1.Introduction

Video games are popular all over the world. They are enjoyed by all ages. Video game industry is huge and the spending on video games per year is huge too. Sales of different types of games vary widely between countries due to local preferences. According to the market research firm SuperData, as of May 2015, the global games market was worth USD 74.2 billion. By region, North America accounted for 23.6 billion dollars, Asia for 23.1 billion dollars, Europe for 22.1 billion dollars and South America for 4.5 billion dollars. There are different genres, publisher and platforms for video games. This project relates to the sales of these video games based on different regions and analyzes the sales. Also I have analyzed which genre, platform or publisher is the most popular and has maximum number of sales.

2.Overview of the project

In this the main goal was to analyze the sales of video games in different regions. The regions are North America, Europe, Japan, other countries(comined) and then the global sales(total of all the regions). The main idea was to visualize the sales for different genres, publishers and platforms. This would give the basic idea about the most popular genres, publishers and platforms amongst all. Also analyzing the effect of genres on sales in different regions.

3.Data

For this project the data was collected from Kaggle(www.kaggle.com). This data gives us the idea about the sales of video games in different regions of the world. The distribution is with respect to genres, publishers and platforms.

Name: Name of the video game

Platform: Platform on which the game was released or is playable

Year: Year in which the game was released

Genre: Genre the game belongs to

Publisher: Name of the publisher who created the game

NA_Sales: Sales in North America

EU_Sales: Sales in Europe

JP_Sales: Sales in Japan

Other_Sales: Sales in other countries

Global_Sales: Global Sales

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sb

%matplotlib inline

import warnings
warnings.simplefilter("ignore")
```

In [2]:
```
df = pd.read_csv('vgsales.csv')
```

In [3]:
```
df.head(50)
```

Out[3]:

	Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sale
0	1	Wii Sports	Wii	2006.0	Sports	Nintendo	41.49	29.02	3.7
1	2	Super Mario Bros.	NES	1985.0	Platform	Nintendo	29.08	3.58	6.8
2	3	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.85	12.88	3.7
3	4	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.75	11.01	3.2
4	5	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.2
5	6	Tetris	GB	1989.0	Puzzle	Nintendo	23.20	2.26	4.2
6	7	New Super Mario Bros.	DS	2006.0	Platform	Nintendo	11.38	9.23	6.5
7	8	Wii Play	Wii	2006.0	Misc	Nintendo	14.03	9.20	2.9
8	9	New Super Mario Bros. Wii	Wii	2009.0	Platform	Nintendo	14.59	7.06	4.7
9	10	Duck Hunt	NES	1984.0	Shooter	Nintendo	26.93	0.63	0.2
10	11	Nintendogs	DS	2005.0	Simulation	Nintendo	9.07	11.00	1.9
11	12	Mario Kart DS	DS	2005.0	Racing	Nintendo	9.81	7.57	4.1
12	13	Pokemon Gold/Pokemon Silver	GB	1999.0	Role-Playing	Nintendo	9.00	6.18	7.2
13	14	Wii Fit	Wii	2007.0	Sports	Nintendo	8.94	8.03	3.6
14	15	Wii Fit Plus	Wii	2009.0	Sports	Nintendo	9.09	8.59	2.5
15	16	Kinect Adventures!	X360	2010.0	Misc	Microsoft Game Studios	14.97	4.94	0.2
16	17	Grand Theft Auto V	PS3	2013.0	Action	Take-Two Interactive	7.01	9.27	0.9
17	18	Grand Theft Auto: San Andreas	PS2	2004.0	Action	Take-Two Interactive	9.43	0.40	0.4
18	19	Super Mario World	SNES	1990.0	Platform	Nintendo	12.78	3.75	3.5
19	20	Brain Age: Train Your Brain in Minutes a Day	DS	2005.0	Misc	Nintendo	4.75	9.26	4.1
20	21	Pokemon Diamond/Pokemon Pearl	DS	2006.0	Role-Playing	Nintendo	6.42	4.52	6.0
21	22	Super Mario Land	GB	1989.0	Platform	Nintendo	10.83	2.71	4.1
