Movie Data Analysis for Microsoft Studios

Business Understanding

Three recomendations for Microsoft for movie production studio debut Microsoft well known tech company is expanging into the film industry They would like to know what makes a high grossing movie

Scharmaine Chappell

Data Anyalyst

Overview

What we'll do is coordinate internet sourced data from a formadable movie databases site, in order to propose what direction Microsoft studios should go for their studio premier. Key aspects valued are genre, location and gross values in comparison to initial production costs.

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import numpy as np
import string
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

Loading Data pulled from The Numbers

Data Description

Our data is gathered from the The Numbers Website, thenumbers.com. According to their site, they have been around and collecting movie data for analysis since 1997, specifically for the movie industry by Bruce Nash. I have chosen production budgets, domestic and world_wide gross numbers to create our database file to help us see why these factors should be considered. The specific details I curated into the database will be discussed as we go into detail of what's measured and how we use it to address the needs of Microsoft Studios.

In [2]:
```
tn_movie_budgets = pd.read_csv("zippedData/tn.movie_budgets.csv.gz")
tn_movie_budgets.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5782 entries, 0 to 5781
Data columns (total 6 columns):
 #   Column             Non-Null Count  Dtype 
---  ------             --------------  ----- 
 0   id                 5782 non-null   int64 
 1   release_date       5782 non-null   object
 2   movie              5782 non-null   object
 3   production_budget  5782 non-null   object
 4   domestic_gross     5782 non-null   object
 5   worldwide_gross    5782 non-null   object
dtypes: int64(1), object(5)
memory usage: 271.2+ KB
```

In [3]:
```
1 tn_movie_budgets.head()
```

Out[3]:

	id	release_date	movie	production_budget	domestic_gross	worldwide_gross
0	1	Dec 18, 2009	Avatar	$425,000,000	$760,507,625	$2,776,345,279
1	2	May 20, 2011	Pirates of the Caribbean: On Stranger Tides	$410,600,000	$241,063,875	$1,045,663,875
2	3	Jun 7, 2019	Dark Phoenix	$350,000,000	$42,762,350	$149,762,350
3	4	May 1, 2015	Avengers: Age of Ultron	$330,600,000	$459,005,868	$1,403,013,963
4	5	Dec 15, 2017	Star Wars Ep. VIII: The Last Jedi	$317,000,000	$620,181,382	$1,316,721,747

Clean DataFrame

Removing Unwanted Punctuation Charachters, From Float Values

In [4]:
```
1 tn_movie_budgets.production_budget = tn_movie_budgets.production_budget.
2 tn_movie_budgets.head()
```

Out[4]:

	id	release_date	movie	production_budget	domestic_gross	worldwide_gross
0	1	Dec 18, 2009	Avatar	425000000.0	$760,507,625	$2,776,345,279
1	2	May 20, 2011	Pirates of the Caribbean: On Stranger Tides	410600000.0	$241,063,875	$1,045,663,875
2	3	Jun 7, 2019	Dark Phoenix	350000000.0	$42,762,350	$149,762,350
3	4	May 1, 2015	Avengers: Age of Ultron	330600000.0	$459,005,868	$1,403,013,963
4	5	Dec 15, 2017	Star Wars Ep. VIII: The Last Jedi	317000000.0	$620,181,382	$1,316,721,747

In [5]:
```
tn_movie_budgets.domestic_gross = tn_movie_budgets.domestic_gross.apply(
tn_movie_budgets.head()
```

Out[5]:

	id	release_date	movie	production_budget	domestic_gross	worldwide_gross
0	1	Dec 18, 2009	Avatar	425000000.0	760507625.0	$2,776,345,279
1	2	May 20, 2011	Pirates of the Caribbean: On Stranger Tides	410600000.0	241063875.0	$1,045,663,875
2	3	Jun 7, 2019	Dark Phoenix	350000000.0	42762350.0	$149,762,350
3	4	May 1, 2015	Avengers: Age of Ultron	330600000.0	459005868.0	$1,403,013,963
4	5	Dec 15, 2017	Star Wars Ep. VIII: The Last Jedi	317000000.0	620181382.0	$1,316,721,747

In [6]:
```
tn_movie_budgets.worldwide_gross = tn_movie_budgets.worldwide_gross.apply
tn_movie_budgets.head()
```

Out[6]:

	id	release_date	movie	production_budget	domestic_gross	worldwide_gross
0	1	Dec 18, 2009	Avatar	425000000.0	760507625.0	2.776345e+09
1	2	May 20, 2011	Pirates of the Caribbean: On Stranger Tides	410600000.0	241063875.0	1.045664e+09
2	3	Jun 7, 2019	Dark Phoenix	350000000.0	42762350.0	1.497624e+08
3	4	May 1, 2015	Avengers: Age of Ultron	330600000.0	459005868.0	1.403014e+09
4	5	Dec 15, 2017	Star Wars Ep. VIII: The Last Jedi	317000000.0	620181382.0	1.316722e+09

Remove null values

In [7]:
```
1 tn_movie_budgets.dropna()
```

Out[7]:

	id	release_date	movie	production_budget	domestic_gross	worldwide_gross
0	1	Dec 18, 2009	Avatar	425000000.0	760507625.0	2.776345e+09
1	2	May 20, 2011	Pirates of the Caribbean: On Stranger Tides	410600000.0	241063875.0	1.045664e+09
2	3	Jun 7, 2019	Dark Phoenix	350000000.0	42762350.0	1.497624e+08
3	4	May 1, 2015	Avengers: Age of Ultron	330600000.0	459005868.0	1.403014e+09
4	5	Dec 15, 2017	Star Wars Ep. VIII: The Last Jedi	317000000.0	620181382.0	1.316722e+09
...	...	...	...	...	...	...
5777	78	Dec 31, 2018	Red 11	7000.0	0.0	0.000000e+00
5778	79	Apr 2, 1999	Following	6000.0	48482.0	2.404950e+05
5779	80	Jul 13, 2005	Return to the Land of Wonders	5000.0	1338.0	1.338000e+03
5780	81	Sep 29, 2015	A Plague So Pleasant	1400.0	0.0	0.000000e+00
5781	82	Aug 5, 2005	My Date With Drew	1100.0	181041.0	1.810410e+05

