Daniel Sterling

Aaron Suarez

Ivelisse Montero

Genre and Platform Sales Across Regions from 1980-2005

Thesis: Our study will analyze the sales performance of video games across genres and platforms through different regions, identifying shifts in genre and platform popularity and their correlation with the regions they are sold in.

Platform Sales: What platforms have the highest sales and what platforms are most prominent in each region? To find this information we grouped the sales by platforms that were available in our data set. After doing so we created individual tables for global sales as well as each region available. Global Sales showed a lot of consistency with North America, Europe, and Other Regions. PlayStation 2 dominated sales with both PlayStation 1 and PlayStation 3 achieving similar sales, all being within the top 6 consoles. Wii and Xbox360 also had high sales globally, keeping up with the PlayStation 3 sales. The only handheld console in our top 6 was the Nintendo DS. None of the other Consoles had noteworthy sales. Japan Sales showed a major difference with the Xbox 360 barely selling and being nowhere near Japan's top consoles. This is likely due to Microsoft not being a Japanese based company. (Reference pages: 3-8)

Genre Sales: What genres have the highest sales and what genres are the most prominent in each region? We used the same logic when analyzing Genre Sales. First grouping sales by genres, then creating individual tables for our regions. Just like with platform sales we notice similar trends between Global Sales and every region besides Japan. Action, Shooter, and Sports games were our top 3 performing genres globally. In Japan, Action and Sports games still dominated sales, however there is a clear distain for Shooter games. There is also a separate category that has almost no sales globally, but Japan managed to make up for it. Role-Playing games were the top performing genre in Japan, even above Action and Sports titles. Many RPGs originate in Japan, so there is a clear love for the genre within the country. (Reference pages: 9-14)

Major Contributions to Genre Sales: What were the three main genres seen in our data set and what countries contributed most to these genres? Initially we wanted to find out what countries were contributing most to Genre sales, however after visualizing our data set, we can see that all the genres are consistent with the population of the individual regions. Action, Sports, and Shooters were mostly consistent in all the countries, with only Japan's shooter sales being an inconsistency. (Reference pages: 15-19)

Correlation Between Years and Sales: Does the year a game is released affect its sales and what were the peak years for video game sales? As time has moved, overall, gaming has become more and more popular. From 1980-1995, gaming had a small but steady increase in sales. However, from 1995-2005, there is an incredibly increase in sales. Gaming popularity drastically increased during this time frame, with its peak being in the final year, 2005. Factors like populations increasing, as well as gaming becoming more and more successful, likely affect this information. It is clear from our line graph and our heat map that gaming was at its peak from 2000 to 2005. (Reference pages: 20-23)

Call to Action: Based of our Analysis of the Data Set we have concluded that if a new game were to included elements of Action, Sports, and Shooter games, while also ensuring that they focus marketing on North America and Europe, that game would have the best foundation for success.

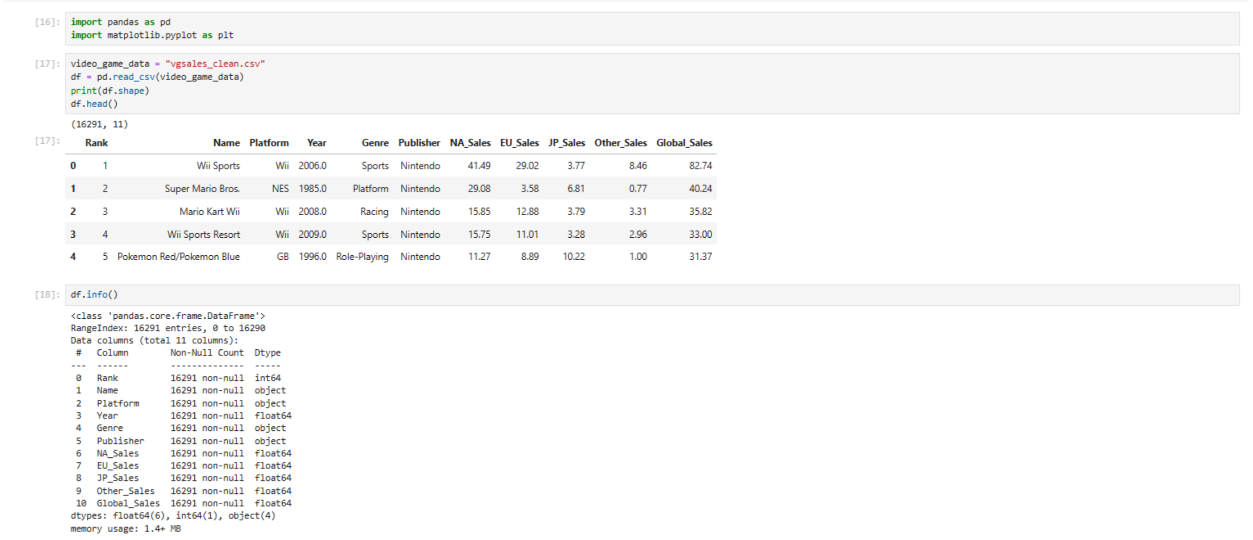
Bias: The major bias of our data set is platforms. Certain games could have likely found more success had they been made available on other Platforms. This is the case with Japan's Platform Sales.

Limitation: The major limitation to our data set is not having indications for a game being part of a long running series. This would heavily affect the popularity of a release.

Future Work: Companies like Microsoft would benefit from looking at sales in Japan, such as Role-Playing Games (RPGs), which would increase their overall sales output. America and Europe both have identical sales trends for both Platforms and Genres, so marketing similarly in both of those regions will benefit game developers.

Data Visualizations:

Platform Sales:



```
[16]: import pandas as pd
      import matplotlib.pyplot as plt
```

```
[17]: video_game_data = "vgsales_clean.csv"
      df = pd.read_csv(video_game_data)
      print(df.shape)
      df.head()
```

```
(16291, 11)
```

[17]:

	Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
0	1	Wii Sports	Wii	2006.0	Sports	Nintendo	41.49	29.02	3.77	8.46	82.74
1	2	Super Mario Bros.	NES	1985.0	Platform	Nintendo	29.08	3.58	6.81	0.77	40.24
2	3	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.85	12.88	3.79	3.31	35.82
3	4	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.75	11.01	3.28	2.96	33.00
4	5	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	1.00	31.37

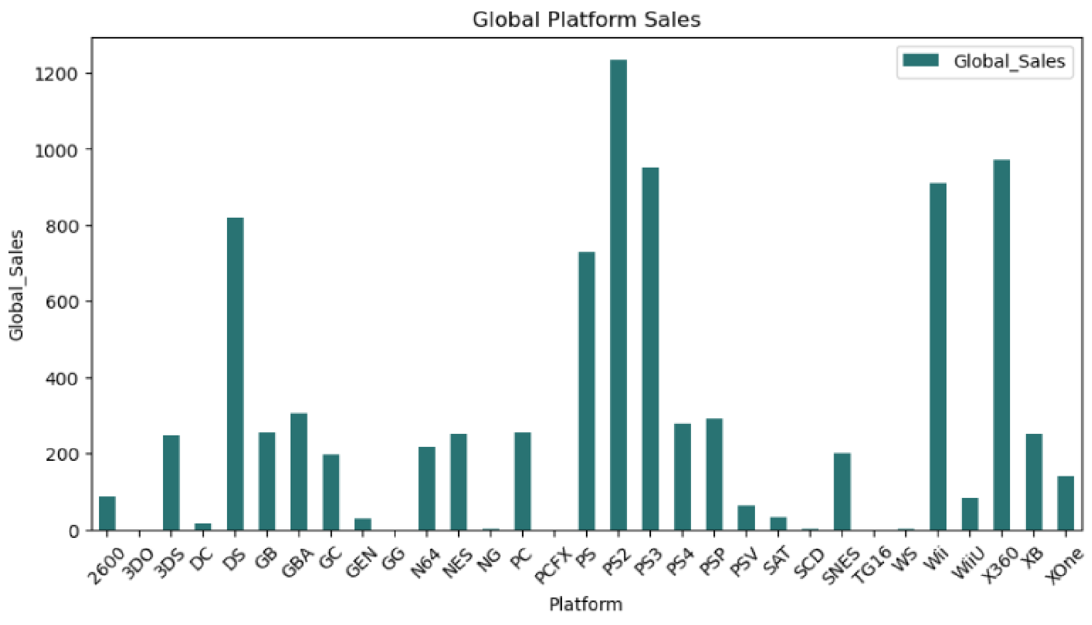
```
[18]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16291 entries, 0 to 16290
Data columns (total 11 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Rank          16291 non-null  int64  
 1   Name          16291 non-null  object 
 2   Platform      16291 non-null  object 
 3   Year          16291 non-null  float64
 4   Genre         16291 non-null  object 
 5   Publisher     16291 non-null  object 
 6   NA_Sales      16291 non-null  float64
 7   EU_Sales      16291 non-null  float64
 8   JP_Sales      16291 non-null  float64
 9   Other_Sales   16291 non-null  float64
 10  Global_Sales  16291 non-null  float64
dtypes: float64(6), int64(1), object(4)
memory usage: 1.4+ MB
```

