final_project

July 28, 2023

Final Project: Mean Gross Revenue of each genre in Disney's catalogue.

Introduction

Questions of Interest

In this analysis, I want to compare the mean total gross return for each genre of movie in the disney movie dataframe. I'm interested in this question because it can give an insight into what style of movie has been the most successful for the disney franchise.

It will be interesting seeing if one genre rules them all by having the greatest mean by a considerable margin, I'd like to think that having a mean of over 2X the second highest will fulfill that criteria.

Once I've determined the highest grossing genre of film I'd like to find out which decade has the highest inflated gross from that genre.

My prediction is that the Adventure genre will "rule them all" and in the 1990's

Dataset description

For this project, we will be using 5 tables regarding Disney's box office success. The data was obtain from Data World Which follows a Creative Common Attribution 4.0 International LicenseLinks to an external site Here.

The tables are:

disney-characters.csv

disney-director.csv

disney-voice-actors.csv

disney_revenue_1991-2016.csv

disney_movies_total_gross.csv

The table I will be using is

disney_movie_total_gross.csv

Methods & Results

Since I am only computing the Mean gross and inflated gross revenue of each genre, I'll be using the total gross table from the dataframes. However, before moving further, let us import the tables and do some basic visualizations.

```
[1]: # Lets import all the required libraries needed for this analysis
     import altair as alt
     import pandas as pd
     import numpy as np

     # import all the required files
     total_gross = pd.read_csv('data/disney_movies_total_gross.csv')
     # let's see what the table looks like
     total_gross
```

```
[1]:                          movie_title  release_date      genre MPAA_rating  \
     0      Snow White and the Seven Dwarfs  Dec 21, 1937    Musical           G
     1                           Pinocchio   Feb 9, 1940  Adventure           G
     2                            Fantasia  Nov 13, 1940    Musical           G
     3                   Song of the South  Nov 12, 1946  Adventure           G
     4                          Cinderella  Feb 15, 1950      Drama           G
     ..                                 ...           ...        ...         ...
     574          The Light Between Oceans   Sep 2, 2016      Drama       PG-13
     575                    Queen of Katwe  Sep 23, 2016      Drama          PG
     576                    Doctor Strange   Nov 4, 2016  Adventure       PG-13
     577                             Moana  Nov 23, 2016  Adventure          PG
     578      Rogue One: A Star Wars Story  Dec 16, 2016  Adventure       PG-13

          total_gross inflation_adjusted_gross
     0    $184,925,485             $5,228,953,251
     1     $84,300,000             $2,188,229,052
     2     $83,320,000             $2,187,090,808
     3     $65,000,000             $1,078,510,579
     4     $85,000,000               $920,608,730
     ..            ...                        ...
     574   $12,545,979               $12,545,979
     575    $8,874,389                $8,874,389
     576  $232,532,923              $232,532,923
     577  $246,082,029              $246,082,029
     578  $529,483,936              $529,483,936

     [579 rows x 6 columns]
```

We can see that the movies are in chronological order but let's see what other info we can find.

```
[2]: total_gross.info()
     print(total_gross['movie_title'].dtype)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 579 entries, 0 to 578
Data columns (total 6 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
```

```
0   movie_title              579 non-null    object
1   release_date             579 non-null    object
2   genre                    562 non-null    object
3   MPAA_rating              523 non-null    object
4   total_gross              579 non-null    object
5   inflation_adjusted_gross 579 non-null    object
dtypes: object(6)
memory usage: 27.3+ KB
object
```

