Final Project Submission

- Student name: Nahum Odemba
- Student pace: Full Time
- Scheduled project review date/time: 27-August-2022
- Institution: Moringa School
- Instructor name: Antony Muiko

FILM INDUSTRY ANALYSIS

Data Preparation

In [113]:

```
#First I will import all the necessary libraries
import pandas as pd
import sqlite3
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

%matplotlib inline
```

The cell below reads the `.csv` and `.db` files using the imported libraries

In [114]:

```
#This step is to load datasets from the computer to our jupyter notebook
budget_df = pd.read_csv("tn.movie_budgets.csv", index_col = 0)
imdb = sqlite3.connect('im.db')
```

Checking the Contenst of our Data

The cell below checks the first five rows of the `budget_df` file

In [115]:

```
budget_df.head()
```

Out[115]:

	release_date	movie	production_budget	domestic_gross	worldwide_gross
id					
1	Dec 18, 2009	Avatar	$425,000,000	$760,507,625	$2,776,345,279
2	May 20, 2011	Pirates of the Caribbean: On Stranger Tides	$410,600,000	$241,063,875	$1,045,663,875
3	Jun 7, 2019	Dark Phoenix	$350,000,000	$42,762,350	$149,762,350
4	May 1, 2015	Avengers: Age of Ultron	$330,600,000	$459,005,868	$1,403,013,963
5	Dec 15, 2017	Star Wars Ep. VIII: The Last Jedi	$317,000,000	$620,181,382	$1,316,721,747

In [116]:

```
budget_df.shape
```

Out[116]:

```
(5782, 5)
```

Select Table Names

In the cell below we check the names of the tables in our `IMDB` database using pd.read_sql

In [117]:

```
pd.read_sql("""SELECT name
                      AS 'Table Names'
                      FROM sqlite_master
                      WHERE type='table';""", imdb)
```

Out[117]:

	Table Names
0	movie_basics
1	directors
2	known_for
3	movie_akas
4	movie_ratings
5	persons
6	principals
7	writers

In the cell below we check the contents of `movie_basics` table pd.read_sql

In [118]:

```
pd.read_sql("""SELECT * FROM  movie_basics""", imdb)
```

Out[118]:

	movie_id	primary_title	original_title	start_year	runtime_minutes	genres
0	tt0063540	Sunghursh	Sunghursh	2013	175.0	Action,Crime,Drama
1	tt0066787	One Day Before the Rainy Season	Ashad Ka Ek Din	2019	114.0	Biography,Drama
2	tt0069049	The Other Side of the Wind	The Other Side of the Wind	2018	122.0	Drama
3	tt0069204	Sabse Bada Sukh	Sabse Bada Sukh	2018	NaN	Comedy,Drama
4	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy
...	...	...	...	...	...	...
146139	tt9916538	Kuambil Lagi Hatiku	Kuambil Lagi Hatiku	2019	123.0	Drama
146140	tt9916622	Rodolpho Teóphilo - O Legado de um Pioneiro	Rodolpho Teóphilo - O Legado de um Pioneiro	2015	NaN	Documentary
146141	tt9916706	Dankyavar Danka	Dankyavar Danka	2013	NaN	Comedy
146142	tt9916730	6 Gunn	6 Gunn	2017	116.0	None
146143	tt9916754	Chico Albuquerque - Revelações	Chico Albuquerque - Revelações	2013	NaN	Documentary

146144 rows × 6 columns

In the cell below we check the contents of `movie_ratings` table pd.read_sql

In [119]:

```
first_query = pd.read_sql("""SELECT * FROM movie_ratings
                            JOIN movie_basics
                            USING(movie_id)
                            WHERE numvotes > 200
                            AND averagerating > 9""", imdb)
first_query.head()
```

Out[119]:

	movie_id	averagerating	numvotes	primary_title	original_title	start_year	runtime_minutes
0	tt9680166	9.6	624	Yeh Suhaagraat Impossible	Yeh Suhaagraat Impossible	2019	92.0
1	tt7738784	9.4	9629	Peranbu	Peranbu	2018	147.0
2	tt8203706	9.4	500	American Deep State	American Deep State	2019	62.0
3	tt6487784	9.2	227	Generation Freedom	Generation Freedom	2019	98.0
4	tt5813916	9.3	100568	The Mountain II	Dag II	2016	135.0

1. Answering the first Business Question

=>Which movie genres have the highest rating and votes?