[19]:
```
# Group Global_Sales
total_platform_sales_Global = df.groupby('Platform')['Global_Sales'].sum().reset_index()
print(total_platform_sales_Global)
```

```
   Platform  Global_Sales
0      2600         86.57
1       3DO          0.10
2       3DS        246.27
3        DC         15.97
4        DS        818.91
5        GB        254.42
6       GBA        305.62
7        GC        197.14
8       GEN         28.36
9        GG          0.04
10      N64        218.21
11      NES        251.07
12       NG          1.44
13       PC        254.70
14     PCFX          0.03
15       PS        727.39
16      PS2       1233.46
17      PS3        949.35
18      PS4        278.10
19      PSP        291.71
20      PSV         61.60
21      SAT         33.59
22      SCD          1.87
23     SNES        200.05
24     TG16          0.16
25       WS          1.42
26      Wii        909.81
27     WiiU         81.86
28     X360        969.60
29       XB        252.09
30     XOne        141.06
```

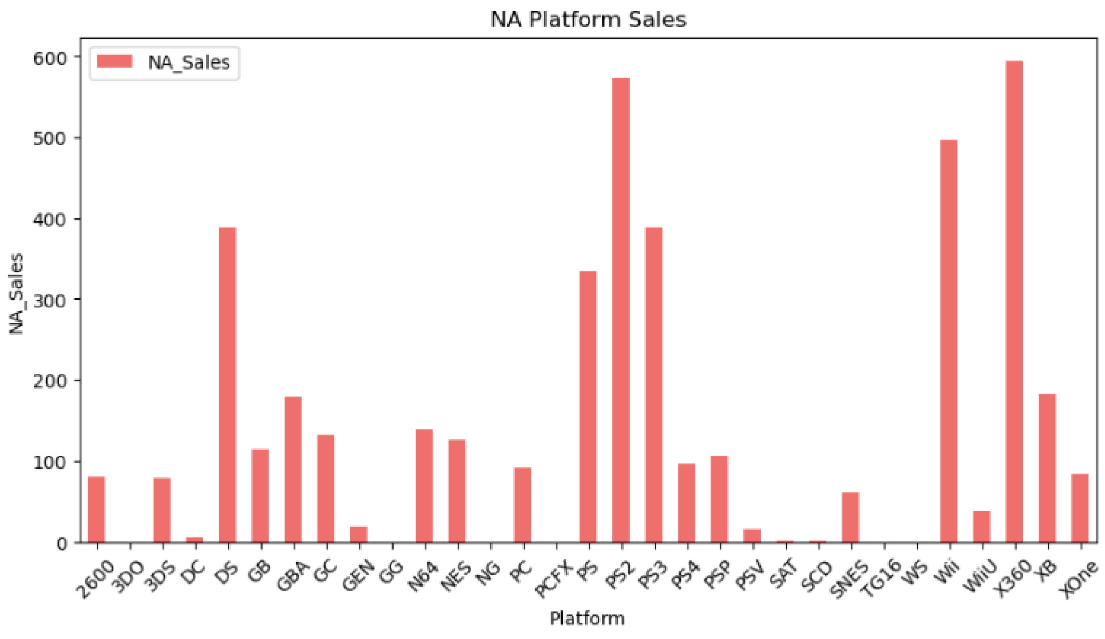
[20]:
```
# Bar Chart 1 NA Platform Sales

total_platform_sales_Global.plot(x='Platform', y='Global_Sales', kind='bar', figsize=(10,5), color='#297373')
plt.title('Global Platform Sales')
plt.xlabel('Platform')
plt.ylabel('Global_Sales')
plt.xticks(rotation=45)
plt.show()
```



[21]:
```
# Group NA_Sales
total_platform_sales_NA = df.groupby('Platform')['NA_Sales'].sum().reset_index()
print(total_platform_sales_NA)
```

```
   Platform  NA_Sales
0      2600     80.78
1       3DO      0.00
2       3DS     78.03
3        DC      5.43
4        DS    388.53
5        GB    113.64
6       GBA    178.43
7        GC    131.94
8       GEN     19.27
9        GG      0.00
10      N64    138.91
11      NES    125.94
12       NG      0.00
13       PC     92.04
14     PCFX      0.00
15       PS    334.71
16      PS2    572.92
17      PS3    388.90
18      PS4     96.80
19      PSP    107.09
20      PSV     16.07
21      SAT      0.72
22      SCD      1.00
23     SNES     61.23
24     TG16      0.00
25       WS      0.00
26      Wii    497.37
27     WiiU     38.32
28     X360    594.33
29       XB    182.06
30     XOne     83.19
```

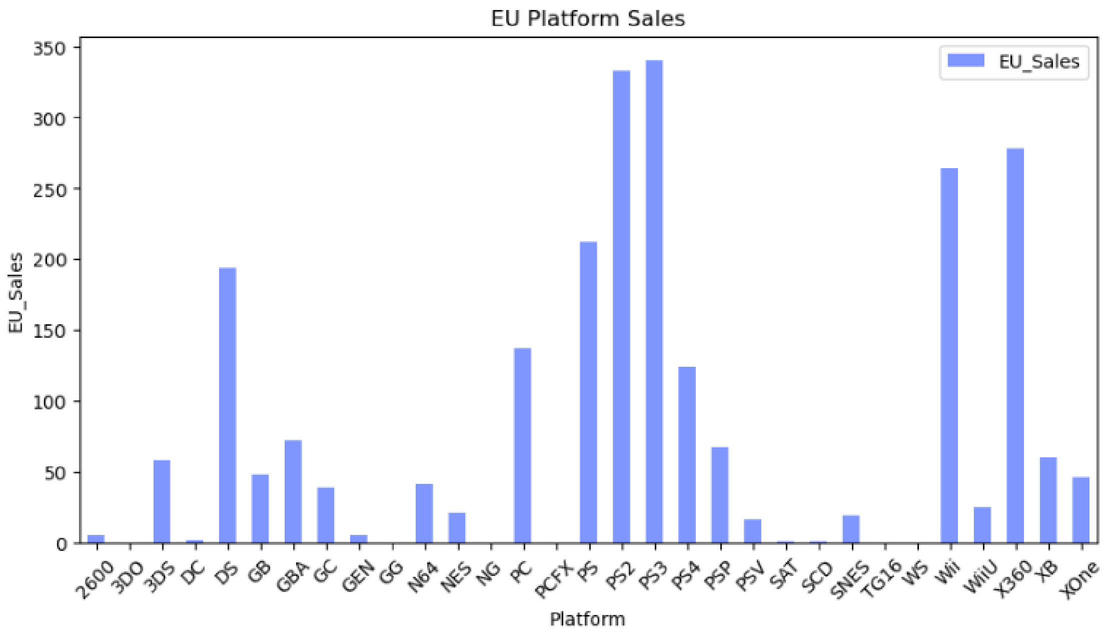
[22]:
```
# Bar Chart 1 NA Platform Sales

total_platform_sales_NA.plot(x='Platform', y='NA_Sales', kind='bar', figsize=(10,5), color='#ef6f6c')
plt.title('NA Platform Sales')
plt.xlabel('Platform')
plt.ylabel('NA_Sales')
plt.xticks(rotation=45)
plt.show()
```



```
[23]: # Group EU_Sales
total_platform_sales_EU = df.groupby('Platform')['EU_Sales'].sum().reset_index()
print(total_platform_sales_EU)
```

```
   Platform  EU_Sales
0      2600      4.86
1       3DO      0.00
2       3DS     58.29
3        DC      1.69
4        DS    194.05
5        GB     47.51
6       GBA     72.49
7        GC     38.32
8       GEN      5.52
9        GG      0.00
10      N64     41.03
11      NES     21.15
12       NG      0.00
13       PC    137.35
14     PCFX      0.00
15       PS    212.38
16      PS2    332.63
17      PS3    340.47
18      PS4    123.70
19      PSP     67.16
20      PSV     16.27
21      SAT      0.54
22      SCD      0.36
23     SNES     19.04
24     TG16      0.00
25       WS      0.00
26      Wii    264.35
27     WiiU     24.23
28     X360    278.00
29       XB     59.65
30     XOne     45.65
```