22	23	Super Mario Bros. 3	NES	1988.0	Platform	Nintendo	9.54	3.44	3.8
23	24	Grand Theft Auto V	X360	2013.0	Action	Take-Two Interactive	9.63	5.31	0.0
24	25	Grand Theft Auto: Vice City	PS2	2002.0	Action	Take-Two Interactive	8.41	5.49	0.4

	Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sale
25	26	Pokemon Ruby/Pokemon Sapphire	GBA	2002.0	Role-Playing	Nintendo	6.06	3.90	5.3
26	27	Pokemon Black/Pokemon White	DS	2010.0	Role-Playing	Nintendo	5.57	3.28	5.6
27	28	Brain Age 2: More Training in Minutes a Day	DS	2005.0	Puzzle	Nintendo	3.44	5.36	5.3
28	29	Gran Turismo 3: A-Spec	PS2	2001.0	Racing	Sony Computer Entertainment	6.85	5.09	1.8
29	30	Call of Duty: Modern Warfare 3	X360	2011.0	Shooter	Activision	9.03	4.28	0.1
30	31	Pokémon Yellow: Special Pikachu Edition	GB	1998.0	Role-Playing	Nintendo	5.89	5.04	3.1
31	32	Call of Duty: Black Ops	X360	2010.0	Shooter	Activision	9.67	3.73	0.1
32	33	Pokemon X/Pokemon Y	3DS	2013.0	Role-Playing	Nintendo	5.17	4.05	4.3
33	34	Call of Duty: Black Ops 3	PS4	2015.0	Shooter	Activision	5.77	5.81	0.3
34	35	Call of Duty: Black Ops II	PS3	2012.0	Shooter	Activision	4.99	5.88	0.6
35	36	Call of Duty: Black Ops II	X360	2012.0	Shooter	Activision	8.25	4.30	0.0
36	37	Call of Duty: Modern Warfare 2	X360	2009.0	Shooter	Activision	8.52	3.63	0.0
37	38	Call of Duty: Modern Warfare 3	PS3	2011.0	Shooter	Activision	5.54	5.82	0.4
38	39	Grand Theft Auto III	PS2	2001.0	Action	Take-Two Interactive	6.99	4.51	0.3
39	40	Super Smash Bros. Brawl	Wii	2008.0	Fighting	Nintendo	6.75	2.61	2.6
40	41	Call of Duty: Black Ops	PS3	2010.0	Shooter	Activision	5.98	4.44	0.4
41	42	Animal Crossing: Wild World	DS	2005.0	Simulation	Nintendo	2.55	3.52	5.3
42	43	Mario Kart 7	3DS	2011.0	Racing	Nintendo	4.74	3.91	2.6
43	44	Halo 3	X360	2007.0	Shooter	Microsoft Game Studios	7.97	2.83	0.1
44	45	Grand Theft Auto V	PS4	2014.0	Action	Take-Two Interactive	3.80	5.81	0.3

	Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sale
45	46	Pokemon HeartGold/Pokemon SoulSilver	DS	2009.0	Action	Nintendo	4.40	2.77	3.9
46	47	Super Mario 64	N64	1996.0	Platform	Nintendo	6.91	2.85	1.9
47	48	Gran Turismo 4	PS2	2004.0	Racing	Sony Computer Entertainment	3.01	0.01	1.1
48	49	Super Mario Galaxy	Wii	2007.0	Platform	Nintendo	6.16	3.40	1.2
49	50	Pokemon Omega Ruby/Pokemon Alpha Sapphire	3DS	2014.0	Role-Playing	Nintendo	4.23	3.37	3.0

In [4]:
```
df.shape
```

Out[4]:
```
(16598, 11)
```

In [5]:
```
df.describe()
```

Out[5]:

	Rank	Year	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_S
count	16598.000000	16327.000000	16598.000000	16598.000000	16598.000000	16598.000000	16598.000
mean	8300.605254	2006.406443	0.264667	0.146652	0.077782	0.048063	0.537
std	4791.853933	5.828981	0.816683	0.505351	0.309291	0.188588	1.555
min	1.000000	1980.000000	0.000000	0.000000	0.000000	0.000000	0.010
25%	4151.250000	2003.000000	0.000000	0.000000	0.000000	0.000000	0.060
50%	8300.500000	2007.000000	0.080000	0.020000	0.000000	0.010000	0.170
75%	12449.750000	2010.000000	0.240000	0.110000	0.040000	0.040000	0.470
max	16600.000000	2020.000000	41.490000	29.020000	10.220000	10.570000	82.740

In [6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16598 entries, 0 to 16597
Data columns (total 11 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Rank          16598 non-null  int64  
 1   Name          16598 non-null  object 
 2   Platform      16598 non-null  object 
 3   Year          16327 non-null  float64
 4   Genre         16598 non-null  object 
 5   Publisher     16540 non-null  object 
 6   NA_Sales      16598 non-null  float64
 7   EU_Sales      16598 non-null  float64
 8   JP_Sales      16598 non-null  float64
 9   Other_Sales   16598 non-null  float64
 10  Global_Sales  16598 non-null  float64
dtypes: float64(6), int64(1), object(4)
memory usage: 1.4+ MB
```

