project_1

November 11, 2019

1 Investigate the TMDB Movies Dataset

by Kruthika Krishnamurthy

1.1 Table of Contents

[67]:
```
# Import the necessary packages
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

%matplotlib inline
```

[68]:
```
# Create the DataFrame frm the CSV file
tmdb = pd.read_csv("tmdb_movies.csv")
```

[69]:
```
# See the first 5 rows the of the DF
tmdb.head(1)
```

[69]:
```
      id    imdb_id  popularity     budget     revenue  original_title  \
0  135397  tt0369610   32.985763  150000000  1513528810  Jurassic World

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

                        homepage         director            tagline  ...  \
0  http://www.jurassicworld.com/  Colin Trevorrow  The park is open.  ...

                                          overview runtime  \
0  Twenty-two years after the events of Jurassic ...     124

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

```
                                production_companies release_date vote_count  \
0  Universal Studios|Amblin Entertainment|Legenda...     6/9/2015       5562

   vote_average  release_year    budget_adj   revenue_adj
0           6.5          2015  137999939.3  1.392446e+09

[1 rows x 21 columns]
```

Data Wrangling

1.1.1 General Properties

Cross check the datatypes of each columns

[70]:
```
tmdb.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
```

Here we can see that "Homepage", "id", "imdb_id" "tagline" columns are not required for our analysis.

So we will drop these columns in the following steps.

Check for number of null value in each columns

[71]:
```
tmdb.isnull().sum()
```

[71]:
```
id                        0
imdb_id                  10
popularity                0
budget                    0
revenue                   0
original_title            0
cast                     76
homepage               7930
director                 44
tagline                2824
keywords               1493
overview                  4
runtime                   0
genres                   23
production_companies   1030
release_date              0
vote_count                0
vote_average              0
release_year              0
budget_adj                0
revenue_adj               0
dtype: int64
```

Here we can see that a number of columns have null values in them. Since these columns are of no relevance to us we will not alter them.

Check if duplicate entry present

[72]:
```
tmdb.duplicated().sum()
```

[72]:
```
1
```

We can see that one of the row is duplicated. We will remove this duplicate in the following steps.

1.1.2 Data Cleaning

In the General Properties we saw that the following 4 columns were not required. So they are being removed here.

[73]:
```
tmdb.drop(['homepage','id','imdb_id','tagline'],axis=1,inplace = True)
tmdb.head(1)
```

[73]:
```
   popularity     budget     revenue  original_title  \
0   32.985763  150000000  1513528810  Jurassic World

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

                                            keywords  \
0  monster|dna|tyrannosaurus rex|velociraptor|island
```

```
                                                overview  runtime  \
0  Twenty-two years after the events of Jurassic ...      124

                                        genres  \
0  Action|Adventure|Science Fiction|Thriller

                                  production_companies release_date  vote_count  \
0  Universal Studios|Amblin Entertainment|Legenda...     6/9/2015        5562

