Data Bootcamp - **Final Project**

Team Members

Lastname, Firstname - Net ID

- Singh, Avantika - as13594
- Stokely, William - wcs307
- Bansal, Anmol - ab8778

Project Overview

The term "video game" was developed to distinguish this class of electronic games that were played to some type of video display. Video games are defined based on their platform, which include arcade games, console games, and personal computer (PC) games. More recently, the industry has expanded onto mobile gaming through smartphones and tablet computers, virtual and augmented reality systems, and remote cloud gaming. Video games are classified into a wide range of genres based on their type of gameplay and purpose.

Since the 2010s, the commercial importance of the video game industry has been increasing. The global market size of the thriving video game industry has more than doubled over the past decade and now easily exceeds one hundred billion dollars.

To help video game developers and publishers alike navigate this fast evolving landscape, **Avanmoliam Partners (Avantika, Anmol, & William)** provides consulting services to these entities.

The dataset that we have used for our analysis has data scrapped from Video Game Charts, which is a business intelligence and research firm that publishes sales data for video games. The data is further combined with ratings data from Metacritic. Unfortunately, there are some missing observations (which we will be exploring in the data cleaning section) as Metacritic covers only a subset of the gaming platforms in the dataset.

Use Case : Our use case for this data is two-fold:

1. Advise the Publishers on choosing the best platforms to bring their particular game to market, given the genre and target market-region of the video games they create.

1. Advise the game publishers on which genre's they should prioritize to diversify their games portfolio.

Importing Python Libraries

In [2]:
```
!pip install geopandas
```

In [3]:
```
!pip install wordcloud
```

```
Requirement already satisfied: wordcloud in c:\users\avantika\anaconda3\lib\site-package
s (1.8.1)
Requirement already satisfied: numpy>=1.6.1 in c:\users\avantika\anaconda3\lib\site-pack
ages (from wordcloud) (1.20.3+vanilla)
Requirement already satisfied: pillow in c:\users\avantika\anaconda3\lib\site-packages
(from wordcloud) (8.0.1)
Requirement already satisfied: matplotlib in c:\users\avantika\anaconda3\lib\site-packag
es (from wordcloud) (3.3.2)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 in c:\users\avan
tika\anaconda3\lib\site-packages (from matplotlib->wordcloud) (2.4.7)
```

```
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\avantika\anaconda3\lib\site
-packages (from matplotlib->wordcloud) (1.3.0)
Requirement already satisfied: certifi>=2020.06.20 in c:\users\avantika\anaconda3\lib\si
te-packages (from matplotlib->wordcloud) (2020.6.20)
Requirement already satisfied: cycler>=0.10 in c:\users\avantika\anaconda3\lib\site-pack
ages (from matplotlib->wordcloud) (0.10.0)
Requirement already satisfied: python-dateutil>=2.1 in c:\users\avantika\anaconda3\lib\s
ite-packages (from matplotlib->wordcloud) (2.8.1)
Requirement already satisfied: six in c:\users\avantika\anaconda3\lib\site-packages (fro
m cycler>=0.10->matplotlib->wordcloud) (1.16.0)
```

