Final Project Submission

Please fill out:

- Student name: Marife Ramoran
- Student pace:
- Scheduled project review date/time: 15th October 2023
- Instructor name: Hardik Idnani
- Blog post URL:

Overview

This project involves the analysis of data to fulfill Microsoft's objective of producing a highly successful movie. By conducting a descriptive examination of movies from the past century, we aim to identify recurring patterns that contribute to the formula for a blockbuster film. Microsoft can then leverage this analysis to track trends and understand the key factors behind the success of blockbuster movies.

Business Problem

Microsoft has taken notice of major corporations venturing into the realm of producing unique video content, and they are eager to join the fray. As part of this endeavor, Microsoft has made the strategic decision to establish a brand-new film studio. However, they currently lack expertise in the field of filmmaking. Your role involves conducting research to identify the most successful film genres currently dominating the box office. Subsequently, you will be tasked with transforming this research into practical recommendations that can guide the head of Microsoft's newly established movie studio in making informed decisions about the types of films to produce.

In [210]:
```
# Import library
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

%matplotlib inline
```

In [211]:
```
# You can execute your code here to investigate the data.
title_basic_df = pd.read_csv('./zippedData/imdb.title.basics.csv.gz
movie_budget_df = pd.read_csv('./zippedData/tn.movie_budgets.csv.gz
title_rating_df = pd.read_csv('./zippedData/imdb.title.ratings.csv.
bom_movie_gross_df = pd.read_csv('./zippedData/bom.movie_gross.csv.
```

Data Preparation

Explain and provide details of the procedure taken to get the data ready for analysis.

Questions:

1. How did you resolve missing values or outliers?
2. Did you eliminate or introduce any variables?
3. Why do these selections align with the data and the business issue at hand?

In [212]:
```
# Run this code to clean the data
rating_basic_df = pd.merge(title_basic_df, title_rating_df,on = ['t
```

In [213]:
```
# Run this code to identify how many datas are missing on rating_ba
rating_basic_df.isna().sum()
```

Out[213]:
```
tconst               0
primary_title        0
original_title       0
start_year           0
runtime_minutes    7620
genres              804
averagerating         0
numvotes              0
dtype: int64
```

In [214]:
```
# I will create a new DataFrame variable to display the outcome.
rating_basic_filled_df = rating_basic_df
```

In [215]:
```
# I need to convert the data type to a float so that I can perform
rating_basic_filled_df['runtime_minutes'] = pd.to_numeric(rating_ba
```

In [216]:
```
# Now that the data is in a format I can work with, I will replace
rating_basic_filled_df['runtime_minutes'] = rating_basic_df['runtim
```

In [217]:
```
# I am renaming the 'primary_title' column in the `rating_basic_fil
rating_basic_filled_df.rename(columns = {'primary_title':'title'},
```

Data Merging

In [218]:
```
# We can begin by merging the bom_movie_gross_df and movie_budget_d
movie_budget_df.rename(columns = {'movie':'title'}, inplace = True)
movie_gross_budget_df = pd.merge(bom_movie_gross_df, movie_budget_d
merged_df = pd.merge(movie_gross_budget_df, rating_basic_filled_df,
```

In [219]:
```
# I want to remove any duplicates that share the same title and rel
merged_df = merged_df.drop_duplicates(subset= ['title', 'release_da
```

In [220]:
```
# This error appeared as an issue that requires resolution before w
merged_df[merged_df['foreign_gross'] == '1,019.4']
merged_df[merged_df['foreign_gross'] == '1,163.0']
merged_df[merged_df['foreign_gross'] == '1,010.0']
merged_df[merged_df['foreign_gross'] == '1,369.5']
```

Out[220]:

	title	studio	domestic_gross_x	foreign_gross	year	id	release_date	productio
1302	Avengers: Infinity War	BV	678800000.0	1,369.5	2018	7	Apr 27, 2018	$30

