INVESTMENT ANALYSIS INTO THE MOVIE BUSINESS

INTRODUCTION

Investing in movie business is profitable and glamourous and is a high risk busines as well.This project explores the three data sets from assorted movie recommendation websites for analysis and its aim is to give insights on whether it is a viable business venture.This will also enable us to identify which genres to invest in and which are more profitable in the industry. In this project the sole purpose is to advise Microsoft whether to proceed and invest in the movie industry and the possible areas to look at in terms of genres, estimated income and acceptance of the product into the market.

In [1]:
```
#Importing libraries for data exploratory process.
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
df1=pd.read_csv('bom.movie_gross.csv')
df2=pd.read_csv('title.basics.csv')
df3=pd.read_csv('title.ratings.csv')
```

In [3]:
```
df1.head()
```

Out[3]:

	title	studio	domestic_gross	foreign_gross	year
0	Toy Story 3	BV	415000000.0	652000000	2010
1	Alice in Wonderland (2010)	BV	334200000.0	691300000	2010
2	Harry Potter and the Deathly Hallows Part 1	WB	296000000.0	664300000	2010
3	Inception	WB	292600000.0	535700000	2010
4	Shrek Forever After	P/DW	238700000.0	513900000	2010

In [4]:
```
df2.head()
```

Out[4]:

	tconst	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

In [5]:
```
df3.head()
```

Out[5]:

	tconst	averagerating	numvotes
0	tt10356526	8.3	31
1	tt10384606	8.9	559
2	tt1042974	6.4	20
3	tt1043726	4.2	50352
4	tt1060240	6.5	21

In [6]:
```
#merging data with common columns
#merging df2 to df3
df2=df2.merge(df3, on=['tconst'], how = 'left')
```

In [7]:
```
df2.head()
```

Out[7]:

	tconst	primary_title	original_title	start_year	runtime_minutes	genres	averageratii
0	tt0063540	Sunghursh	Sunghursh	2013	175.0	Action,Crime,Drama	7
1	tt0066787	One Day Before the Rainy Season	Ashad Ka Ek Din	2019	114.0	Biography,Drama	7
2	tt0069049	The Other Side of the Wind	The Other Side of the Wind	2018	122.0	Drama	6
3	tt0069204	Sabse Bada Sukh	Sabse Bada Sukh	2018	NaN	Comedy,Drama	6
4	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy	6

In [8]:
```
#merging df1 to df2
#Looping to see how many values of the title column in df1 are in values of primary_tit
list_of_title = []
iterate = 0
for i in df1['title']:
    if i in df2['primary_title'].values:
        iterate += 1
        list_of_title.append(i)
len(list_of_title)
```

Out[8]: 2606

In [9]:
```
#Renaming column in df1 to match the one in df2
df1.rename(columns={'title':'primary_title'}, inplace = True)
```

In [10]:
```
df1.head()
```

Out[10]:

	primary_title	studio	domestic_gross	foreign_gross	year
0	Toy Story 3	BV	415000000.0	652000000	2010
1	Alice in Wonderland (2010)	BV	334200000.0	691300000	2010
2	Harry Potter and the Deathly Hallows Part 1	WB	296000000.0	664300000	2010
3	Inception	WB	292600000.0	535700000	2010
4	Shrek Forever After	P/DW	238700000.0	513900000	2010

In [11]:
```
df2.head()
```

Out[11]:

	tconst	primary_title	original_title	start_year	runtime_minutes	genres	averageratii
0	tt0063540	Sunghursh	Sunghursh	2013	175.0	Action,Crime,Drama	7
1	tt0066787	One Day Before the Rainy Season	Ashad Ka Ek Din	2019	114.0	Biography,Drama	7
2	tt0069049	The Other Side of the Wind	The Other Side of the Wind	2018	122.0	Drama	6
3	tt0069204	Sabse Bada Sukh	Sabse Bada Sukh	2018	NaN	Comedy,Drama	6
4	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy	6