   vote_average  release_year    budget_adj   revenue_adj
0           6.5          2015  137999939.3  1.392446e+09
```

Drop the duplicate columns

[74]:
```
tmdb.drop_duplicates(inplace = True)
tmdb.duplicated().sum().any()
```

[74]:
```
False
```

Cross check the duplicate values after removing.

[75]:
```
tmdb.duplicated().sum()
```

[75]:
```
0
```

The duplicates are removed.

This marks the end of the Data Wrangling phase. ## Exploratory Data Analysis

In this phase we will answer couple of research questions that are relevant to this Data.

1.1.3 Which year has the highest number of movie releases?

We will group the dataset by the ***release_year*** and use an arbitrary index ***popularity*** to count the number of movie releases per year.

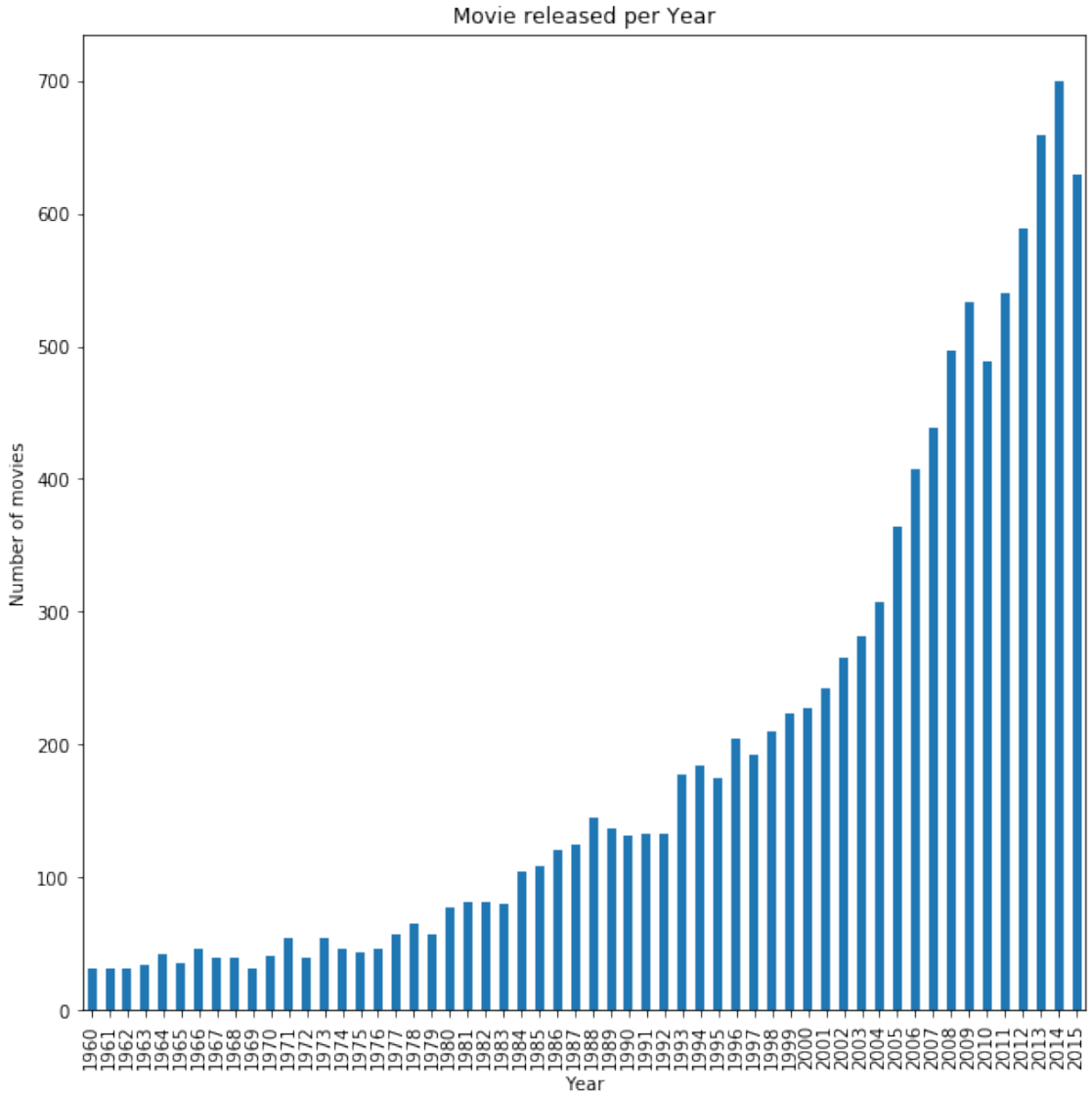
[76]:
```
df_movie_count = tmdb.groupby('release_year').count().popularity
df_movie_count[:6]
```

[76]:
```
release_year
1960    32
1961    31
1962    32
1963    34
1964    42
1965    35
Name: popularity, dtype: int64
```

Here we see the number of movies released every year from 1960 through 2015. We will plot this out below.

[109]:
```
df_movie_count.plot(kind = 'bar',figsize=[10,10])
plt.xlabel('Year')
plt.ylabel('Number of movies')
```