We're missing some values in the genre column which will affect our data, so let's drop those from our data as we don't know the genre. We want our genre and total_gross columns to each have the same number of values.

```
[3]: # we will drop all the NaN values from the genre column only
     total_gross = total_gross.dropna(subset=['genre'])
     total_gross
```

```
[3]:                      movie_title  release_date      genre MPAA_rating  \
     0    Snow White and the Seven Dwarfs  Dec 21, 1937    Musical           G
     1                          Pinocchio   Feb 9, 1940  Adventure           G
     2                           Fantasia  Nov 13, 1940    Musical           G
     3                  Song of the South  Nov 12, 1946  Adventure           G
     4                         Cinderella  Feb 15, 1950      Drama           G
     ..                               ...           ...        ...         ...
     574        The Light Between Oceans   Sep 2, 2016      Drama       PG-13
     575                  Queen of Katwe  Sep 23, 2016      Drama          PG
     576                  Doctor Strange   Nov 4, 2016  Adventure       PG-13
     577                           Moana  Nov 23, 2016  Adventure          PG
     578     Rogue One: A Star Wars Story  Dec 16, 2016  Adventure       PG-13

          total_gross inflation_adjusted_gross
     0    $184,925,485           $5,228,953,251
     1     $84,300,000           $2,188,229,052
     2     $83,320,000           $2,187,090,808
     3     $65,000,000           $1,078,510,579
     4     $85,000,000             $920,608,730
     ..            ...                      ...
     574   $12,545,979             $12,545,979
     575    $8,874,389              $8,874,389
     576  $232,532,923            $232,532,923
     577  $246,082,029            $246,082,029
     578  $529,483,936            $529,483,936

     [562 rows x 6 columns]
```

```
[4]: # check for NaN values in the new dataset
     total_gross.info()
```

```
print(total_gross['movie_title'].dtype)
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 562 entries, 0 to 578
Data columns (total 6 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   movie_title              562 non-null    object
 1   release_date             562 non-null    object
 2   genre                    562 non-null    object
 3   MPAA_rating              513 non-null    object
 4   total_gross              562 non-null    object
 5   inflation_adjusted_gross 562 non-null    object
dtypes: object(6)
memory usage: 30.7+ KB
object
```

We now have 562 values in the genre, total, and inflation columns!

Let's take the genre column in isolation from the rest and examine it. We will be breaking the genre group down to each type of genre.

[5]:
```
total_gross
```

[5]:
```
                           movie_title  release_date        genre MPAA_rating  \
0      Snow White and the Seven Dwarfs  Dec 21, 1937      Musical           G
1                            Pinocchio   Feb 9, 1940    Adventure           G
2                             Fantasia  Nov 13, 1940      Musical           G
3                     Song of the South  Nov 12, 1946    Adventure           G
4                           Cinderella  Feb 15, 1950        Drama           G
..                                 ...           ...          ...         ...
574            The Light Between Oceans   Sep 2, 2016        Drama       PG-13
575                      Queen of Katwe  Sep 23, 2016        Drama          PG
576                      Doctor Strange   Nov 4, 2016    Adventure       PG-13
577                               Moana  Nov 23, 2016    Adventure          PG
578      Rogue One: A Star Wars Story  Dec 16, 2016    Adventure       PG-13

        total_gross inflation_adjusted_gross
0      $184,925,485           $5,228,953,251
1       $84,300,000           $2,188,229,052
2       $83,320,000           $2,187,090,808
3       $65,000,000           $1,078,510,579
4       $85,000,000             $920,608,730
..              ...                      ...
574     $12,545,979              $12,545,979
575      $8,874,389               $8,874,389
576    $232,532,923             $232,532,923
577    $246,082,029             $246,082,029
```

```
578   $529,483,936                $529,483,936
```

```
[562 rows x 6 columns]
```

Let's check the data types to make sure we can work with the values correctly.

`[6]:` `total_gross.dtypes`

```
[6]: movie_title                object
     release_date               object
     genre                      object
     MPAA_rating                object
     total_gross                object
     inflation_adjusted_gross   object
     dtype: object
```

We see that the inflation_adjusted_gross columns are objects but we want them to be float. Let's change that!

```
[7]: # changing the object to int

     total_gross['inflation_adjusted_gross'] =␣
      ↪total_gross['inflation_adjusted_gross'].str.replace('[\$\,]', '').
      ↪astype('float')

     total_gross
```

```
<ipython-input-7-fe804591aa4f>:3: FutureWarning: The default value of regex will
change from True to False in a future version.
  total_gross['inflation_adjusted_gross'] =
total_gross['inflation_adjusted_gross'].str.replace('[\$\,]',
'').astype('float')
<ipython-input-7-fe804591aa4f>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  total_gross['inflation_adjusted_gross'] =
total_gross['inflation_adjusted_gross'].str.replace('[\$\,]',
'').astype('float')
```

```
[7]:                           movie_title  release_date      genre MPAA_rating  \
     0     Snow White and the Seven Dwarfs  Dec 21, 1937    Musical           G
     1                           Pinocchio   Feb 9, 1940  Adventure           G
     2                            Fantasia  Nov 13, 1940    Musical           G
     3                   Song of the South  Nov 12, 1946  Adventure           G
     4                          Cinderella  Feb 15, 1950      Drama           G
```

```
 ..                                  …          …           …          …
574           The Light Between Oceans   Sep 2, 2016      Drama       PG-13
575                    Queen of Katwe   Sep 23, 2016      Drama          PG
576                   Doctor Strange    Nov 4, 2016   Adventure       PG-13
577                            Moana   Nov 23, 2016   Adventure          PG
578     Rogue One: A Star Wars Story   Dec 16, 2016   Adventure       PG-13

        total_gross  inflation_adjusted_gross
0       $184,925,485                5.228953e+09
1        $84,300,000                2.188229e+09
2        $83,320,000                2.187091e+09
3        $65,000,000                1.078511e+09
4        $85,000,000                9.206087e+08
 ..              …                           …
574      $12,545,979                1.254598e+07
575       $8,874,389                8.874389e+06
576     $232,532,923                2.325329e+08
577     $246,082,029                2.460820e+08
578     $529,483,936                5.294839e+08

[562 rows x 6 columns]
```

