Investigate_a_Dataset

December 20, 2017

1 Project: Analysising the TMDB Movies Dataset. Analysing factors that effect the revenue of a movie

1.1 Table of Contents

Introduction
Data Wrangling
Exploratory Data Analysis
Conclusions
Introduction

This data set contains information about 10,000 movies collected from The Movie Database. It has information about a movie's genre, budget, revenue, cast etc. I'm going to focus on variables that can affect the revenue of a movie, like its genre and the month it was released. Is considering the month of release of a movie important for maximum returns? Which genre is more likely to earn more profit?

```
In [118]: import pandas as pd
          import numpy as np
          import matplotlib.pyplot as plt
          import seaborn as sns
          from datetime import datetime

          %matplotlib inline
```

Data Wrangling

1.1.1 General Properties

```
In [119]: df_m=pd.read_csv('tmdb-movies.csv')
          df_m.head(10)
```

```
Out[119]:        id    imdb_id  popularity     budget     revenue  \
          0  135397  tt0369610   32.985763  150000000  1513528810
          1   76341  tt1392190   28.419936  150000000   378436354
          2  262500  tt2908446   13.112507  110000000   295238201
          3  140607  tt2488496   11.173104  200000000  2068178225
          4  168259  tt2820852    9.335014  190000000  1506249360
```

```
5  281957  tt1663202    9.110700  135000000   532950503
6   87101  tt1340138    8.654359  155000000   440603537
7  286217  tt3659388    7.667400  108000000   595380321
8  211672  tt2293640    7.404165   74000000  1156730962
9  150540  tt2096673    6.326804  175000000   853708609

                original_title  \
0                Jurassic World
1            Mad Max: Fury Road
2                      Insurgent
3  Star Wars: The Force Awakens
4                     Furious 7
5                  The Revenant
6            Terminator Genisys
7                   The Martian
8                       Minions
9                    Inside Out

                                                 cast  \
0  Chris Pratt|Bryce Dallas Howard|Irrfan Khan|Vi...
1  Tom Hardy|Charlize Theron|Hugh Keays-Byrne|Nic...
2  Shailene Woodley|Theo James|Kate Winslet|Ansel...
3  Harrison Ford|Mark Hamill|Carrie Fisher|Adam D...
4  Vin Diesel|Paul Walker|Jason Statham|Michelle ...
5  Leonardo DiCaprio|Tom Hardy|Will Poulter|Domhn...
6  Arnold Schwarzenegger|Jason Clarke|Emilia Clar...
7  Matt Damon|Jessica Chastain|Kristen Wiig|Jeff ...
8  Sandra Bullock|Jon Hamm|Michael Keaton|Allison...
9  Amy Poehler|Phyllis Smith|Richard Kind|Bill Ha...

                                       homepage  \
0                    http://www.jurassicworld.com/
1                      http://www.madmaxmovie.com/
2    http://www.thedivergentseries.movie/#insurgent
3  http://www.starwars.com/films/star-wars-episod...
4                          http://www.furious7.com/
5      http://www.foxmovies.com/movies/the-revenant
6                    http://www.terminatormovie.com/
7       http://www.foxmovies.com/movies/the-martian
8                      http://www.minionsmovie.com/
9               http://movies.disney.com/inside-out

                       director  \
0                Colin Trevorrow
1                  George Miller
2               Robert Schwentke
3                    J.J. Abrams
4                      James Wan
```

```
5  Alejandro GonzÃąlez IÃ±Ã¡rritu
6                      Alan Taylor
7                     Ridley Scott
8        Kyle Balda|Pierre Coffin
9                      Pete Docter

                                          tagline      ...       \
0                              The park is open.      ...
1                             What a Lovely Day.      ...
2                       One Choice Can Destroy You      ...
3                    Every generation has a story.      ...
4                               Vengeance Hits Home      ...
5  (n. One who has returned, as if from the dead.)      ...
6                                 Reset the future      ...
7                                   Bring Him Home      ...
8     Before Gru, they had a history of bad bosses      ...
9         Meet the little voices inside your head.      ...

                                            overview runtime  \
0  Twenty-two years after the events of Jurassic ...     124
1  An apocalyptic story set in the furthest reach...     120
2  Beatrice Prior must confront her inner demons ...     119
3  Thirty years after defeating the Galactic Empi...     136
4  Deckard Shaw seeks revenge against Dominic Tor...     137
5  In the 1820s, a frontiersman, Hugh Glass, sets...     156
6  The year is 2029. John Connor, leader of the r...     125
7  During a manned mission to Mars, Astronaut Mar...     141
8  Minions Stuart, Kevin and Bob are recruited by...      91
9  Growing up can be a bumpy road, and it's no ex...      94

                                       genres  \
0  Action|Adventure|Science Fiction|Thriller
1  Action|Adventure|Science Fiction|Thriller
2         Adventure|Science Fiction|Thriller
3   Action|Adventure|Science Fiction|Fantasy
4                     Action|Crime|Thriller
5           Western|Drama|Adventure|Thriller
6  Science Fiction|Action|Thriller|Adventure
7             Drama|Adventure|Science Fiction
8          Family|Animation|Adventure|Comedy
9                    Comedy|Animation|Family

                                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
```

```
5  Regency Enterprises|Appian Way|CatchPlay|Anony...     12/25/15       3929
6            Paramount Pictures|Skydance Productions      6/23/15       2598
7  Twentieth Century Fox Film Corporation|Scott F...      9/30/15       4572
8      Universal Pictures|Illumination Entertainment      6/17/15       2893
9  Walt Disney Pictures|Pixar Animation Studios|W...       6/9/15       3935

   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           7.2          2015  1.241999e+08  4.903142e+08
6           5.8          2015  1.425999e+08  4.053551e+08
7           7.6          2015  9.935996e+07  5.477497e+08
8           6.5          2015  6.807997e+07  1.064192e+09
9           8.0          2015  1.609999e+08  7.854116e+08

[10 rows x 21 columns]
```