In [8]:
```
sorted(list(df.Year.unique()))
```

Out[8]:
```
[1980.0,
 1981.0,
 1982.0,
 1983.0,
 1984.0,
 1985.0,
 1986.0,
 1987.0,
 1988.0,
 1989.0,
 1990.0,
 1991.0,
 1992.0,
 1993.0,
 1994.0,
 1995.0,
 1996.0,
 1997.0,
 1998.0,
 1999.0,
 2000.0,
 2001.0,
 2002.0,
 2003.0,
 2004.0,
 2005.0,
 2006.0,
 2007.0,
 2008.0,
 2009.0,
 2010.0,
 2011.0,
 2012.0,
 2013.0,
 2014.0,
 2015.0,
 nan,
 2016.0,
 2017.0,
 2020.0]
```

In [9]:
```
df.Genre.value_counts()
```

Out[9]:
```
Action          3316
Sports          2346
Misc            1739
Role-Playing    1488
Shooter         1310
Adventure       1286
Racing          1249
Platform         886
Simulation       867
Fighting         848
Strategy         681
Puzzle           582
Name: Genre, dtype: int64
```

In [10]:
```
df['Global_Sales'].min()
```

Out[10]:
```
0.01
```

In [11]:
```
test = df[df['Global_Sales']>0.01]
test[test['Global_Sales'] != test['NA_Sales']+test['EU_Sales']+test['JP_Sales']+test[
```

Out[11]:
```
6761
```

In [12]:
```
df[df.duplicated()].shape[0]
```

Out[12]:
```
0
```

In [13]:
```
df_copy = df.copy()
```

In [14]:
```
df_copy.dropna(axis=0, how='any',inplace=True)
```

In [15]:
```
df_copy.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 16291 entries, 0 to 16597
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: 1.5+ MB
```

In [16]:
```
df_copy['Global_Sales'] = df_copy['NA_Sales'] + df_copy['EU_Sales'] + df_copy['JP_Sale
```

In [17]:
```
df_copy[df_copy['Global_Sales'] != df_copy['NA_Sales']+df_copy['EU_Sales']+df_copy['JP
```

Out[17]:
```
(0, 11)
```

In [18]:
```
# store the file
df_copy.reset_index(drop=True)
df_copy.to_csv('clean_vgsales.csv')
```

In [19]:
```
#Load data
clean_df = pd.read_csv('clean_vgsales.csv')
```

In [20]:
```
clean_df['Publisher'].value_counts().describe()
```

Out[20]:
```
count     576.000000
mean       28.282986
std       115.417374
min         1.000000
25%         1.000000
50%         3.000000
75%        10.000000
max      1339.000000
Name: Publisher, dtype: float64
```