5782 rows × 6 columns

LET'S TALK DOLLARS

Money is what we spend and what we get back. We expect to get back at least what we spent, hopefully if all goes well more. The question is are we willing to wait for it to build up, want it back right away, or both? Let's go with Make your money back fast, but continue to build on it through generations.

Create total_gross Column

Add our domestic_gross and worldwide_gross values first. This gives a look at the big picture.

In [8]:

```
1  tn_movie_budgets['total_gross'] = tn_movie_budgets['domestic_gross'] + tn
2  tn_movie_budgets.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5782 entries, 0 to 5781
Data columns (total 7 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   id                 5782 non-null   int64  
 1   release_date       5782 non-null   object 
 2   movie              5782 non-null   object 
 3   production_budget  5782 non-null   float64
 4   domestic_gross     5782 non-null   float64
 5   worldwide_gross    5782 non-null   float64
 6   total_gross        5782 non-null   float64
dtypes: float64(4), int64(1), object(2)
memory usage: 316.3+ KB
```

Top 25 Highest Grossing Movies

This includes a juxtaposition of production_budget(yellow) and all gross values (total_gross(black), worldwide_gross(green), domestic_gross(blue) of the top 25 highest grossing movies. This will show us what movies are making or losing money over all.

Narrow down our dataset to the top 25(Top25) highest grossing movies

In [9]:
```
Top25=tn_movie_budgets.sort_values(["total_gross"], ascending = False).h
Top25
```

Out[9]:

	id	release_date	movie	production_budget	domestic_gross	worldwide_gross	tota
0	1	Dec 18, 2009	Avatar	425000000.0	760507625.0	2.776345e+09	3.536
5	6	Dec 18, 2015	Star Wars Ep. VII: The Force Awakens	306000000.0	936662225.0	2.053311e+09	2.989
42	43	Dec 19, 1997	Titanic	200000000.0	659363944.0	2.208208e+09	2.867
6	7	Apr 27, 2018	Avengers: Infinity War	300000000.0	678815482.0	2.048134e+09	2.726
33	34	Jun 12, 2015	Jurassic World	215000000.0	652270625.0	1.648855e+09	2.301
26	27	May 4, 2012	The Avengers	225000000.0	623279547.0	1.517936e+09	2.141
41	42	Feb 16, 2018	Black Panther	200000000.0	700059566.0	1.348258e+09	2.048
4	5	Dec 15, 2017	Star Wars Ep. VIII: The Last Jedi	317000000.0	620181382.0	1.316722e+09	1.936
66	67	Apr 3, 2015	Furious 7	190000000.0	353007020.0	1.518723e+09	1.871
3	4	May 1, 2015	Avengers: Age of Ultron	330600000.0	459005868.0	1.403014e+09	1.862
43	44	Jun 15, 2018	Incredibles 2	200000000.0	608581744.0	1.242521e+09	1.851
134	35	Mar 17, 2017	Beauty and the Beast	160000000.0	504014165.0	1.259200e+09	1.763
112	13	Jun 22, 2018	Jurassic World: Fallen Kingdom	170000000.0	417719760.0	1.305773e+09	1.723
260	61	Jul 15, 2011	Harry Potter and the Deathly Hallows: Part II	125000000.0	381193157.0	1.341693e+09	1.722
155	56	Nov 22, 2013	Frozen	150000000.0	400738009.0	1.272470e+09	1.673
47	48	May 3, 2013	Iron Man 3	200000000.0	408992272.0	1.215392e+09	1.624
44	45	Dec 16, 2016	Rogue One: A Star Wars Story	200000000.0	532177324.0	1.049103e+09	1.581
95	96	Mar 8, 2019	Captain Marvel	175000000.0	426525952.0	1.123062e+09	1.549

	id	release_date	movie	production_budget	domestic_gross	worldwide_gross	tota
16	17	May 6, 2016	Captain America: Civil War	250000000.0	408084349.0	1.140069e+09	1.548
74	75	Jul 18, 2008	The Dark Knight	185000000.0	533720947.0	1.001996e+09	1.535
10	11	Jul 20, 2012	The Dark Knight Rises	275000000.0	448139099.0	1.084439e+09	1.532
425	26	Dec 17, 2003	The Lord of the Rings: The Return of the King	94000000.0	377845905.0	1.141403e+09	1.519
45	46	Jun 17, 2016	Finding Dory	200000000.0	486295561.0	1.021215e+09	1.507
303	4	May 19, 1999	Star Wars Ep. I: The Phantom Menace	115000000.0	474544677.0	1.027045e+09	1.501
672	73	Jul 10, 2015	Minions	74000000.0	336045770.0	1.160336e+09	1.496