The genre column has multiple values and the values need to be seperated. The number of multiple values in the column are different for each row

```
In [120]: df_m.shape #Size of the dataframe

Out[120]: (10866, 21)

In [121]: df_m.dtypes

Out[121]: id                      int64
          imdb_id                object
          popularity            float64
          budget                  int64
          revenue                 int64
          original_title         object
          cast                   object
          homepage               object
          director               object
          tagline                object
          keywords               object
          overview               object
          runtime                 int64
          genres                 object
          production_companies   object
          release_date           object
          vote_count              int64
          vote_average          float64
          release_year            int64
          budget_adj            float64
```

```
revenue_adj             float64
dtype: object
```

The datatypes of the column seem to be alright except of the release_date. I'll change it in the Data Cleaning section.

```
In [122]: df_m.nunique() #calculate unique values in each column
```

```
Out[122]: 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 [123]: df_m.describe()
          #computes statistics of numerical columns of the dataframe
```

```
Out[123]:                   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 [124]: df_m.duplicated().sum() #find duplicate rows
```

```
Out[124]: 1
```

We need to remove this duplicate row because it will not contribute to our analysis.

```
In [125]: np.count_nonzero(df_m.isnull()) #find total null values
```

```
Out[125]: 13434
```

Many null values in the dataset. I'll remove this in the next section.

1.2 2. Data Cleaning

1.2.1 Dropping columns, null values and duplicate rows

The following columns can't be used for any kind of analysis I want to perform.

```
In [126]: df_m.drop(['homepage', 'tagline','overview'], axis=1, inplace=True)

          #drop rows with null values in these columns.
          df_m.dropna(subset=['genres', 'original_title', 'production_companies','runtime'], inp
```

```
In [127]: df_m.drop_duplicates(inplace=True)
          df_m.duplicated().sum()#Number of duplicate rows
```

```
Out[127]: 0
```

```
In [128]: np.count_nonzero(df_m.isnull()) #Total null values
```

```
Out[128]: 1184
```

1.2.2 Separating values in Genre column

```
In [129]: def safe_access(container, i):
              """Return this value if it exists at index i and if it doesn't, return a null valu
              result = container.split('|')
              try:
                  return result[i]
              except IndexError or KeyError:
                  return pd.np.nan
```

```
In [130]: df_mm=df_m[df_m['genres'].str.contains('|')]#returns rows which have multiple values i
          df_mm.head()
```