The `imdb` and dataset will be used to answer this question

But I will have to join the `movie_basics` and `movie_ratings` tables to solve this problem

Joining the tables

In the cell below we join `movie_basics` table with `movie_ratings` using a shared column

In [120]:

```
basics_ratings = pd.read_sql("""SELECT * FROM movie_basics
                            JOIN movie_ratings
                            USING(movie_id)""", imdb)

basics_ratings
```

Out[120]:

	movie_id	primary_title	original_title	start_year	runtime_minutes	genres
0	tt0063540	Sunghursh	Sunghursh	2013	175.0	Action,Crime,Drama
1	tt0066787	One Day Before the Rainy Season	Ashad Ka Ek Din	2019	114.0	Biography,Drama
2	tt0069049	The Other Side of the Wind	The Other Side of the Wind	2018	122.0	Drama
3	tt0069204	Sabse Bada Sukh	Sabse Bada Sukh	2018	NaN	Comedy,Drama
4	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy
...	...	...	...	...	...	...
73851	tt9913084	Diabolik sono io	Diabolik sono io	2019	75.0	Documentary
73852	tt9914286	Sokagin Çocuklari	Sokagin Çocuklari	2019	98.0	Drama,Family
73853	tt9914642	Albatross	Albatross	2017	NaN	Documentary
73854	tt9914942	La vida sense la Sara Amat	La vida sense la Sara Amat	2019	NaN	None
73855	tt9916160	Drømmeland	Drømmeland	2019	72.0	Documentary

73856 rows × 8 columns

Data Cleaning

In [121]:

```
#I will start with checking if there is any duplicated movies
basics_ratings['primary_title'].duplicated().sum()
```

Out[121]:

3863

In [122]:

```
#drop duplicated values
basics_ratings_no_duplicates = basics_ratings.drop_duplicates(subset=['primary_title
basics_ratings_no_duplicates.head()
```

Out[122]:

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

In [123]:

```
#confirming no duplicates left
basics_ratings_no_duplicates['primary_title'].duplicated().sum()
```

Out[123]:

0

In [124]:

```
#The second step is to clean the budget_df by changing the data types
#Let us start by checking the budget datatypes
budget_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 5782 entries, 1 to 82
Data columns (total 5 columns):
 #   Column             Non-Null Count  Dtype 
---  ------             --------------  ----- 
 0   release_date       5782 non-null   object
 1   movie              5782 non-null   object
 2   production_budget  5782 non-null   object
 3   domestic_gross     5782 non-null   object
 4   worldwide_gross    5782 non-null   object
dtypes: object(5)
memory usage: 271.0+ KB
```

In [125]:

```
#The production_budget, worldwide_gross, and domestic_gross are object types
#We have to change them to int type
budget_df['production_budget'] = budget_df['production_budget'].str.replace(',', '')
budget_df['domestic_gross'] = budget_df['domestic_gross'].str.replace(',', '').str.r
budget_df['worldwide_gross'] = budget_df['worldwide_gross'].str.replace(',', '').str
budget_df.head()
```

...

In [126]:

```
#Filtering basics_ratings_no_duplicates further to have only
#movies with numvotes more than 100 and rating above 6

filtered1 = basics_ratings_no_duplicates[basics_ratings_no_duplicates['averagerating
filtered2 = filtered1[filtered1['numvotes'] > 100]
filtered2
```

Out[126]:

	movie_id	primary_title	original_title	start_year	runtime_minutes	
2	tt0069049	The Other Side of the Wind	The Other Side of the Wind	2018	122.0	
4	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Dra
6	tt0137204	Joe Finds Grace	Joe Finds Grace	2017	83.0	Adventure,Animat
7	tt0146592	Pál Adrienn	Pál Adrienn	2010	136.0	
10	tt0162942	Children of the Green Dragon	A zöld sárkány gyermekei	2010	89.0	
...	...	...	...	...	...	
73840	tt9904844	Ott Tänak: The Movie	Ott Tänak: The Movie	2019	125.0	D
73841	tt9905412	Ottam	Ottam	2019	120.0	
73842	tt9905462	Pengalila	Pengalila	2019	111.0	
73849	tt9911774	Padmavyuhathile Abhimanyu	Padmavyuhathile Abhimanyu	2019	130.0	
73852	tt9914286	Sokagin Çocuklari	Sokagin Çocuklari	2019	98.0	D