Values Visualized

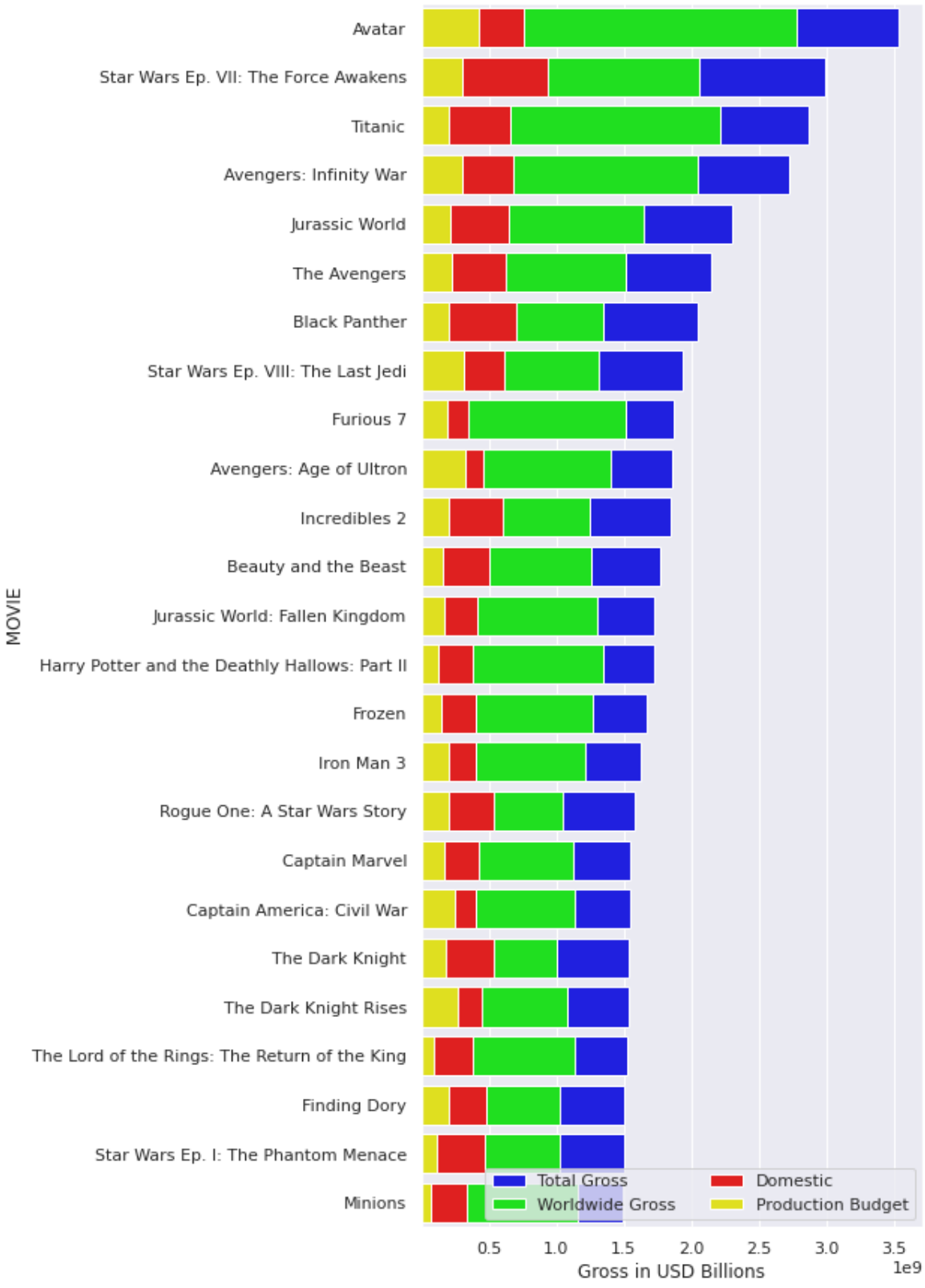
total_gross(blue), worldwide_gross(lime), domestic_gross(grey), production_budget(yellow)

In [10]:

```
sns.set_theme(style="darkgrid")

# # Initialize  matplotlib e
f, ax = plt.subplots(figsize=(6, 15), sharey=True)

# Load dataset
tn_top25_info = tn_movie_budgets.sort_values(['total_gross'], ascending

#Plot total gross values
sns.set_color_codes("bright")
sns.barplot(x="total_gross", y="movie", data=tn_top25_info,
            label="Total Gross", color="blue")

# Plot the worldwild gross values per title
sns.set_color_codes("bright")
sns.barplot(x="worldwide_gross", y="movie", data=tn_top25_info,
            label="Worldwide Gross", color="lime")

# Plot the domestic gross values per title
sns.set_color_codes("bright")
sns.barplot(x="domestic_gross", y="movie", data=tn_top25_info,
            label="Domestic", color="red")

#Plot budget
sns.set_color_codes("bright")
sns.barplot(x="production_budget", y="movie", data=tn_top25_info,
            label="Production Budget", color="yellow")

# Add legend
ax.legend(ncol=2, loc="lower right", frameon=True)
ax.set(xlim=len(tn_top25_info), ylabel=" MOVIE ",
       xlabel="Gross in USD Billions")
```

Out[10]:

```
[(25.0, 3713695549.2),
 Text(0, 0.5, ' MOVIE '),
 Text(0.5, 0, 'Gross in USD Billions')]
```



Observation

This table shows us the top 25 gross values of our entire database. Seeing how it's based on the total gross(total_gross aka in the black) for each of these movies, we could stop here and just want the $3.5+B Avatar has made so far. But, if we take a look at production values we can visually see that sometimes it makes a difference and sometimes it doesn't.