```
plt.title('Movie released per Year')
plt.grid(False)
```



CONCLUSION - 700 movies were produced in 2014 which was the highest.

1.1.4 What is the average runtime of the movies which are sorted based on their ratings?

Let us first split the ratings into 4 categories 1. Below Average *(Rating <= 5)* 2. Average *(Rating between 5 and 7)* 3. Good *(Rating of 7 or 8)* 4. Highly Recommended *(Rating > 8)*

[78]:
```
bin_edges = [1,5,7,8,10]
bin_names = ['below_average', 'average', 'good', 'highly_recommended']
```

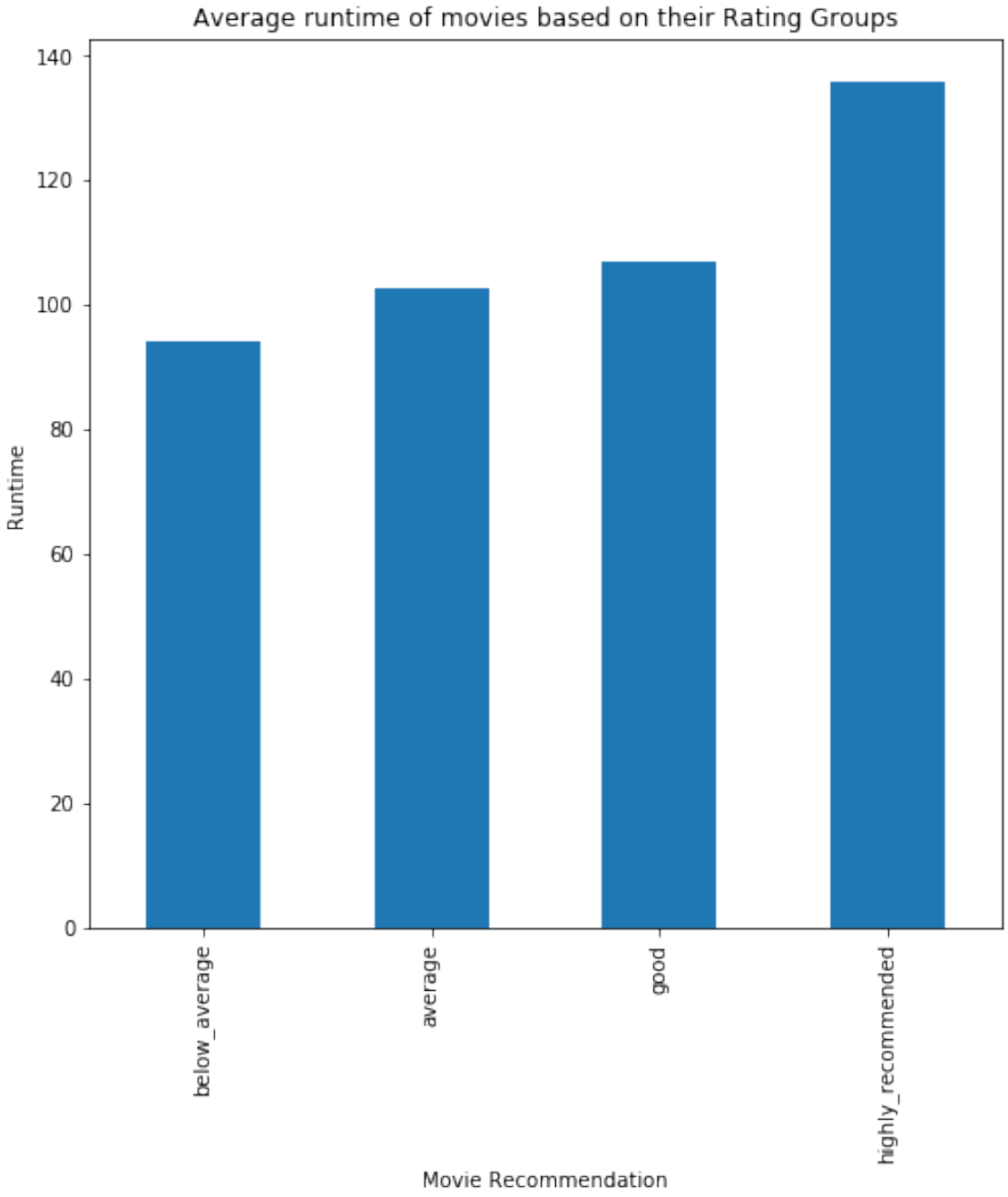
Add movie column name as Highly_recommended

[94]:
```
tmdb['Movie_Recomandation']=pd.cut(tmdb['vote_average'␣
↪],bin_edges,labels=bin_names)
```

Plot a graph for Movie_Recommendation VS Runtime
Movie_Recommendation is based on Ratings by viewers

[128]:
```
tmdb_runtime=tmdb.groupby('Movie_Recomandation').mean().runtime
tmdb_runtime.plot(kind='bar',figsize=(8,8))
plt.xlabel('Movie Recommendation')
plt.ylabel('Runtime')
plt.title('Average runtime of movies based on their Rating Groups')
```

[128]:
```
Text(0.5, 1.0, 'Average runtime of movies based on their Rating Groups')
```



CONCLUSION - Highly rated movie by viewers will always run longer in theaters ie, those movies which are rated above 8 have an average runtime of 139 days.

1.1.5 Which genres are most liked by viewers?

Create a dataframe copy to work with.

[99]:
```
df1 = tmdb.copy()
```

Creating a function to split genres seperated by ' | ' and create new column in a seperate data frame

[101]:
```
def getGenre():
    new = []
    for c in df1.index:
        a = str(df1['genres'][c]).split('|')
        for d in a:
            new.append(d)
    temp = pd.Series(new)
    count = temp.value_counts(ascending=False)
    genres = list(dict.fromkeys(new))