13977 rows × 8 columns

In [127]:

```
#inner join will ensure we keep columns in all the joined dataframes
joined_df= filtered2.join(budget_df, how ="inner")
joined_df.head()
```

Out[127]:

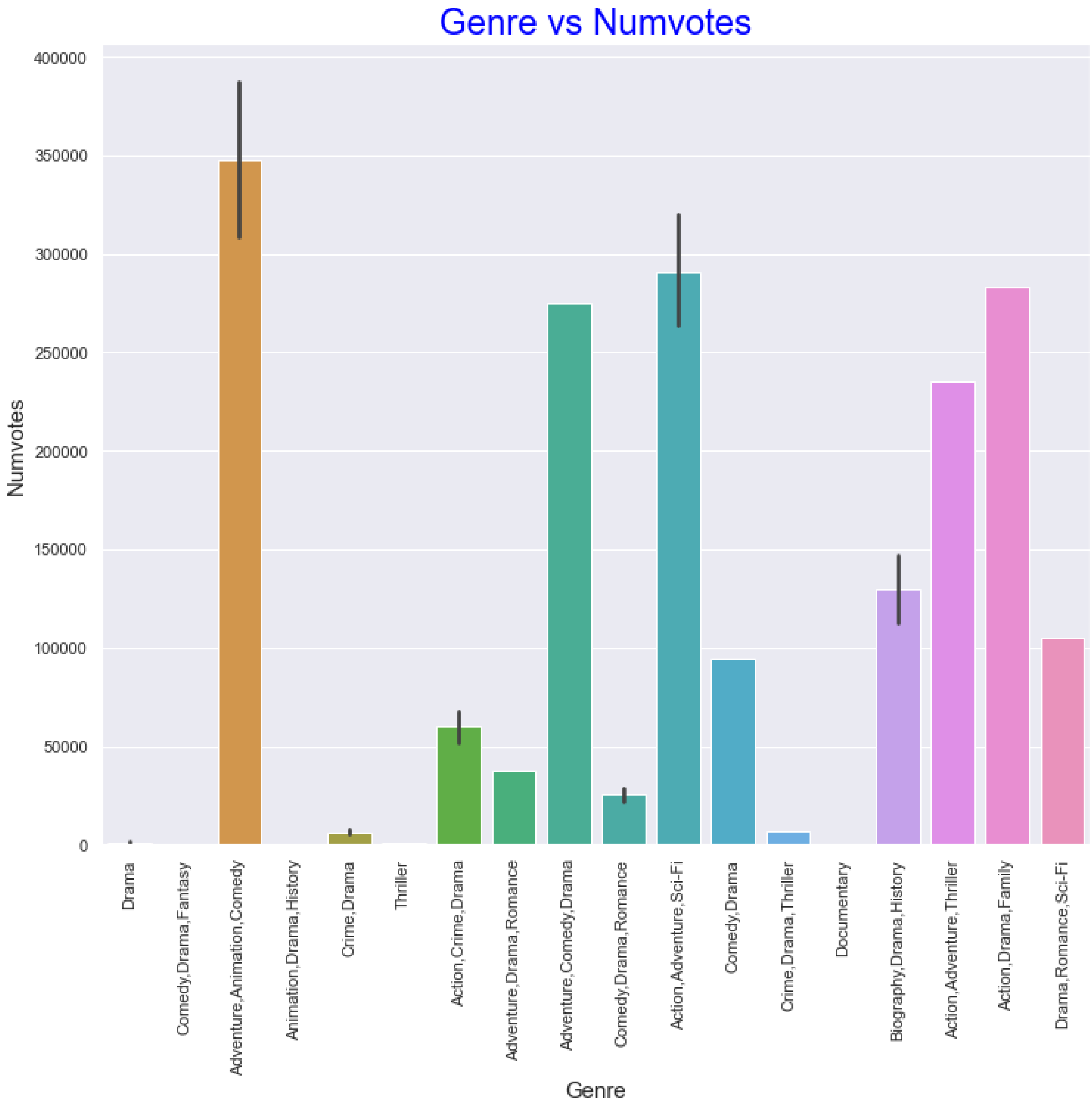
	movie_id	primary_title	original_title	start_year	runtime_minutes	genres	averagerating	nun
2	tt0069049	The Other Side of the Wind	The Other Side of the Wind	2018	122.0	Drama	6.9	
2	tt0069049	The Other Side of the Wind	The Other Side of the Wind	2018	122.0	Drama	6.9	
2	tt0069049	The Other Side of the Wind	The Other Side of the Wind	2018	122.0	Drama	6.9	
2	tt0069049	The Other Side of the Wind	The Other Side of the Wind	2018	122.0	Drama	6.9	
2	tt0069049	The Other Side of the Wind	The Other Side of the Wind	2018	122.0	Drama	6.9	