Out[130]:

	id	imdb_id	popularity	budget	revenue	original_title	cast	director	keywords	runtime	genres	production_companies	release_date	vote_count	vote_average	release_year	budget_adj	revenue_adj
0	135397	tt0369610	32.985763	150000000	1513528810	Jurassic World	Chris Pratt|Bryce Dallas Howard|Irrfan Khan|Vi...	Colin Trevorrow	monster|dna|tyrannosaurus rex|velociraptor|island	124	Action|Adventure|Science Fiction|Thriller	Universal Studios|Amblin Entertainment|Legenda...	6/9/15	5562	6.5	2015	1.379999e+08	1.392446e+09
1	76341	tt1392190	28.419936	150000000	378436354	Mad Max: Fury Road	Tom Hardy|Charlize Theron|Hugh Keays-Byrne|Nic...	George Miller	future|chase|post-apocalyptic|dystopia|australia	120	Action|Adventure|Science Fiction|Thriller	Village Roadshow Pictures|Kennedy Miller Produ...	5/13/15	6185	7.1	2015	1.379999e+08	3.481613e+08
2	262500	tt2908446	13.112507	110000000	295238201	Insurgent	Shailene Woodley|Theo James|Kate Winslet|Ansel...	Robert Schwentke	based on novel|revolution|dystopia|sequel|dyst...	119	Adventure|Science Fiction|Thriller	Summit Entertainment|Mandeville Films|Red Wago...	3/18/15	2480	6.3	2015	1.012000e+08	2.716190e+08
3	140607	tt2488496	11.173104	200000000	2068178225	Star Wars: The Force Awakens	Harrison Ford|Mark Hamill|Carrie Fisher|Adam D...	J.J. Abrams	android|spaceship|jedi|space opera|3d	136	Action|Adventure|Science Fiction|Fantasy	Lucasfilm|Truenorth Productions|Bad Robot	12/15/15	5292	7.5	2015	1.839999e+08	1.902723e+09
4	168259	tt2820852	9.335014	190000000	1506249360	Furious 7	Vin Diesel|Paul Walker|Jason Statham|Michelle ...	James Wan	car race|speed|revenge|suspense|car	137	Action|Crime|Thriller	Universal Pictures|Original Film|Media Rights ...	4/1/15	2947	7.3	2015	1.747999e+08	1.385749e+09

```
In [131]: df_mm.shape==df_m.shape #Do all rows have multiple values in genres column?

Out[131]: True
```

Apparenlty, all columns have multiple values in genre column. I'll take the first five in the values.

```
In [132]: df1=df_mm.copy() #Copies data column
          df2=df_mm.copy()
          df3=df_mm.copy()
          df4=df_mm.copy()
          df5=df_mm.copy()
```

The following lines get each genre from for each movie and adds the row to a data frame. If a value doesn't exist, it stores a null. Later these null values will be deleted and the all dataframes will be appended

```
In [133]: df1['genres']=df1['genres'].apply(lambda x: safe_access(x,0))
          df2['genres']=df2['genres'].apply(lambda x: safe_access(x,1))
          df3['genres']=df3['genres'].apply(lambda x: safe_access(x,2))
          df4['genres']=df4['genres'].apply(lambda x: safe_access(x,3))
          df5['genres']=df5['genres'].apply(lambda x: safe_access(x,4))

In [134]: print("null values in df1\n", np.count_nonzero(df1['genres'].isnull()))
          print("null values in df2\n", np.count_nonzero(df2['genres'].isnull()))
          print("null values in df3\n", np.count_nonzero(df3['genres'].isnull()))
          print("null values in df4\n", np.count_nonzero(df4['genres'].isnull()))
          print("null values in df5\n", np.count_nonzero(df5['genres'].isnull()))

null values in df1
 0
null values in df2
 1982
null values in df3
 5101
null values in df4
 7972
null values in df5
 9318
```

Need to drop all rows which have null values in the genres column in the new dataframe (df1, df2..) to avoid redundancy.

```
In [135]: df1.dropna(subset=['genres'],inplace=True) #drop null values from the genres column
          df2.dropna(subset=['genres'],inplace=True)
          df3.dropna(subset=['genres'],inplace=True)
          df4.dropna(subset=['genres'],inplace=True)
          df5.dropna(subset=['genres'],inplace=True)
```

```
In [136]: print("null values in df1\n", np.count_nonzero(df1['genres'].isnull()))
          print("null values in df2\n", np.count_nonzero(df2['genres'].isnull()))
          print("null values in df3\n", np.count_nonzero(df3['genres'].isnull()))
          print("null values in df4\n", np.count_nonzero(df4['genres'].isnull()))
          print("null values in df5\n", np.count_nonzero(df5['genres'].isnull()))
```

```
null values in df1
 0
null values in df2
 0
null values in df3
 0
null values in df4
 0
null values in df5
 0
```

```
In [137]: df1=df1.append(df2, ignore_index=True)
          df3=df3.append(df4,ignore_index=True)
          df3=df3.append(df5,ignore_index=True)
          new_df=df1.append(df3,ignore_index=True)
          new_df.shape
```

```
Out[137]: (24757, 18)
```