Let's See the Net Return Values for These Top Grossers

Net ReturnsTable

Domestic, Worldwide & the percentage of difference between the two gross values per movie(net_domestic,net_beyond, net_compare_per)

In [11]:

```
#find return on investement using total_gross & production budget
tn_movie_budgets['net_domestic'] =tn_movie_budgets['domestic_gross'] - t
tn_movie_budgets['net_beyond'] =tn_movie_budgets['worldwide_gross'] - tn
tn_movie_budgets.sort_values("total_gross", ascending=False).head(25)

```

Out[11]:

	id	release_date	movie	production_budget	domestic_gross	worldwide_gross	
0	1	Dec 18, 2009	Avatar	425000000.0	760507625.0	2.776345e+09	3.5
5	6	Dec 18, 2015	Star Wars Ep. VII: The Force Awakens	306000000.0	936662225.0	2.053311e+09	2.9
42	43	Dec 19, 1997	Titanic	200000000.0	659363944.0	2.208208e+09	2.8
6	7	Apr 27, 2018	Avengers: Infinity War	300000000.0	678815482.0	2.048134e+09	2.7
33	34	Jun 12, 2015	Jurassic World	215000000.0	652270625.0	1.648855e+09	2.3
26	27	May 4, 2012	The Avengers	225000000.0	623279547.0	1.517936e+09	2.1
41	42	Feb 16, 2018	Black Panther	200000000.0	700059566.0	1.348258e+09	2.0
4	5	Dec 15, 2017	Star Wars Ep. VIII: The Last Jedi	317000000.0	620181382.0	1.316722e+09	1.9
66	67	Apr 3, 2015	Furious 7	190000000.0	353007020.0	1.518723e+09	1.8
3	4	May 1, 2015	Avengers: Age of Ultron	330600000.0	459005868.0	1.403014e+09	1.8
43	44	Jun 15, 2018	Incredibles 2	200000000.0	608581744.0	1.242521e+09	1.8
134	35	Mar 17, 2017	Beauty and the Beast	160000000.0	504014165.0	1.259200e+09	1.7
112	13	Jun 22, 2018	Jurassic World: Fallen Kingdom	170000000.0	417719760.0	1.305773e+09	1.7
260	61	Jul 15, 2011	Harry Potter and the Deathly Hallows: Part II	125000000.0	381193157.0	1.341693e+09	1.7
155	56	Nov 22, 2013	Frozen	150000000.0	400738009.0	1.272470e+09	1.6
47	48	May 3, 2013	Iron Man 3	200000000.0	408992272.0	1.215392e+09	1.6

	id	release_date	movie	production_budget	domestic_gross	worldwide_gross	
44	45	Dec 16, 2016	Rogue One: A Star Wars Story	200000000.0	532177324.0	1.049103e+09	1.5
95	96	Mar 8, 2019	Captain Marvel	175000000.0	426525952.0	1.123062e+09	1.5
16	17	May 6, 2016	Captain America: Civil War	250000000.0	408084349.0	1.140069e+09	1.5
74	75	Jul 18, 2008	The Dark Knight	185000000.0	533720947.0	1.001996e+09	1.5
10	11	Jul 20, 2012	The Dark Knight Rises	275000000.0	448139099.0	1.084439e+09	1.5
425	26	Dec 17, 2003	The Lord of the Rings: The Return of the King	94000000.0	377845905.0	1.141403e+09	1.5
45	46	Jun 17, 2016	Finding Dory	200000000.0	486295561.0	1.021215e+09	1.5
303	4	May 19, 1999	Star Wars Ep. I: The Phantom Menace	115000000.0	474544677.0	1.027045e+09	1.5
672	73	Jul 10, 2015	Minions	74000000.0	336045770.0	1.160336e+09	1.4

Top 25 Highest Grossing Net Returns Visualized

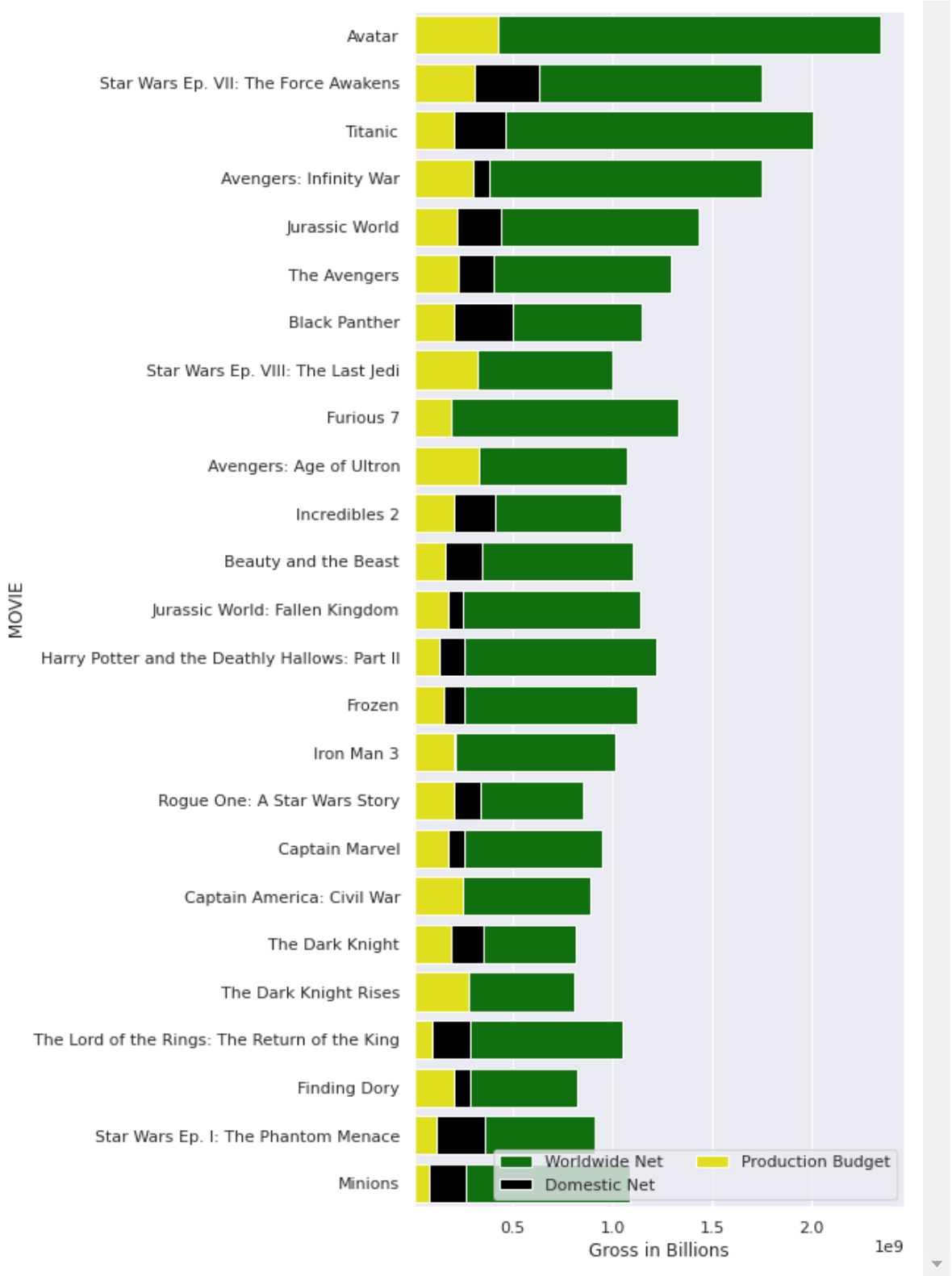
Here we see production_budget(yellow) for these 25 Top Grossing movies layered above the net values for both domestic(net_domestic(black)) and worldwide(net_beyond(green) layered underneath.

