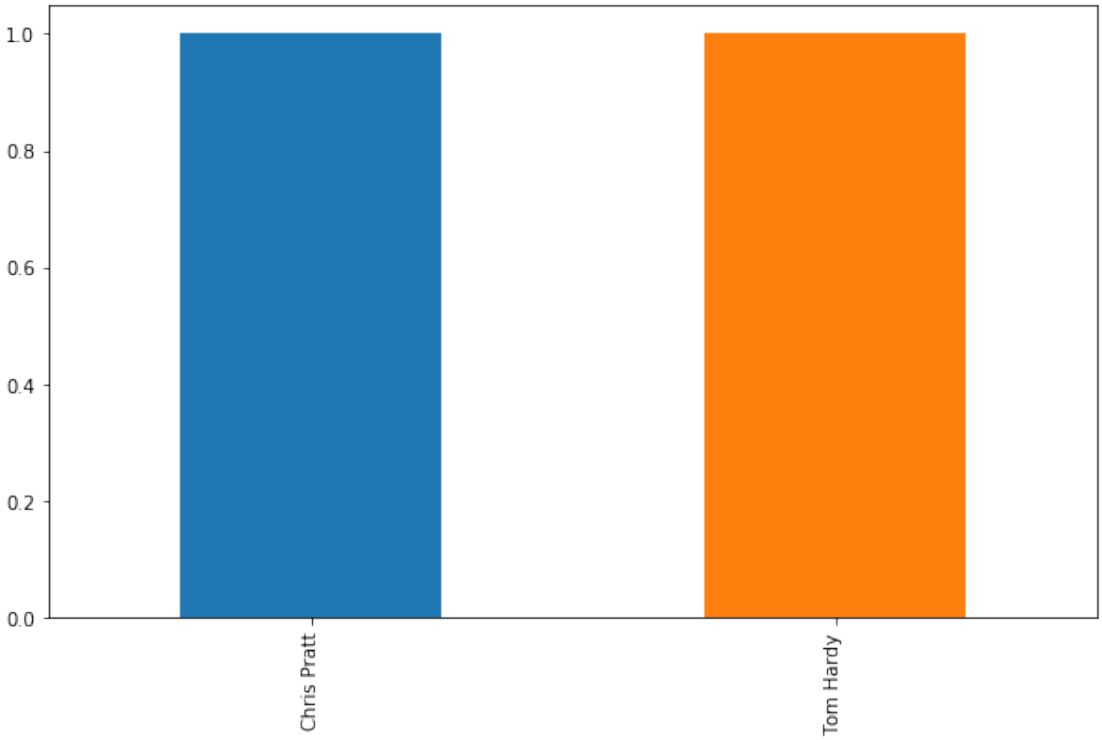

```
In [95]: df.popularity.describe()

Out[95]: count    11353.000000
         mean         0.664078
         std          1.020655
         min          0.000188
         25%          0.212029
         50%          0.388667
         75%          0.731793
         max         32.985763
         Name: popularity, dtype: float64

In [97]: df.popularity.hist();
```



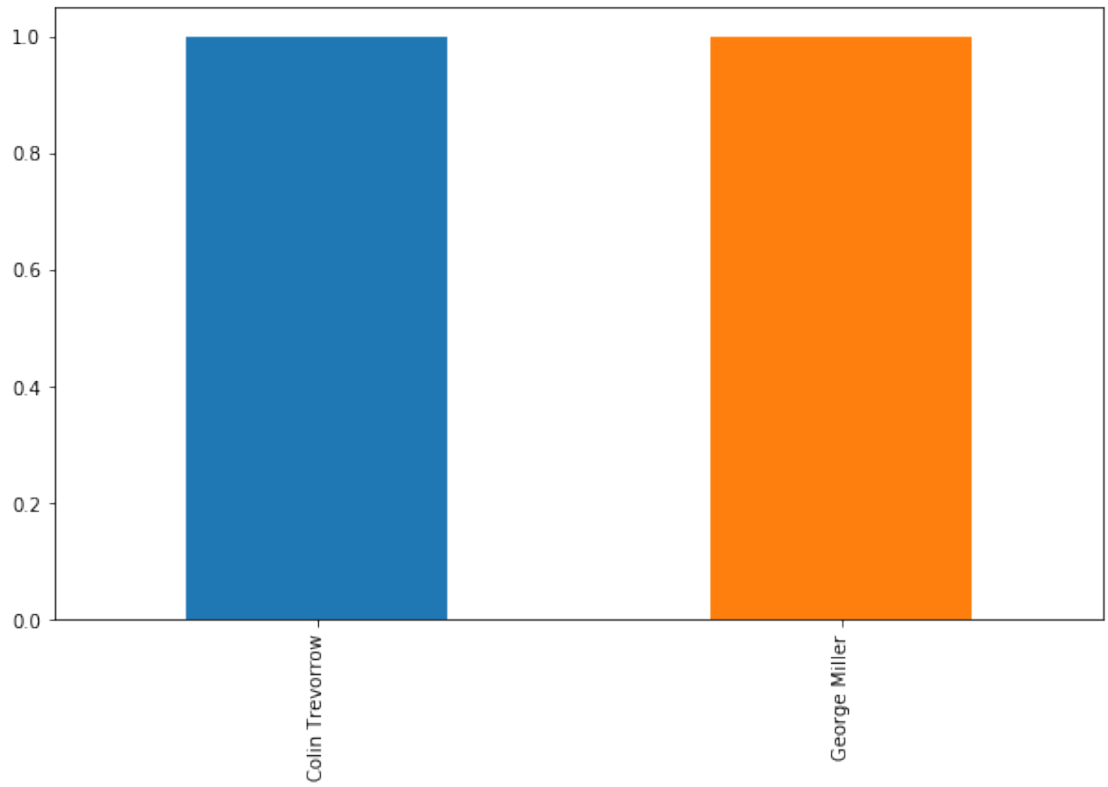

```
In [128]: df_pc = df[df['popularity'] > 25]

In [129]: df_pc.hist(figsize = (10, 15));
```


budget
+1.5e8
budget_adj
+1.3799993e8
id
popularity
release_year
revenue
1e9
revenue_adj
1e9
runtime
vote_average
vote_count

```
In [130]: df_pc['cast'].value_counts().plot(kind = 'bar',figsize = (10, 6));
```




```
In [131]: df_pc['director'].value_counts().plot(kind = 'bar',figsize = (10, 6));
```



Conclusions

Based on our analysis. Action movies tends to be the most voted and most popular movie among the viewers. However, further analysis shows that for the fact that action movies are more popular doesn't mean that it generates more revenue than other genres. Based on revenues, Drama and Comedy movies brings in more revenue than action movies. Furthermore, viewers voted more for movies produced in 2013 than popular movies in 2015. In conclusion, based on our analysis, movies produced by collin trevorrow tends to be popular among viewers with 75% and movies where chris pratt is being casted also gain more popularity among viewers.