1.2.3 Changing datatype of release_date to timestamp from String

```
In [138]: type(new_df['release_date'][0])==str #Checks if release_date column is of type string
```

```
Out[138]: True
```

```
In [139]: new_df['release_date']=pd.to_datetime(new_df['release_date'])
          type(new_df['release_date'][0]) #data type of release_date column
```

```
Out[139]: pandas._libs.tslib.Timestamp
```

1.2.4 Adding new column release_month for further analysis

```
In [140]: new_df['release_month']=new_df['release_date'].apply(lambda x: x.to_datetime().month)
          new_df.head()
```

```
/opt/conda/lib/python3.6/site-packages/pandas/core/series.py:2355: FutureWarning: to_datetime is
  mapped = lib.map_infer(values, f, convert=convert_dtype)
```

```
Out[140]:        id    imdb_id  popularity     budget     revenue  \
          0  135397  tt0369610   32.985763  150000000  1513528810
          1   76341  tt1392190   28.419936  150000000   378436354
          2  262500  tt2908446   13.112507  110000000   295238201
```

```
3  140607  tt2488496   11.173104  200000000  2068178225
4  168259  tt2820852    9.335014  190000000  1506249360

                original_title  \
0                Jurassic World
1            Mad Max: Fury Road
2                     Insurgent
3  Star Wars: The Force Awakens
4                     Furious 7

                                                cast           director  \
0  Chris Pratt|Bryce Dallas Howard|Irrfan Khan|Vi...    Colin Trevorrow
1  Tom Hardy|Charlize Theron|Hugh Keays-Byrne|Nic...      George Miller
2  Shailene Woodley|Theo James|Kate Winslet|Ansel...   Robert Schwentke
3  Harrison Ford|Mark Hamill|Carrie Fisher|Adam D...        J.J. Abrams
4  Vin Diesel|Paul Walker|Jason Statham|Michelle ...          James Wan

                                          keywords  runtime     genres  \
0  monster|dna|tyrannosaurus rex|velociraptor|island      124     Action
1   future|chase|post-apocalyptic|dystopia|australia      120     Action
2  based on novel|revolution|dystopia|sequel|dyst...      119  Adventure
3              android|spaceship|jedi|space opera|3d      136     Action
4                car race|speed|revenge|suspense|car      137     Action

                               production_companies release_date  vote_count  \
0  Universal Studios|Amblin Entertainment|Legenda...   2015-06-09        5562
1  Village Roadshow Pictures|Kennedy Miller Produ...   2015-05-13        6185
2  Summit Entertainment|Mandeville Films|Red Wago...   2015-03-18        2480
3          Lucasfilm|Truenorth Productions|Bad Robot   2015-12-15        5292
4  Universal Pictures|Original Film|Media Rights ...   2015-04-01        2947

   vote_average  release_year    budget_adj   revenue_adj  release_month
0           6.5          2015  1.379999e+08  1.392446e+09              6
1           7.1          2015  1.379999e+08  3.481613e+08              5
2           6.3          2015  1.012000e+08  2.716190e+08              3
3           7.5          2015  1.839999e+08  1.902723e+09             12
4           7.3          2015  1.747999e+08  1.385749e+09              4
```

In [141]:
```
#changes month number to month name in release_month
new_df['release_month']= new_df['release_month'].map({1:'January', 2:'February', 3:'Ma
new_df.head()
```