In [4]:
```
!pip install mapclassify
```

```
Requirement already satisfied: mapclassify in c:\users\avantika\anaconda3\lib\site-packa
ges (2.4.2)
Requirement already satisfied: numpy>=1.3 in c:\users\avantika\anaconda3\lib\site-packag
es (from mapclassify) (1.20.3+vanilla)
Requirement already satisfied: pandas>=1.0 in c:\users\avantika\anaconda3\lib\site-packa
ges (from mapclassify) (1.2.4)
Requirement already satisfied: scikit-learn in c:\users\avantika\anaconda3\lib\site-pack
ages (from mapclassify) (0.23.2)
Requirement already satisfied: scipy>=1.0 in c:\users\avantika\anaconda3\lib\site-packag
es (from mapclassify) (1.5.2)
Requirement already satisfied: networkx in c:\users\avantika\anaconda3\lib\site-packages
(from mapclassify) (2.5)
Requirement already satisfied: pytz>=2017.3 in c:\users\avantika\anaconda3\lib\site-pack
ages (from pandas>=1.0->mapclassify) (2020.1)
Requirement already satisfied: python-dateutil>=2.7.3 in c:\users\avantika\anaconda3\lib
\site-packages (from pandas>=1.0->mapclassify) (2.8.1)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\avantika\anaconda3\lib\s
ite-packages (from scikit-learn->mapclassify) (2.1.0)
Requirement already satisfied: joblib>=0.11 in c:\users\avantika\anaconda3\lib\site-pack
ages (from scikit-learn->mapclassify) (0.17.0)
Requirement already satisfied: decorator>=4.3.0 in c:\users\avantika\anaconda3\lib\site-
packages (from networkx->mapclassify) (4.4.2)
Requirement already satisfied: six>=1.5 in c:\users\avantika\anaconda3\lib\site-packages
(from python-dateutil>=2.7.3->pandas>=1.0->mapclassify) (1.16.0)
```

In [5]:
```
!pip install plotly
```

```
Requirement already satisfied: plotly in c:\users\avantika\anaconda3\lib\site-packages
(4.14.3)
Requirement already satisfied: retrying>=1.3.3 in c:\users\avantika\anaconda3\lib\site-p
ackages (from plotly) (1.3.3)
Requirement already satisfied: six in c:\users\avantika\anaconda3\lib\site-packages (fro
m plotly) (1.16.0)
```

In [6]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.ticker as tick
import plotly.express as px
import seaborn as sns

from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split,cross_val_score
from sklearn.linear_model import LogisticRegression
from sklearn import metrics
from sklearn.metrics import roc_curve
from sklearn.metrics import confusion_matrix,accuracy_score

import geopandas as gpd
from wordcloud import WordCloud, ImageColorGenerator
```

In [7]:
```
# Setting display options of the dataframe
pd.options.display.width=None
pd.options.display.max_columns = None