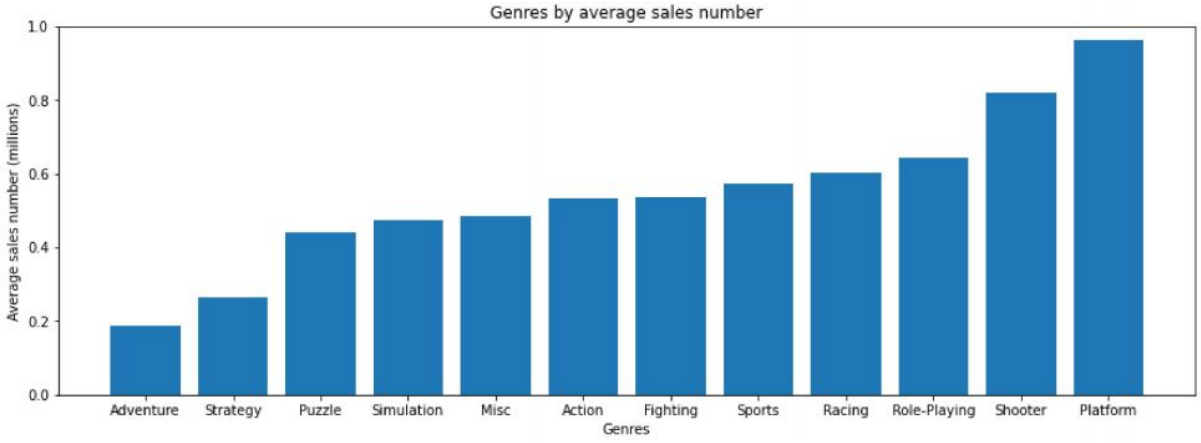
In [21]:
```
# find the list of the publishers who had published more than 3 games
publishers_more_than_3_list = list(clean_df['Publisher'].value_counts()[clean_df['Publ
```

In [22]:
```
# I only need the data who are from these publishers
clean_df = clean_df[clean_df['Publisher'].isin(publishers_more_than_3_list)]
```

In [23]:
```
# let's see the average Global_Sales by genres
clean_df.groupby('Genre')['Global_Sales'].mean().sort_values()
```

Out[23]:
```
Genre
Adventure       0.187558
Strategy        0.265721
Puzzle          0.441418
Simulation      0.475684
Misc            0.483500
Action          0.534082
Fighting        0.537800
Sports          0.574259
Racing          0.600978
Role-Playing    0.641563
Shooter         0.819281
Platform        0.963213
Name: Global_Sales, dtype: float64
```

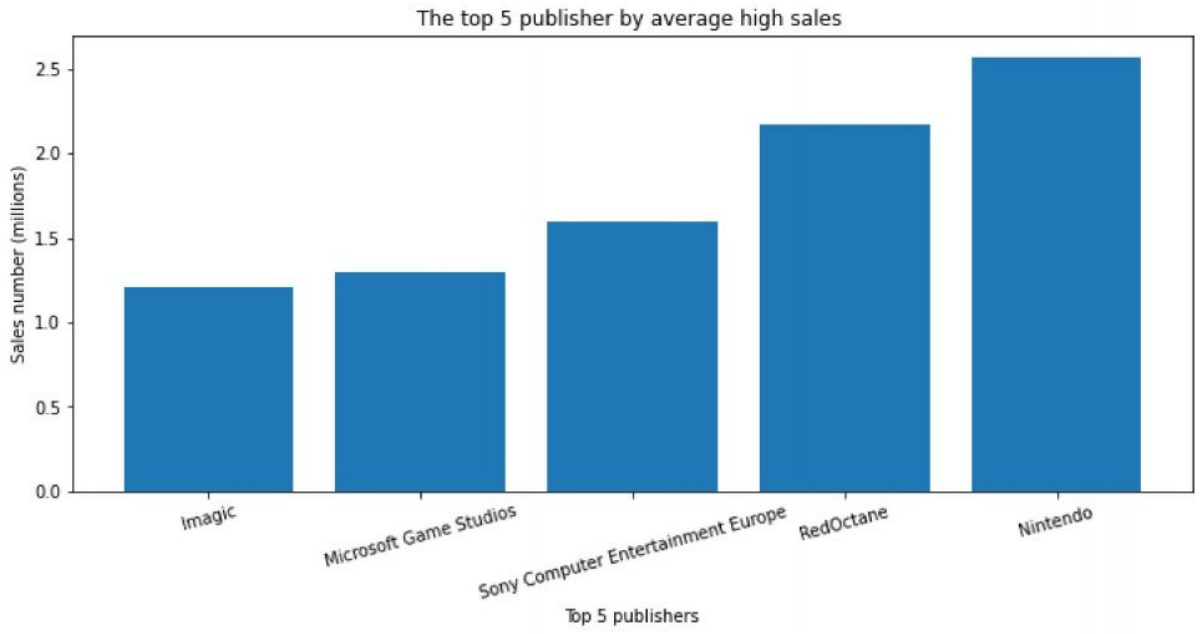
In [24]:
```
# Visulization
plt.figure(figsize=(15,5))
plt.bar(range(0,12), list(clean_df.groupby('Genre')['Global_Sales'].mean().sort_values
plt.title('Genres by average sales number')
plt.xlabel('Genres')
plt.ylabel('Average sales number (millions)');
plt.ylim(0,1);
```