Out[141]:
```
       id    imdb_id  popularity     budget     revenue  \
0  135397  tt0369610   32.985763  150000000  1513528810
1   76341  tt1392190   28.419936  150000000   378436354
2  262500  tt2908446   13.112507  110000000   295238201
3  140607  tt2488496   11.173104  200000000  2068178225
4  168259  tt2820852    9.335014  190000000  1506249360
```

```
                original_title  \
0                Jurassic World
1            Mad Max: Fury Road
2                     Insurgent
3  Star Wars: The Force Awakens
4                     Furious 7

                                                cast           director  \
0  Chris Pratt|Bryce Dallas Howard|Irrfan Khan|Vi...    Colin Trevorrow
1  Tom Hardy|Charlize Theron|Hugh Keays-Byrne|Nic...      George Miller
2  Shailene Woodley|Theo James|Kate Winslet|Ansel...   Robert Schwentke
3  Harrison Ford|Mark Hamill|Carrie Fisher|Adam D...        J.J. Abrams
4  Vin Diesel|Paul Walker|Jason Statham|Michelle ...          James Wan

                                        keywords  runtime     genres  \
0  monster|dna|tyrannosaurus rex|velociraptor|island      124     Action
1   future|chase|post-apocalyptic|dystopia|australia      120     Action
2  based on novel|revolution|dystopia|sequel|dyst...      119  Adventure
3              android|spaceship|jedi|space opera|3d      136     Action
4                car race|speed|revenge|suspense|car      137     Action

                               production_companies release_date  vote_count  \
0  Universal Studios|Amblin Entertainment|Legenda...   2015-06-09        5562
1  Village Roadshow Pictures|Kennedy Miller Produ...   2015-05-13        6185
2  Summit Entertainment|Mandeville Films|Red Wago...   2015-03-18        2480
3         Lucasfilm|Truenorth Productions|Bad Robot    2015-12-15        5292
4  Universal Pictures|Original Film|Media Rights ...   2015-04-01        2947

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

1.2.5 Shorten few genre values because they are too long

```
In [142]: new_df['genres']=new_df['genres'].replace({'Science Fiction':"Sci Fiction", "Documenta
          new_df['genres'].unique()
```

```
Out[142]: array(['Action', "Adv're", 'Western', 'Sci Fiction', 'Drama', 'Family',
                 'Comedy', 'Crime', 'Romance', 'War', 'Mystery', 'Thriller',
                 'Fantasy', 'History', 'Animation', 'Horror', 'Music', "Doc'tary",
                 'TV Movie', 'Foreign'], dtype=object)
```

Adventure –> Adv're Science Fiction –> Sci Fiction Documentart –> Doc'tary

3. Exploratory Data Analysis

1.2.6 Which Genre has the highest revenue?

I'm gonna groupby genre of the movies and find the revenue_adj mean and compare them. Even if a single movie has multiple genres, this method will give the right result because for each each movie, each corresponding genre gets equal weight of the revenue. And the number of movies with a particular genre shouldn't affect much because I'm taking the average.

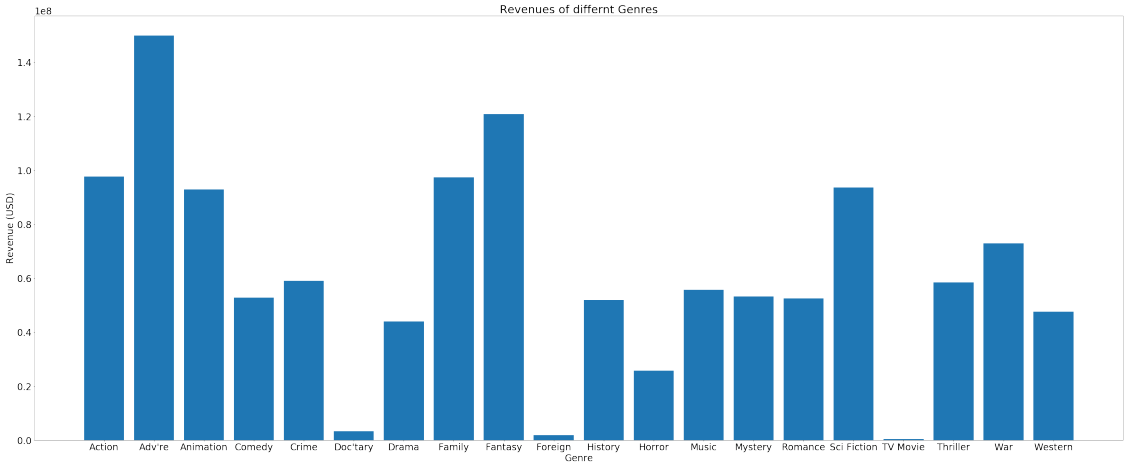
```
In [144]: mean_revenues=new_df.groupby('genres')['revenue_adj'].mean()
          mean_revenues
```