Now that we have the column as the right data type lets

```
[8]: #grouping the genre column into each genre name
     total_gross_genre = total_gross.groupby(by='genre')['inflation_adjusted_gross'].
      ↪mean().sort_values(ascending=False)
     total_gross_genre = total_gross_genre.reset_index()
     total_gross_genre
```

```
[8]:                    genre  inflation_adjusted_gross
     0               Musical                6.035979e+08
     1             Adventure                1.903974e+08
     2                Action                1.374734e+08
     3     Thriller/Suspense                8.965379e+07
     4                Comedy                8.466773e+07
     5       Romantic Comedy                7.777708e+07
     6               Western                7.381571e+07
     7                 Drama                7.189302e+07
     8   Concert/Performance                5.741084e+07
     9           Black Comedy                5.224349e+07
     10               Horror                2.341385e+07
     11          Documentary                1.271803e+07
```

As a first visulization Let's take this information and plot it on a chart but before we do let's change the total_gross column name so its different than the main dataframe.

```
[9]: total_gross_genre = total_gross_genre.
      ↪rename(columns={'inflation_adjusted_gross': 'total_gross_revenue'})
     total_gross_genre
```

```
[9]:                   genre  total_gross_revenue
     0              Musical         6.035979e+08
     1            Adventure         1.903974e+08
     2               Action         1.374734e+08
     3      Thriller/Suspense        8.965379e+07
     4               Comedy         8.466773e+07
     5      Romantic Comedy         7.777708e+07
     6              Western         7.381571e+07
     7                Drama         7.189302e+07
     8    Concert/Performance        5.741084e+07
     9          Black Comedy         5.224349e+07
     10              Horror         2.341385e+07
     11         Documentary         1.271803e+07
```

```
[10]: chart = alt.Chart(total_gross_genre, width=500, height=300).mark_bar().encode(
                   x=alt.X('genre:N', title='Disney Movie Genre',),
                   y=alt.Y('total_gross_revenue:Q', title= 'Total Gross␣
      ↪Revenue')
                   ).properties(title="Total Gross Revenue of each Disney movie␣
      ↪genre")
      chart
```

```
[10]: alt.Chart(…)
```

The data suggests that the Musical is highest grossing genre of Disney movies but what if that
number is skewed by the inflated rate because of its age? For our second visulization lets try
something fun! We will compare the mean total gross value using the inflation_adjusted_gross
column and compare each decade! Then determine if the numbers may be skewed.

Let's re-import the original dataframe into a new copy that will import the release_date into its
datatime

```
[11]: inflated_gross_revenue = pd.read_csv('data/disney_movies_total_gross.csv',␣
      ↪parse_dates = ['release_date'])
      inflated_gross_revenue
```

```
[11]:                        movie_title release_date       genre MPAA_rating  \
      0    Snow White and the Seven Dwarfs   1937-12-21     Musical           G
      1                         Pinocchio   1940-02-09   Adventure           G
      2                          Fantasia   1940-11-13     Musical           G
      3                 Song of the South   1946-11-12   Adventure           G
      4                        Cinderella   1950-02-15       Drama           G
      ..                              ...          ...         ...         ...
      574        The Light Between Oceans   2016-09-02       Drama       PG-13
```

```
575                 Queen of Katwe    2016-09-23      Drama         PG
576                 Doctor Strange    2016-11-04  Adventure      PG-13
577                          Moana    2016-11-23  Adventure         PG
578   Rogue One: A Star Wars Story    2016-12-16  Adventure      PG-13


        total_gross inflation_adjusted_gross
0      $184,925,485           $5,228,953,251
1       $84,300,000           $2,188,229,052
2       $83,320,000           $2,187,090,808
3       $65,000,000           $1,078,510,579
4       $85,000,000             $920,608,730
..              …                        …
574     $12,545,979              $12,545,979
575      $8,874,389               $8,874,389
576    $232,532,923             $232,532,923
577    $246,082,029             $246,082,029
578    $529,483,936             $529,483,936

[579 rows x 6 columns]
```

```
[12]: inflated_gross_revenue.dtypes
```

```
[12]: movie_title                          object
      release_date                 datetime64[ns]
      genre                                object
      MPAA_rating                          object
      total_gross                          object
      inflation_adjusted_gross             object
      dtype: object
```