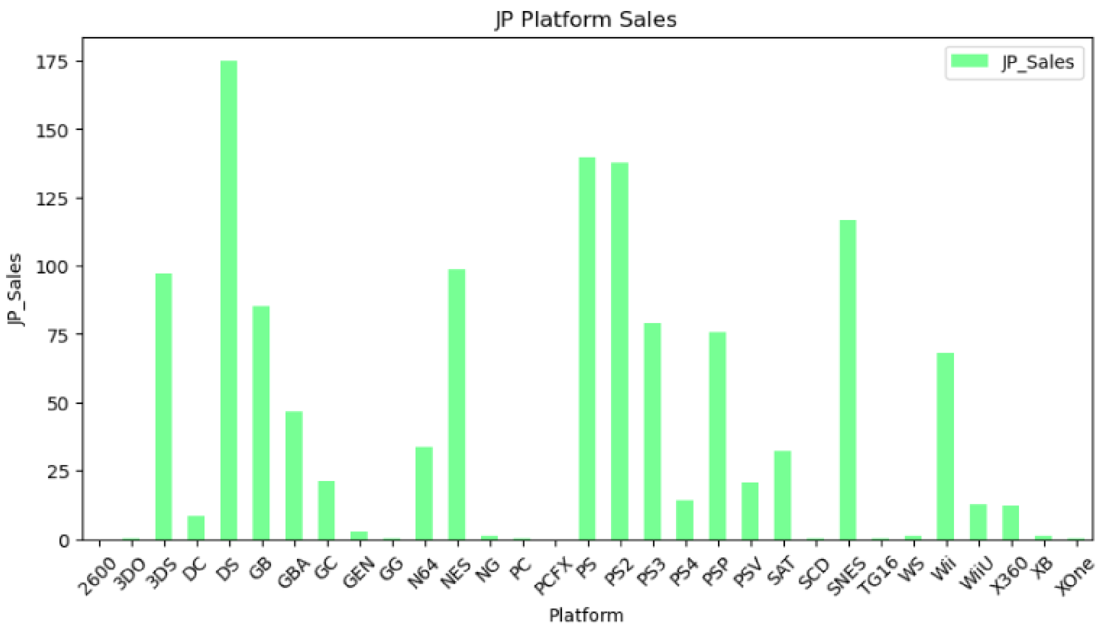
```
[24]: # Bar Chart 2 EU Platform Sales

total_platform_sales_EU.plot(x='Platform', y='EU_Sales', kind='bar', figsize=(10,5), color='#7f96ff')
plt.title('EU Platform Sales')
plt.xlabel('Platform')
plt.ylabel('EU_Sales')
plt.xticks(rotation=45)
plt.show()
```



[25]:
```
# Group JP_Sales
total_platform_sales_JP = df.groupby('Platform')['JP_Sales'].sum().reset_index()
print(total_platform_sales_JP)
```

```
   Platform  JP_Sales
0      2600      0.00
1       3DO      0.10
2       3DS     97.30
3        DC      8.56
4        DS    175.02
5        GB     85.12
6       GBA     46.56
7        GC     21.34
8       GEN      2.67
9        GG      0.04
10      N64     33.76
11      NES     98.65
12       NG      1.44
13       PC      0.17
14     PCFX      0.03
15       PS    139.78
16      PS2    137.54
17      PS3     79.21
18      PS4     14.30
19      PSP     75.89
20      PSV     20.86
21      SAT     32.26
22      SCD      0.45
23     SNES    116.55
24     TG16      0.16
25       WS      1.42
26      Wii     68.28
27     WiiU     12.79
28     X360     12.30
29       XB      1.38
30     XOne      0.34
```

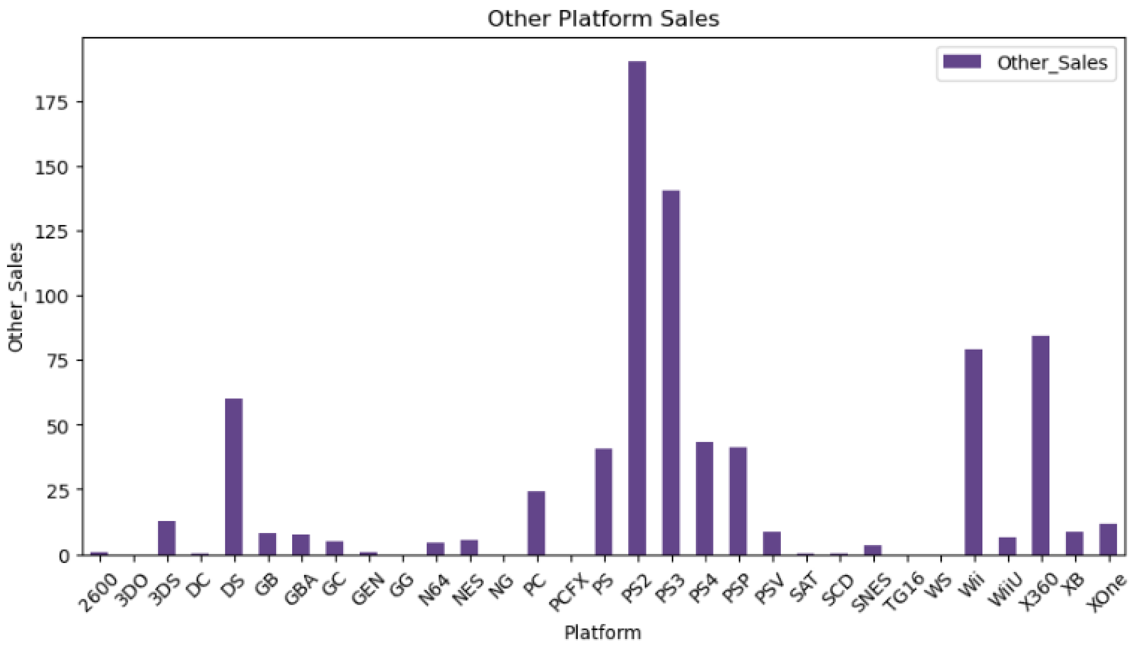
[26]:
```
# Bar Chart 3 JP Platform Sales

total_platform_sales_JP.plot(x='Platform', y='JP_Sales', kind='bar', figsize=(10,5), color='#77ff94')
plt.title('JP Platform Sales')
plt.xlabel('Platform')
plt.ylabel('JP_Sales')
plt.xticks(rotation=45)
plt.show()
```



```
[27]: # Group Other_Sales
total_platform_sales_Other = df.groupby('Platform')['Other_Sales'].sum().reset_index()
print(total_platform_sales_Other)
```

```
   Platform  Other_Sales
0      2600         0.84
1       3DO         0.00
2       3DS        12.55
3        DC         0.27
4        DS        60.29
5        GB         8.16
6       GBA         7.51
7        GC         5.13
8       GEN         0.89
9        GG         0.00
10      N64         4.31
11      NES         5.31
12       NG         0.00
13       PC        24.33
14     PCFX         0.00
15       PS        40.69
16      PS2       190.47
17      PS3       140.81
18      PS4        43.36
19      PSP        41.52
20      PSV         8.41
21      SAT         0.07
22      SCD         0.05
23     SNES         3.22
24     TG16         0.00
25       WS         0.00
26      Wii        79.20
27     WiiU         6.45
28     X360        84.67
29       XB         8.48
30     XOne        11.92
```

```
[28]: # Bar Chart 4 Other Platform Sales

total_platform_sales_Other.plot(x='Platform', y='Other_Sales', kind='bar', figsize=(10,5), color='#63458a')
plt.title('Other Platform Sales')
plt.xlabel('Platform')
plt.ylabel('Other_Sales')
plt.xticks(rotation=45)
plt.show()
```



Genre Sales:

```
[399]: # Dependencies and Setup
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)
```

```
[400]: # File to Load
data_filepath = ("vgsales_clean.csv")
video_game_data = pd.read_csv(data_filepath)
```

```
[401]: #Read Video Games Sales Data File and store into Pandas DataFrames
print(video_game_data.shape)
video_game_data.head()
```

```
(16291, 11)
```

[401]:

	Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
0	1	Wii Sports	Wii	2006.0	Sports	Nintendo	41.49	29.02	3.77	8.46	82.74
1	2	Super Mario Bros.	NES	1985.0	Platform	Nintendo	29.08	3.58	6.81	0.77	40.24
2	3	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.85	12.88	3.79	3.31	35.82
3	4	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.75	11.01	3.28	2.96	33.00
4	5	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	1.00	31.37