```
Out[144]: genres
          Action          9.763580e+07
          Adv're          1.498139e+08
          Animation       9.287552e+07
          Comedy          5.271744e+07
          Crime           5.902647e+07
          Doc'tary        3.292831e+06
          Drama           4.394511e+07
          Family          9.738196e+07
          Fantasy         1.207935e+08
          Foreign         1.918406e+06
          History         5.198219e+07
          Horror          2.579533e+07
          Music           5.562651e+07
          Mystery         5.322556e+07
          Romance         5.243856e+07
          Sci Fiction     9.363840e+07
          TV Movie        4.325119e+05
          Thriller        5.843933e+07
          War             7.295850e+07
          Western         4.754193e+07
          Name: revenue_adj, dtype: float64
```

```
In [148]: locations=[i for i in range(mean_revenues.count())]
          print(locations)
          labels= mean_revenues.keys()
          height=mean_revenues.values
          plt.bar(locations, height, tick_label=labels) #Plot the bar chart
          #Set title of the bar chart
          plt.title("Revenues of differnt Genres")

          #x-axis label
          plt.xlabel("Genre")

          #y-axis label
          plt.ylabel("Revenue (USD)")
          fig_size = plt.rcParams["figure.figsize"]
          fig_size[0]=50
```

```
        fig_size[1]=20
        print(fig_size)
        plt.rcParams["figure.figsize"]= fig_size #Set the size of the plot
        plt.rcParams.update({'font.size': 24}) #Sets the fontsize
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]
[50, 20]
```

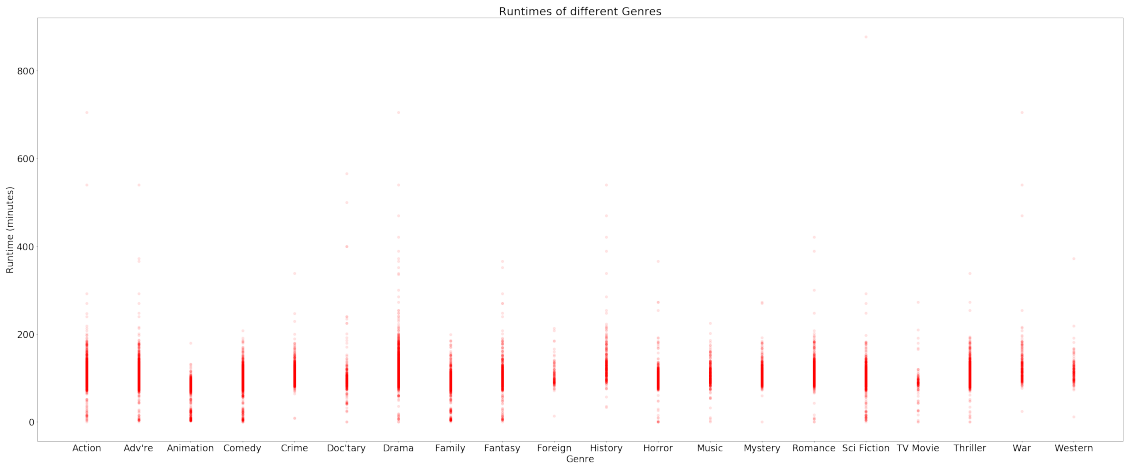


Adventure movies have the highest revenue and TV Movie have the least revenue.

1.2.7 Does runtime of a movie depend on its genre? Is there any significant difference between the runtimes of each genre?

I'm going to plot the runtime of each movie corresponding to its genre. If the distribution is along same/close horizontal axis, there is no significant difference in the runtime of each genre.