Let's change all necessary data types

```
[13]: # change the column datatype
      inflated_gross_revenue['inflation_adjusted_gross'] =␣
       ↪inflated_gross_revenue['inflation_adjusted_gross'].str.replace('[\$\,]', '').
       ↪astype('float')
      inflated_gross_revenue
```

```
<ipython-input-13-32968fe62485>:2: FutureWarning: The default value of regex
will change from True to False in a future version.
  inflated_gross_revenue['inflation_adjusted_gross'] =
inflated_gross_revenue['inflation_adjusted_gross'].str.replace('[\$\,]',
'').astype('float')
```

```
[13]:                       movie_title release_date      genre MPAA_rating  \
      0    Snow White and the Seven Dwarfs    1937-12-21    Musical           G
      1                         Pinocchio    1940-02-09  Adventure           G
```

```
2                          Fantasia  1940-11-13   Musical         G
3                 Song of the South  1946-11-12  Adventure        G
4                        Cinderella  1950-02-15     Drama         G
..                              ...         ...       ...       ...
574          The Light Between Oceans  2016-09-02     Drama     PG-13
575                   Queen of Katwe  2016-09-23     Drama        PG
576                  Doctor Strange  2016-11-04  Adventure     PG-13
577                           Moana  2016-11-23  Adventure        PG
578     Rogue One: A Star Wars Story  2016-12-16  Adventure     PG-13

        total_gross  inflation_adjusted_gross
0     $184,925,485              5.228953e+09
1      $84,300,000              2.188229e+09
2      $83,320,000              2.187091e+09
3      $65,000,000              1.078511e+09
4      $85,000,000              9.206087e+08
..             ...                       ...
574    $12,545,979              1.254598e+07
575     $8,874,389              8.874389e+06
576   $232,532,923              2.325329e+08
577   $246,082,029              2.460820e+08
578   $529,483,936              5.294839e+08

[579 rows x 6 columns]
```