In [12]:
```
df = df1.merge(df2)
```

Exploratory data analysis for the merged dataframe

In [13]:
```
df
```

Out[13]:

	primary_title	studio	domestic_gross	foreign_gross	year	tconst	original_title	start_year	run
0	Toy Story 3	BV	415000000.0	652000000	2010	tt0435761	Toy Story 3	2010	
1	Inception	WB	292600000.0	535700000	2010	tt1375666	Inception	2010	
2	Shrek Forever After	P/DW	238700000.0	513900000	2010	tt0892791	Shrek Forever After	2010	
3	The Twilight Saga: Eclipse	Sum.	300500000.0	398000000	2010	tt1325004	The Twilight Saga: Eclipse	2010	
4	Iron Man 2	Par.	312400000.0	311500000	2010	tt1228705	Iron Man 2	2010	
...	...	...	...	...	...	...	...	...	...
3361	Souvenir	Strand	11400.0	NaN	2018	tt2389092	Souvenir	2014	
3362	Souvenir	Strand	11400.0	NaN	2018	tt3478898	Souvenir	2014	

	primary_title	studio	domestic_gross	foreign_gross	year	tconst	original_title	start_year	rur
3363	Beauty and the Dogs	Osci.	8900.0	NaN	2018	tt6776572	Aala Kaf Ifrit	2017	
3364	The Quake	Magn.	6200.0	NaN	2018	tt6523720	Skjelvet	2018	
3365	An Actor Prepares	Grav.	1700.0	NaN	2018	tt5718046	An Actor Prepares	2018	

3366 rows × 12 columns

In [14]:
```
df.head()
```

Out[14]:

	primary_title	studio	domestic_gross	foreign_gross	year	tconst	original_title	start_year	runtim
0	Toy Story 3	BV	415000000.0	652000000	2010	tt0435761	Toy Story 3	2010	
1	Inception	WB	292600000.0	535700000	2010	tt1375666	Inception	2010	
2	Shrek Forever After	P/DW	238700000.0	513900000	2010	tt0892791	Shrek Forever After	2010	
3	The Twilight Saga: Eclipse	Sum.	300500000.0	398000000	2010	tt1325004	The Twilight Saga: Eclipse	2010	
4	Iron Man 2	Par.	312400000.0	311500000	2010	tt1228705	Iron Man 2	2010	

In [15]:
```
df.tail()
```

Out[15]:

	primary_title	studio	domestic_gross	foreign_gross	year	tconst	original_title	start_year	rur
3361	Souvenir	Strand	11400.0	NaN	2018	tt2389092	Souvenir	2014	
3362	Souvenir	Strand	11400.0	NaN	2018	tt3478898	Souvenir	2014	
3363	Beauty and the Dogs	Osci.	8900.0	NaN	2018	tt6776572	Aala Kaf Ifrit	2017	
3364	The Quake	Magn.	6200.0	NaN	2018	tt6523720	Skjelvet	2018	
3365	An Actor Prepares	Grav.	1700.0	NaN	2018	tt5718046	An Actor Prepares	2018	

In [16]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3366 entries, 0 to 3365
Data columns (total 12 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   primary_title    3366 non-null   object 
 1   studio           3363 non-null   object 
 2   domestic_gross   3342 non-null   float64
 3   foreign_gross    2043 non-null   object 
 4   year             3366 non-null   int64  
```

```
 5   tconst           3366 non-null   object
 6   original_title   3366 non-null   object
 7   start_year       3366 non-null   int64
 8   runtime_minutes  3198 non-null   float64
 9   genres           3326 non-null   object
 10  averagerating    3027 non-null   float64
 11  numvotes         3027 non-null   float64
dtypes: float64(4), int64(2), object(6)
memory usage: 341.9+ KB
```

In [17]:
```
df.describe().T
```

Out[17]:

	count	mean	std	min	25%	50%	75%	m
domestic_gross	3342.0	2.986560e+07	6.532329e+07	100.0	136250.0	1950000.0	31400000.0	700100000
year	3366.0	2.014059e+03	2.445261e+00	2010.0	2012.0	2014.0	2016.0	201
start_year	3366.0	2.013958e+03	2.530699e+00	2010.0	2012.0	2014.0	2016.0	202
runtime_minutes	3198.0	1.049009e+02	2.482994e+01	2.0	92.0	103.0	117.0	62
averagerating	3027.0	6.457582e+00	1.012277e+00	1.6	5.9	6.6	7.1	
numvotes	3027.0	6.170030e+04	1.255132e+05	5.0	2117.0	13109.0	62765.5	184106

In [18]:
```
df.isnull().sum()
```

Out[18]:
```
primary_title         0
studio                3
domestic_gross       24
foreign_gross      1323
year                  0
tconst                0
original_title        0
start_year            0
runtime_minutes     168
genres               40
averagerating       339
numvotes            339
dtype: int64
```

In [19]:
```
df.isnull().any()
```

Out[19]:
```
primary_title      False
studio              True
domestic_gross      True
foreign_gross       True
year               False
tconst             False
original_title     False
start_year         False
runtime_minutes     True
genres              True
averagerating       True
numvotes            True
dtype: bool
```

In [20]:
```
df.duplicated().sum()
```

Out[20]:
```
0
```

DATA CLEANING

For data cleaning prosses, we have missing data in some of the columns that affect our analysis significantly: 1.The numvotes and averagerating column we will keep and replace with 0.0 because for our type of analysis it is significant. Hypothetically it could mean the users did not like the movie or it did not have much viewership. 2.The genres, we will drop the rows because they are not significant to the entire data and being categorical makes it difficult to place the genre of the movie. 3.Runtime munites we will replace with median because most of the data is ranging within the median. 4.Foreign Gross has a very significant amount of missing data and therefore we will drop it,replacing it would alter the results 5.Domestic gross has small missing data which we will drop because its standard deviation is high and replacing it might change the results of the analysis.

In [21]:
```
#Dropping the foreign_gross columns
df.drop('foreign_gross', axis=1, inplace= True)
```

In [22]:
```
#Dropping rows with NaN values for genres and domestic_gross columns