In [221]:
```
# Now we need to address and resolve the error
merged_df['foreign_gross'][824] = '1017600000'
merged_df['foreign_gross'][825] = '1163000000'
merged_df['foreign_gross'][1170] = '1010000000'
merged_df['foreign_gross'][1302] = '1369500000'
```

```
/var/folders/zr/m5b0ldvs7695p_h8w70ndwc40000gn/T/ipykernel_1736/30
24590889.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pa
ndas-docs/stable/user_guide/indexing.html#returning-a-view-versus-
a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/in
dexing.html#returning-a-view-versus-a-copy)
  merged_df['foreign_gross'][825] = '1163000000'
/var/folders/zr/m5b0ldvs7695p_h8w70ndwc40000gn/T/ipykernel_1736/30
24590889.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pa
ndas-docs/stable/user_guide/indexing.html#returning-a-view-versus-
a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/in
dexing.html#returning-a-view-versus-a-copy)
  merged_df['foreign_gross'][1170] = '1010000000'
/var/folders/zr/m5b0ldvs7695p_h8w70ndwc40000gn/T/ipykernel_1736/30
24590889.py:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pa
ndas-docs/stable/user_guide/indexing.html#returning-a-view-versus-
a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/in
dexing.html#returning-a-view-versus-a-copy)
  merged_df['foreign_gross'][1302] = '1369500000'
```

In [222]:
```
# Now, it's possible to modify the data type of the 'foreign_gross'
merged_df['foreign_gross'] = pd.to_numeric(merged_df['foreign_gross
```

In [223]:
```
# As certain foreign gross data was absent, I filled it using the m
merged_df['foreign_gross'] = merged_df['foreign_gross'].fillna(valu
```

Data Cleaning

In [224]:
```
# I would like to eliminate the comma with the genre
def split_comma(x):
    return x.split(",")
```

In [225]:
```
# I need to determine whether there is any missing data in the genr
merged_df['genres'].isna().sum()
```

Out[225]: 1

In [226]:
```
# I have identified 1 missing data where we can choose between drop
# In this scenario, I have filled it with "missing" since it should
merged_df['genres'] = merged_df['genres'].fillna (value = 'missing'
```

In [227]:
```
# In here, I am using the map method to remove the comma
merged_df['genres'] = merged_df ['genres'].map(split_comma)
```

In [228]:
```
# I am defining another function to remove the dollar sign and comm
def remove_dollar_comma(x):
    x = x.replace("," , "")
    return x.replace("$", "")
```

In [229]:
```
# Using the new function that was created, we can now proceed with
merged_df['production_budget'] = merged_df['production_budget'].map
```

In [230]:
```
# We can now change the type of column production_budget to int
merged_df['production_budget'] = pd.to_numeric(merged_df['productio
```

Feature Engineering

In [231]:
```
# I intend to remove the existing 'worldwide_gross' column
# then create a new one to be accurate especially in cases where th
# This will produce consistent data
merged_df.drop("worldwide_gross", axis = 1)
```

Out[231]:

	title	studio	domestic_gross_x	foreign_gross	year	id	release_date	prod
0	Toy Story 3	BV	415000000.0	652000000.0	2010	47	Jun 18, 2010	
1	Inception	WB	292600000.0	535700000.0	2010	38	Jul 16, 2010	
2	Shrek Forever After	P/DW	238700000.0	513900000.0	2010	27	May 21, 2010	
3	The Twilight Saga: Eclipse	Sum.	300500000.0	398000000.0	2010	53	Jun 30, 2010	
4	Iron Man 2	Par.	312400000.0	311500000.0	2010	15	May 7, 2010	
...	...	...	...	...	...	...	...	
1407	Destroyer	Annapurna	1500000.0	4000000.0	2018	5	Dec 25, 2018	
1408	Gotti	VE	4300000.0	43000000.0	2018	64	Jun 15, 2018	
1409	Bilal: A New Breed of Hero	VE	491000.0	1700000.0	2018	100	Feb 2, 2018	
1410	Mandy	RLJ	1200000.0	43000000.0	2018	71	Sep 14, 2018	
1412	Lean on Pete	A24	1200000.0	43000000.0	2018	13	Apr 6, 2018	

1166 rows × 16 columns

In [232]:
```
# I want to calculate the global earnings and place them in a new c
# This represents the total of both 'domestic_gross' and 'foreign_g
merged_df['worldwide_gross'] = merged_df['domestic_gross_x'] + merg
```