We want to make a new column that will group the movies into the year they were made.

```
[14]:  #add a column
       inflated_gross_revenue = inflated_gross_revenue.assign(year =␣
       ↪inflated_gross_revenue['release_date'].dt.year)
       inflated_gross_revenue
```

```
[14]:                         movie_title release_date      genre MPAA_rating  \
       0   Snow White and the Seven Dwarfs  1937-12-21    Musical         G
       1                        Pinocchio  1940-02-09  Adventure         G
       2                         Fantasia  1940-11-13    Musical         G
       3                Song of the South  1946-11-12  Adventure         G
       4                       Cinderella  1950-02-15      Drama         G
       ..                             ...         ...        ...       ...
       574         The Light Between Oceans  2016-09-02      Drama     PG-13
       575                  Queen of Katwe  2016-09-23      Drama        PG
       576                 Doctor Strange  2016-11-04  Adventure     PG-13
       577                          Moana  2016-11-23  Adventure        PG
       578    Rogue One: A Star Wars Story  2016-12-16  Adventure     PG-13

             total_gross  inflation_adjusted_gross  year
       0    $184,925,485              5.228953e+09  1937
```

```
1     $84,300,000                2.188229e+09  1940
2     $83,320,000                2.187091e+09  1940
3     $65,000,000                1.078511e+09  1946
4     $85,000,000                9.206087e+08  1950
..            …                         …     …
574   $12,545,979                1.254598e+07  2016
575    $8,874,389                8.874389e+06  2016
576  $232,532,923                2.325329e+08  2016
577  $246,082,029                2.460820e+08  2016
578  $529,483,936                5.294839e+08  2016

[579 rows x 7 columns]
```

```
[15]: from floor_div_into_decade import decades

      inflated_gross_revenue = decades(inflated_gross_revenue, 'decade', 'year')
      inflated_gross_revenue
```

```
[15]:      index                      movie_title release_date      genre  \
      0        0  Snow White and the Seven Dwarfs   1937-12-21    Musical
      1        1                        Pinocchio   1940-02-09  Adventure
      2        2                         Fantasia   1940-11-13    Musical
      3        3                Song of the South   1946-11-12  Adventure
      4        4                       Cinderella   1950-02-15      Drama
      ..      …                              …            …          …
      574    574        The Light Between Oceans   2016-09-02      Drama
      575    575                  Queen of Katwe   2016-09-23      Drama
      576    576                  Doctor Strange   2016-11-04  Adventure
      577    577                           Moana   2016-11-23  Adventure
      578    578   Rogue One: A Star Wars Story   2016-12-16  Adventure

           MPAA_rating    total_gross  inflation_adjusted_gross  year  decades
      0              G  $184,925,485              5.228953e+09  1937     1930
      1              G   $84,300,000              2.188229e+09  1940     1940
      2              G   $83,320,000              2.187091e+09  1940     1940
      3              G   $65,000,000              1.078511e+09  1946     1940
      4              G   $85,000,000              9.206087e+08  1950     1950
      ..            …             …                        …     …        …
      574        PG-13   $12,545,979              1.254598e+07  2016     2010
      575           PG    $8,874,389              8.874389e+06  2016     2010
      576        PG-13  $232,532,923              2.325329e+08  2016     2010
      577           PG  $246,082,029              2.460820e+08  2016     2010
      578        PG-13  $529,483,936              5.294839e+08  2016     2010

[579 rows x 9 columns]
```

Let's run !black on our code in the py file.

```
[53]: !black floor_div_into_decade.py
```

All done!
1 file left unchanged.

```
[54]: !flake8 floor_div_into_decade.py
```

```
[55]: # separate the year col into decades
      #inflated_gross_revenue = inflated_gross_revenue.assign(decade =␣
      ↪(inflated_gross_revenue['year']//10) * 10)
      #inflated_gross_revenue
```

let's make a function that will group the year column and compare the mean of the inflation so that we can add other dataframes.

The dataframe has been imported using parse_dates so that the dates are in a datetime type.

The value of the gross revenue is a float

A new column named year has been created to hold the year extracted from the datatime value of the release year

The year column is converted to a string

The year column has been reduced down to the decade from the years

Let's import the function into our notebook.

Now that we have our years grouped lets group the year.

Let's plot this as well!

```
[17]: #import & run custom function from py file
      from data_grouping import data_grouping

      inflated_gross_revenue_year = data_grouping(inflated_gross_revenue, 'decades',␣
      ↪'inflation_adjusted_gross')
      inflated_gross_revenue_year
```

```
[17]:    decades          mean
      0     1930  5.228953e+09
      1     1940  1.817943e+09
      2     1950  6.766075e+08
      3     1960  4.270692e+08
      4     1970  1.181057e+08
      5     1980  7.600902e+07
      6     1990  7.518349e+07
      7     2000  9.181107e+07
      8     2010  1.529127e+08
```

```
[18]: # bar chart to show mean
      chart2 = alt.Chart(inflated_gross_revenue_year, width=500, height=300).
       ↪mark_bar().encode(
                      x=alt.X('decades:N', title='Disney Movie Decade(s)'),
                      y=alt.Y('mean:Q', title= 'Total Gross Revenue with Adjusted␣
       ↪Inflation')
                      ).properties(title="Total Gross Revenue of Disney movies for␣
       ↪each Decade")
      chart2
```

[18]: alt.Chart(…)