# Setting Pandas scientific notation for floats in Python
pd.set_option('display.float_format', lambda x: '%.5f' % x)
```

Data Loading

In [8]:
```
path="https://github.com/GitHub847362/2021SpringDataBootcamp/raw/main/Video_Games_Sales_
df=pd.read_csv(path)
df
```

Out[8]:

	Name	Platform	Year_of_Release	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales
0	Wii Sports	Wii	2006.00000	Sports	Nintendo	41.36000	28.96000	3.77000
1	Super Mario Bros.	NES	1985.00000	Platform	Nintendo	29.08000	3.58000	6.81000
2	Mario Kart Wii	Wii	2008.00000	Racing	Nintendo	15.68000	12.76000	3.79000
3	Wii Sports Resort	Wii	2009.00000	Sports	Nintendo	15.61000	10.93000	3.28000
4	Pokemon Red/Pokemon Blue	GB	1996.00000	Role-Playing	Nintendo	11.27000	8.89000	10.22000
...	...	...	...	...	...	...	...	...
16714	Samurai Warriors: Sanada Maru	PS3	2016.00000	Action	Tecmo Koei	0.00000	0.00000	0.01000
16715	LMA Manager 2007	X360	2006.00000	Sports	Codemasters	0.00000	0.01000	0.00000
16716	Haitaka no Psychedelica	PSV	2016.00000	Adventure	Idea Factory	0.00000	0.00000	0.01000
16717	Spirits & Spells	GBA	2003.00000	Platform	Wanadoo	0.01000	0.00000	0.00000
16718	Winning Post 8 2016	PSV	2016.00000	Simulation	Tecmo Koei	0.00000	0.00000	0.01000

16719 rows × 16 columns

Metadata

The dataset include 16 features that are described as follows

1. **Name** - The games name
2. **Platform** - Platform of the games release (i.e. PC,PS4, etc.)
3. **Year** - Year of the game's release

4. **Genre** - Genre of the game
5. **Publisher** - Publisher of the game
6. **NA_Sales** - Sales in North America (in millions)
7. **EU_Sales** - Sales in Europe (in millions)
8. **JP_Sales** - Sales in Japan (in millions)
9. **Other_Sales** - Sales in the rest of the world (in millions)
10. **Global_Sales** - Total worldwide sales.
11. **Critic_score** - Aggregate score compiled by Metacritic staff
12. **Critic_count** - The number of critics used in coming up with the Critic_score
13. **User_score** - Score by Metacritic's subscribers
14. **User_count** - Number of users who gave the user_score
15. **Developer** - Party responsible for creating the game
16. **Rating** - The ESRB ratings

In [9]:
```
df.columns
```

Out[9]:
```
Index(['Name', 'Platform', 'Year_of_Release', 'Genre', 'Publisher', 'NA_Sales',
       'EU_Sales', 'JP_Sales', 'Other_Sales', 'Global_Sales', 'Critic_Score',
       'Critic_Count', 'User_Score', 'User_Count', 'Developer', 'Rating'],
      dtype='object')
```

In [10]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16719 entries, 0 to 16718
Data columns (total 16 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   Name             16717 non-null  object 
 1   Platform         16719 non-null  object 
 2   Year_of_Release  16450 non-null  float64
 3   Genre            16717 non-null  object 
 4   Publisher        16665 non-null  object 
 5   NA_Sales         16719 non-null  float64
 6   EU_Sales         16719 non-null  float64
 7   JP_Sales         16719 non-null  float64
 8   Other_Sales      16719 non-null  float64
 9   Global_Sales     16719 non-null  float64
 10  Critic_Score     8137 non-null   float64
 11  Critic_Count     8137 non-null   float64
 12  User_Score       10015 non-null  object 
 13  User_Count       7590 non-null   float64
 14  Developer        10096 non-null  object 
 15  Rating           9950 non-null   object 
dtypes: float64(9), object(7)
memory usage: 2.0+ MB
```

In [11]:
```
# Categorial variables
df.select_dtypes([np.object]).columns
```

```
<ipython-input-11-94be335111e2>:2: DeprecationWarning: `np.object` is a deprecated alias
for the builtin `object`. To silence this warning, use `object` by itself. Doing this wi
ll not modify any behavior and is safe.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/relea
se/1.20.0-notes.html#deprecations
  df.select_dtypes([np.object]).columns
```

Out[11]:
```
Index(['Name', 'Platform', 'Genre', 'Publisher', 'User_Score', 'Developer',
       'Rating'],
```

```
      dtype='object')
```

In [12]:
```
# Numerical Variables
df.select_dtypes([np.float64]).columns
```

Out[12]:
```
Index(['Year_of_Release', 'NA_Sales', 'EU_Sales', 'JP_Sales', 'Other_Sales',
       'Global_Sales', 'Critic_Score', 'Critic_Count', 'User_Count'],
      dtype='object')
```

Data Cleaning and Filtering

1. Handling NaN/missing values in data

In [13]:
```
# Modify the dataset to replace 'empty' values everywhere in the dataset with NaN (np.N
df = df.replace('empty',np.NaN)
```

In [14]:
```
# Column wise summary of nulls (or NaN)
df.isna().sum()
```

Out[14]:
```
Name                 2
Platform             0
Year_of_Release    269
Genre                2
Publisher           54
NA_Sales             0
EU_Sales             0
JP_Sales             0
Other_Sales          0
Global_Sales         0
Critic_Score      8582
Critic_Count      8582
User_Score        6704
User_Count        9129
Developer         6623
Rating            6769
dtype: int64
```

We observe that there is approximately 30% of data missing in features such as Critic_Score, Critic_Count, User_Score and User_Count. This may have been due to a problem in the ETL process or a data collection issue. So inorder to be able to conduct a more precise data analysis we begin with dealing with NAN values .