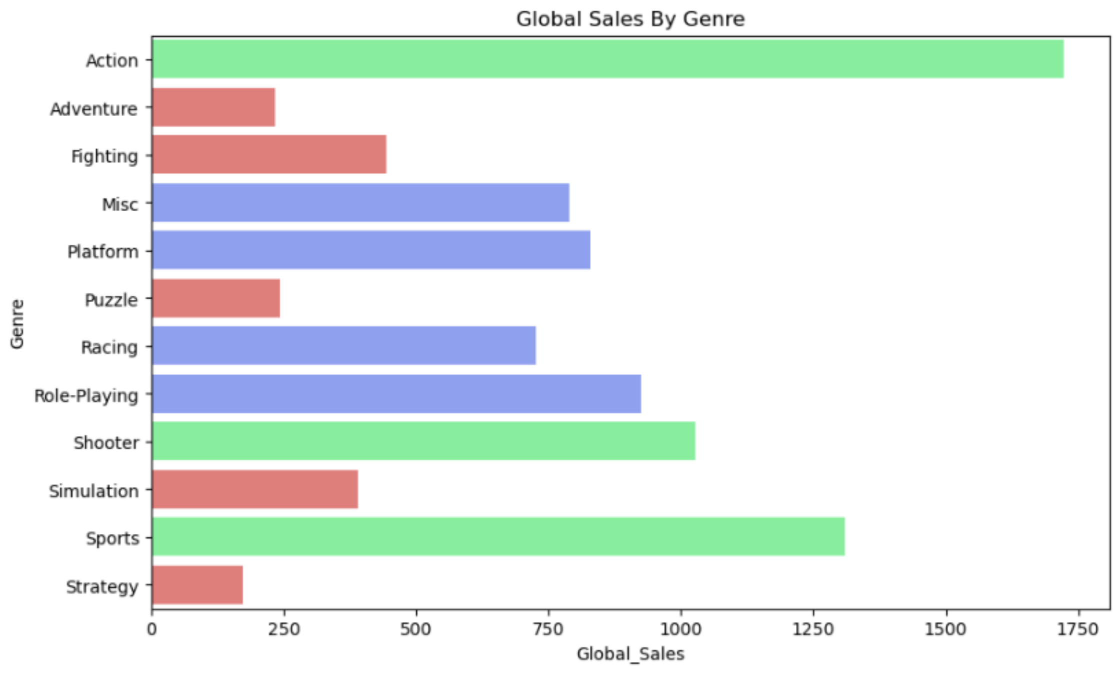
```
[402]: Data_filepath= pd.DataFrame(video_game_data)
Data_filepath.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16291 entries, 0 to 16290
Data columns (total 11 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Rank          16291 non-null  int64  
 1   Name          16291 non-null  object 
 2   Platform      16291 non-null  object 
 3   Year          16291 non-null  float64
 4   Genre         16291 non-null  object 
 5   Publisher     16291 non-null  object 
 6   NA_Sales      16291 non-null  float64
 7   EU_Sales      16291 non-null  float64
 8   JP_Sales      16291 non-null  float64
 9   Other_Sales   16291 non-null  float64
 10  Global_Sales  16291 non-null  float64
dtypes: float64(6), int64(1), object(4)
memory usage: 1.4+ MB
```

```
[403]: # Calculate the total amount in each genre by global sales
       total_platform_genres = pd.DataFrame(video_game_data.groupby('Genre')['Global_Sales'].sum().reset_index())
       print(total_platform_genres)
```

```
           Genre  Global_Sales
0         Action       1722.84
1      Adventure        234.59
2       Fighting        444.05
3           Misc        789.87
4       Platform        829.13
5         Puzzle        242.21
6         Racing        726.76
7   Role-Playing        923.83
8        Shooter       1026.20
9     Simulation        389.98
10        Sports       1309.24
11      Strategy        173.27
```

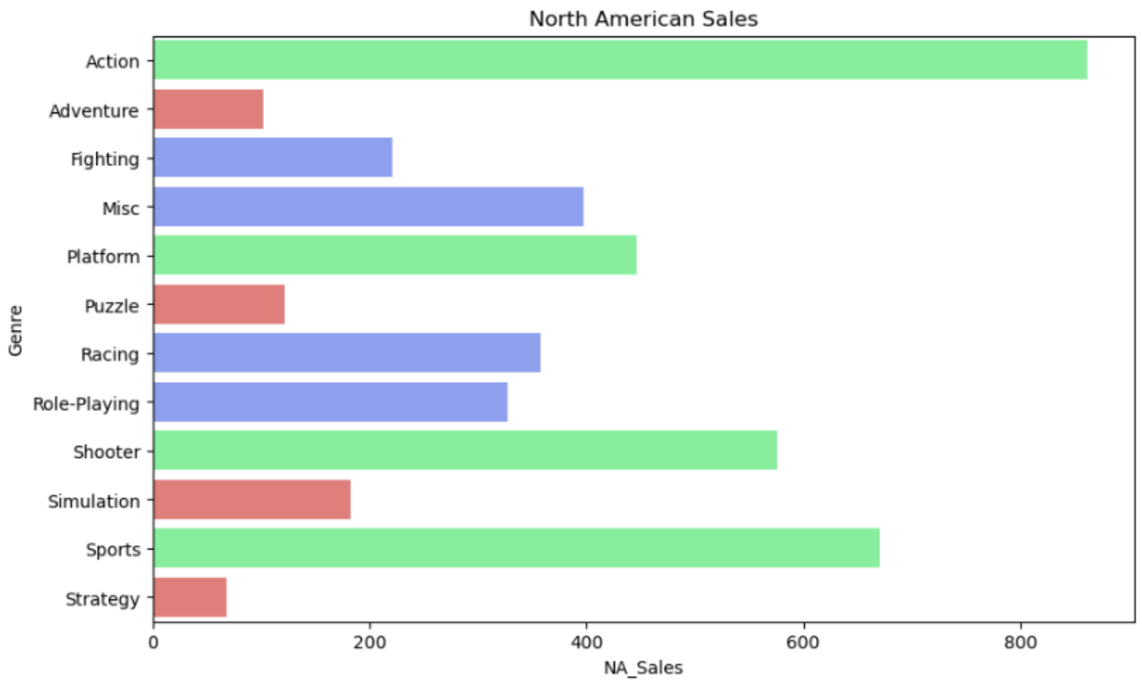
```
[404]: # Grouped Bar Chart for Global Sales By Genre
       clrs = ['#EF6F6C' if (x < 500) else '#7F96FF' if (x < 1000) else '#77FF94' for x in total_platform_genres['Global_Sales']]
       plt.figure(figsize=(10, 6))
       sns.barplot(data = total_platform_genres, x ='Global_Sales', y = 'Genre', errorbar = None, palette = clrs,)
       plt.title('Global Sales By Genre')
       plt.xlabel('Global_Sales')
       plt.ylabel('Genre')
       plt.show()
```



```
[405]: # Calculate the total amount in each genre by NA sales
       total_platform_genres = pd.DataFrame(Data_filepath.groupby('Genre')['NA_Sales'].sum().reset_index())
       print(total_platform_genres)
```

```
           Genre  NA_Sales
0         Action    861.77
1      Adventure    101.93
2       Fighting    220.74
3           Misc    396.92
4       Platform    445.99
5         Puzzle    122.01
6         Racing    356.93
7   Role-Playing    326.50
8        Shooter    575.16
9     Simulation    181.78
10        Sports    670.09
11      Strategy     67.83
```

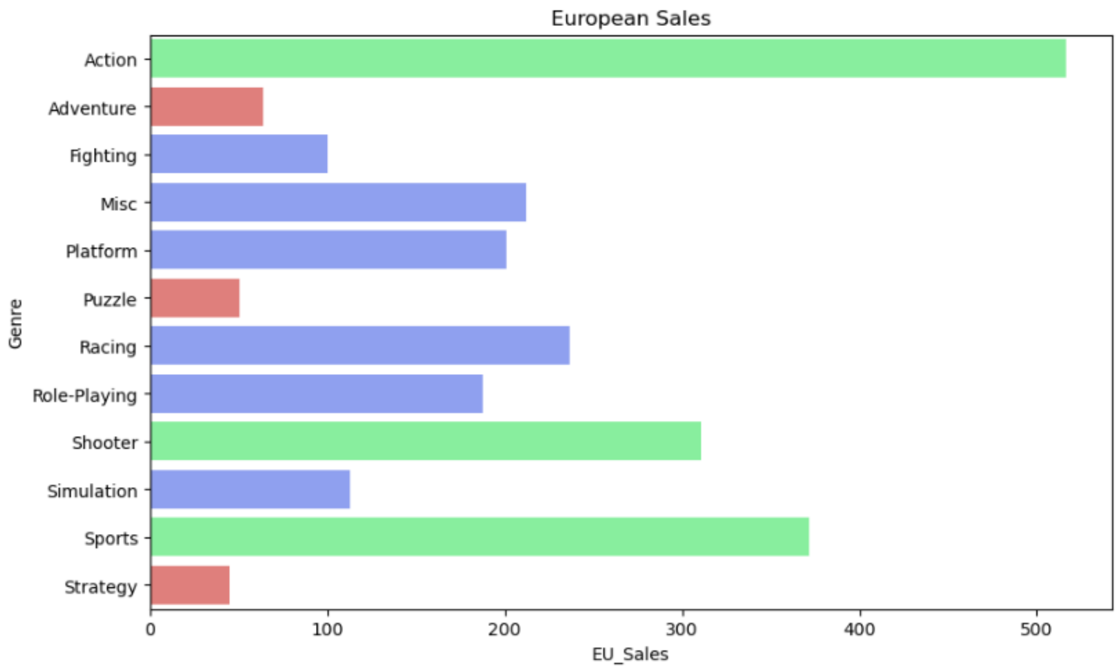
```
[406]: # Grouped Bar Chart for NA Sales By Genre
       clrs = ['#EF6F6C' if (x < 200) else '#7F96FF' if (x < 400) else '#77FF94' for x in total_platform_genres['NA_Sales']]
       plt.figure(figsize=(10, 6))
       sns.barplot(data = total_platform_genres, x ='NA_Sales', y = 'Genre', errorbar = None, palette = clrs,)
       plt.title('North American Sales')
       plt.xlabel('NA_Sales')
       plt.ylabel('Genre')
       plt.show()
```