df=df.dropna(subset=['genres','domestic_gross','studio'])
```

In [25]:
```
#Replacing runtime NaN values with median
runtime_median = df['runtime_minutes'].median()
df.loc[:,'runtime_minutes'].fillna(runtime_median, inplace = True)
```

In [26]:
```
#Replacing the numvotes and averagerating columns NaN values with 0.0
df.loc[:,'averagerating'].fillna(value=0.0,inplace=True)
df.loc[:,'numvotes'].fillna(value=0.0,inplace=True)
```

In [27]:
```
df
```

Out[27]:

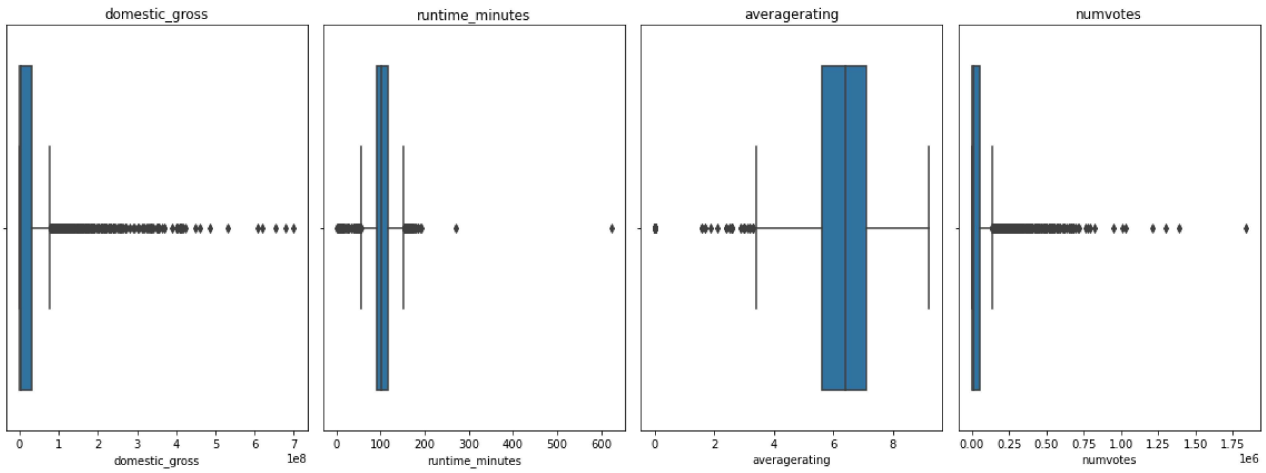
	primary_title	studio	domestic_gross	year	tconst	original_title	start_year	runtime_minutes
0	Toy Story 3	BV	415000000.0	2010	tt0435761	Toy Story 3	2010	103.0
1	Inception	WB	292600000.0	2010	tt1375666	Inception	2010	148.0
2	Shrek Forever After	P/DW	238700000.0	2010	tt0892791	Shrek Forever After	2010	93.0
3	The Twilight Saga: Eclipse	Sum.	300500000.0	2010	tt1325004	The Twilight Saga: Eclipse	2010	124.0