In [12]:

```
sns.set_theme(style="darkgrid")

## Initialize  matplotlib
f, ax = plt.subplots(figsize=(6, 15), sharey=True)

# Load dataset
tn_top25_dnet = tn_movie_budgets.sort_values(['total_gross'], ascending

# Plot the worldwild gross values per title
sns.set_color_codes("pastel")
sns.barplot(x="net_beyond", y="movie", data=tn_top25_dnet,
            label="Worldwide Net", color="green")

#Plot total gross values
sns.set_color_codes("pastel")
sns.barplot(x="net_domestic", y="movie", data=tn_top25_dnet,
            label="Domestic Net", color="black")

#Plot budget
sns.set_color_codes("muted")
sns.barplot(x="production_budget", y="movie", data=tn_top25_dnet,
            label="Production Budget", color="yellow",)


# Add legend
ax.legend(ncol=2, loc="lower right", frameon=True)
ax.set(xlim=len(tn_top25_dnet), ylabel=" MOVIE ",
       xlabel="Gross in Billions")
```

Out[12]:

```
[(25.0, 2468912542.95),
 Text(0, 0.5, ' MOVIE '),
 Text(0.5, 0, 'Gross in Billions')]
```



Observation

With production_budget values still attached, we see the need to go beyond the homeland and be able to share it with the world. AvatarMany factors come into play with going worldwide. First currency exchange rates. Again, we're talking atout money. With our data being in US Dollars, we will accept net returns as is. Another is language barriers. There are many ways of saying that action speaks louder than words, whoever or wherever you heard it from, it's true in the movie

industry. I'm not going to advise you on how to write your Action movie, or the parts, but if we look at our Top25 we see each one is about some kind of struggle. With struggle comes action. The World Needs Action! Conflict! Inner or Outer! Love and Hate!

Breaking Down Data by Domestic vs Worldwide Gross Values

Top 25 Highest Domestic Gross Value Movies (domestic_gross(red))