In [25]:
```
# let's see the average Global_Sales by publishers
clean_df.groupby('Publisher')['Global_Sales'].mean().sort_values()[-5:]
```

Out[25]:
```
Publisher
Imagic                                  1.205000
Microsoft Game Studios                  1.300423
Sony Computer Entertainment Europe      1.592000
RedOctane                               2.172500
Nintendo                                2.563549
Name: Global_Sales, dtype: float64
```

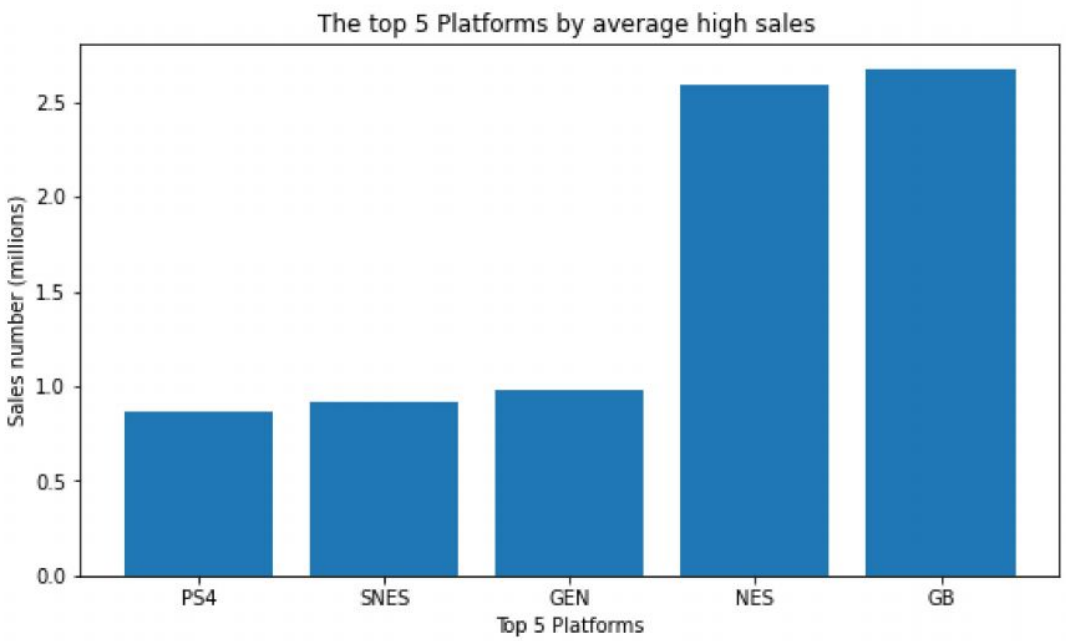
In [26]:
```
#Visualization
plt.figure(figsize=(12,5))
plt.bar([1, 2, 3,4,5], [1.205, 1.30042328, 1.592 , 2.1725, 2.56354885], tick_label=['
       'Sony Computer Entertainment Europe', 'RedOctane', 'Nintendo'])
plt.title('The top 5 publisher by average high sales ')
plt.xlabel('Top 5 publishers')
plt.ylabel('Sales number (millions)');
plt.xticks(rotation=15);
```