Exploratory Data Analysis

In the cell below we will plot a bar plot to show the relationship between movie genres and numvotes

In [136]:

```
sns.set(rc = {'figure.figsize': (12,10)} )
ay = sns.barplot(x = 'genres', y = 'numvotes', data = joined_df)
plt.title('Genre vs Numvotes', size = 24, color = 'blue')
ay.set_ylabel('Numvotes', size = 15)
ay.set_xlabel('Genre', size = 15)
plt.xticks(rotation = 90)
plt.savefig("Movie genre vs Numvotes.png", dpi = 80);
```



Analysis

The following movie genres have high ratings and number of votes

1.Adventure, Animation, Comedy

2. Action, Adventure, Sci-Fi

3. Action, Drama, Family

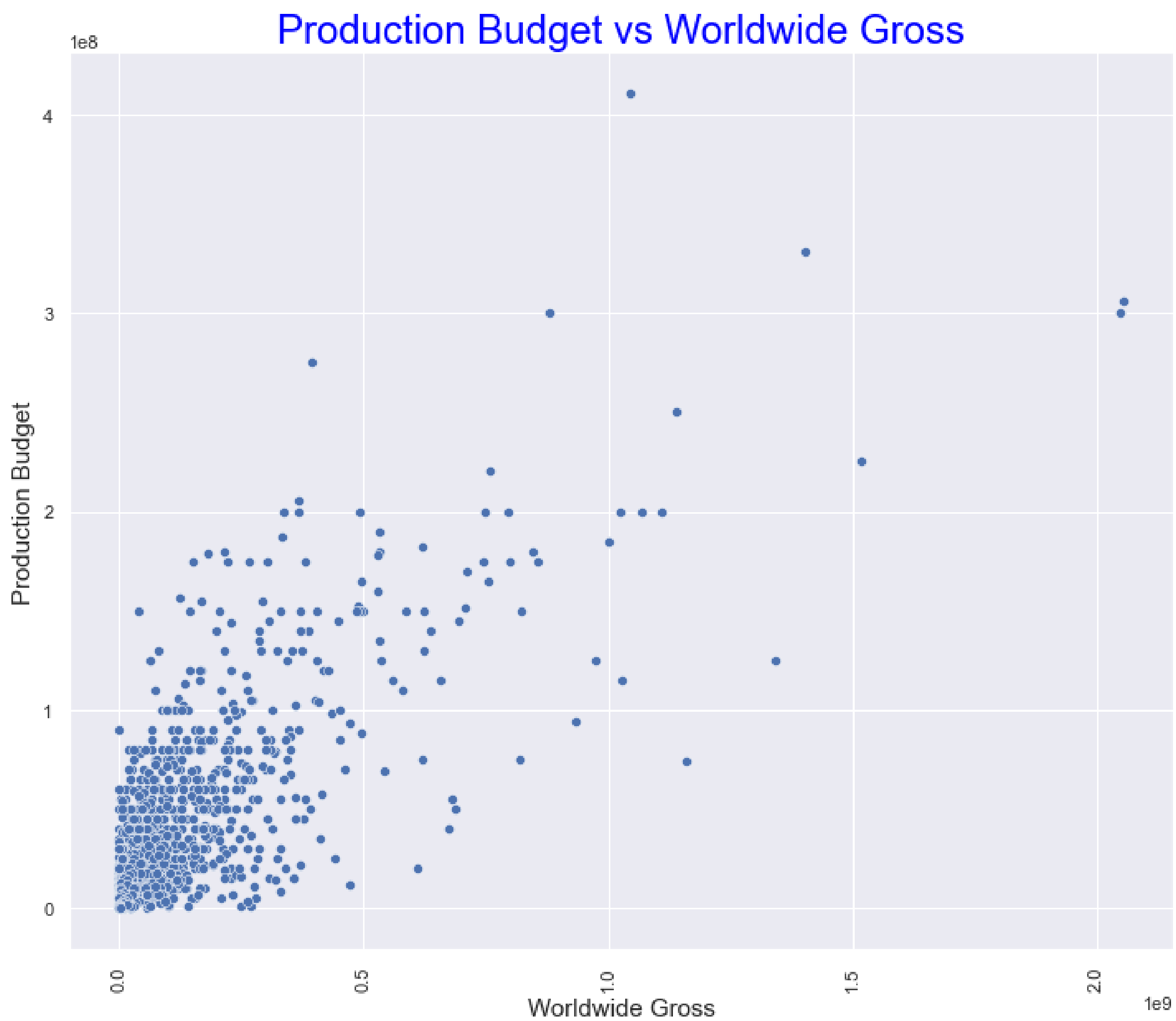