In [233]:
```
# I want to create a new column that is called 'blockbuster' that w
# A straightforward criterion for this would be if the overall budg
# contains all expenses from production to marketing, qualifies the
merged_df['blockbuster'] = (merged_df['worldwide_gross']) >= (2 * m
```

In [234]:
```
# I want to refine the selection to include only the movies classif
# then arrange them based on their global earnings
blockbusters_df = merged_df[merged_df['blockbuster'] == True]
blockbusters_sort_df = blockbusters_df.sort_values(['worldwide_gros
```

In [235]:
```
# Now I want to identify who made the top 10 studios that have prod
top_studios = blockbusters_sort_df['studio'].value_counts().head(10
```

In [236]:
```
# I would like to determine the most prevalent combinations of genr
top_genres = blockbusters_sort_df['genres'].value_counts().head(10)
```

In [237]:
```
# I want to split the genres using explode method
top_genres = blockbusters_sort_df.explode('genres')
```

In [238]:
```
# I want to identify the films that generated the highest earnings.
top_10_num = blockbusters_sort_df.head(10).drop(['year','release_da
```

In [239]:
```
# Here I want to sort the production budget of the top 10 box offic
top_10_num_sorted = top_10_num.sort_values(by='production_budget',
```

Data Modelling

Explain and provide a rationale for the procedure employed in analyzing or constructing data models.

Questions:

1. What methods were employed for data analysis or modeling?
2. How was the initial approach refined through iterations to enhance its effectiveness?
3. How can these choices be justified in light of the data and the business problem at hand?

In [240]:
```
# Run the code to model the data
colors = ['navy', 'indigo', 'crimson', 'peru', 'silver', 'teal', 'o
```

In [241]:
```
top_10_num
```

Out[241]:

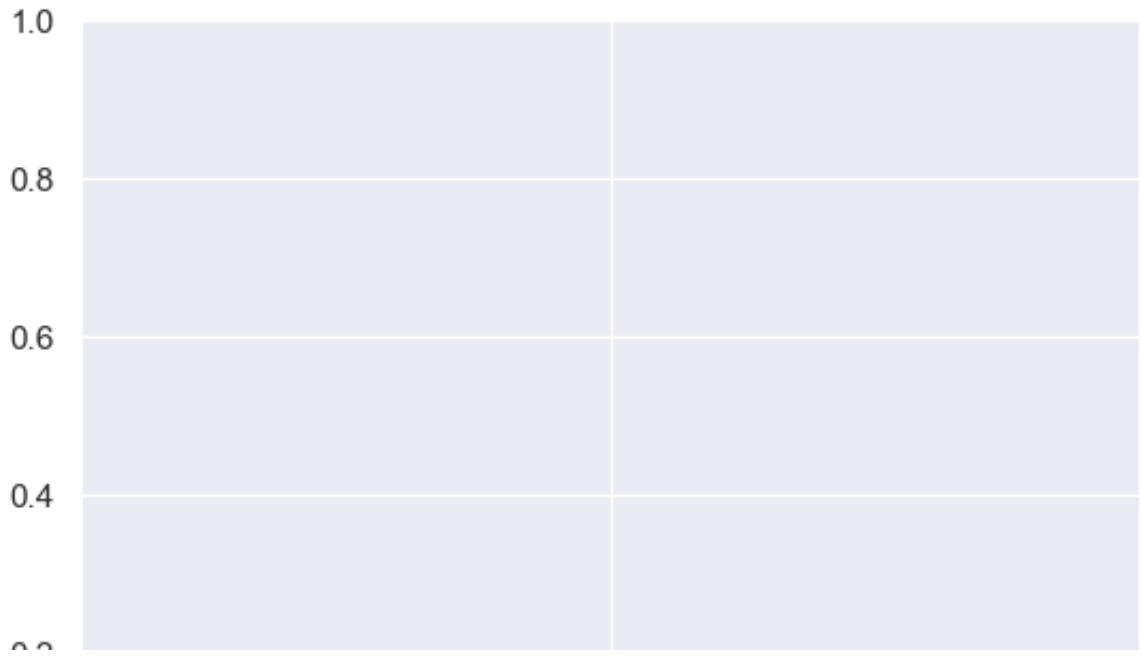
	title	studio	domestic_gross_x	foreign_gross	production_budget	worldwide_gr
1302	Avengers: Infinity War	BV	678800000.0	1.369500e+09	300000000	2.048300e
824	Jurassic World	Uni.	652300000.0	1.017600e+09	215000000	1.669900e
825	Furious 7	Uni.	353000000.0	1.163000e+09	190000000	1.516000e
826	Avengers: Age of Ultron	BV	459000000.0	9.464000e+08	330600000	1.405400e
1303	Black Panther	BV	700100000.0	6.469000e+08	200000000	1.347000e
1304	Jurassic World: Fallen Kingdom	Uni.	417700000.0	8.918000e+08	170000000	1.309500e
520	Frozen	BV	400700000.0	8.757000e+08	150000000	1.276400e
1305	Incredibles 2	BV	608600000.0	6.342000e+08	200000000	1.242800e
1170	The Fate of the Furious	Uni.	226000000.0	1.010000e+09	250000000	1.236000e
523	Iron Man 3	BV	409000000.0	8.058000e+08	200000000	1.214800e