[407]:
```
# Calculate the total amount in each genre by EU Sales
total_platform_genres = pd.DataFrame(video_game_data.groupby('Genre')['EU_Sales'].sum().reset_index())
print(total_platform_genres)
```

```
           Genre  EU_Sales
0         Action    516.48
1      Adventure     63.74
2       Fighting    100.00
3           Misc    211.77
4       Platform    200.65
5         Puzzle     50.52
6         Racing    236.31
7   Role-Playing    187.57
8        Shooter    310.45
9     Simulation    113.02
10        Sports    371.34
11      Strategy     44.84
```

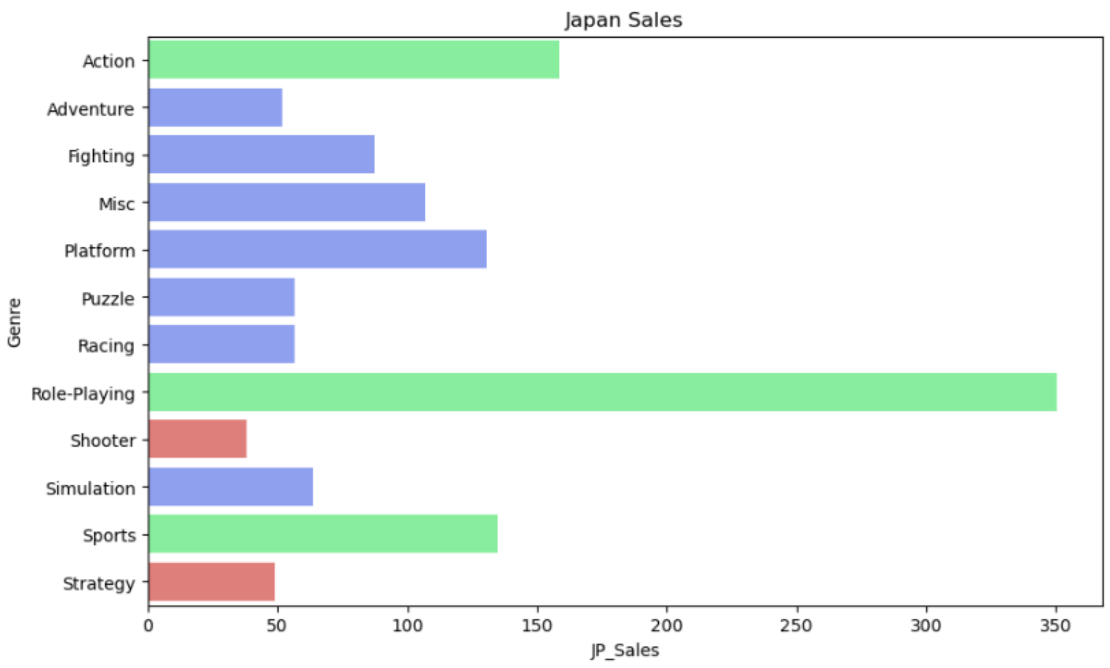
[408]:
```
# Grouped Bar Chart for EU Sales By Genre
clrs = ['#EF6F6C' if (x < 100) else '#7F96FF' if (x < 300) else '#77FF94' for x in total_platform_genres['EU_Sales']]
plt.figure(figsize=(10, 6))
sns.barplot(data = total_platform_genres, x ='EU_Sales', y = 'Genre', errorbar = None, palette = clrs,)
plt.title('European Sales')
plt.xlabel('EU_Sales')
plt.ylabel('Genre')
plt.show()
```



[409]:
```python
# Calculate the total amount in each genre by JP Sales
total_platform_genres = pd.DataFrame(video_game_data.groupby('Genre')['JP_Sales'].sum().reset_index())
print(total_platform_genres)
```

```
           Genre  JP_Sales
0         Action    158.65
1      Adventure     51.99
2       Fighting     87.15
3           Misc    106.67
4       Platform    130.65
5         Puzzle     56.68
6         Racing     56.61
7   Role-Playing    350.29
8        Shooter     38.18
9     Simulation     63.54
10        Sports    134.76
11      Strategy     49.10
```

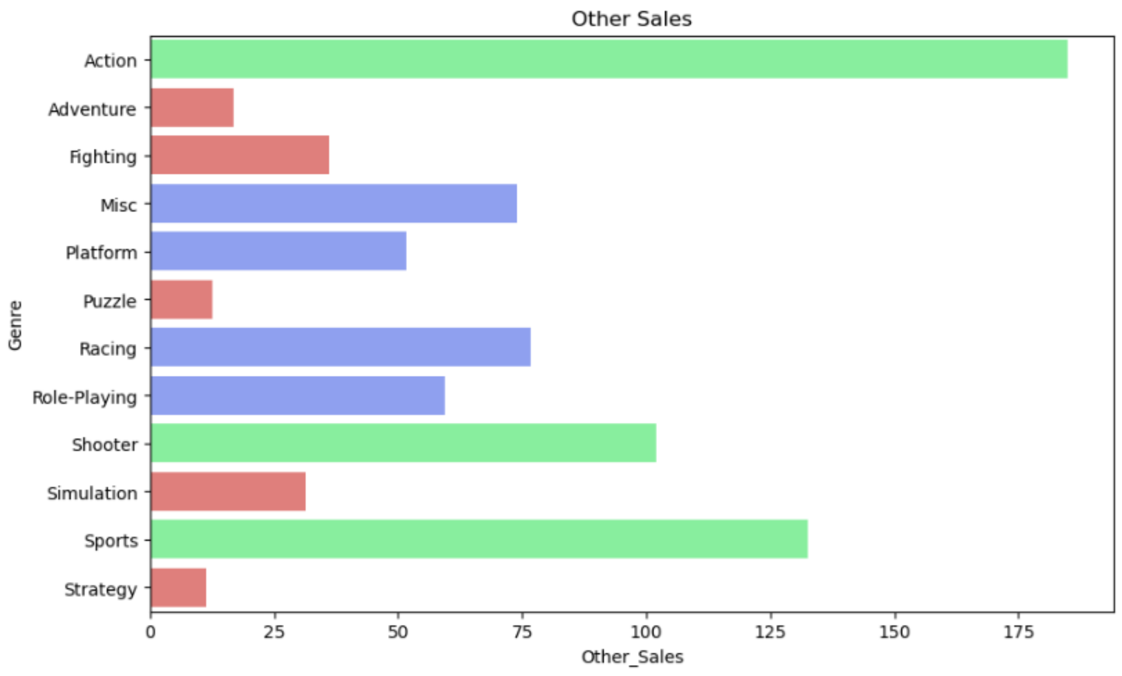
[410]:
```python
# Grouped Bar Chart for JP Sales By Genre
clrs = ['#EF6F6C' if (x < 50) else '#7F96FF' if (x < 131) else '#77FF94' for x in total_platform_genres['JP_Sales']]
plt.figure(figsize=(10, 6))
sns.barplot(data = total_platform_genres, x ='JP_Sales', y = 'Genre', errorbar = None, palette = clrs,)
plt.title('Japan Sales')
plt.xlabel('JP_Sales')
plt.ylabel('Genre')
plt.show()
```



[411]:
```
# Calculate the total amount in each genre by Other Sales
total_platform_genres = pd.DataFrame(video_game_data.groupby('Genre')['Other_Sales'].sum().reset_index())
print(total_platform_genres)
```

```
           Genre  Other_Sales
0         Action       184.92
1      Adventure        16.70
2       Fighting        36.19
3           Misc        73.92
4       Platform        51.51
5         Puzzle        12.47
6         Racing        76.68
7   Role-Playing        59.38
8        Shooter       101.90
9     Simulation        31.36
10        Sports       132.65
11      Strategy        11.23
```

[412]:
```
# Grouped Bar Chart for Other Sales By Genre
clrs = ['#EF6F6C' if (x < 50) else '#7F96FF' if (x < 100) else '#77FF94' for x in total_platform_genres['Other_Sales']]
plt.figure(figsize=(10, 6))
sns.barplot(data = total_platform_genres, x ='Other_Sales', y = 'Genre', errorbar = None, palette = clrs,)
plt.title('Other Sales')
plt.xlabel('Other_Sales')
plt.ylabel('Genre')
plt.show()
```



Major Contributions to Genre Sales:

[128]:
```
import folium
import pandas as pd
import geopandas as gpd
import geoviews as gv
from bokeh.io import output_notebook
from bokeh.plotting import show
```