4	Iron Man 2	Par.	312400000.0	2010	tt1228705	Iron Man 2	2010	124.0
...	...	...	...	...	...	...	...	...
3361	Souvenir	Strand	11400.0	2018	tt2389092	Souvenir	2014	86.0
3362	Souvenir	Strand	11400.0	2018	tt3478898	Souvenir	2014	86.0
3363	Beauty and the Dogs	Osci.	8900.0	2018	tt6776572	Aala Kaf Ifrit	2017	100.0
3364	The Quake	Magn.	6200.0	2018	tt6523720	Skjelvet	2018	106.0
3365	An Actor Prepares	Grav.	1700.0	2018	tt5718046	An Actor Prepares	2018	97.0

3301 rows × 11 columns

In [28]:
```
df.isnull().sum()
```

Out[28]:
```
primary_title      0
studio             0
domestic_gross     0
year               0
tconst             0
original_title     0
start_year         0
runtime_minutes    0
genres             0
averagerating      0
numvotes           0
dtype: int64
```

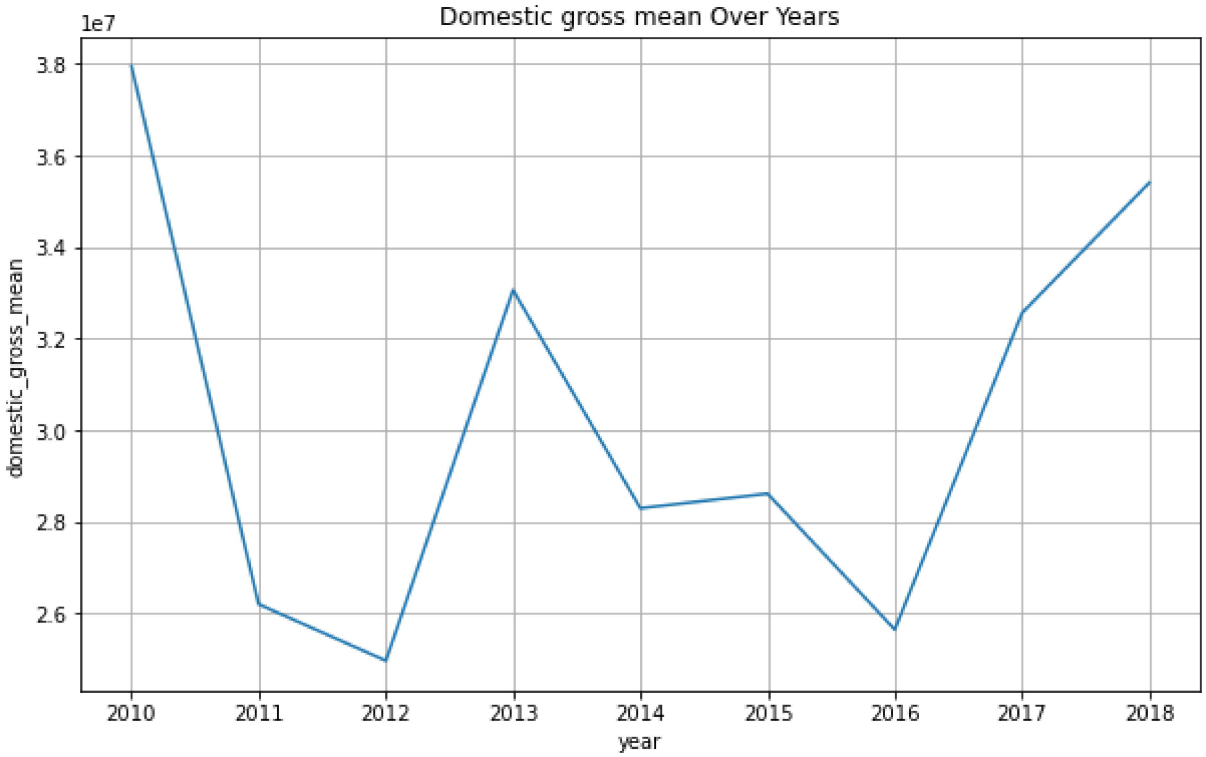
In [29]:
```
#Identifying outliers