2. Answering the Second Question

Does High Production Cost Translate to High Income?

To answer this question, we have to plot a scatter plot to show the relationship between `production_budget` and `worldwide_gross`

In [129]:

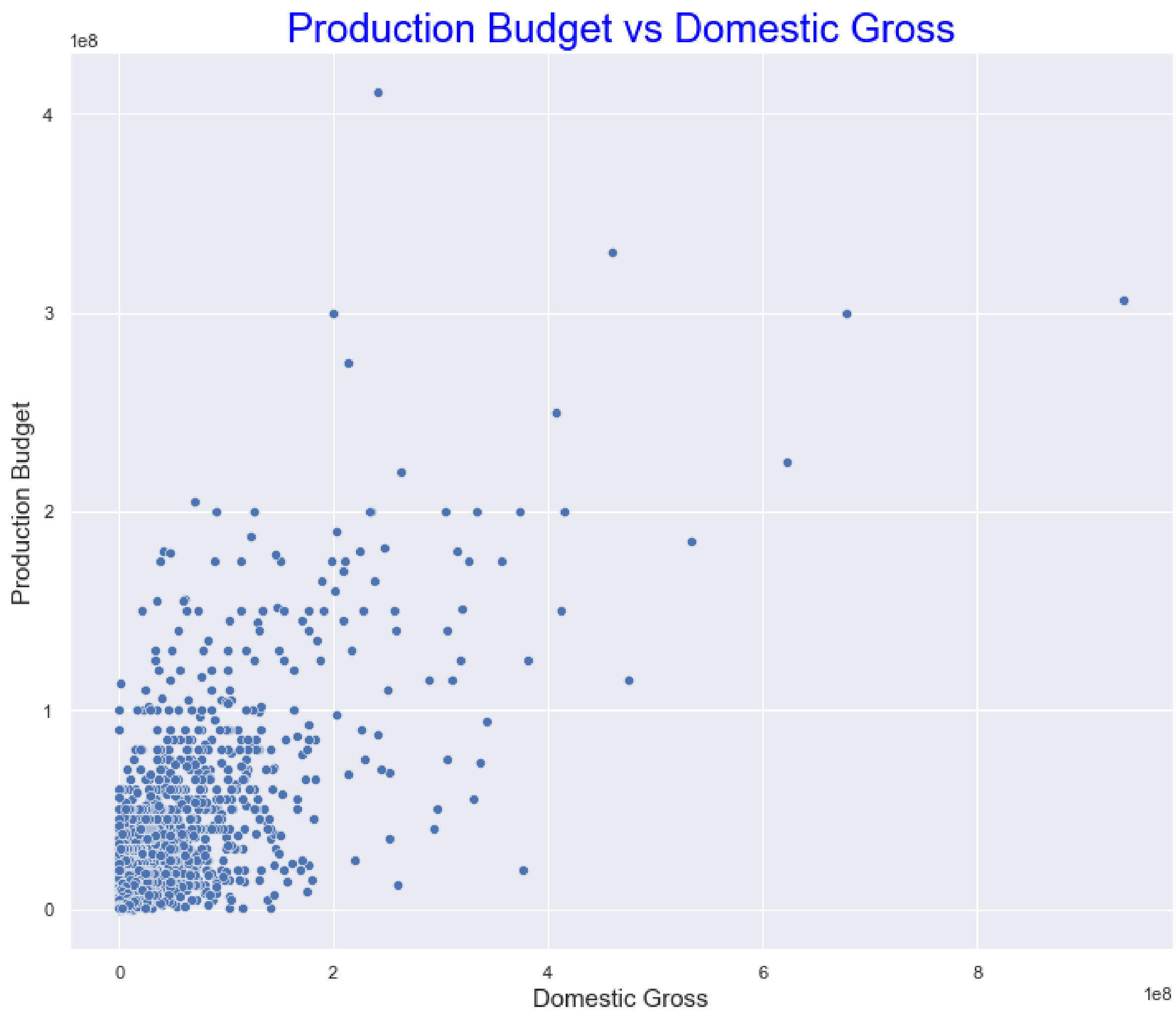
```
sns.set(rc = {'figure.figsize': (12,10)} )
ay = sns.scatterplot(x = 'worldwide_gross', y = 'production_budget', data = joined_d
plt.xticks(rotation = 90)
plt.title('Production Budget vs Worldwide Gross', size = 24, color = 'blue')
ay.set_ylabel('Production Budget', size = 15)
ay.set_xlabel('Worldwide Gross', size = 15)
plt.savefig("Production Budget vs Worldwide Gross.png", dpi = 80);
```