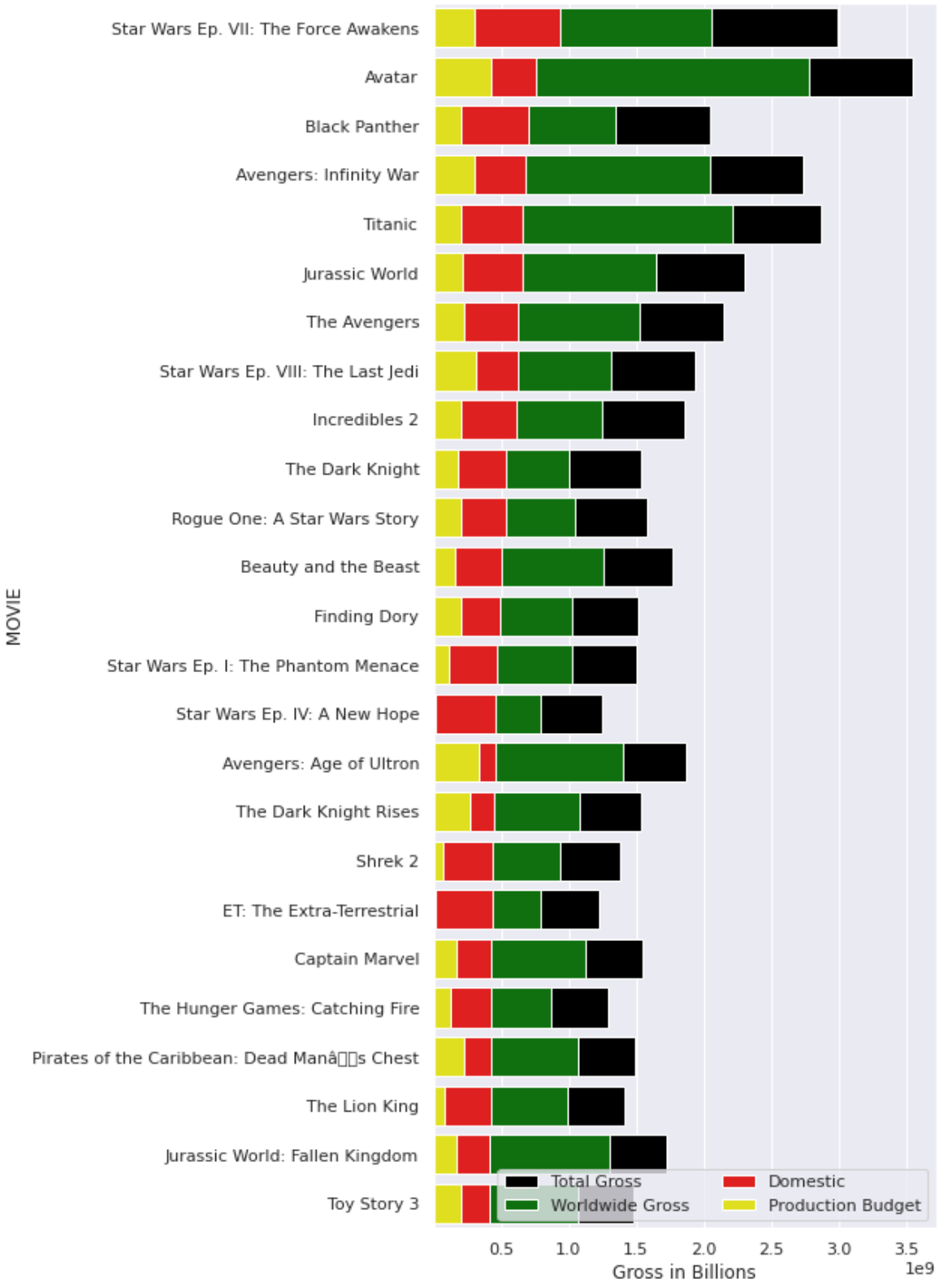
Same variables:total_gross(black), worldwide_gross(green) and production_budget(yellow) - But is across our entire dataset, not just the Top 25 Highest Grossing

In [13]:

```
sns.set_theme(style="darkgrid")

# # Initialize  matplotlib
f, ax = plt.subplots(figsize=(6, 15), sharey=True)

# Load dataset
tn_top25_dgross = tn_movie_budgets.sort_values(["domestic_gross"], ascen

#Plot total gross values
sns.set_color_codes("pastel")
sns.barplot(x="total_gross", y="movie", data=tn_top25_dgross ,
            label="Total Gross", color="black")

# Plot the worldwild gross values per title
sns.set_color_codes("pastel")
sns.barplot(x="worldwide_gross", y="movie", data=tn_top25_dgross ,
            label="Worldwide Gross", color="green")

# Plot the domestic gross values per title
sns.set_color_codes("bright")
sns.barplot(x="domestic_gross", y="movie", data=tn_top25_dgross,
            label="Domestic", color="red")

#Plot budget
sns.set_color_codes("muted")
sns.barplot(x="production_budget", y="movie", data=tn_top25_dgross,
            label="Production Budget", color="yellow",)


# Add legend
ax.legend(ncol=2, loc="lower right", frameon=True)
ax.set(xlim=len(tn_top25_dgross ), ylabel=" MOVIE ",
       xlabel="Gross in Billions")
```

Out[13]:

```
[(25.0, 3713695549.2),
 Text(0, 0.5, ' MOVIE '),
 Text(0.5, 0, 'Gross in Billions')]
```


Star Wars Ep. VII: The Force Awakens
Avatar
Black Panther
Avengers: Infinity War
Titanic
Jurassic World
The Avengers
Star Wars Ep. VIII: The Last Jedi
Incredibles 2
The Dark Knight
Rogue One: A Star Wars Story
Beauty and the Beast
Finding Dory
Star Wars Ep. I: The Phantom Menace
Star Wars Ep. IV: A New Hope
Avengers: Age of Ultron
The Dark Knight Rises
Shrek 2
ET: The Extra-Terrestrial
Captain Marvel
The Hunger Games: Catching Fire
Pirates of the Caribbean: Dead Manå□□s Chest
The Lion King
Jurassic World: Fallen Kingdom
Toy Story 3
MOVIE
Total Gross
Worldwide Gross
Domestic
Production Budget
0.5
1.0
1.5
2.0
2.5
3.0
3.5
1e9
Gross in Billions

Worldwide Top 25 Highest Grossing Values(worldwide_gross)