From this data we can see that the movies from the 1930's to the 1940's are 8X grossing the latest
decades. My guess is that a few over-inflated musicals are in this decade. If this is correct, let's
remove the movies from 1930 - 1980 compare the results. First let's find out what fraction of the
highest grossing films are the 1930 - 1980 decade.

```
[19]: inflated_gross_revenue
```

```
[19]:      index                   movie_title release_date    genre  \
      0        0  Snow White and the Seven Dwarfs   1937-12-21    Musical
      1        1                      Pinocchio   1940-02-09  Adventure
      2        2                       Fantasia   1940-11-13    Musical
      3        3              Song of the South   1946-11-12  Adventure
      4        4                     Cinderella   1950-02-15      Drama
      ..     …                           …           …        …
      574    574        The Light Between Oceans   2016-09-02      Drama
      575    575               Queen of Katwe   2016-09-23      Drama
      576    576               Doctor Strange   2016-11-04  Adventure
      577    577                        Moana   2016-11-23  Adventure
      578    578    Rogue One: A Star Wars Story   2016-12-16  Adventure

          MPAA_rating   total_gross  inflation_adjusted_gross  year  decades
      0             G  $184,925,485              5.228953e+09  1937     1930
      1             G   $84,300,000              2.188229e+09  1940     1940
      2             G   $83,320,000              2.187091e+09  1940     1940
      3             G   $65,000,000              1.078511e+09  1946     1940
      4             G   $85,000,000              9.206087e+08  1950     1950
      ..          …             …                       …      …        …
      574       PG-13   $12,545,979              1.254598e+07  2016     2010
      575          PG    $8,874,389              8.874389e+06  2016     2010
      576       PG-13  $232,532,923              2.325329e+08  2016     2010
      577          PG  $246,082,029              2.460820e+08  2016     2010
      578       PG-13  $529,483,936              5.294839e+08  2016     2010

      [579 rows x 9 columns]
```

```
[20]: # test for movies grossing over 1 billion dollars
      inflated_gross_revenue[inflated_gross_revenue['inflation_adjusted_gross'] >␣
      ↪1000000000]
```

```
[20]:    index              movie_title release_date      genre MPAA_rating  \
      0      0  Snow White and the Seven Dwarfs   1937-12-21    Musical           G
      1      1                      Pinocchio   1940-02-09  Adventure           G
      2      2                       Fantasia   1940-11-13    Musical           G
      3      3              Song of the South   1946-11-12  Adventure           G
      6      6              Lady and the Tramp   1955-06-22      Drama           G
      8      8                 101 Dalmatians   1961-01-25     Comedy           G

          total_gross  inflation_adjusted_gross  year  decades
      0  $184,925,485              5.228953e+09  1937     1930
      1   $84,300,000              2.188229e+09  1940     1940
      2   $83,320,000              2.187091e+09  1940     1940
      3   $65,000,000              1.078511e+09  1946     1940
      6   $93,600,000              1.236036e+09  1955     1950
      8  $153,000,000              1.362871e+09  1961     1960
```

This information clearly shows that the adjusted grossing revenue favors movies from the oldest decades. So its safe to say these numbers skew our results. Let's remove the 1930 - 1980 decades to get a clearer picture of the highest grossing genre.

```
[21]: #filter out the 1930 - 1980 subset

      inflated_gross_revenue_filtered =␣
      ↪inflated_gross_revenue[(inflated_gross_revenue['decades'] >= 1980)]
      inflated_gross_revenue_filtered
```

```
[21]:     index                movie_title release_date      genre MPAA_rating  \
      24     24              Midnight Madness   1980-02-08        NaN         NaN
      25     25   The Last Flight of Noah's Ark   1980-06-25        NaN         NaN
      26     26           The Devil and Max Devlin   1981-01-01        NaN         NaN
      27     27                           Amy   1981-03-20      Drama         NaN
      28     28              The Fox and the Hound   1981-07-10     Comedy         NaN
      ..    ...                           ...          ...        ...         ...
      574   574           The Light Between Oceans   2016-09-02      Drama       PG-13
      575   575                 Queen of Katwe   2016-09-23      Drama          PG
      576   576                 Doctor Strange   2016-11-04  Adventure       PG-13
      577   577                         Moana   2016-11-23  Adventure          PG
      578   578   Rogue One: A Star Wars Story   2016-12-16  Adventure       PG-13

          total_gross  inflation_adjusted_gross  year  decades
      24    $2,900,000                9088096.0  1980     1980
      25   $11,000,000               34472116.0  1980     1980
      26   $16,000,000               48517980.0  1981     1980
```

```
27          $0                              0.0  1981       1980
28   $43,899,231                    133118889.0  1981       1980
..            …                              …     …          …
574  $12,545,979                    12545979.0  2016       2010
575   $8,874,389                     8874389.0  2016       2010
576  $232,532,923                  232532923.0  2016       2010
577  $246,082,029                  246082029.0  2016       2010
578  $529,483,936                  529483936.0  2016       2010

[555 rows x 9 columns]
```