Another scatter plot with `domestic_gross` instead of `worldwide_gross` can give further insight in this relationship

In [130]:

```
sns.set(rc = {'figure.figsize': (12,10)} )
ay = sns.scatterplot(x = 'domestic_gross', y = 'production_budget', data = joined_df
plt.title('Production Budget vs Domestic Gross', size =24, color = 'blue')
ay.set_ylabel('Production Budget', size = 15)
ay.set_xlabel('Domestic Gross', size = 15)
plt.savefig("Production Budget vs Domestic Gross.png", dpi = 80);
```



Analysis

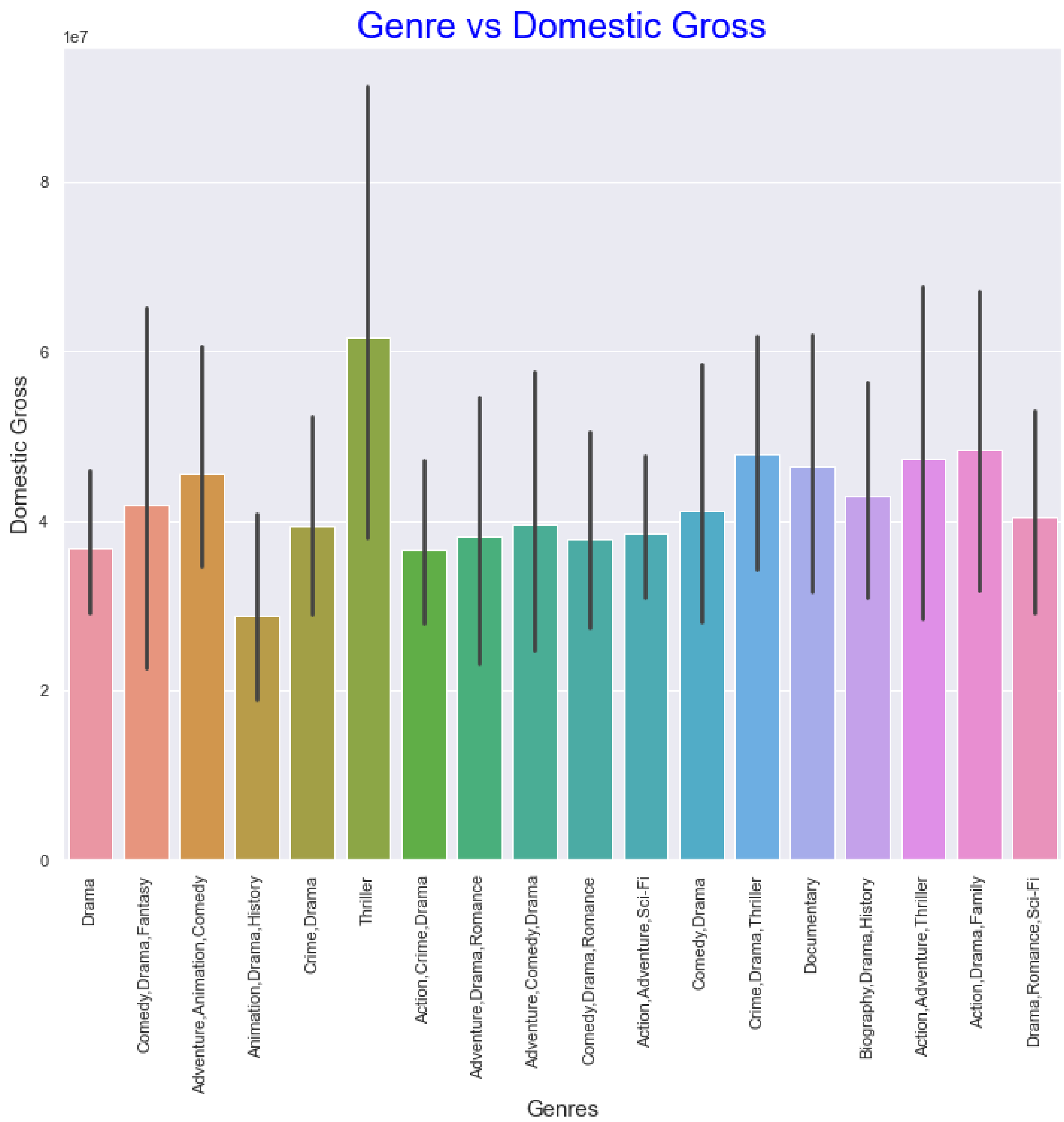
High production budget does not translate to high income

3. Answering the Third Question

Does the movie genre determine its income?

In [131]:

```
sns.set(rc = {'figure.figsize': (12,10)} )
ay = sns.barplot(x = 'genres', y = 'domestic_gross', data = joined_df)
plt.title('Genre vs Domestic Gross', size =24, color = 'blue')
ay.set_ylabel('Domestic Gross', size = 15)
ay.set_xlabel('Genres', size = 15)
plt.xticks(rotation = 90)
plt.savefig("Genre vs Domestic Gross.png", dpi = 80);
```



In [132]:

```
#Second we will compare the movie genre with the profit
#The profit will be calculated by substracting the production budget from worldwide
joined_df['Profit'] = joined_df['worldwide_gross'] - joined_df['production_budget']
```