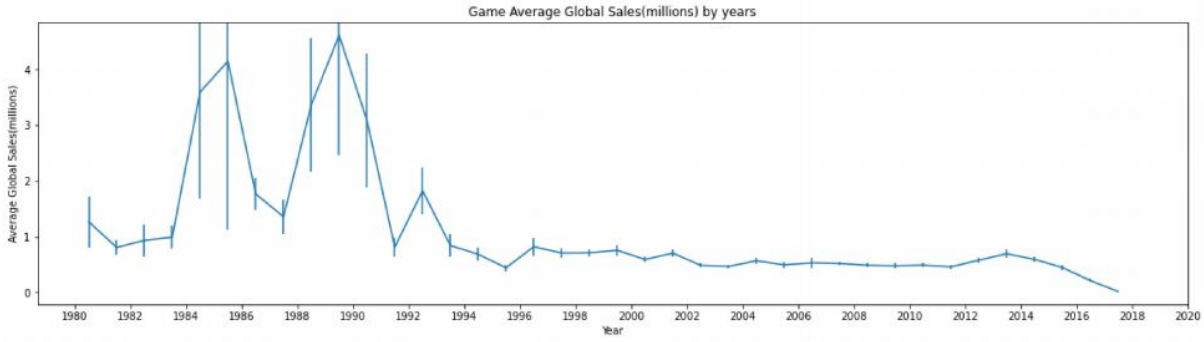
In [27]:
```
# let's see the average Global_Sales by platforms
clean_df.groupby('Platform')['Global_Sales'].mean().sort_values()[-5:].values
```

Out[27]:
```
array([0.8608805 , 0.92192488, 0.97875   , 2.59473684, 2.67357895])
```

In [28]:
```
#Visualization
plt.figure(figsize=(9,5))
plt.bar([1, 2, 3,4,5], [0.8608805, 0.92192488, 0.97875, 2.59473684, 2.67357895], tick_
plt.title('The top 5 Platforms by average high sales ')
plt.xlabel('Top 5 Platforms')
plt.ylabel('Sales number (millions)');
```



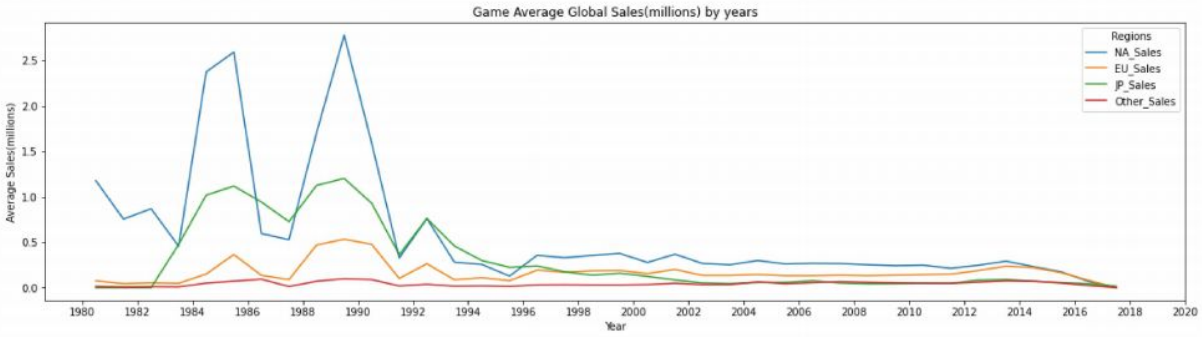
In [29]:
```
# Visualization

plt.figure(figsize=(20,5))
# set bin edges, compute centers
bin_size = 1
xbin_edges = np.arange(1980, clean_df['Year'].max()+bin_size, bin_size)
xbin_centers = (xbin_edges + bin_size/2)[:-1]

# compute statistics in each bin
data_xbins = pd.cut(clean_df['Year'], xbin_edges, right = False, include_lowest = True
y_means = clean_df['Global_Sales'].groupby(data_xbins).mean()
y_sems = clean_df['Global_Sales'].groupby(data_xbins).sem()  #std

# plot the summarized data
plt.errorbar(x = xbin_centers, y = y_means, yerr = y_sems)
plt.xlabel('Year')
plt.ylabel('Average Global Sales(millions)');
plt.xticks(range(1980,2021,2), range(1980,2021,2));
plt.title('Game Average Global Sales(millions) by years');
```