let's remove the NaN genres

```python
[22]:  # remove NaN values from the genre col
       inflated_gross_revenue_filtered = inflated_gross_revenue_filtered.
         ↪dropna(subset=['genre'])
       inflated_gross_revenue_filtered
```

```
[22]:      index                movie_title release_date       genre MPAA_rating  \
      27      27                        Amy   1981-03-20       Drama         NaN
      28      28        The Fox and the Hound   1981-07-10      Comedy         NaN
      29      29                  Condorman   1981-08-07      Action         NaN
      30      30              Night Crossing   1982-02-05       Drama         NaN
      31      31                       Tron   1982-07-09      Action         NaN
      ..      …                          …            …           …          …
      574    574   The Light Between Oceans   2016-09-02       Drama       PG-13
      575    575              Queen of Katwe   2016-09-23       Drama          PG
      576    576               Doctor Strange   2016-11-04   Adventure       PG-13
      577    577                       Moana   2016-11-23   Adventure          PG
      578    578   Rogue One: A Star Wars Story   2016-12-16   Adventure       PG-13

           total_gross  inflation_adjusted_gross  year  decades
      27            $0                       0.0  1981     1980
      28   $43,899,231               133118889.0  1981     1980
      29            $0                       0.0  1981     1980
      30    $4,500,000                12903059.0  1982     1980
      31   $26,918,576                77184895.0  1982     1980
      ..            …                         …     …        …
      574  $12,545,979                12545979.0  2016     2010
      575   $8,874,389                 8874389.0  2016     2010
      576  $232,532,923              232532923.0  2016     2010
      577  $246,082,029              246082029.0  2016     2010
      578  $529,483,936              529483936.0  2016     2010

      [541 rows x 9 columns]
```

14

```
[23]: from data_grouping import data_grouping

      inflated_gross_revenue_new_year =␣
      ↪data_grouping(inflated_gross_revenue_filtered, 'decades',␣
      ↪'inflation_adjusted_gross')
      inflated_gross_revenue_new_year
```

```
[23]:    decades          mean
      0     1980  7.835296e+07
      1     1990  7.794864e+07
      2     2000  9.288921e+07
      3     2010  1.529127e+08
```

One more look at the chart without the 1930 - 1980 values.

```
[24]: chart3 = alt.Chart(inflated_gross_revenue_new_year, width=500, height=300).
      ↪mark_bar().encode(
                       x=alt.X('decades:N', title='Disney Movie Decade(s)'),
                       y=alt.Y('mean:Q', title= 'Total Gross Revenue with Adjusted␣
      ↪Inflation')
                       ).properties(title="Total Gross Revenue of Disney movies for␣
      ↪each Decade")
      chart3
```

```
[24]: alt.Chart(…)
```

That looks a bit more adjusted. We can now find the highest grossing genre from this chart.

```
[25]: high_gross = inflated_gross_revenue_filtered.
      ↪groupby(by='genre')['inflation_adjusted_gross'].mean().
      ↪sort_values(ascending=False)
      high_gross = high_gross.reset_index()
      high_gross
```

```
[25]:                    genre  inflation_adjusted_gross
      0              Adventure              1.649406e+08
      1                 Action              1.374734e+08
      2                Musical              9.806013e+07
      3       Thriller/Suspense              8.965379e+07
      4         Romantic Comedy              7.777708e+07
      5                 Comedy              7.518772e+07
      6                Western              7.381571e+07
      7      Concert/Performance              5.741084e+07
      8                  Drama              5.421310e+07
      9            Black Comedy              5.224349e+07
      10                Horror              2.341385e+07
      11           Documentary              1.271803e+07
```

```
[26]: # plot the new data
      chart4 = alt.Chart(high_gross, width=500, height=300).mark_bar().encode(
                      x=alt.X('genre:N', title='Disney Movie Genre',),
                      y=alt.Y('inflation_adjusted_gross:Q', title= 'Total Gross␣
      ↪Revenue')
                      ).properties(title="Total Gross Revenue of each Disney movie␣
      ↪genre")
      chart4
```

```
[26]: alt.Chart(…)
```