In [242]:
```
# I selected the top 10 genres to establish a foundational understa
# with the potential to create a blockbuster hit.
fig, ax = plt.subplots()

genre_name = top_genres.index

ax.barh(genre_name,list(top_genres), color = colors)
ax.set_xlabel('Amount of Movies')
ax.set_ylabel('Combination of Genres')
ax.set_title('The Genres of Box Office Hits 2010-2018')

plt.savefig(".\image\genres_bar.png", dpi = 150, bbox_inches = 'tig

plt.show()
```

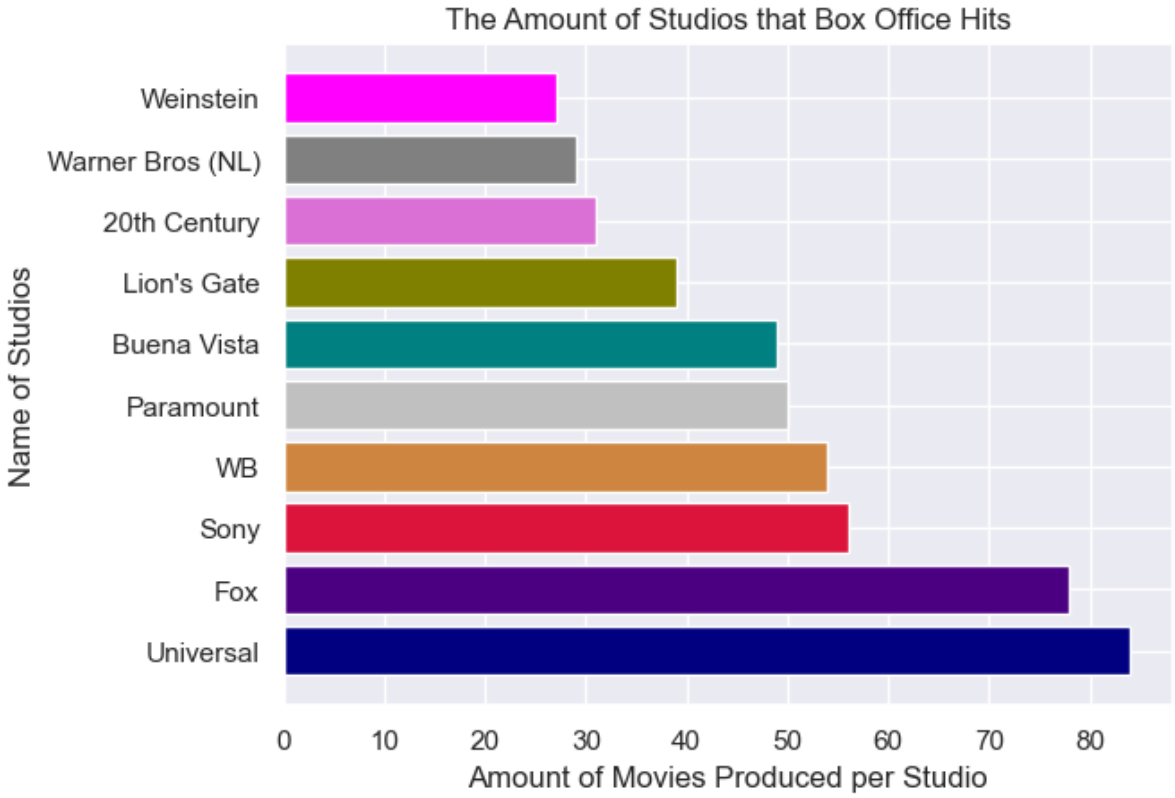
```
---------------------------------------------------------------------
---------
ValueError                                Traceback (most recent c
all last)
Cell In[242], line 7
```

```
      3 fig, ax = plt.subplots()
      5 genre_name = top_genres.index
----> 7 ax.barh(genre_name,list(top_genres), color = colors)
      8 ax.set_xlabel('Amount of Movies')
      9 ax.set_ylabel('Combination of Genres')

File ~/anaconda3/lib/python3.11/site-packages/matplotlib/axes/_axes.py:2649, in Axes.barh(self, y, width, height, left, align, data, **kwargs)
   2539 r"""
   2540 Make a horizontal bar plot.
   2541
   (...)
   2646 :doc:`/gallery/lines_bars_and_markers/horizontal_barchart_distribution`.
   2647 """
   2648 kwargs.setdefault('orientation', 'horizontal')
-> 2649 patches = self.bar(x=left, height=height, width=width, bottom=y,
   2650                    align=align, data=data, **kwargs)
   2651 return patches

File ~/anaconda3/lib/python3.11/site-packages/matplotlib/__init__.py:1442, in _preprocess_data.<locals>.inner(ax, data, *args, **kwargs)
   1439 @functools.wraps(func)
   1440 def inner(ax, *args, data=None, **kwargs):
   1441     if data is None:
-> 1442         return func(ax, *map(sanitize_sequence, args), **kwargs)
   1444     bound = new_sig.bind(ax, *args, **kwargs)
   1445     auto_label = (bound.arguments.get(label_namer)
   1446                   or bound.kwargs.get(label_namer))

File ~/anaconda3/lib/python3.11/site-packages/matplotlib/axes/_axes.py:2417, in Axes.bar(self, x, height, width, bottom, align, **kwargs)
   2414     if yerr is not None:
   2415         yerr = self._convert_dx(yerr, y0, y, self.convert_yunits)
-> 2417 x, height, width, y, linewidth, hatch = np.broadcast_arrays(
   2418     # Make args iterable too.
   2419     np.atleast_1d(x), height, width, y, linewidth, hatch)
   2421 # Now that units have been converted, set the tick locations.
   2422 if orientation == 'vertical':

File <__array_function__ internals>:200, in broadcast_arrays(*args, **kwargs)

File ~/anaconda3/lib/python3.11/site-packages/numpy/lib/stride_tricks.py:540, in broadcast_arrays(subok, *args)
```