```
In [146]: plt.scatter(new_df['genres'],new_df['runtime'], color='r', alpha=0.1) #plot a scatter
          plt.title('Runtimes of different Genres')
          plt.ylabel("Runtime (minutes)")
          plt.xlabel("Genre")
          plt.rcParams['figure.figsize'] = [50,20]
          plt.rcParams.update({'font.size': 30})
```



There is no significant difference in these visualizations. We can see that most of the distribution is from 150 - 200 minutes (Darker the point/regioin, more movie runtimes are concentrated there).

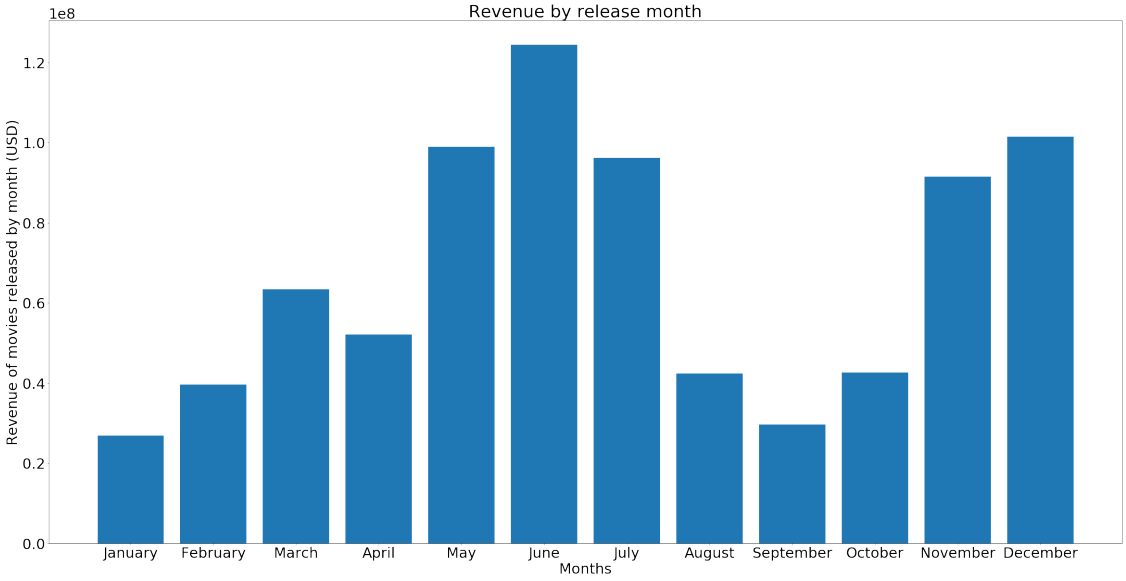
1.2.8 Which month is the best for the movie release?

This is a very intersting question because the production companies can target this month for maximum returns. Taking the mean revenue values for each month and plotting a bar graph should give a good idea to answer the question.

```
In [149]: months_dict = {"January":1, "February":2, "March":3, "April":4, "May":5, "June":6, "Ju

          def month_value(date):
              return months_dict[date]

In [150]: month_revenues=new_df.groupby('release_month')['revenue_adj'].mean()
          month_revenues

Out[150]: release_month
          April        5.210252e+07
          August       4.236415e+07
          December     1.014656e+08
          February     3.965993e+07
          January      2.691735e+07
          July         9.613436e+07
          June         1.243722e+08
          March        6.334589e+07
          May          9.901106e+07
          November     9.147676e+07
          October      4.265785e+07
          September    2.968905e+07
          Name: revenue_adj, dtype: float64
```

Need to sort the month list in ascending order of the months in the calendar and not their first letter

```
In [151]: keys=month_revenues.keys()
          sorted_month=sorted(keys, key=month_value)
          #This sorts the month in the right order using the dictionary month_dict defined above
          sorted_month

Out[151]: ['January',
           'February',
           'March',
           'April',
           'May',
           'June',
           'July',
           'August',
           'September',
           'October',
           'November',
           'December']

In [153]: locations=[i for i in range(month_revenues.count())]
          height=[month_revenues[i] for i in sorted_month] #to have the values in order.
          #Can't use the series keys directly because it sorts by first character
          labels=sorted_month
          plt.bar(locations, height, tick_label=labels)
          plt.rcParams['figure.figsize'] = [40,20]
          plt.xlabel('Months')
          plt.ylabel('Revenue of movies released by month (USD)')
          plt.title('Revenue by release month')
          plt.rcParams.update({'font.size': 30})
```



The movies released in June seem have the highest return compared to other months.

4. Conclusions

1: Adventure movies seem to have higher revenue than movies of other genre.

2: Runtime of a movie doesn't really depend on the genre.

3: Movies released in the month of June have higher revenue than movies released in any other month.

Limitation of the analysis

The cast of the movie was not considered which can also have a huge impact on the revenue of a movie and may be the movies with higher revenue (adventure or released in June) casted many favorite celebrities.

The movies in this dataset have similar runtime and their might be factors that affect the runtime of a movie.

References

Stack Overflow

Python Documentation

Idea of safe_access from Sohier's code

```
In [154]: from subprocess import call
          call(['python', '-m', 'nbconvert', 'Investigate_a_Dataset.ipynb'])
```

```
Out[154]: 0
```