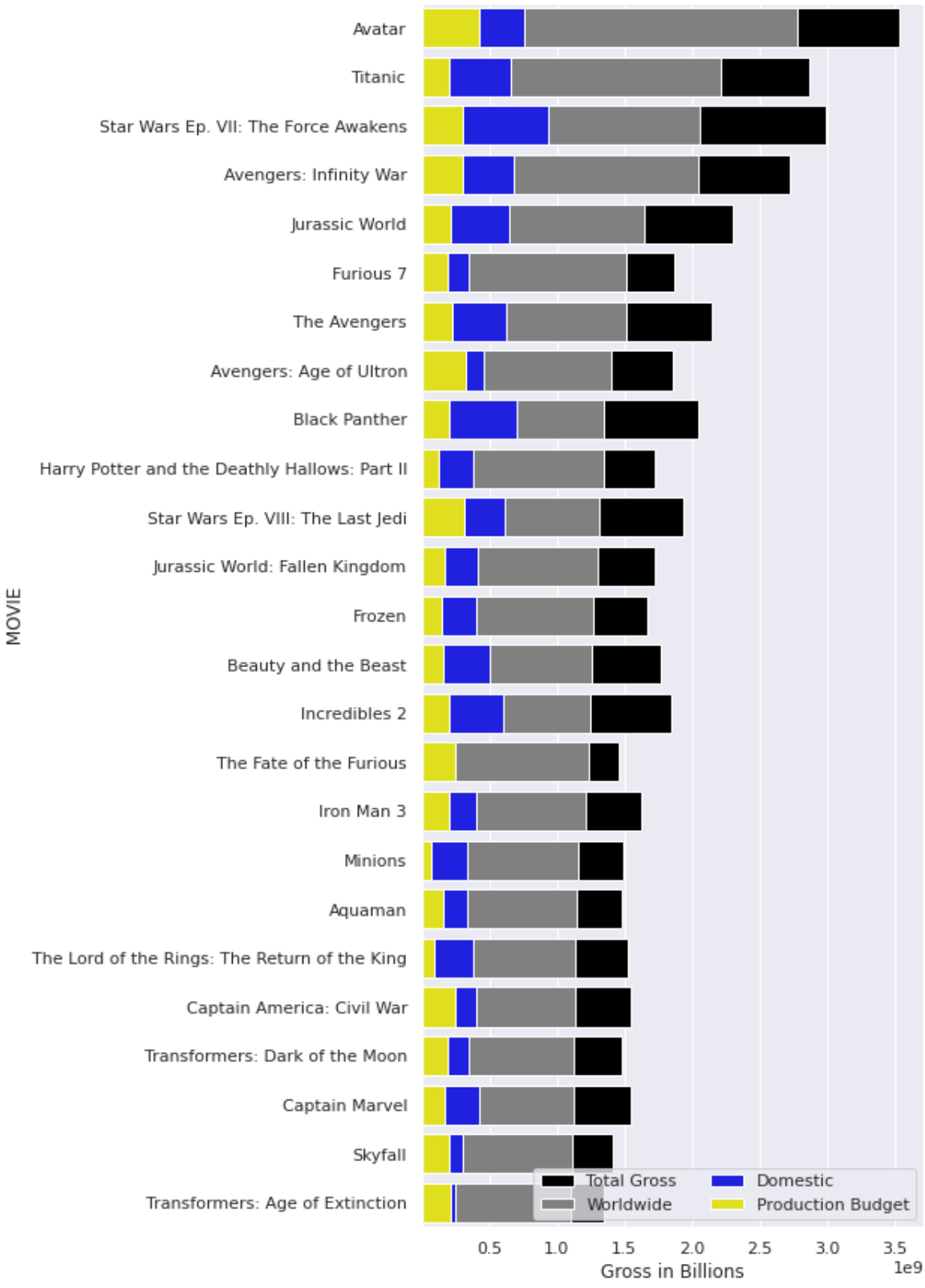
Plotting same variables and complete datset, but sorting by worldwide_gross shown in grey

In [14]:

```
sns.set_theme(style="darkgrid")

# # Initialize  matplotlib e
f, ax = plt.subplots(figsize=(6, 15), sharey=True)

# Load dataset
tn_top25_wwgross_info = tn_movie_budgets.sort_values(['worldwide_gross']

#Plot total gross values
sns.set_color_codes("pastel")
sns.barplot(x="total_gross", y="movie", data=tn_top25_wwgross_info,
            label="Total Gross", color="black")

# Plot the worldwild gross values per title
sns.set_color_codes("pastel")
sns.barplot(x="worldwide_gross", y="movie", data=tn_top25_wwgross_info,
            label="Worldwide", color="grey")

# Plot the domestic gross values per title
sns.set_color_codes("pastel")
sns.barplot(x="domestic_gross", y="movie", data=tn_top25_wwgross_info,
            label="Domestic", color="blue")

#Plot budget
sns.set_color_codes("pastel")
sns.barplot(x="production_budget", y="movie", data=tn_top25_wwgross_info
            label="Production Budget", color="yellow")

# Add legend
ax.legend(ncol=2, loc="lower right", frameon=True)
ax.set(xlim=len(tn_top25_wwgross_info), ylabel=" MOVIE ",
       xlabel="Gross in Billions")
```

Out[14]:

```
[(25.0, 3713695549.2),
 Text(0, 0.5, ' MOVIE '),
 Text(0.5, 0, 'Gross in Billions')]
```


Avatar
Titanic
Star Wars Ep. VII: The Force Awakens
Avengers: Infinity War
Jurassic World
Furious 7
The Avengers
Avengers: Age of Ultron
Black Panther
Harry Potter and the Deathly Hallows: Part II
Star Wars Ep. VIII: The Last Jedi
Jurassic World: Fallen Kingdom
Frozen
Beauty and the Beast
Incredibles 2
The Fate of the Furious
Iron Man 3
Minions
Aquaman
The Lord of the Rings: The Return of the King
Captain America: Civil War
Transformers: Dark of the Moon
Captain Marvel
Skyfall
Transformers: Age of Extinction
MOVIE
Total Gross
Worldwide
Domestic
Production Budget
0.5
1.0
1.5
2.0
2.5
3.0
3.5
Gross in Billions
1e9

Observation

Even as we change and review the gross values separately, we are able to gather that it's necessary to start domestically, but may not always bring recuperation. Start domestically, then reach beyond your borders.

To Reiterate

We will review the net percentages of both the domestic and worldwide gross values and then compare the two as a percentage. Whether we pull the top 25 of all our data or just those of the top 25 domestic and top 25 worldwide, we are able to see that stories being retold, broken down, or continuing saga's are what is having the most impact on around the world.