    return [genres, count]
```

The Function ***getGenre*** returns 2 lists ie - 1. A list of all Genres 2. A list of counts for all Genres This can be seen in the following 2 blocks

[114]:
```
# List of all genres (sample 3 elements)
getGenre()[0][:3]
```

[114]:
```
['Action', 'Adventure', 'Science Fiction']
```

[113]:
```
# Count of movies per genre (sample 3 elements)
getGenre()[1][:3]
```

[113]:
```
Drama       4760
Comedy      3793
Thriller    2907
dtype: int64
```

[104]:
```
votes = np.array(df1['vote_average'])
genres = getGenre()[0]
genres_list = list(map(str, df1['genres']))
votes_df = pd.DataFrame(index = genres, columns = ['sum', 'avg'])
j = 0
votes_df.fillna(0, inplace = True)
for i in genres_list:
        a = i.split('|')
        votes_df.loc[a,"sum"] = votes_df.loc[a,"sum"] + votes[j]
        j += 1
```

Created a seperate Dataframe ***votes_df*** having three colums - 1. **Genres** have 20 unique flavors of movies, 2. **Sum** is the addition of votes for each flavor of movies, 3. **Avg** = (the total votes per geners) / (No of movies in each genres)

[116]:
```
votes_df[0:3]
```

[116]:

	sum	avg
Action	13798.0	5.787752
Adventure	8738.6	5.940585

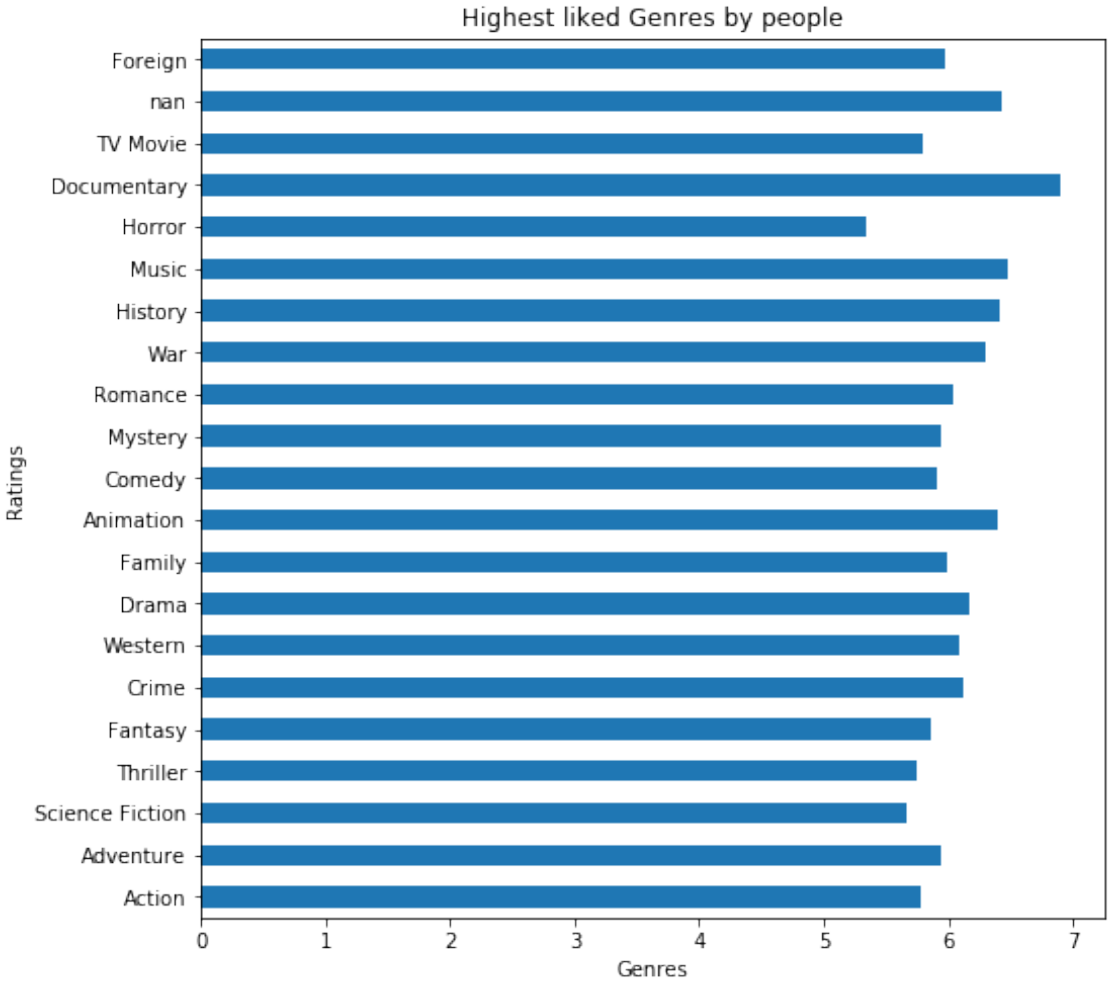
```
Science Fiction   6963.0  5.665582
```

Calculate the *avg* field as follows

[105]:
```
votes_df["avg"] = votes_df["sum"] / getGenre()[1]
```

Plotting graph for highest liked genres on basis of viewer's votes

[126]:
```
votes_df['avg'].plot(kind ='barh',figsize=(8,8))
plt.xlabel('Genres')
plt.ylabel('Ratings')
plt.title('Highest liked Genres by people')
plt.grid(False)
```



CONCLUSION - Documentaries are the most liked genre by viwers.

1.1.6 Which flavor of movies have the most number of releases?