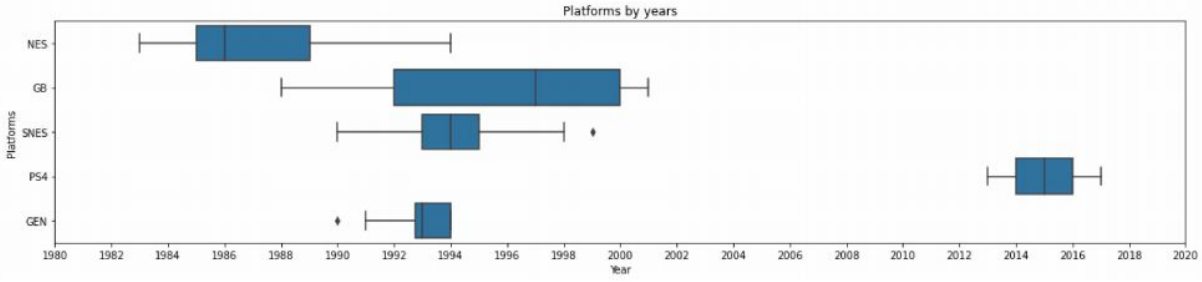
In [30]:
```
# Visualization

plt.figure(figsize=(20,5))
# set bin edges, compute centers
bin_size = 1
xbin_edges = np.arange(1980, clean_df['Year'].max()+bin_size, bin_size)
```

```
xbin_centers = (xbin_edges + bin_size/2)[:-1]

# compute statistics in each bin
data_xbins = pd.cut(clean_df['Year'], xbin_edges, right = False, include_lowest = True
y_means = clean_df['NA_Sales'].groupby(data_xbins).mean()
plt.errorbar(x = xbin_centers, y = y_means)
y_means = clean_df['EU_Sales'].groupby(data_xbins).mean()
plt.errorbar(x = xbin_centers, y = y_means)
y_means = clean_df['JP_Sales'].groupby(data_xbins).mean()
plt.errorbar(x = xbin_centers, y = y_means)
y_means = clean_df['Other_Sales'].groupby(data_xbins).mean()
plt.errorbar(x = xbin_centers, y = y_means)

# plot the summarized data
plt.xlabel('Year')
plt.ylabel('Average Sales(millions)');
plt.xticks(range(1980,2021,2), range(1980,2021,2));
plt.title('Game Average Global Sales(millions) by years');
plt.legend(title="Regions", labels=['NA_Sales','EU_Sales','JP_Sales','Other_Sales'])
```

Out[30]:
```
<matplotlib.legend.Legend at 0x1f51f284dc0>
```



In [31]:
```
# Visualization
#I only need the data from top platform
Top5platform_df = clean_df[clean_df['Platform'].isin(['PS4', 'SNES', 'GEN', 'NES', 'GB
plt.figure(figsize=(20,4))
base_color = sb.color_palette()[0]
sb.boxplot(data = Top5platform_df, x = 'Year', y = 'Platform', color = base_color)
plt.xlabel('Year')
plt.ylabel('Platforms');
plt.xticks(range(1980,2021,2), range(1980,2021,2));
plt.title('Platforms by years');
```



In [32]:
```
# import statmodel to analyze the relationship between independent variables and deper
import statsmodels.api as sm;
```

In [33]:
```
df_new = clean_df.copy()
```

4.Results

From the above vizualizations we can clearly say that DC and Play Station are the most popular platforms amongst all followed by xbox. Action genre is the most popular genre of all and is followed by sports and fighting respectively. We can also see that Daito is the most popular followed by TYO and Miwasa respectively.

From the above tests we can say that the genres less popular cause significant change in sales in all regions as compared to the ones that are more popular.

5.Conclusion

By the above data we can say that action games on DC or playstation for that matter are the most popular and are the ones responsible for maximum sales all over the globe. Also as these games are so abundant and popular(ranking wise), variation in the sales of one or two such games would not cause significant change in the overall sales.