In [133]:

```
joined_df
```

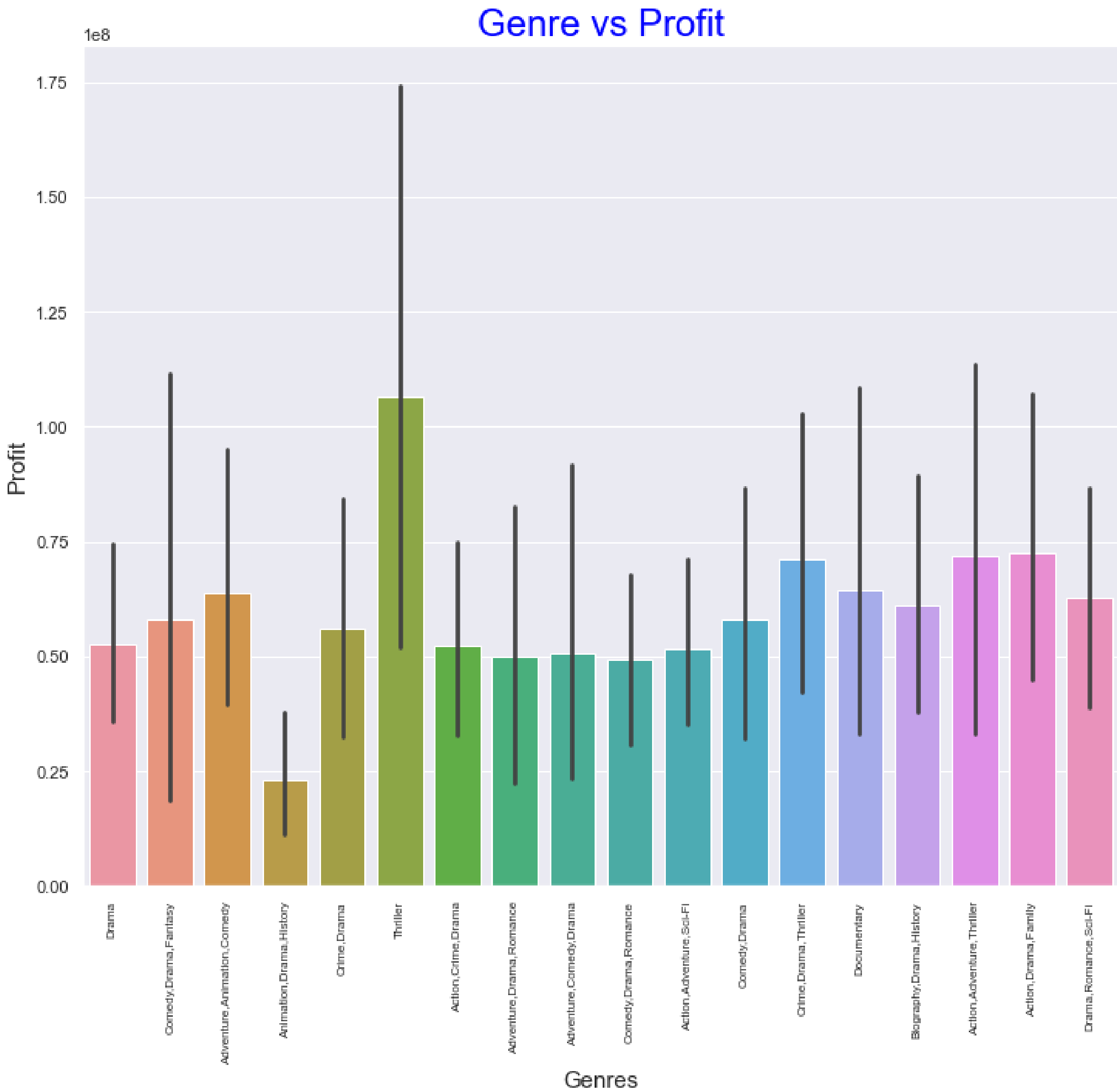
Out[133]:

	movie_id	primary_title	original_title	start_year	runtime_minutes	genres
2	tt0069049	The Other Side of the Wind	The Other Side of the Wind	2018	122.0	Drama
2	tt0069049	The Other Side of the Wind	The Other Side of the Wind	2018	122.0	Drama
2	tt0069049	The Other Side of the Wind	The Other Side of the Wind	2018	122.0	Drama
2	tt0069049	The Other Side of the Wind	The Other Side of the Wind	2018	122.0	Drama
2	tt0069049	The Other Side of the Wind	The Other Side of the Wind	2018	122.0	Drama
...	...	...	...	...	...	...
100	tt0443465	Before We Go	Before We Go	2014	95.0	Comedy,Drama,Romance
100	tt0443465	Before We Go	Before We Go	2014	95.0	Comedy,Drama,Romance
100	tt0443465	Before We Go	Before We Go	2014	95.0	Comedy,Drama,Romance
100	tt0443465	Before We Go	Before We Go	2014	95.0	Comedy,Drama,Romance
100	tt0443465	Before We Go	Before We Go	2014	95.0	Comedy,Drama,Romance

1676 rows × 14 columns

In [134]:

```
sns.set(rc = {'figure.figsize': (12,10)} )
ay = sns.barplot(x = 'genres', y = 'Profit', data = joined_df)
plt.title('Genre vs Profit', size =24, color = 'blue')
ay.set_ylabel('Profit', size = 15)
ay.set_xlabel('Genres', size = 15)
plt.xticks(rotation = 90, size= 8)
plt.savefig("Genre vs Profit.png", dpi = 80);
```



Analysis

Thriller genre makes more money followed by Action, Drama, Family, and Action, Adventure, Thriller