In [15]:
```
#Drop the two missing values from NAME and Genre
df = df.dropna(axis='index', how='all', subset=['Name'])
df = df.dropna(axis='index', how='all', subset=['Genre'])
```

In [16]:
```
#Drop the 269 rows with missing values for Year_of_Release
df = df.dropna(axis='index', how='all', subset=['Year_of_Release'])
```

In [17]:
```
#Drop the 54 rows with missing values for Publisher
df = df.dropna(axis='index', how='all', subset=['Publisher'])
```

2. Dropping columns with more that 30% missing data.

In [18]:
```
# Removing columns: Critic_Score, Critic_Count, User_Score, User_Count, Developer, Rati

cols = ['Critic_Score', 'Critic_Count', 'User_Score', 'User_Count', 'Developer', 'Ratin
print('(Rows,Cols) Before dropping columns :', df.shape)
```

```
df = df.drop(cols, axis=1)
print('(Rows,Cols) After dropping columns :', df.shape)
```

```
(Rows,Cols) Before dropping columns : (16416, 16)
(Rows,Cols) After dropping columns : (16416, 10)
```

3. Removing duplicate values.

In [19]:
```
# Removing duplicate lines from the data set
df = df.drop_duplicates(subset=None, keep='first', inplace=False)
```

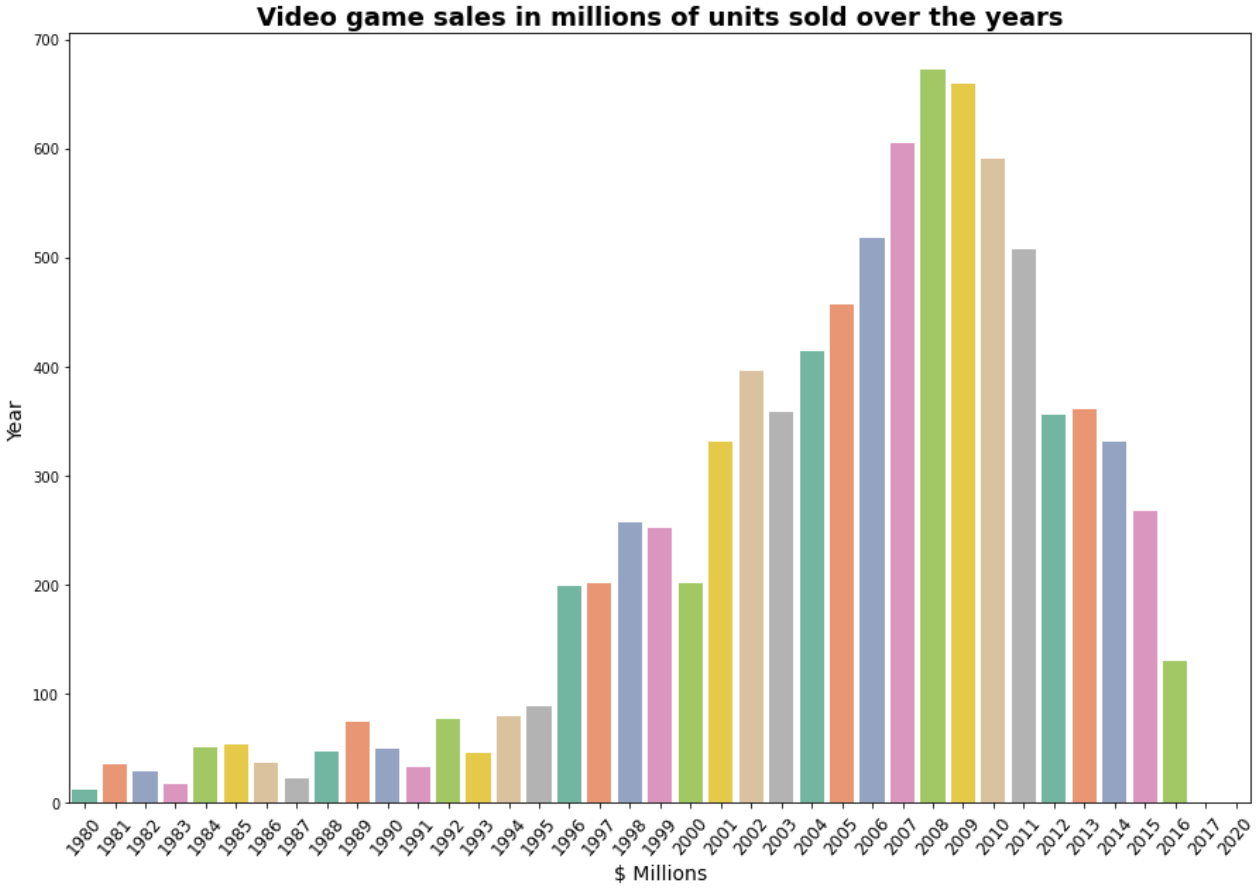
4. Altering the existing column datatypes.