Net Percentages Tables

Domestic, Worldwide & the percentage of difference between the two gross values per movie(net_domestic(red),net_beyond(green), net_compare_per(blue))

In [15]:

```
#Finding the percentages
tn_movie_budgets['net_dom_per'] = (tn_movie_budgets['domestic_gross'] /
tn_movie_budgets['net_bey_per'] = (tn_movie_budgets['worldwide_gross'] /
tn_movie_budgets['net_compare_per'] = (tn_movie_budgets['net_dom_per'] /
tn_movie_budgets.head()
```

Out[15]:

	id	release_date	movie	production_budget	domestic_gross	worldwide_gross	total_g
0	1	Dec 18, 2009	Avatar	425000000.0	760507625.0	2.776345e+09	3.536853
1	2	May 20, 2011	Pirates of the Caribbean: On Stranger Tides	410600000.0	241063875.0	1.045664e+09	1.286728
2	3	Jun 7, 2019	Dark Phoenix	350000000.0	42762350.0	1.497624e+08	1.925247
3	4	May 1, 2015	Avengers: Age of Ultron	330600000.0	459005868.0	1.403014e+09	1.862020
4	5	Dec 15, 2017	Star Wars Ep. VIII: The Last Jedi	317000000.0	620181382.0	1.316722e+09	1.936903

In [16]:

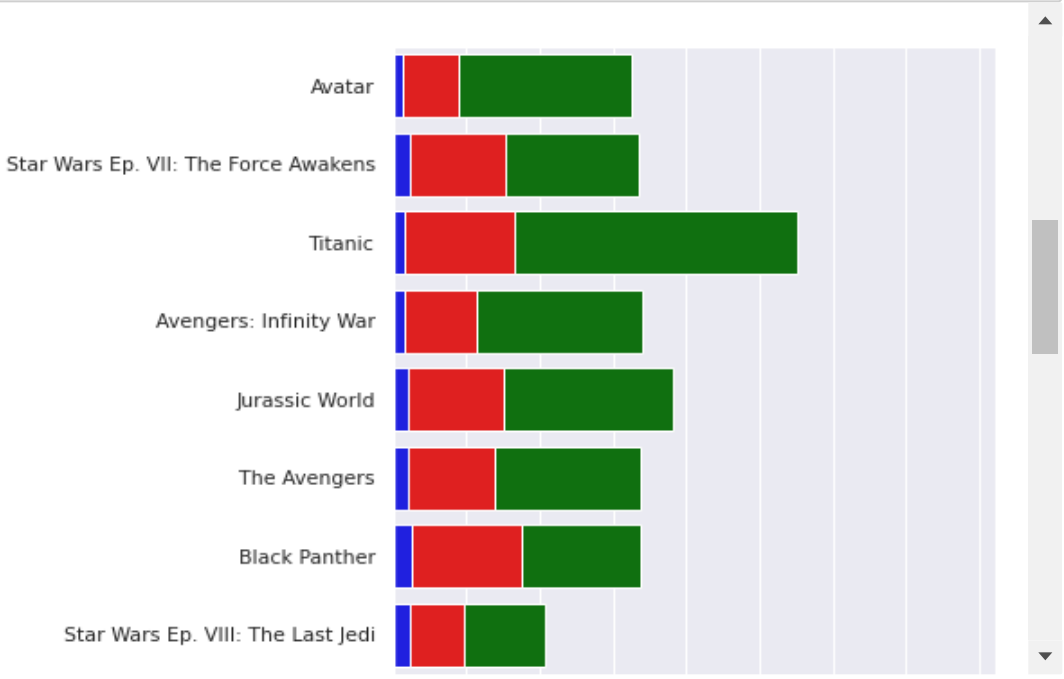
```
tn_movie_net_per_returns = tn_movie_budgets[['movie','release_date','tot
tn_movie_net_per_returns.sort_values('total_gross', ascending = False).h
```

Out[16]:

	movie	release_date	total_gross	net_dom_per	net_bey_per	net_compare_per
0	Avatar	Dec 18, 2009	3.536853e+09	178.942971	653.257713	27.392401
5	Star Wars Ep. VII: The Force Awakens	Dec 18, 2015	2.989973e+09	306.098766	671.016739	45.617158

In [17]:

```
sns.set_theme(style="darkgrid")

# # Initialize  matplotlib e
f, ax = plt.subplots(figsize=(6, 20), sharey=True)

# Load dataset
top25_netpers = tn_movie_net_per_returns.sort_values('total_gross', ascer

# Plot the worldwild gross values per title
sns.set_color_codes("bright")
sns.barplot(x="net_bey_per", y="movie", data=top25_netpers,
            label="Worldwide", color="green")

# Plot the domestic gross values per title
sns.set_color_codes("pastel")
sns.barplot(x="net_dom_per", y="movie", data=top25_netpers,
            label="Domestic", color="red")

#Plot total gross values
sns.set_color_codes("pastel")
sns.barplot(x="net_compare_per", y="movie", data=top25_netpers,
            label="Net Comparison Percentages", color="blue")

# Add legend
ax.legend(ncol=2, loc="lower right", frameon=True)
ax.set(xlim=(len(top25_netpers) / 100), ylabel=" MOVIE ",
       xlabel="Percentage of Total Gross")
```



In Conclusion

We have seen through these visualizations that the greatest impact Microsoft Studios can make would be to bring to life a story of epic proportions, tugging on the hearts of humanity by producing an action movie, with conflict, and challenging relationships. Possibly a story of long ago, yet to be

retold, or not so recently. The right one will warrant a desire for more. Begin distribution at home, then go worldwide and reap the benefits of crossing not only language barriers, but those barriers that mankind uses to separate themselves.

Thank you, Scharmaine Chappell

In [ ]:

```
1
```