columns_to_identify= ['domestic_gross','runtime_minutes','averagerating','numvotes']

#setting up the subplots
fig, axes = plt.subplots(nrows=1, ncols=len(columns_to_identify), figsize =(16,6))

#creating box plot for each column
for i, column in enumerate(columns_to_identify):
    sns.boxplot(x=df[column], ax=axes[i])
    axes[i].set_title(column)

plt.tight_layout()
plt.show()
```



Data Visualization

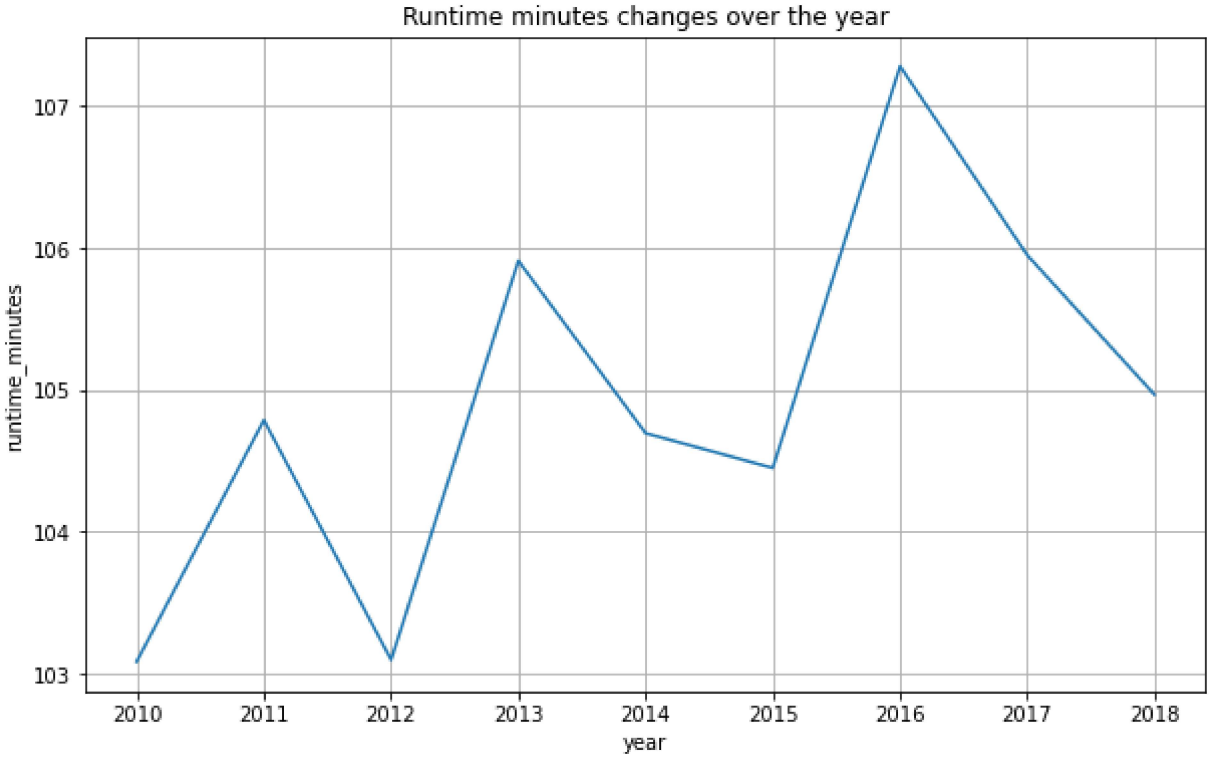
In [30]:
```
# line graph showing domestic gross trend yearly
#grouping the data by year and calculating the mean of domestic gross

domestic_gross_mean = df.groupby('year')['domestic_gross'].mean().reset_index()

#plotting the line graph

plt.figure(figsize=(10,6))
plt.plot(domestic_gross_mean['year'],domestic_gross_mean['domestic_gross'])
```

```
plt.title('Domestic gross mean Over Years')
plt.xlabel('year')
plt.ylabel('domestic_gross_mean')
plt.grid(True)
plt.show()
```



In [31]:

```
# line graph showing runtime trend yearly
#grouping the data by year and calculating the mean of runtime minutes

runtime_mean = df.groupby('year')['runtime_minutes'].mean().reset_index()