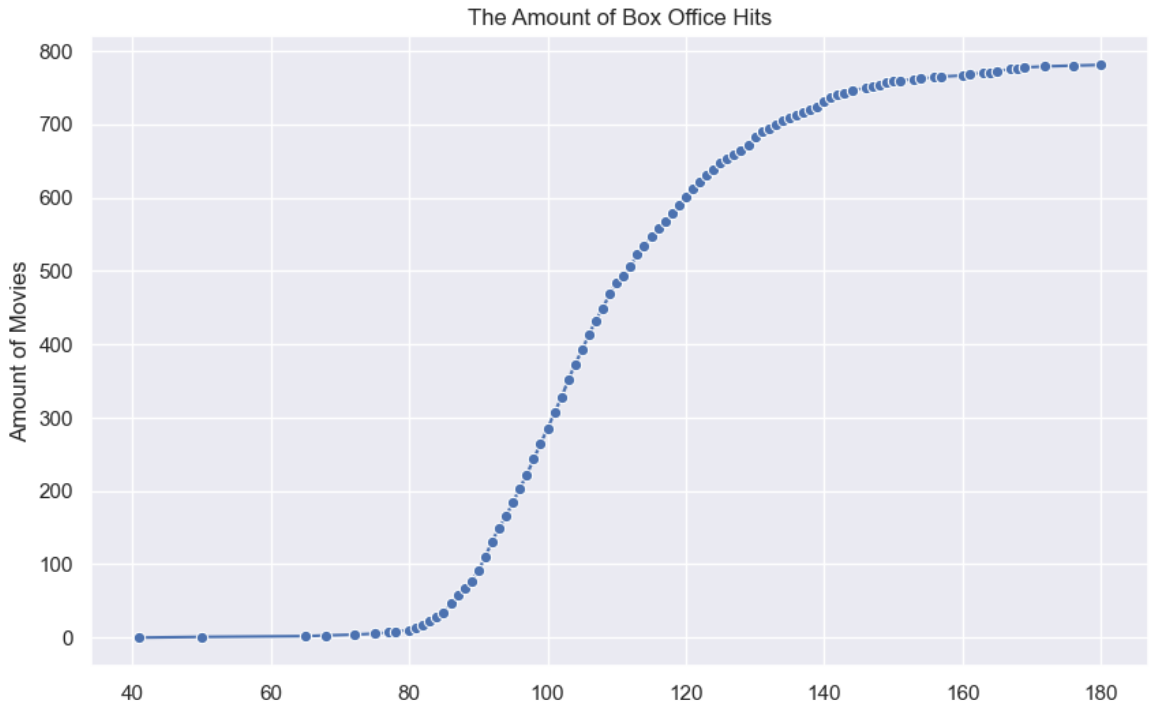
```
    533 # nditer is not used here to avoid the limit of 32 arrays.
    534 # Otherwise, something like the following one-liner would
suffice:
    535 # return np.nditer(args, flags=['multi_index', 'zerosize_o
k'],
    536 #                  order='C').itviews
    538 args = [np.array(_m, copy=False, subok=subok) for _m in
args]
--> 540 shape = _broadcast_shape(*args)
    542 if all(array.shape == shape for array in args):
    543     # Common case where nothing needs to be broadcasted.
    544     return args

File ~/anaconda3/lib/python3.11/site-packages/numpy/lib/stride_tri
cks.py:422, in _broadcast_shape(*args)
    417 """Returns the shape of the arrays that would result from
broadcasting the
    418 supplied arrays against each other.
    419 """
    420 # use the old-iterator because np.nditer does not handle s
ize 0 arrays
    421 # consistently
--> 422 b = np.broadcast(*args[:32])
    423 # unfortunately, it cannot handle 32 or more arguments dir
ectly
    424 for pos in range(32, len(args), 31):
    425     # ironically, np.broadcast does not properly handle np
.broadcast
    426     # objects (it treats them as scalars)
    427     # use broadcasting to avoid allocating the full array

ValueError: shape mismatch: objects cannot be broadcast to a singl
e shape.  Mismatch is between arg 2 with shape (18,) and arg 3 wit
h shape (2009,).
```