In [20]:
```
# We observe that the Year_of_Release is coming in as float64; instead, we convert to i
df.Year_of_Release = df.Year_of_Release.astype(int)
```

Data Exploration and Analysis

Let's begin with looking at the trends in video game sales over the years

In [21]:
```
sales_over_years = df.groupby(['Year_of_Release']).sum()
x = sales_over_years.index
y = sales_over_years['Global_Sales']
plt.figure(figsize=(15,10))
ax = sns.barplot(x = x,y = y, palette="Set2")
ax.set_xlabel(xlabel='$ Millions', fontsize=14)
ax.set_xticklabels(labels = x, fontsize=12, rotation=50)
ax.set_ylabel(ylabel='Year', fontsize=14)
ax.set_title(label='Video game sales in millions of units sold over the years', fontsiz
plt.show();
```



Rise-and-Fall : Reviewing the graph above, we see over the past several decades that video game unit sales have risen dramatically when comparing games released in the 1980s to those released into the late 2000s. We see that the late 1990s saw a large increase in the number of games released per year. However, unit sales appear to have peaked in 2008 and have since declined down toward pre-2000 levels. This raises some interesting questions which must be considered:

1. **What led to the explosion in the late 1990s ?** We believe that one of the most important changes that occurred during the mid-90's was the transition from raster graphics to 3-D graphics. This gave rise to genres which previously did not exist, like first-person shooter, real-time strategy, and MMO. Consoles such as the first Playstation and Nintendo 64 sold heavily during this time period and ushered in a more modern form of video games, with 3-D graphics and processing that previous consoles were unable to perform. While certainly not exhaustive, other important reasons for the increase in the late 90s include a decline in popularity of arcades and expanded investment by major investors.