[129]:
```
GAME_DATA_PATH = "vgsales_clean.csv"
MAP_CENTER = [0, 0]
MAP_ZOOM = 2
EU_LAT = 54.5260
EU_LON = 15.2551
JP_LAT = 36.2048
JP_LON = 138.2529
NA_LAT = 39.30
NA_LON = -94.71
EU_COLOR = 'blue'
JP_COLOR = 'green'
NA_COLOR = 'red'
LOC_DATA = {
    'EU_Sales': {'lat': 54.53, 'lon': 15.26, 'color': '#7F96FF',},
    'JP_Sales': {'lat': 36.20, 'lon': 138.25, 'color': '#77FF94',},
    'NA_Sales': {'lat': 39.30, 'lon': -94.71, 'color': '#EF6F6C',},
}
```

[130]:
```
df = pd.read_csv(GAME_DATA_PATH )
print(df.shape)
df.head()
```

```
(16291, 11)
```

[130]:

	Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
0	1	Wii Sports	Wii	2006.0	Sports	Nintendo	41.49	29.02	3.77	8.46	82.74
1	2	Super Mario Bros.	NES	1985.0	Platform	Nintendo	29.08	3.58	6.81	0.77	40.24
2	3	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.85	12.88	3.79	3.31	35.82
3	4	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.75	11.01	3.28	2.96	33.00
4	5	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	1.00	31.37

```
[131]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16291 entries, 0 to 16290
Data columns (total 11 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Rank          16291 non-null  int64  
 1   Name          16291 non-null  object 
 2   Platform      16291 non-null  object 
 3   Year          16291 non-null  float64
 4   Genre         16291 non-null  object 
 5   Publisher     16291 non-null  object 
 6   NA_Sales      16291 non-null  float64
 7   EU_Sales      16291 non-null  float64
 8   JP_Sales      16291 non-null  float64
 9   Other_Sales   16291 non-null  float64
 10  Global_Sales  16291 non-null  float64
dtypes: float64(6), int64(1), object(4)
memory usage: 1.4+ MB
```

```
[132]: df_melted = df.melt(id_vars=['Rank','Genre'], value_vars=['NA_Sales', 'EU_Sales', 'JP_Sales'], var_name='region', value_name='sales')
print(df_melted.shape)
df_melted.head(3)
```

```
(48873, 4)
```

[132]:

	Rank	Genre	region	sales
0	1	Sports	NA_Sales	41.49
1	2	Platform	NA_Sales	29.08
2	3	Racing	NA_Sales	15.85

```
[133]: total_genre_sales = df_melted.groupby(['Genre','region']).sum().reset_index()
for region, vals in LOC_DATA.items():
    mask_s = total_genre_sales.region == region
    total_genre_sales.loc[mask_s, 'lat'] = vals['lat']
    total_genre_sales.loc[mask_s, 'lon'] = vals['lon']
    total_genre_sales.loc[mask_s, 'color'] = vals['color']
print(total_genre_sales.shape)
total_genre_sales.head()
```

```
(36, 7)
```

[133]:

	Genre	region	Rank	sales	lat	lon	color
0	Action	EU_Sales	25955792	516.48	54.53	15.26	#7F96FF
1	Action	JP_Sales	25955792	158.65	36.20	138.25	#77FF94
2	Action	NA_Sales	25955792	861.77	39.30	-94.71	#EF6F6C
3	Adventure	EU_Sales	14704318	63.74	54.53	15.26	#7F96FF
4	Adventure	JP_Sales	14704318	51.99	36.20	138.25	#77FF94

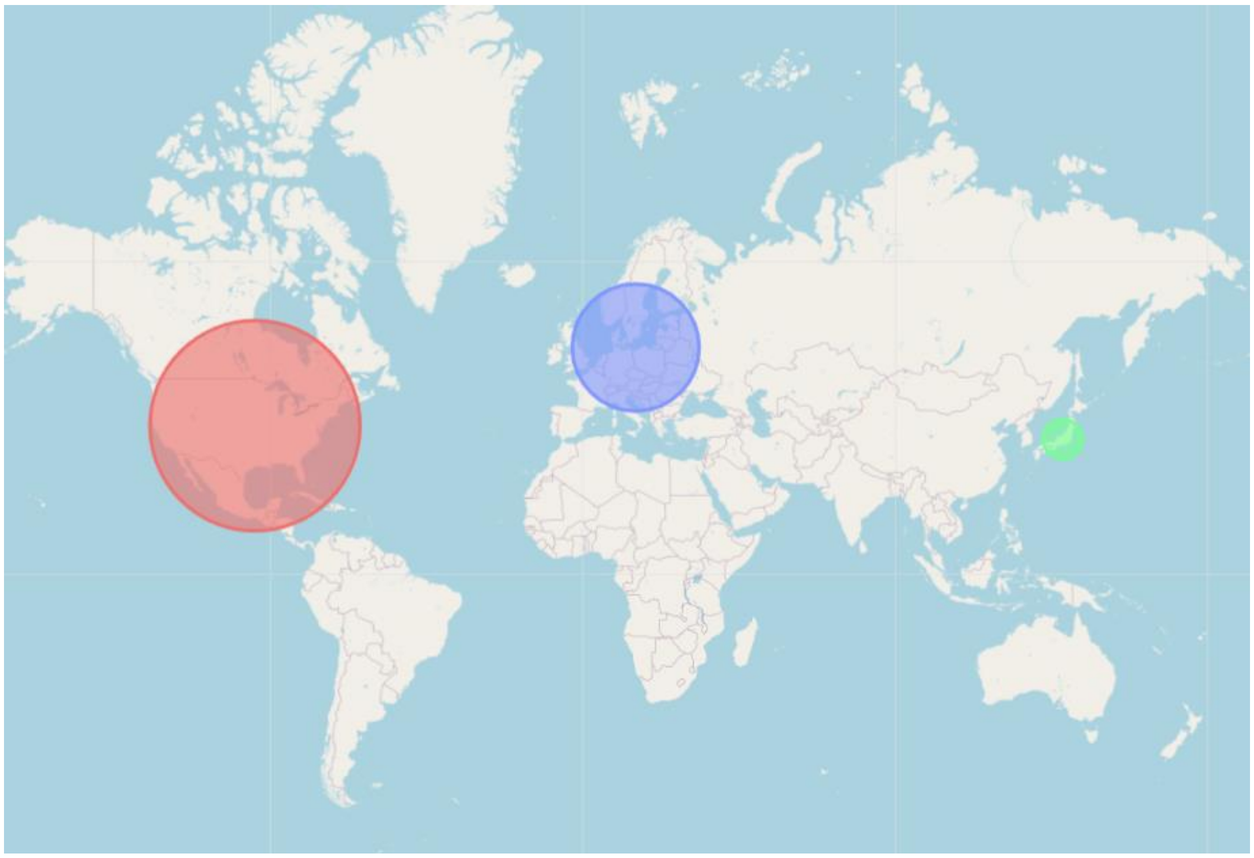
```
[134]: # Action data frame

action_df = total_genre_sales.loc[total_genre_sales['Genre']=='Action'].copy()
print(action_df.shape)
action_df.head(3)
```

```
(3, 7)
```

[134]:

	Genre	region	Rank	sales	lat	lon	color
0	Action	EU_Sales	25955792	516.48	54.53	15.26	#7F96FF
1	Action	JP_Sales	25955792	158.65	36.20	138.25	#77FF94
2	Action	NA_Sales	25955792	861.77	39.30	-94.71	#EF6F6C

```
[135]: # Create a map centered around the average latitude and longitude

my_map=folium.Map(location=MAP_CENTER, zoom_start=MAP_ZOOM)

# Loop through the DataFrame and add CircleMarkers to the map
for idx, row in action_df.iterrows():
    folium.CircleMarker(
        location=[row['lat'], row['lon']],  # Set the location based on lat and lon
        radius=row['sales'] / 10,  # Size is based on the 'size' column, divided to scale it properly
        color=row['color'],  # Marker color
        fill=True,
        fill_color=row['color'],  # Fill color
        fill_opacity=0.6,
        # popup=f"Size: {row['size']}",  # Add popup with size info
    ).add_to(my_map)

my_map
```