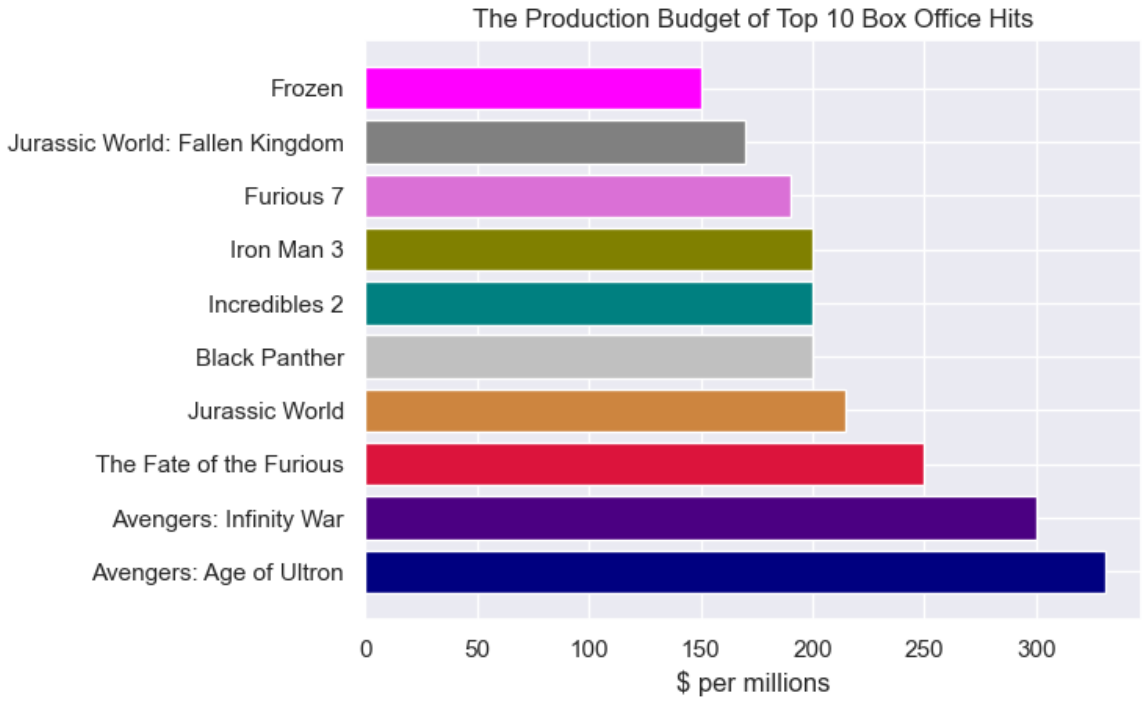
In [243]:
```
# I chose the top 10 to determine if there are existing affiliation
# which could lead to potential collaborations in creating new, ori
fig, ax = plt.subplots()

studio_name = ['Universal','Fox','Sony','WB','Paramount','Buena Vis

ax.barh(studio_name,list(top_studios), color = colors)
ax.set_xlabel('Amount of Movies Produced per Studio')
ax.set_ylabel('Name of Studios')
ax.set_title('The Amount of Studios that Box Office Hits')

plt.savefig(".\image\studio_bar.png", dpi = 150, bbox_inches = 'tig