#plotting the line graph

plt.figure(figsize=(10,6))
plt.plot(runtime_mean['year'],runtime_mean['runtime_minutes'])
plt.title('Runtime minutes changes over the year')
plt.xlabel('year')
plt.ylabel('runtime_minutes')
plt.grid(True)
plt.show()
```



In [32]:
```
#Grouping each genre in relation to domestic gross, runtime, averagerating and numvotes
df.groupby('genres').agg({'domestic_gross':'mean','runtime_minutes':'mean','averagerati
```

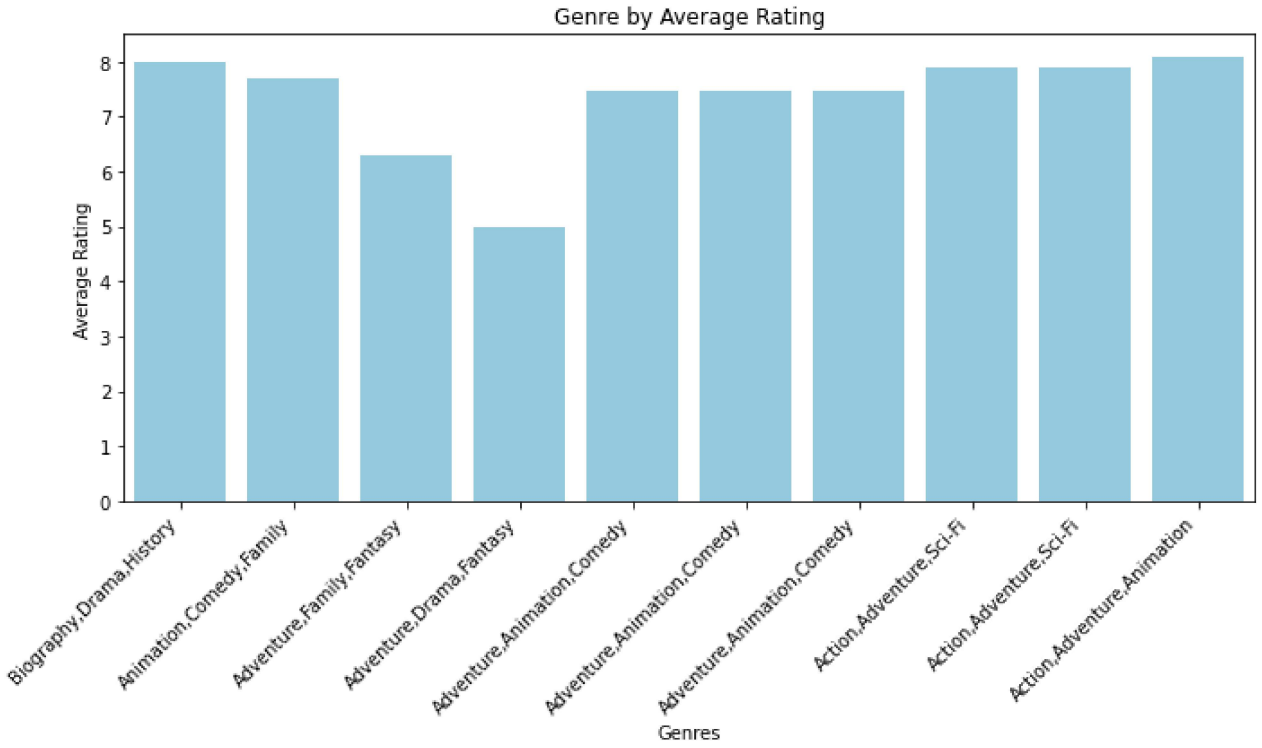
Out[32]:

	genres	domestic_gross	runtime_minutes	averagerating	numvotes
0	Action	1.580743e+07	115.136364	5.004545	5691.272727
1	Action,Adventure	5.408333e+04	113.666667	5.866667	4892.333333
2	Action,Adventure,Animation	9.930275e+07	100.227273	7.354545	124986.818182
3	Action,Adventure,Biography	6.005725e+07	128.750000	7.000000	191598.000000
4	Action,Adventure,Comedy	9.913976e+07	110.718750	6.271875	181259.937500
...	...	...	...	...	...
323	Sci-Fi	8.302002e+07	83.000000	2.020000	704.200000
324	Sport	5.300000e+06	114.000000	7.900000	77.000000
325	Thriller	1.715469e+07	94.918919	3.832432	1244.351351
326	Thriller,Western	2.110000e+04	95.000000	6.400000	7874.000000
327	Western	1.070000e+07	103.000000	0.000000	0.000000

328 rows × 5 columns

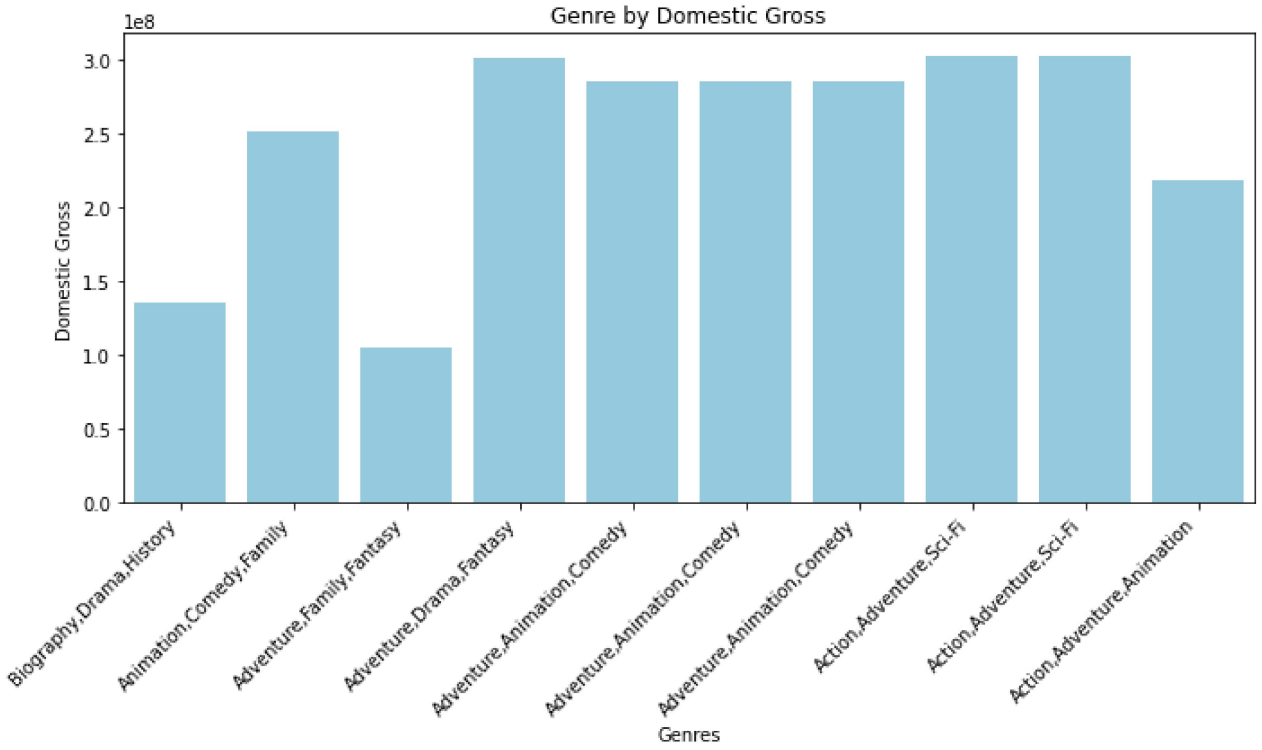
In [33]:
```
# A bar plot for genre vs average rating
#sorting in Descending order
df.sort_values(by='averagerating', ascending=False)

#Selecting the top 10
top_data=df.head(10)
```

```
#Creating the bar chart
plt.figure(figsize=(10,6))
sns.barplot(x='genres', y='averagerating', data=top_data, color='skyblue', ci=None, ord
plt.xlabel('Genres')
plt.ylabel('Average Rating')
plt.title('Genre by Average Rating')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

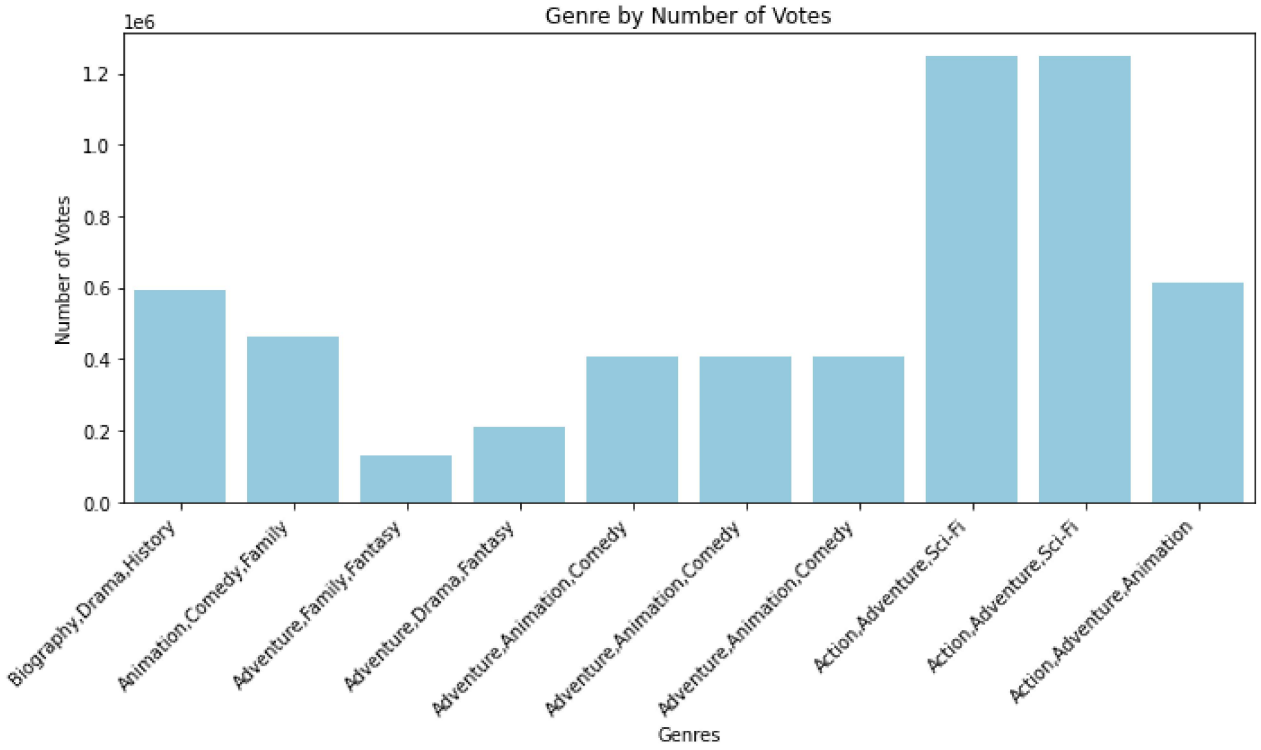


In [34]:

```
# A bar plot for genre vs Domestic gross
#sorting in Descending order
df.sort_values(by='domestic_gross', ascending=False)

#Selecting the top 10
top_data=df.head(10)

#Creating the bar chart
plt.figure(figsize=(10,6))
sns.barplot(x='genres', y='domestic_gross', data=top_data, color='skyblue', ci=None, or
plt.xlabel('Genres')
plt.ylabel('Domestic Gross')
plt.title('Genre by Domestic Gross')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



In [33]:
```
# A bar plot for genre vs Number of Votes
#sorting in Descending order
df.sort_values(by='numvotes', ascending=False)

#Selecting the top 10
top_data=df.head(10)

#Creating the bar chart
plt.figure(figsize=(10,6))
sns.barplot(x='genres', y='numvotes', data=top_data,color='skyblue', ci=None, order=top
plt.xlabel('Genres')
plt.ylabel('Number of Votes')
plt.title('Genre by Number of Votes')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



Recommendations

The above analysis indicates and shows the different distribution of ratings, number of votes and domestic gross against different genres. The movies with highest rankings based on the mentioned critea, I would recommend investment gravitated towards; Action,Adventure,Drama,Sci-Fi,Comedy and Animation or and preferably a combination of these genres. At the begining the film industry shows drastic changes in income from one year to the next, however, from 2016 it has taken an upward trajectory therefore it is business to be considered and invest in and shows a possibility of growth in coming future.

Conclusion

The film indstry has great possibilities of growth because it dynamic, evolving, vast and robust. It has diverse genres that give coverage to a diverse target group based on preferences from comedy,action,adventure,Sci-Fi among others.The industry is able to generate income widely in this case both domestic and foreign income. In conclusion it a viable business venture.