We see that Adventure is the highest grossing genre. Not by much, and not enough the "rule them all!" We will use Adventure only for the rest of the excercise.

```
[27]: # drop unecessary cols
      inflated_gross_revenue_index = inflated_gross_revenue_filtered.
      ↪drop(columns=(['movie_title', 'release_date', 'MPAA_rating', 'total_gross',␣
      ↪'year']))
      inflated_gross_revenue_index
```

```
[27]:       index       genre   inflation_adjusted_gross   decades
      27        27        Drama                        0.0      1980
      28        28       Comedy                133118889.0      1980
      29        29       Action                        0.0      1980
      30        30        Drama                 12903059.0      1980
      31        31       Action                 77184895.0      1980
      ..        …          …                          …         …
      574      574        Drama                 12545979.0      2010
      575      575        Drama                  8874389.0      2010
      576      576    Adventure                232532923.0      2010
      577      577    Adventure                246082029.0      2010
      578      578    Adventure                529483936.0      2010

      [541 rows x 4 columns]
```

```
[28]: #multi-indexing genre and year cols
      inflated_gross_revenue_index.set_index(['genre', 'decades'], inplace=True)
      inflated_gross_revenue_index
```

```
[28]:                       index   inflation_adjusted_gross
      genre    decades
      Drama    1980          27                        0.0
      Comedy   1980          28                133118889.0
      Action   1980          29                        0.0
      Drama    1980          30                 12903059.0
      Action   1980          31                 77184895.0
      …                       …                          …
```

```
Drama      2010    574            12545979.0
           2010    575             8874389.0
Adventure  2010    576           232532923.0
           2010    577           246082029.0
           2010    578           529483936.0
```

[541 rows x 2 columns]

[29]: `# organizing for better style`
`inflated_gross_revenue_index.sort_index(inplace=True)`
`inflated_gross_revenue_index`

[29]:
```
                    index   inflation_adjusted_gross
genre     decades
Action    1980        29                        0.0
          1980        31                 77184895.0
          1980        62                 60097074.0
          1980        68                 12009960.0
          1990        90                 29485923.0
...                  ...                        ...
Western   2000       328                 89042541.0
          2000       387                 81547672.0
          2000       398                 91340921.0
          2000       401                 30416359.0
          2010       534                 92597388.0
```

[541 rows x 2 columns]

[30]: `# isolate Adventure for anaylsis`
`inflated_gross_revenue_index.loc['Adventure'].reset_index()`

[30]:
```
     decades   index   inflation_adjusted_gross
0      1980      39                 33900697.0
1      1980      40                 25215934.0
2      1980      41                 50553142.0
3      1980      48                 53637367.0
4      1980      49                 42183197.0
..      ...     ...                        ...
119    2010     572                 55483770.0
120    2010     573                 76233151.0
121    2010     576                232532923.0
122    2010     577                246082029.0
123    2010     578                529483936.0
```

[124 rows x 3 columns]

```
[31]: inflated_gross_revenue_index.groupby(by='decades')['inflation_adjusted_gross'].
      ↪mean().reset_index()
```

```
[31]:    decades  inflation_adjusted_gross
      0     1980              7.835296e+07
      1     1990              7.794864e+07
      2     2000              9.288921e+07
      3     2010              1.529127e+08
```

The data suggests that Disney made the most gross revenue from its Adventure genre in the 2010's.

Discussion

In this exercise I tried to find which genre of Disney movie was the highest grossing then in which decade was it also the highest grossing. The data suggested that Musicals were the clear winner but it seemed fishy as it was over 8X more and the rest of the field was much closer to 0.5X so I decided to investigate if the data needed a little more wrangling. I then found that the earliest movies were over-inflated to a crazy high degree so I decided to take out the first decade. This left me with Adventure being the highest grossing genre and in the 1980's.

My guess wasn't very far off but it was nice to see that it was so close!

Findings like this can help suggest a few things:

Maybe the 1980's was an era of society wanting to escape from reality and enjoy an adventure

Maybe a particular movie was an all-time high grossing film that carried the rest

It tells me that the highest grossing movies were probably in the Adventure genre

It tells me that maybe there was a higher grossing movie in another genre but the mean of that genre wasn't as high

Some other questions I'd like answered are

What were the top 3 grossing movies in each decade?

Were those top 3 movies in the adventure genre for the 1980's

What was the top decade for each genre

Resources

The charts and code were all done by moi!

Resources Used

<a href=""https://www.youtube.com/watch?v=tcRGa2soc-c>YouTube

HTML formatting

anything python

```
[ ]:
```

```
[ ]:
```