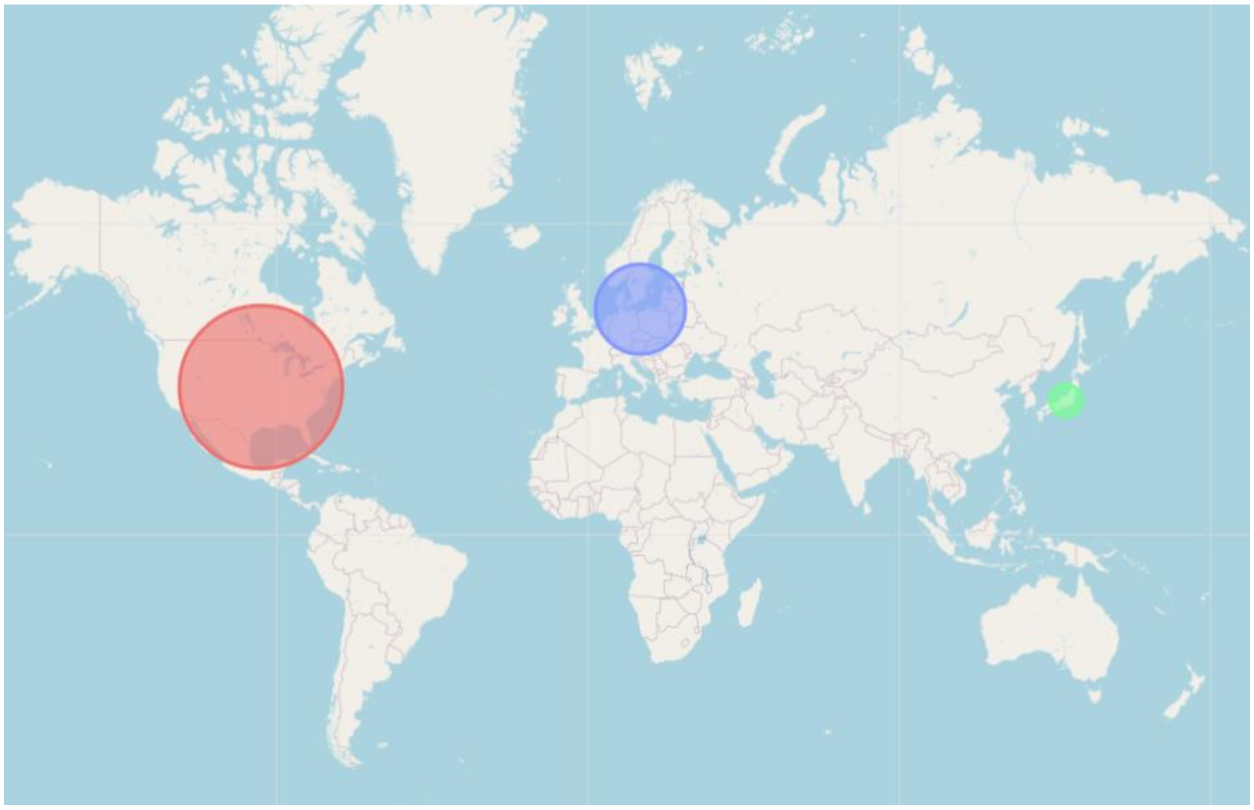
```
[136]: # Sports data frame

sports_df = total_genre_sales.loc[total_genre_sales['Genre']=='Sports'].copy()
print(sports_df.shape)
sports_df.head(3)
```

```
(3, 7)
```

[136]:

	Genre	region	Rank	sales	lat	lon	color
30	Sports	EU_Sales	17105195	371.34	54.53	15.26	#7F96FF
31	Sports	JP_Sales	17105195	134.76	36.20	138.25	#77FF94
32	Sports	NA_Sales	17105195	670.09	39.30	-94.71	#EF6F6C

```
[137]: # Create a map centered around the average latitude and longitude

my_map=folium.Map(location=MAP_CENTER, zoom_start=MAP_ZOOM)

# Loop through the DataFrame and add CircleMarkers to the map
for idx, row in sports_df.iterrows():
    folium.CircleMarker(
        location=[row['lat'], row['lon']],  # Set the location based on lat and lon
        radius=row['sales'] / 10,  # Size is based on the 'size' column, divided to scale it properly
        color=row['color'],  # Marker color
        fill=True,
        fill_color=row['color'],  # Fill color
        fill_opacity=0.6,
        # popup=f"Size: {row['size']}",  # Add popup with size info
    ).add_to(my_map)

my_map
```



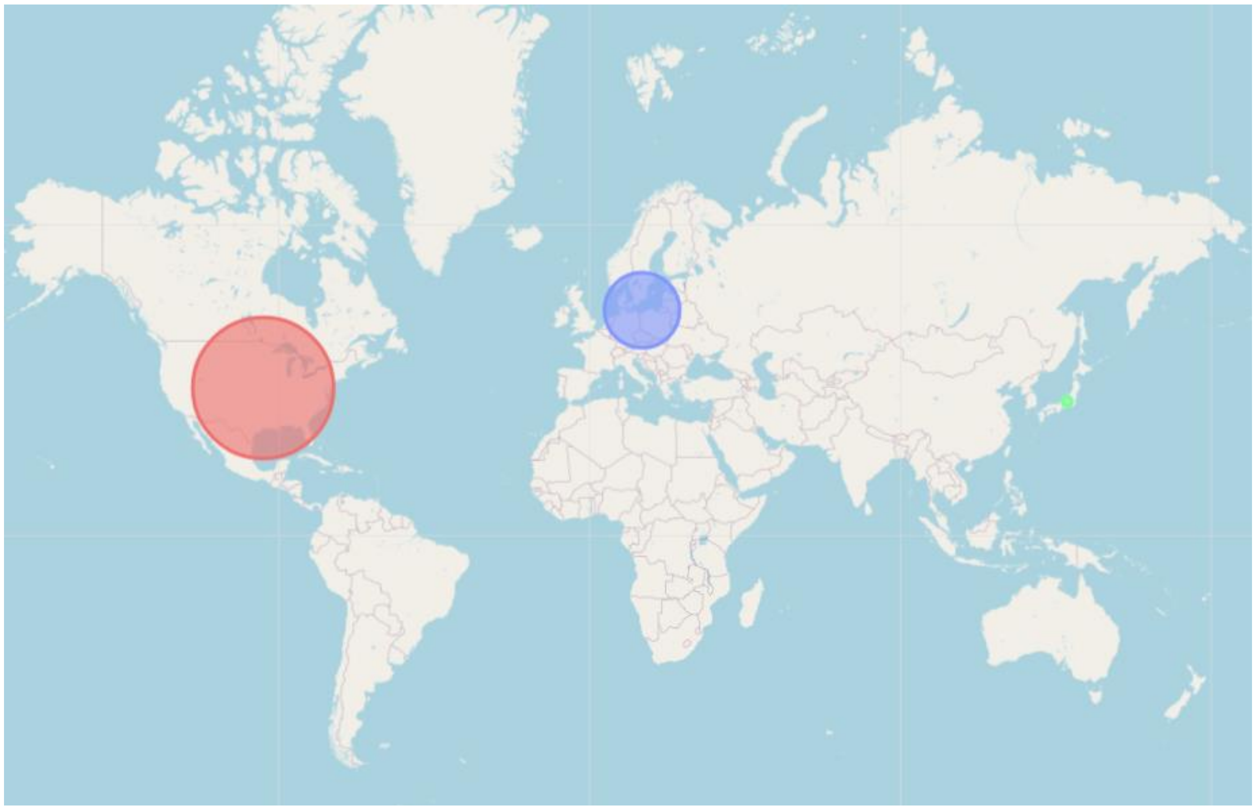
[138]:
```python
# Shooter sales data frame
shooter_df = total_genre_sales.loc[total_genre_sales['Genre']=='Shooter'].copy()
print(shooter_df.shape)
shooter_df.head(3)
```

```
(3, 7)
```

[138]:

	Genre	region	Rank	sales	lat	lon	color
24	Shooter	EU_Sales	9399409	310.45	54.53	15.26	#7F96FF
25	Shooter	JP_Sales	9399409	38.18	36.20	138.25	#77FF94
26	Shooter	NA_Sales	9399409	575.16	39.30	-94.71	#EF6F6C

[139]:
```python
# Create a map centered around the average latitude and longitude

my_map=folium.Map(location=MAP_CENTER, zoom_start=MAP_ZOOM)

# Loop through the DataFrame and add CircleMarkers to the map
for idx, row in shooter_df.iterrows():
    folium.CircleMarker(
        location=[row['lat'], row['lon']],  # Set the location based on lat and lon
        radius=row['sales'] / 10,  # Size is based on the 'size' column, divided to scale it properly
        color=row['color'],  # Marker color
        fill=True,
        fill_color=row['color'],  # Fill color
        fill_opacity=0.6,
        # popup=f"Size: {row['size']}",  # Add popup with size info
    ).add_to(my_map)

my_map
```



Correlation Between Years and Sales:

```
[161]: import pandas as pd
       import seaborn as sns
       import matplotlib.patches
       import matplotlib.pyplot as plt
       import numpy as np
       import warnings
       warnings.simplefilter(action='ignore', category=FutureWarning)
```

```
[162]: # File to Load
       data_filepath = ("vgsales_clean.csv")
       video_game_data = pd.read_csv(data_filepath)
```

```
[163]: #Read Video Games Sales Data File and store into Pandas DataFrames
       print(video_game_data.shape)
       video_game_data.head()
```

```
(16291, 11)
```

[163]:

	Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
0	1	Wii Sports	Wii	2006.0	Sports	Nintendo	41.49	29.02	3.77	8.46	82.74
1	2	Super Mario Bros.	NES	1985.0	Platform	Nintendo	29.08	3.58	6.81	0.77	40.24
2	3	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.85	12.88	3.79	3.31	35.82
3	4	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.75	11.01	3.28	2.96	33.00
4	5	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	1.00	31.37

```
[164]: Data_filepath= pd.DataFrame(video_game_data)
       Data_filepath.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16291 entries, 0 to 16290
Data columns (total 11 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Rank          16291 non-null  int64  
 1   Name          16291 non-null  object 
 2   Platform      16291 non-null  object 
 3   Year          16291 non-null  float64
 4   Genre         16291 non-null  object 
 5   Publisher     16291 non-null  object 
 6   NA_Sales      16291 non-null  float64
 7   EU_Sales      16291 non-null  float64
 8   JP_Sales      16291 non-null  float64
 9   Other_Sales   16291 non-null  float64
 10  Global_Sales  16291 non-null  float64
dtypes: float64(6), int64(1), object(4)
memory usage: 1.4+ MB
```

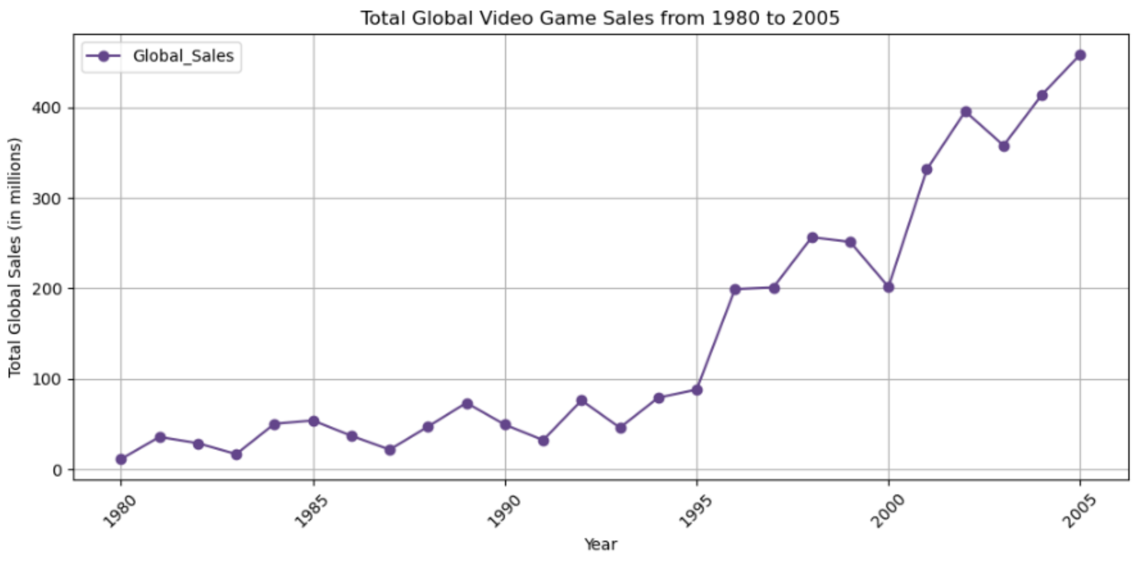
[165]:

```
total_platform_year = pd.DataFrame(video_game_data.groupby('Year')['Global_Sales'].sum().reset_index())
print(total_platform_year.shape)
print(total_platform_year)
```

```
(39, 2)
      Year  Global_Sales
0   1980.0         11.38
1   1981.0         35.77
2   1982.0         28.86
3   1983.0         16.79
4   1984.0         50.36
5   1985.0         53.94
6   1986.0         37.07
7   1987.0         21.74
8   1988.0         47.22
9   1989.0         73.45
10  1990.0         49.39
11  1991.0         32.23
12  1992.0         76.16
13  1993.0         45.98
14  1994.0         79.17
15  1995.0         88.11
16  1996.0        199.15
17  1997.0        200.98
18  1998.0        256.47
19  1999.0        251.27
20  2000.0        201.56
21  2001.0        331.47
22  2002.0        395.52
23  2003.0        357.85
24  2004.0        414.01
25  2005.0        458.51
26  2006.0        521.04
27  2007.0        609.92
28  2008.0        678.90
29  2009.0        667.30
30  2010.0        600.29
31  2011.0        515.80
32  2012.0        363.49
33  2013.0        368.11
34  2014.0        337.03
35  2015.0        264.44
36  2016.0         70.90
37  2017.0          0.05
38  2020.0          0.29
```

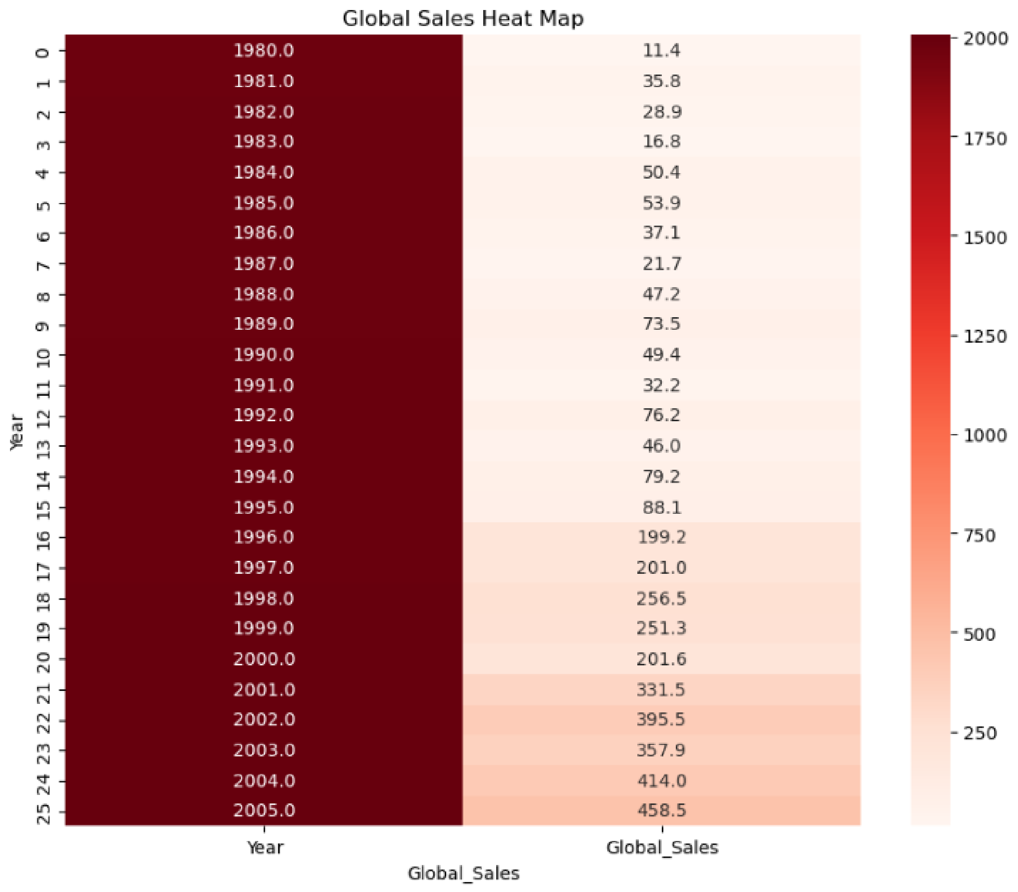
[166]:
```
# Filter for a specific range of years
start_year = 1980
end_year = 2005
filtered_sales = total_platform_year[(total_platform_year['Year'] >= start_year) & (total_platform_year['Year'] <= end_year)]
```

[167]:
```
# Line Chart for Global Sales By Year

plt.figure(figsize=(10, 5))
plt.plot(filtered_sales['Year'], filtered_sales['Global_Sales'], marker='o', color ='#63458A', label='Global_Sales')
plt.title(f'Total Global Video Game Sales from {start_year} to {end_year}')
plt.xlabel('Year')
plt.ylabel('Total Global Sales (in millions)')
plt.xticks(rotation=45)  # Rotate x-axis labels for better readability
plt.grid()
plt.legend()
plt.tight_layout()  # Adjust layout to make room for labels
plt.show()
```



```
[168]: # Filter for a specific range of years
       start_year = 1980
       end_year = 2005
       filtered_sales = total_platform_year[(total_platform_year['Year'] >= start_year) & (total_platform_year['Year'] <= end_year)]
```

```
[187]: # Heat map

       custom_palette = sns.color_palette(["#63458A", "#297373"])
       plt.figure(figsize=(10, 8)),(filtered_sales['Year'], filtered_sales['Global_Sales'])
       sns.heatmap(filtered_sales, annot=True, fmt=".1f", cmap="Reds")
       plt.title('Global Sales Heat Map')
       plt.xlabel('Global_Sales')
       plt.ylabel('Year')
       plt.show()
```



Work Cited:

Upadorprofzs. “Eda - Video Game Sales.” *Kaggle*, Kaggle, 21 July 2020, www.kaggle.com/code/upadorprofzs/eda-video-game-sales.

“Chatgpt.” ChatGPT, chatgpt.com/. Accessed 13 Dec. 2024,

https://chatgpt.com.