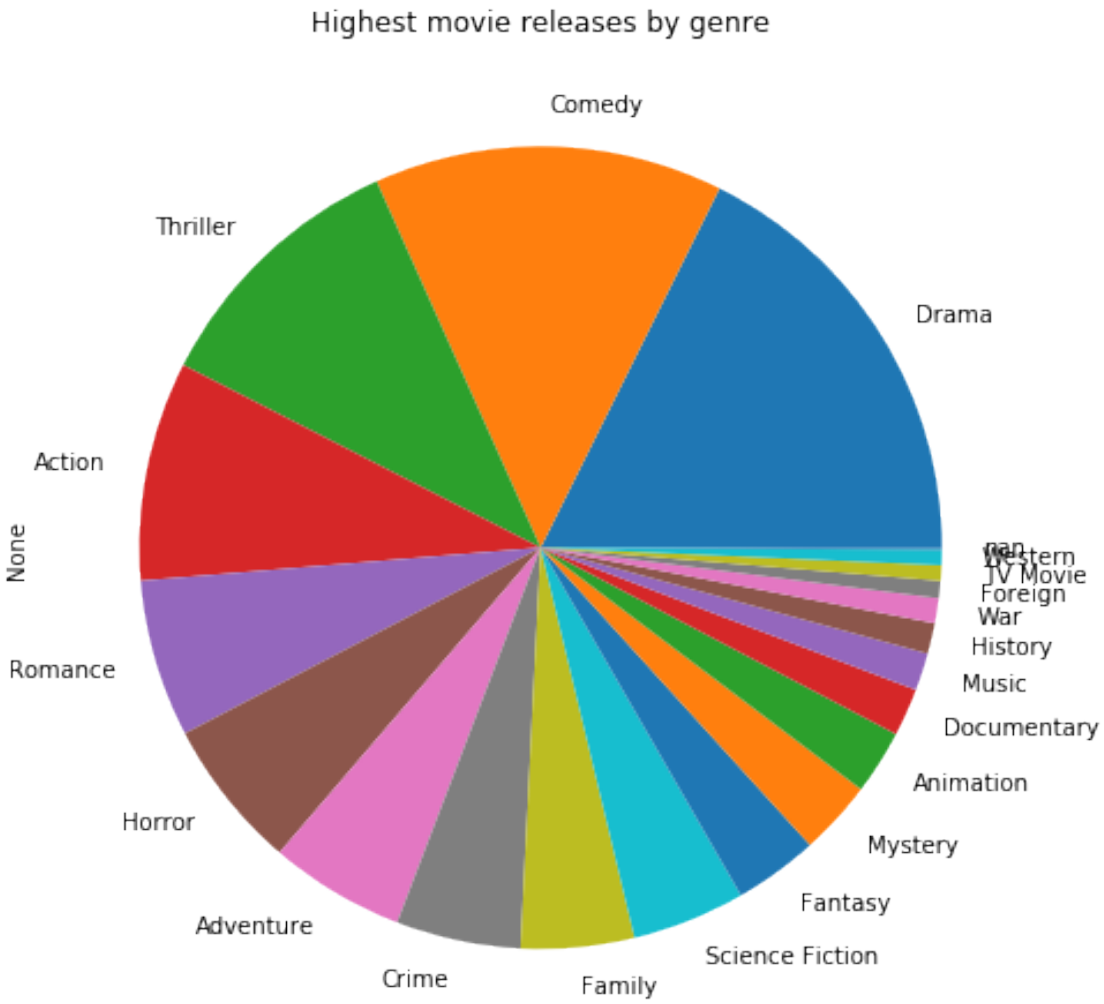
Let us use the ***getGenre*** function again and this time use the counts to plot the number of movie releases by genre.

[129]:
```
getGenre()[1][:3]
```

[129]:
```
Drama       4760
Comedy      3793
Thriller    2907
dtype: int64
```

[122]:
```
getGenre()[1].plot(kind='pie',figsize=(8,8))
plt.title('Highest movie releases by genre')
```

[122]:
```
Text(0.5, 1.0, 'Highest movie releases by genre')
```



CONCLUSION - Drama is the highest released genre having 4760 movie releases. ## Conclusions

700 movies were produced in 2014 which was the highest.

Highly rated movie by viewers will always run longer in theaters ie, those movies which are rated above 8 have an average runtime of 139 days.

Documentaries are the most liked genre by viwers.

Drama is the highest released genre having 4760 movie releases.

[ ]: