Business Problem

Microsoft sees all the big companies creating original video content and they want to get in on the fun. They have decided to create a new movie studio, but they don't know anything about creating movies. You are charged with exploring what types of films are currently doing the best at the box office.In the following project, I will attemept to translate those findings into actionable insights that the head of Microsoft's new movie studio can use to help decide what type of films to create.

In [58]:
```
#Start by importing the necessary libraries

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [59]:
```
#Open csv file as a Pandas DataFrame to load the data

df = pd.read_csv('title_basics.csv')
df
```

Out[59]:

	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
...	...	...	...	...	...	...
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	NaN
146143	tt9916754	Chico Albuquerque - Revelações	Chico Albuquerque - Revelações	2013	NaN	Documentary

146144 rows × 6 columns

In [60]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 146144 entries, 0 to 146143
Data columns (total 6 columns):
 #   Column           Non-Null Count   Dtype  
---  ------           --------------   -----  
 0   tconst           146144 non-null  object 
 1   primary_title    146143 non-null  object 
 2   original_title   146122 non-null  object 
 3   start_year       146144 non-null  int64  
 4   runtime_minutes  114405 non-null  float64
 5   genres           140736 non-null  object 
dtypes: float64(1), int64(1), object(4)
memory usage: 6.7+ MB
```

In [61]:
```
#The columns primary_title and original_title have have almost identical rows,
#Having noted similarities in the two columns primary title and original title drop the primary_title

# Drop a column using axis=1
column_to_drop = 'primary_title'
newdf = df.drop('primary_title', axis=1)

# Display the resulting DataFrame
print(newdf)
```

```
          tconst                              original_title  start_year  \
0       tt0063540                                   Sunghursh        2013
1       tt0066787                                Ashad Ka Ek Din        2019
2       tt0069049                    The Other Side of the Wind        2018
3       tt0069204                                Sabse Bada Sukh        2018
4       tt0100275                          La Telenovela Errante        2017
...           ...                                           ...         ...
146139  tt9916538                          Kuambil Lagi Hatiku        2019
146140  tt9916622  Rodolpho Teóphilo - O Legado de um Pioneiro        2015
146141  tt9916706                                Dankyavar Danka        2013
146142  tt9916730                                         6 Gunn        2017
146143  tt9916754                Chico Albuquerque - Revelações        2013

        runtime_minutes                genres
0                 175.0     Action,Crime,Drama
1                 114.0        Biography,Drama
2                 122.0                  Drama
3                   NaN           Comedy,Drama
4                  80.0   Comedy,Drama,Fantasy
...                 ...                    ...
146139            123.0                  Drama
146140              NaN            Documentary
146141              NaN                 Comedy
146142            116.0                    NaN
146143              NaN            Documentary

[146144 rows x 5 columns]
```

In [62]:
```
# Proceed to rename the column original_title to title
newdf = newdf.rename(columns={'original_title': 'title'})

# Display the DataFrame after renaming the column
newdf.head()
```

Out[62]:

	tconst	title	start_year	runtime_minutes	genres
0	tt0063540	Sunghursh	2013	175.0	Action,Crime,Drama
1	tt0066787	Ashad Ka Ek Din	2019	114.0	Biography,Drama
2	tt0069049	The Other Side of the Wind	2018	122.0	Drama
3	tt0069204	Sabse Bada Sukh	2018	NaN	Comedy,Drama
4	tt0100275	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy

In [63]:
```
#Open csv file as a Pandas DataFrame to load the data

df1 = pd.read_csv('title_ratings.csv')
df1
```

Out[63]:

	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
...	...	...	...
73851	tt9805820	8.1	25
73852	tt9844256	7.5	24
73853	tt9851050	4.7	14
73854	tt9886934	7.0	5
73855	tt9894098	6.3	128

73856 rows × 3 columns

In [64]:
```
#Open csv file as a Pandas DataFrame to load the data

df2 = pd.read_csv('bom_movie_gross.csv')
df2
```

Out[64]:

	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
...	...	...	...	...	...
3382	The Quake	Magn.	6200.0	NaN	2018
3383	Edward II (2018 re-release)	FM	4800.0	NaN	2018
3384	El Pacto	Sony	2500.0	NaN	2018
3385	The Swan	Synergetic	2400.0	NaN	2018
3386	An Actor Prepares	Grav.	1700.0	NaN	2018

3387 rows × 5 columns

In [65]:
```
# Combing title_basics and title_ratings

#To merge the dataframes, we use the following code:
merged_df = pd.merge(newdf, df1, on='tconst', how='outer')
merged_df.head()
```

Out[65]:

	tconst	title	start_year	runtime_minutes	genres	averagerating	numvotes
0	tt0063540	Sunghursh	2013	175.0	Action,Crime,Drama	7.0	77.0
1	tt0066787	Ashad Ka Ek Din	2019	114.0	Biography,Drama	7.2	43.0
2	tt0069049	The Other Side of the Wind	2018	122.0	Drama	6.9	4517.0
3	tt0069204	Sabse Bada Sukh	2018	NaN	Comedy,Drama	6.1	13.0
4	tt0100275	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy	6.5	119.0

In [66]:
```
merged_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 146144 entries, 0 to 146143
Data columns (total 7 columns):
 #   Column           Non-Null Count   Dtype  
---  ------           --------------   -----  
 0   tconst           146144 non-null  object 
 1   title            146122 non-null  object 
 2   start_year       146144 non-null  int64  
 3   runtime_minutes  114405 non-null  float64
 4   genres           140736 non-null  object 
 5   averagerating    73856 non-null   float64
 6   numvotes         73856 non-null   float64
dtypes: float64(3), int64(1), object(3)
memory usage: 7.8+ MB
```

In [67]:
```
# Check for missing values
merged_df.isna().sum()
```

Out[67]:
```
tconst                0
title                22
start_year            0
runtime_minutes    31739
genres              5408
averagerating      72288
numvotes           72288
dtype: int64
```

In [68]:
```
# Drop null values

merged_df.dropna(inplace=True)
merged_df.head()
```

Out[68]:

	tconst	title	start_year	runtime_minutes	genres	averagerating	numvotes
0	tt0063540	Sunghursh	2013	175.0	Action,Crime,Drama	7.0	77.0
1	tt0066787	Ashad Ka Ek Din	2019	114.0	Biography,Drama	7.2	43.0
2	tt0069049	The Other Side of the Wind	2018	122.0	Drama	6.9	4517.0
4	tt0100275	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy	6.5	119.0
7	tt0137204	Joe Finds Grace	2017	83.0	Adventure,Animation,Comedy	8.1	263.0

In [69]:
```
merged_df.isna().sum()
```

Out[69]:
```
tconst             0
title              0
start_year         0
runtime_minutes    0
genres             0
averagerating      0
numvotes           0
dtype: int64
```

In [70]:
```
merged_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 65720 entries, 0 to 146134
Data columns (total 7 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   tconst           65720 non-null  object 
 1   title            65720 non-null  object 
 2   start_year       65720 non-null  int64  
 3   runtime_minutes  65720 non-null  float64
 4   genres           65720 non-null  object 
 5   averagerating    65720 non-null  float64
 6   numvotes         65720 non-null  float64
dtypes: float64(3), int64(1), object(3)
memory usage: 4.0+ MB
```

In [71]:
```
#To merge merged_df and df2 we use the following code
merged_df4 = pd.merge(merged_df, df2, on='title', how='outer')
merged_df4.head()
```

Out[71]:

	tconst	title	start_year	runtime_minutes	genres	averagerating	numvotes	studio	domestic_gross	foreign_gross	yea
0	tt0063540	Sunghursh	2013.0	175.0	Action,Crime,Drama	7.0	77.0	NaN	NaN	NaN	NaN
1	tt0066787	Ashad Ka Ek Din	2019.0	114.0	Biography,Drama	7.2	43.0	NaN	NaN	NaN	NaN
2	tt0069049	The Other Side of the Wind	2018.0	122.0	Drama	6.9	4517.0	NaN	NaN	NaN	NaN
3	tt0100275	La Telenovela Errante	2017.0	80.0	Comedy,Drama,Fantasy	6.5	119.0	NaN	NaN	NaN	NaN
4	tt0137204	Joe Finds Grace	2017.0	83.0	Adventure,Animation,Comedy	8.1	263.0	NaN	NaN	NaN	NaN

In [73]:
```
merged_df4.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 66975 entries, 0 to 66974
Data columns (total 11 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   tconst           65720 non-null  object 
 1   title            66975 non-null  object 
 2   start_year       65720 non-null  float64
 3   runtime_minutes  65720 non-null  float64
 4   genres           65720 non-null  object 
 5   averagerating    65720 non-null  float64
 6   numvotes         65720 non-null  float64
 7   studio           3651 non-null   object 
 8   domestic_gross   3623 non-null   float64
 9   foreign_gross    2209 non-null   object 
 10  year             3656 non-null   float64
dtypes: float64(6), object(5)
memory usage: 5.6+ MB
```

In [74]:
```
# Check for missing values
merged_df4.isna().sum()
```

Out[74]:
```
tconst             1255
title                 0
start_year         1255
runtime_minutes    1255
genres             1255
averagerating      1255
numvotes           1255
studio            63324
domestic_gross    63352
foreign_gross     64766
year              63319
dtype: int64
```

In [75]:
```
# Drop null values

merged_df4.dropna(inplace=True)
merged_df4.head()
```

Out[75]:

	tconst	title	start_year	runtime_minutes	genres	averagerating	numvotes	studio	domestic_gross	foreign_gross	y
31	tt0337692	On the Road	2012.0	124.0	Adventure,Drama,Romance	6.1	37886.0	IFC	744000.0	8000000	201
32	tt4339118	On the Road	2014.0	89.0	Drama	6.0	6.0	IFC	744000.0	8000000	201
33	tt5647250	On the Road	2016.0	121.0	Drama	5.7	127.0	IFC	744000.0	8000000	201
38	tt0359950	The Secret Life of Walter Mitty	2013.0	114.0	Adventure,Comedy,Drama	7.3	275300.0	Fox	58200000.0	129900000	201
42	tt0365907	A Walk Among the Tombstones	2014.0	114.0	Action,Crime,Drama	6.5	105116.0	Uni.	26300000.0	26900000	201

In [76]:
```
merged_df4.isna().sum()
```

Out[76]:
```
tconst             0
title              0
start_year         0
runtime_minutes    0
genres             0
averagerating      0
numvotes           0
studio             0
domestic_gross     0
foreign_gross      0
year               0
dtype: int64
```

In [77]:
```
merged_df4.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1518 entries, 31 to 64832
Data columns (total 11 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   tconst           1518 non-null   object 
 1   title            1518 non-null   object 
 2   start_year       1518 non-null   float64
 3   runtime_minutes  1518 non-null   float64
 4   genres           1518 non-null   object 
 5   averagerating    1518 non-null   float64
 6   numvotes         1518 non-null   float64
 7   studio           1518 non-null   object 
 8   domestic_gross   1518 non-null   float64
 9   foreign_gross    1518 non-null   object 
 10  year             1518 non-null   float64
dtypes: float64(6), object(5)
memory usage: 142.3+ KB
```

Data cleaning

In this section, we clean the data to generate quality data.

###We look out for unnecessary data

In [78]:
```
# Drop unneeded column

merged_df5 = merged_df4.drop('tconst', axis=1)
merged_df5.head()
```

Out[78]:

	title	start_year	runtime_minutes	genres	averagerating	numvotes	studio	domestic_gross	foreign_gross	year
31	On the Road	2012.0	124.0	Adventure,Drama,Romance	6.1	37886.0	IFC	744000.0	8000000	2012.0
32	On the Road	2014.0	89.0	Drama	6.0	6.0	IFC	744000.0	8000000	2012.0
33	On the Road	2016.0	121.0	Drama	5.7	127.0	IFC	744000.0	8000000	2012.0
38	The Secret Life of Walter Mitty	2013.0	114.0	Adventure,Comedy,Drama	7.3	275300.0	Fox	58200000.0	129900000	2013.0
42	A Walk Among the Tombstones	2014.0	114.0	Action,Crime,Drama	6.5	105116.0	Uni.	26300000.0	26900000	2014.0

We check for missing values
Missing values would make our data inaccurate which may lead to bias in decision making.

In [79]:
```
# Check for missing values

merged_df5.isna().sum()

#There are no missing values
```

Out[79]:
```
title              0
start_year         0
runtime_minutes    0
genres             0
averagerating      0
numvotes           0
studio             0
domestic_gross     0
foreign_gross      0
year               0
dtype: int64
```

We check for duplicates

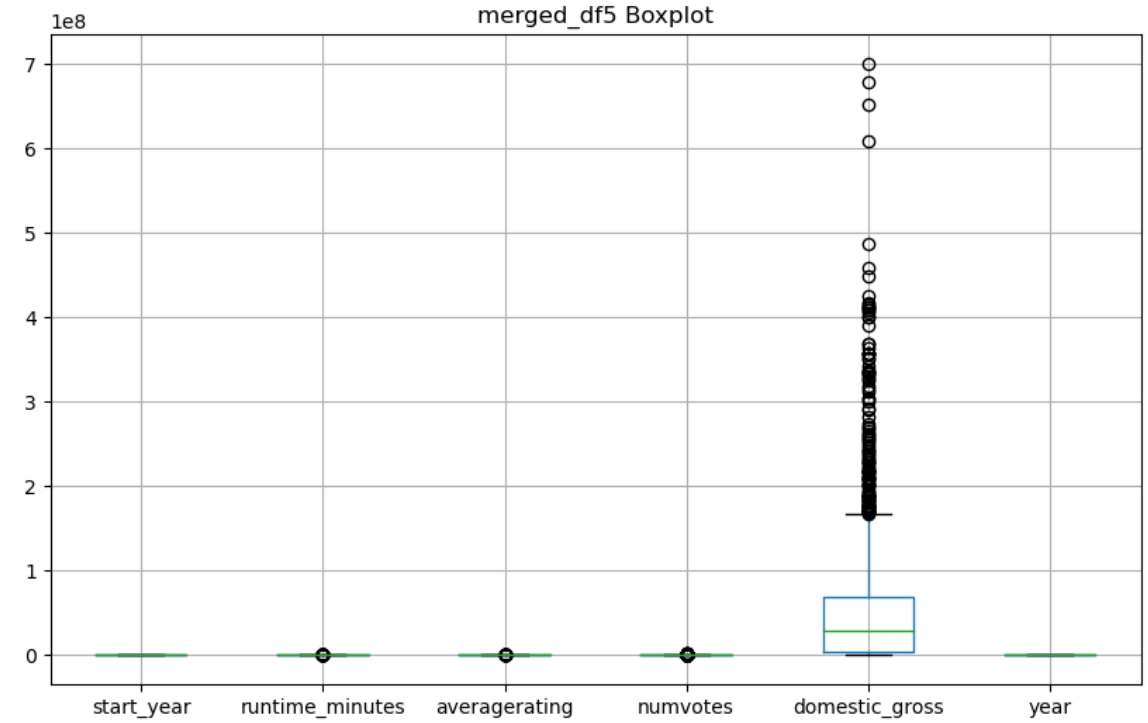
In [80]:
```
#To check for duplicates we use the code
merged_df5.duplicated().sum()

#No duplicates exist, we proceed
```

Out[80]:
```
0
```

Handling outliers

```
Outliers are extreme values that greatly stand out  an overall pattern of values. They may manifest as a typo during
data entry.
They are mainly dependent on context, proporton domain
```

In [81]:
```
# Plot a boxplot
fig, ax = plt.subplots(figsize=(10, 6))  # Adjust the width and height as needed
merged_df5.boxplot(ax=ax)
plt.title('merged_df5 Boxplot')
plt.show()
```



Exploratory Data Analysis

This will help us get a good perspective on the company's data

In [82]:
```
# numvotes analysis
merged_df5.numvotes.value_counts()
```

Out[82]:
```
numvotes
6.0         7
7.0         7
8.0         7
14.0        7
10.0        6
           ..
249501.0    1
10726.0     1
263328.0    1
83532.0     1
2067.0      1
Name: count, Length: 1446, dtype: int64
```

In [83]:
```
# average rating analysis
merged_df5.averagerating.value_counts()
```

Out[83]:
```
averagerating
6.3    75
6.2    75
6.6    72
6.5    67
6.8    66
       ..
8.7     1
1.6     1
2.9     1
3.3     1
2.1     1
Name: count, Length: 64, dtype: int64
```

In [84]:
```
# runtime_minutes analysis
merged_df5.runtime_minutes.value_counts()
```

Out[84]:
```
runtime_minutes
105.0    43
107.0    43
106.0    41
100.0    39
103.0    38
         ..
57.0      1
3.0       1
180.0     1
65.0      1
167.0     1
Name: count, Length: 119, dtype: int64
```

In [85]:
```
# startyear analysis
merged_df5.start_year.value_counts()
```

Out[85]:
```
start_year
2010.0    234
2011.0    194
2016.0    167
2014.0    164
2012.0    161
2013.0    159
2015.0    151
2017.0    151
2018.0    128
2019.0      9
Name: count, dtype: int64
```

In [86]:
```
#To find the unique values in genres we use the below code
merged_df5['genres'].unique()
```

Out[86]:
```
array(['Adventure,Drama,Romance', 'Drama', 'Adventure,Comedy,Drama',
       'Action,Crime,Drama', 'Action,Adventure,Sci-Fi',
       'Action,Comedy,Crime', 'Comedy,Drama', 'Comedy,Family',
       'Adventure,Animation,Comedy', 'Action,Sci-Fi,Thriller', 'Comedy',
       'Action,Adventure,Thriller', 'Horror,Mystery,Thriller',
       'Action,Drama,Family', 'Drama,Romance,Sci-Fi',
       'Biography,Drama,History', 'Action,Comedy,Fantasy',
       'Action,Adventure,Animation', 'Action,Adventure,Fantasy', 'Sci-Fi',
       'Documentary,Drama,Sport', 'Adventure,Drama,Fantasy',
       'Horror,Thriller', 'Action,Crime,Thriller', 'Comedy,Horror',
       'Drama,Mystery,Sci-Fi', 'Comedy,Drama,Music', 'Action,Thriller',
       'Documentary', 'Adventure,Drama,Mystery', 'Drama,Fantasy',
       'Action,Adventure,Comedy', 'Action,Adventure,Crime',
       'Comedy,Romance', 'Action,Adventure,Drama', 'Comedy,Drama,Romance',
       'Drama,History,Romance', 'Adventure,Comedy,Family',
       'Drama,Horror,Mystery', 'Drama,Fantasy,Horror', 'Biography,Drama',
       'Adventure,Comedy,Fantasy', 'Biography,Drama,Romance',
       'Biography,Drama,Music', 'Comedy,Family,Fantasy', 'Drama,War',
       'Adventure,Drama,Sci-Fi', 'Action,Adventure,Horror',
       'Action,Drama,Fantasy', 'Animation,Comedy,Family',
       'Crime,Drama,Thriller', 'Animation,Comedy,Crime',
       'Adventure,Animation,Family', 'Drama,Romance',
       'Adventure,Drama,Thriller', 'Adventure,Family,Fantasy',
       'Action,Adventure,Family', 'Action,Mystery,Thriller',
       'Action,Drama,Sci-Fi', 'Drama,Thriller', 'Action,Drama,Thriller',
       'Crime,Drama,Horror', 'Action,Drama', 'Biography,Drama,Sport',
       'Crime,Drama,History', 'Adventure,Drama,Family',
       'Biography,Comedy,Drama', 'Drama,Romance,War',
       'Biography,Crime,Drama', 'Action,Animation,Comedy', 'Action',
       'Adventure', 'Adventure,Drama,History', 'Biography,Drama,Thriller',
       'Biography,Drama,Family', 'Action,Fantasy,Western',
       'Action,Comedy,Sci-Fi', 'Fantasy,Horror,Mystery',
       'Action,Comedy,Romance', 'Action,Family,Fantasy',
       'Action,Fantasy,Horror', 'Action,Crime,Sci-Fi',
       'Action,Fantasy,Thriller', 'Comedy,Fantasy,Horror',
       'Action,Biography,Drama', 'Adventure,Comedy,Sci-Fi',
       'Crime,Drama,Mystery', 'Crime,Mystery,Thriller',
       'Documentary,News', 'Horror', 'Drama,Music,Musical',
       'Mystery,Thriller', 'Drama,Mystery,Thriller',
       'Drama,Fantasy,Romance', 'Biography,Documentary',
       'Adventure,Fantasy', 'Adventure,Biography,Comedy',
       'Biography,Comedy,Crime', 'Comedy,Drama,Family',
       'Action,Drama,Romance', 'Action,Adventure,Western',
       'Action,Crime,Mystery', 'Adventure,Biography,Drama', 'Crime,Drama',
       'Mystery,Sci-Fi,Thriller', 'Comedy,Crime,Drama',
       'Comedy,Family,Romance', 'Adventure,Documentary',
       'Adventure,Comedy,Music', 'Adventure,Comedy', 'Comedy,Sci-Fi',
       'Animation,Crime,Drama', 'Thriller', 'Biography,Documentary,Sport',
       'Action,Comedy', 'Horror,Mystery', 'Fantasy,Horror,Thriller',
       'Action,Comedy,Family', 'Comedy,Drama,Fantasy',
       'Comedy,Crime,Romance', 'Drama,Sport', 'Crime,Thriller',
       'Drama,Music,Romance', 'Action,Comedy,Horror',
       'Comedy,Crime,History', 'Drama,Horror,Thriller',
       'Adventure,Drama,Sport', 'Crime,Horror,Mystery',
       'Comedy,Drama,Musical', 'Drama,Fantasy,Music',
       'Drama,Romance,Thriller', 'Drama,History',
       'Drama,Fantasy,Thriller', 'Drama,Mystery,Western',
       'Action,Drama,Mystery', 'Drama,Thriller,War',
       'Romance,Sci-Fi,Thriller', 'Drama,Sci-Fi',
       'Action,Adventure,Biography', 'Drama,Mystery',
       'Adventure,Drama,Western', 'Comedy,Romance,Sport', 'Family,Sport',
       'Comedy,Mystery', 'Action,Drama,History', 'Drama,Family,Music',
       'Drama,Family', 'Action,Crime', 'Documentary,Music',
       'Adventure,Mystery,Sci-Fi', 'Drama,Sci-Fi,Thriller',
       'Comedy,Drama,Sport', 'Comedy,Crime', 'Biography,Drama,Musical',
       'Comedy,Fantasy', 'Romance,Thriller', 'Comedy,Mystery,Romance',
       'Crime,Drama,Fantasy', 'Thriller,Western', 'Drama,Music',
       'Comedy,Crime,Thriller', 'Comedy,Crime,Mystery',
       'Crime,Drama,Romance', 'Documentary,War', 'Action,History',
       'Drama,History,War', 'Documentary,Drama,Mystery',
       'Action,Biography,Crime', 'Comedy,Crime,Documentary',
       'Comedy,Horror,Romance', 'Drama,Family,Sport',
       'Action,Comedy,Drama', 'Action,Sci-Fi', 'Comedy,Fantasy,Romance',
       'Adventure,History', 'Biography,Documentary,Drama', 'Family',
       'Biography,Documentary,History', 'Crime,Documentary',
       'Comedy,Drama,Mystery', 'Action,Comedy,Sport',
       'Action,Horror,Mystery', 'Documentary,History', 'Mystery',
       'Horror,Mystery,Sci-Fi', 'Action,Mystery,Sci-Fi',
       'Drama,History,Sport', 'Animation,Drama,Sci-Fi',
       'Drama,Musical,Romance', 'Drama,Fantasy,Mystery',
       'Horror,Romance,Thriller', 'Action,Adventure,Mystery',
```

```
       'Drama,Mystery,Romance', 'Mystery,Romance,Thriller',
       'Drama,Western', 'Action,Horror,Sci-Fi', 'Comedy,Music,Romance',
       'Adventure,Crime,Drama', 'Adventure,Comedy,Crime',
       'Action,Comedy,Thriller', 'Biography,Crime,Documentary',
       'Drama,Horror,Sci-Fi', 'Crime', 'Fantasy,Thriller', 'Music',
       'Comedy,Drama,History', 'Adventure,Family,Sci-Fi',
       'Action,Adventure', 'Horror,Sci-Fi,Thriller', 'Drama,Horror',
       'Biography,Drama,War', 'Adventure,Family',
       'Animation,Comedy,Drama', 'Comedy,Western', 'Comedy,Documentary',
       'Comedy,Mystery,Sci-Fi', 'Action,Drama,Sport', 'Fantasy',
       'Adventure,Comedy,Romance', 'Comedy,Music',
       'Documentary,Drama,Music', 'Animation,Biography,Crime',
       'Comedy,Thriller', 'Biography,Documentary,Mystery',
       'Animation,Drama,Fantasy', 'Action,Biography,Comedy',
       'Comedy,Drama,War', 'Drama,History,Thriller',
       'Biography,Documentary,Music', 'Action,Sport',
       'Crime,Horror,Thriller', 'Biography,Documentary,Thriller',
       'Fantasy,Horror', 'Action,Comedy,War', 'Comedy,Drama,Horror',
       'Adventure,Comedy,Horror', 'Documentary,Family',
       'Comedy,Musical,Romance', 'Comedy,Sport'], dtype=object)
```

In [87]:
```
merged_df5['title'].unique()
```

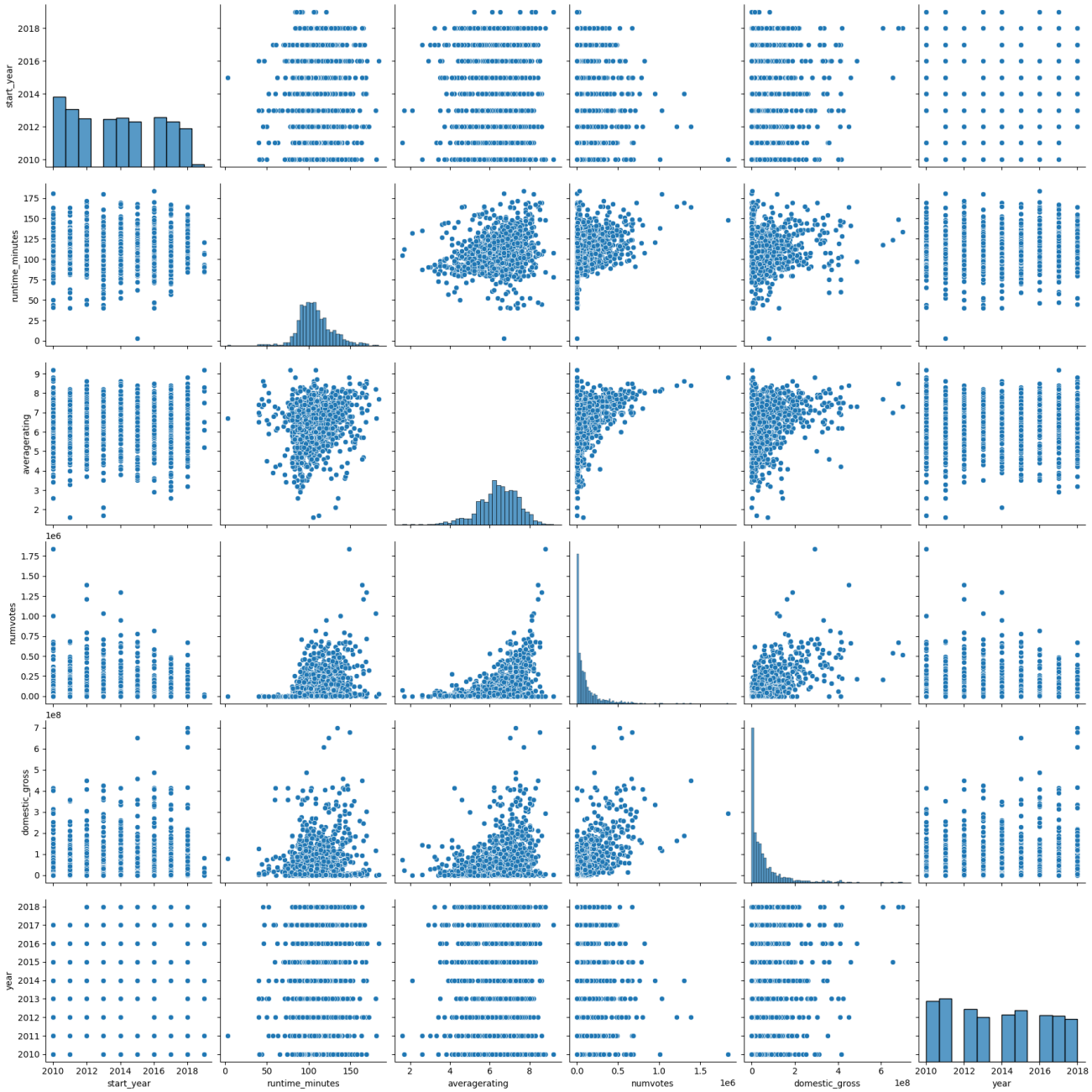
Out[87]:
```
array(['On the Road', 'The Secret Life of Walter Mitty',
       'A Walk Among the Tombstones', ..., 'The Past', "Nobody's Fool",
       'Burn the Stage: The Movie'], dtype=object)
```

In [38]:
```
merged_df5['studio'].unique()
```

Out[38]:
```
array(['IFC', 'Fox', 'Uni.', 'FD', 'BV', 'P/DW', 'LGF', 'Wein.', 'WB',
       'Over.', 'Sony', 'Par.', 'Magn.', 'RAtt.', 'W/Dim.', 'A24', 'CBS',
       'FoxS', 'Rela.', 'WB (NL)', 'Sum.', 'Focus', 'NGE', 'SPC', 'SGem',
       'EOne', 'CE', 'Mira.', 'OMNI/FSR', 'LG/S', 'App.', 'NM', 'Amazon',
       'Free', 'Osci.', 'STX', 'RTWC', 'FInd.', 'BG', 'Gold.', 'ATO',
       'Relbig.', 'BST', 'Anch.', 'MPFT', 'MGM', 'GK', 'Arth.', 'TriS',
       'UTV', 'First', 'Trib.', 'FM', 'Drft.', 'Eros', 'MNE', 'CGld',
       'FOAK', 'Hann.', 'Strand', 'PDA', 'ORF', 'Cohen', 'Viv.', 'Zeit.',
       'Mont.', 'Jan.', 'MBox', 'KL', '3D', 'WOW', 'Rog.', 'ParV', 'FIP',
       'Yash', 'TFA', 'Lorb.', 'NYer', 'Imag.', 'P4', 'Da.', 'AGF',
       'FRun', 'NAV', 'Rocket', 'WHE', 'LD', 'HTR', 'FCW', 'IM', 'Shout!',
       'Studio 8', 'PH', 'AF', 'KE', 'WGUSA', 'Ampl.', 'Aviron',
       'Global Road', 'Abr.', 'EC', 'Imax', 'Greenwich', 'BH Tilt', 'ITL',
       'Good Deed', 'DR', 'Scre.', 'U/P', 'Annapurna', 'VE', 'FR', 'PFR',
       'Orch.', 'Affirm', 'GrtIndia', 'Zee', 'FUN', 'ENTMP', 'MOM',
       'Neon', 'VPD', 'Blue Fox', 'Vita.', 'CL', 'BBC', 'Grindstone',
       'Trafalgar'], dtype=object)
```

In [92]:
```
#plots scatter plots of the numerical variables
sns.pairplot(merged_df5, height = 3)
plt.show()
```

```
C:\Users\Administrator\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed t
o tight
  self._figure.tight_layout(*args, **kwargs)
```



In [107]:
```
# Group by 'genre' and calculate the mean rating for each genre
genres_ratings_mean = merged_df5.groupby('genres')['averagerating'].mean()
genres_ratings_mean

# Sort the result by ratings in descending order
sorted_genres_ratings = genres_ratings_mean.sort_values(ascending=False)

# Display the sorted result
print(sorted_genres_ratings)
```

```
genres
Adventure                9.2
Action,Sport             8.4
Drama,Western            8.4
Adventure,Drama,Sci-Fi   8.3
Crime,Documentary        8.3
                        ...
Sci-Fi                   4.2
Comedy,Drama,Horror      4.2
Fantasy,Horror           3.8
Drama,Mystery,Western    3.4
Comedy,Thriller          2.1
Name: averagerating, Length: 239, dtype: float64
```

In [108]:
```
# Group by 'genre' and calculate the mean rating for each genre
genres_domestic_gross_mean = merged_df5.groupby('genres')['domestic_gross'].mean()
genres_domestic_gross_mean

# Sort the result by ratings in descending order
sorted_genres_domestic_gross = genres_domestic_gross_mean.sort_values(ascending=False)

# Display the sorted result
print(sorted_genres_domestic_gross)
```

```
genres
Documentary,Drama,Sport    412600000.0
Sci-Fi                     412600000.0
Adventure,Drama,Sport      400700000.0
Family                     356500000.0
Comedy,Mystery             254500000.0
                              ...
Action,Horror,Mystery        5500.0
Comedy,Crime,Mystery         5000.0
Comedy,Crime,History         4800.0
Fantasy,Thriller             1400.0
Comedy,Thriller               800.0
Name: domestic_gross, Length: 239, dtype: float64
```

In [111]:
```
# Group by 'genre' and calculate the mean rating for each genre
genres_foreign_gross_mean = merged_df5.groupby('genres')['foreign_gross'].mean()
genres_foreign_gross_mean

# Sort the result by ratings in descending order
sorted_genres_foreign_gross = genres_foreign_gross_mean.sort_values(ascending=False)

# Display the sorted result
print(sorted_genres_foreign_gross)
```

```
genres
Adventure,Drama,Sport            8.757000e+08
Adventure,Fantasy                5.111333e+08
Family                           5.011000e+08
Adventure,Drama,Sci-Fi           4.455500e+08
Sci-Fi                           4.093000e+08
                                         ...
Documentary,Drama,Mystery        2.420000e+05
Biography,Documentary,Thriller   2.020000e+05
Biography,Documentary,Drama      1.050000e+05
Comedy,Documentary               7.400000e+04
Biography,Comedy,Crime           4.070000e+04
Name: foreign_gross, Length: 239, dtype: float64
```

In [102]:
```
# Group by 'genre' and calculate the mean rating for each genre
genres_ratings_mean = merged_df5.groupby('genres')['foreign_gross'].mean()
genres_ratings_mean
```

Out[102]:
```
genres
Action                          3.710000e+07
Action,Adventure                3.470000e+05
Action,Adventure,Animation      2.963357e+08
Action,Adventure,Biography      1.489000e+08
Action,Adventure,Comedy         2.688100e+08
                                    ...
Romance,Sci-Fi,Thriller         3.270255e+07
Romance,Thriller                6.560500e+06
Sci-Fi                          4.093000e+08
Thriller                        4.888530e+07
Thriller,Western                3.000000e+05
Name: foreign_gross, Length: 239, dtype: float64
```

Groupby observations

Genre vs ratings

-The genre with the highest ratings was Adventure with a rating of 9.2 while lowest was Comedy,thriller with a rating of 2.1

Genre vs domestic gross

-The genre with the highest domestic gross was had a gross of 41260000 and was a tie bewteen Docu drama and Sci Fi. The least earing in domestic gross was Comedy Thriller which amounted to 800

Genre vs foreign gross

-The genre that earned the highest foreign gross was Action at 875.7 million while the lowest was Thriller-Western with 300,000

In [46]:
```
# First we are going to covert our categorical data  to numerical data so that we can be able to create a correlation matr
# Convert catergorical data to Numerical
from sklearn.preprocessing import LabelEncoder
en = LabelEncoder()
merged_df5['genres'] = en.fit_transform(merged_df5['genres'])
merged_df5['studio'] = en.fit_transform(merged_df5['studio'])
merged_df5['title'] = en.fit_transform(merged_df5['title'])
merged_df5
```

Out[46]:

	title	start_year	runtime_minutes	genres	averagerating	numvotes	studio	domestic_gross	foreign_gross	year
31	718	2012.0	124.0	76	6.1	37886.0	55	744000.0	8000000	2012.0
32	718	2014.0	89.0	176	6.0	6.0	55	744000.0	8000000	2012.0
33	718	2016.0	121.0	176	5.7	127.0	55	744000.0	8000000	2012.0
38	1179	2013.0	114.0	63	7.3	275300.0	43	58200000.0	129900000	2013.0
42	44	2014.0	114.0	30	6.5	105116.0	113	26300000.0	26900000	2014.0
...	...	...	...	...	...	...	...	...	...	...
61098	937	2018.0	116.0	33	5.0	4753.0	103	20500000.0	236000	2018.0
61418	416	2018.0	127.0	192	7.3	151571.0	1	44100000.0	35300000	2018.0
62133	1152	2018.0	120.0	191	7.4	54.0	99	1300000.0	9300000	2013.0
62388	707	2018.0	110.0	131	4.6	3618.0	90	31700000.0	1800000	2018.0
64832	185	2018.0	84.0	173	8.8	2067.0	108	4200000.0	16100000	2018.0

1518 rows × 10 columns

In [51]:
```
#To check for data types, we use:
merged_df5.dtypes
```

Out[51]:
```
title              int32
start_year       float64
runtime_minutes  float64
genres             int32
averagerating    float64
numvotes         float64
studio             int32
domestic_gross   float64
foreign_gross     object
year             float64
dtype: object
```

In [52]:
```
merged_df5['foreign_gross'] = merged_df5['foreign_gross'].str.replace(',', '').astype(float)
merged_df5
```

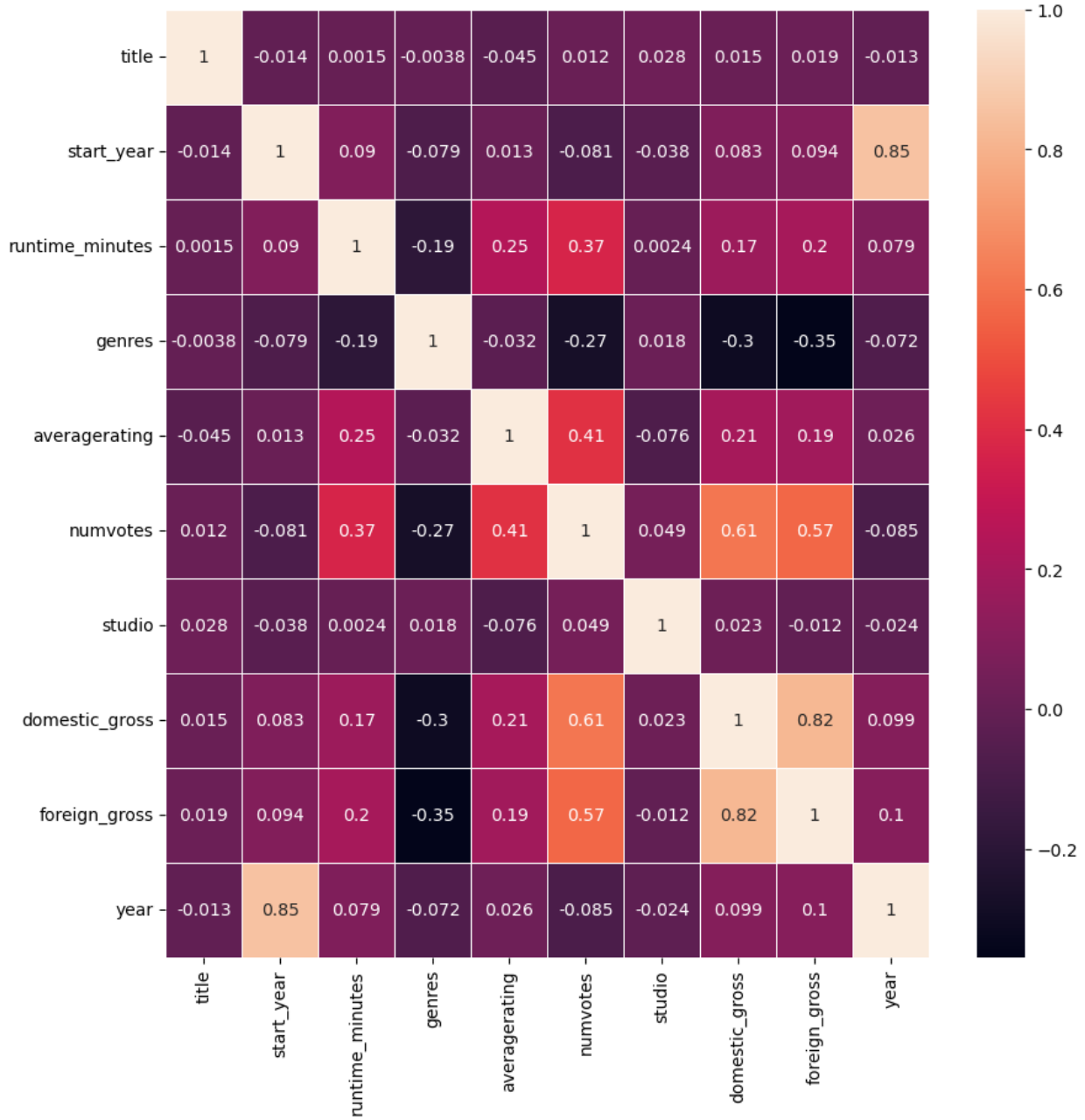
Out[52]:

	title	start_year	runtime_minutes	genres	averagerating	numvotes	studio	domestic_gross	foreign_gross	year
31	718	2012.0	124.0	76	6.1	37886.0	55	744000.0	8000000.0	2012.0
32	718	2014.0	89.0	176	6.0	6.0	55	744000.0	8000000.0	2012.0
33	718	2016.0	121.0	176	5.7	127.0	55	744000.0	8000000.0	2012.0
38	1179	2013.0	114.0	63	7.3	275300.0	43	58200000.0	129900000.0	2013.0
42	44	2014.0	114.0	30	6.5	105116.0	113	26300000.0	26900000.0	2014.0
...	...	...	...	...	...	...	...	...	...	...
61098	937	2018.0	116.0	33	5.0	4753.0	103	20500000.0	236000.0	2018.0
61418	416	2018.0	127.0	192	7.3	151571.0	1	44100000.0	35300000.0	2018.0
62133	1152	2018.0	120.0	191	7.4	54.0	99	1300000.0	9300000.0	2013.0
62388	707	2018.0	110.0	131	4.6	3618.0	90	31700000.0	1800000.0	2018.0
64832	185	2018.0	84.0	173	8.8	2067.0	108	4200000.0	16100000.0	2018.0

1518 rows × 10 columns

In [53]:
```
# Create a correlation matrix
corrMatrix = merged_df5.corr()
corrMatrix

# Vizualise the correlation matrix using a heat map

fig, ax = plt.subplots(figsize=(10,10))
sns.heatmap(corrMatrix, annot=True,  linewidths=.5, ax=ax)
```

Out[53]: <Axes: >



Observations

Genre has the highest correlation with studio. This is because it has a score that is closest to 1.
Genre has the least correaltion with title, start year average rating and the year,since their score is the closest to -1.

Conclusion

Microsft should consider making Adventure films as it ranks the highest with a 9.2 rating.
When considering films with the highest domestic gross, Documentary-Drama-Sport ranks the highest 412.6m
When considering films with the highest foreign gross, Action films have the highest gross at 875.7M while the lowest is