plt.show()
```



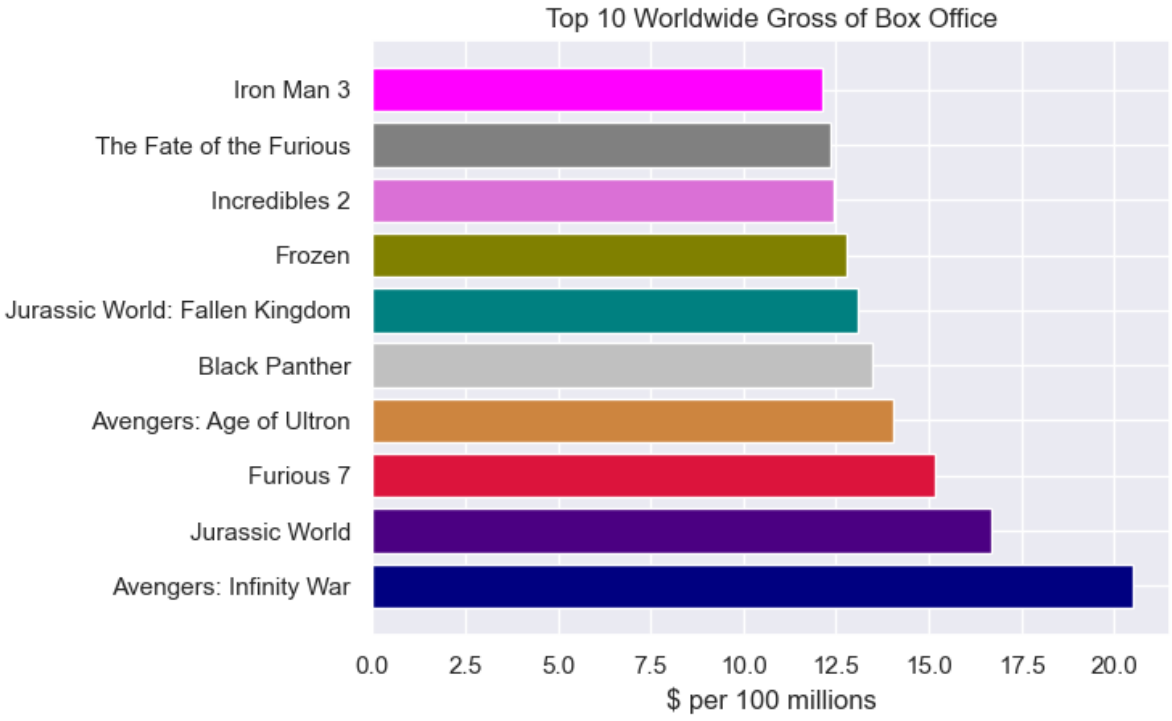
In [244]:
```
# Here is a breakdown of the number of movies and their typical run
# which could influence viewers' decisions to watch these films
sns.set(style="darkgrid")

plt.figure(figsize=(10, 6))  # Adjust the figure size if needed
sns.lineplot(x=sorted(blockbusters_df['runtime_minutes']), y=range(

plt.xlabel('Number of Minutes')
plt.ylabel('Amount of Movies')
plt.title('The Amount of Box Office Hits')


plt.savefig("numberminutes_line_seaborn.png", dpi=500, bbox_inches=

plt.show()
```



In [245]:
```
# Here is the chart displaying the production budget of the top 10 

fig, ax = plt.subplots()

ax.barh(top_10_num_sorted['title'] ,list(top_10_num_sorted['product
ax.set_xlabel('$ per millions')
ax.set_title('The Production Budget of Top 10 Box Office Hits')

plt.savefig(".\image\production_budget_bar.png", dpi = 150, bbox_in

plt.show()
```



In [246]:
```
# Here is a chart that illustrates which titles generated the highe
fig, ax = plt.subplots()

ax.barh(top_10_num['title'] ,list(top_10_num['worldwide_gross']/100
ax.set_xlabel('$ per 100 millions')
ax.set_title('Top 10 Worldwide Gross of Box Office')

plt.savefig(".\image\worldwide_gross_bar.png", dpi = 150, bbox_inch

plt.show()
```



Evaluation

Assess the effectiveness of your work in addressing the specified business problem.

Questions

1. How do you analyze the outcomes?
2. How effectively does your model align with the data?
3. To what extent does it outperform the baseline model?
4. How assured are you of the model's ability to generalize beyond the available data?
5. How convinced are you that implementing this model would be advantageous for the business?

Based on the provided data, we have valuable information to work with. We have insights into successful genres, optimal runtime, and an estimated production budget, which can help us approach the goal of achieving a position among the top 10 highest-grossing movies worldwide. I am confident that this data can serve as a helpful guide for creating a potential blockbuster hit.

Conclusion

Share your final remarks on the project, encompassing findings, constraints, and potential future actions.

---

1. What actions would you suggest the company take based on the outcomes of this project?
2. What are some factors that could limit the comprehensiveness of your analysis in addressing the business challenge?
3. What potential enhancements or strategies could be considered for future iterations of this project?

Based on this analysis, we propose three recommendations for Microsoft's movie studio's inaugural blockbuster film:

- Develop a superhero-themed movie that combines elements of adventure, animation, and comedy (e.g., like "Frozen") or action, adventure, and sci-fi (e.g., akin to "Iron Man 3"). These genres consistently perform well at the box office, with five of the top ten worldwide gross earners falling into the superhero category.
- Consider forming strategic partnerships with established studios like Universal Pictures or 20th Century Fox, renowned for their extensive experience in film production and their track record of producing successful blockbusters.
- Aim for a movie duration within the 90-110 minute range and allocate a production budget in the range of 150 million to 300 million, which represents a reasonable maximum benchmark for investment.

Business Recommendations

Future considerations involve the acquisition of additional data to assess the impact of influential directors, actors, and other crew members on the potential for blockbuster success.

Furthermore, we should explore predictive models for production delays that may impact the budget.

In addition, it's essential to maintain a proactive approach towards industry trends, ensuring timely adoption to capitalize on emerging concepts before their novelty diminishes. Given the time required for movie production, sustained relevance is critical for maximizing profits.