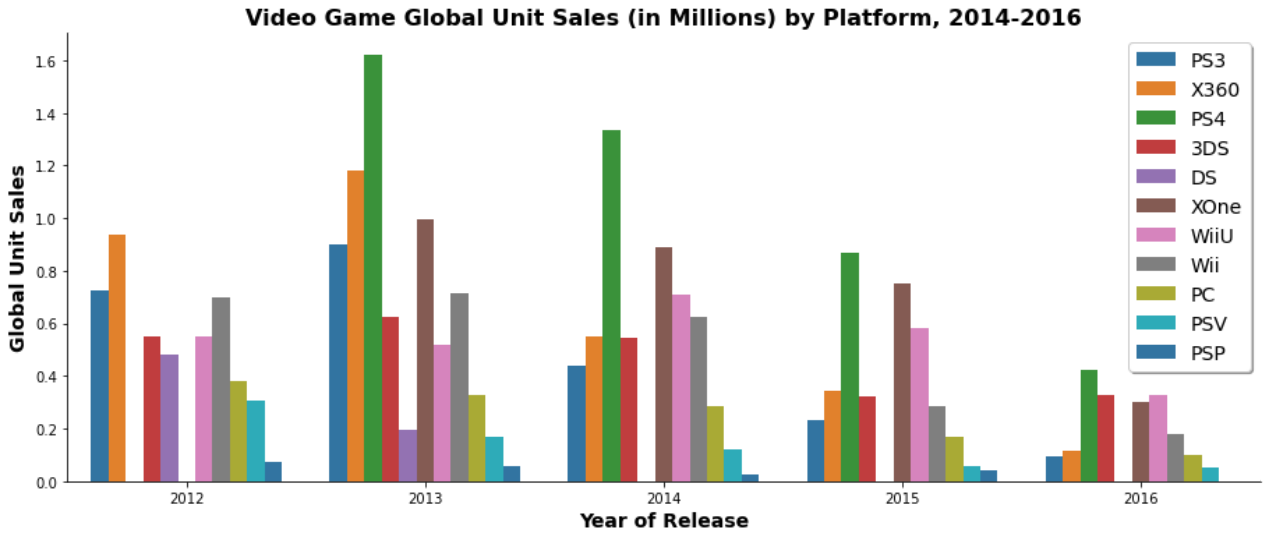
2. **What led to the decline starting in the late 2000s?** Analysis of the video-game market trends show us that the decline is observed one year after the debut of Apple's iPhone, as mobile gaming has risen in prominence (Android and iOS games are not captured in this dataset).

Additional references : History of Gaming

Observing the trends in video game unit sales across various gaming platforms

In [22]:
```
platform_sales=df.loc[(df['Year_of_Release']>2011) & (df['Year_of_Release']<2017)]
plt.figure(figsize=(20,15))
sns.catplot(x="Year_of_Release",y="Global_Sales",kind="bar",
            data = platform_sales,
            hue="Platform",
            palette="tab10",height=5,aspect=2.5,
            ci=None,
            legend=False)
plt.xlabel('Year of Release', fontsize=14, weight='bold')
plt.ylabel('Global Unit Sales', fontsize=14, weight='bold')
plt.title('Video Game Global Unit Sales (in Millions) by Platform, 2014-2016', fontsize
plt.legend(fontsize = 14,
           shadow = True,
           facecolor = 'white');
plt.show()
```

```
<Figure size 1440x1080 with 0 Axes>
```



Three Shrinking Giants : Viewing things on a platform-specific breakdown in the multi-bar graph above, then looking at the most recent 3 years of the dataset for games released during 2014-2016, we see that Sony, Nintendo, and Microsoft all experienced steep declines in year-over-year worldwide unit sales, particularly among their most recent generation of non-handheld platforms (e.g. PlayStayion 4, Xbox One, and Nintendo Wii).

In fact, Nintendo's Wii by 2016 has actually sunk below Nintendo's handheld platform, the 3Ds, in terms of unit sales. While the 3DS has shown resilience in the face of smartphone gaming market growth, Sony's handheld devices have all but fallen off of the map (with no data even reported for the Sony PSP in 2016). PC sales, a small portion of the market, continue to steadily decrease as well.

To video game developers, this suggests that depending on their skill sets, if they are to invest their efforts anywhere, they may wish to invest less in handheld-oriented games (except perhaps for the 3DS which has held somewhat firm) and less in PC games.

Let's dive deeper and look at top-video game genre with the maximum sales for each year over the past two decades :

A video game genre is a category of games that share similar gameplay characteristics. Video game genres have little to do with the storyline or setting itself, but how the player exists within that world. For example, simulation video gaming places players in a virtual world that mimics aspects of the real world, whereas first-person shooter (FPS) games are usually defined by their first-person perspective and battle-heavy gameplay using long-range weaponry.

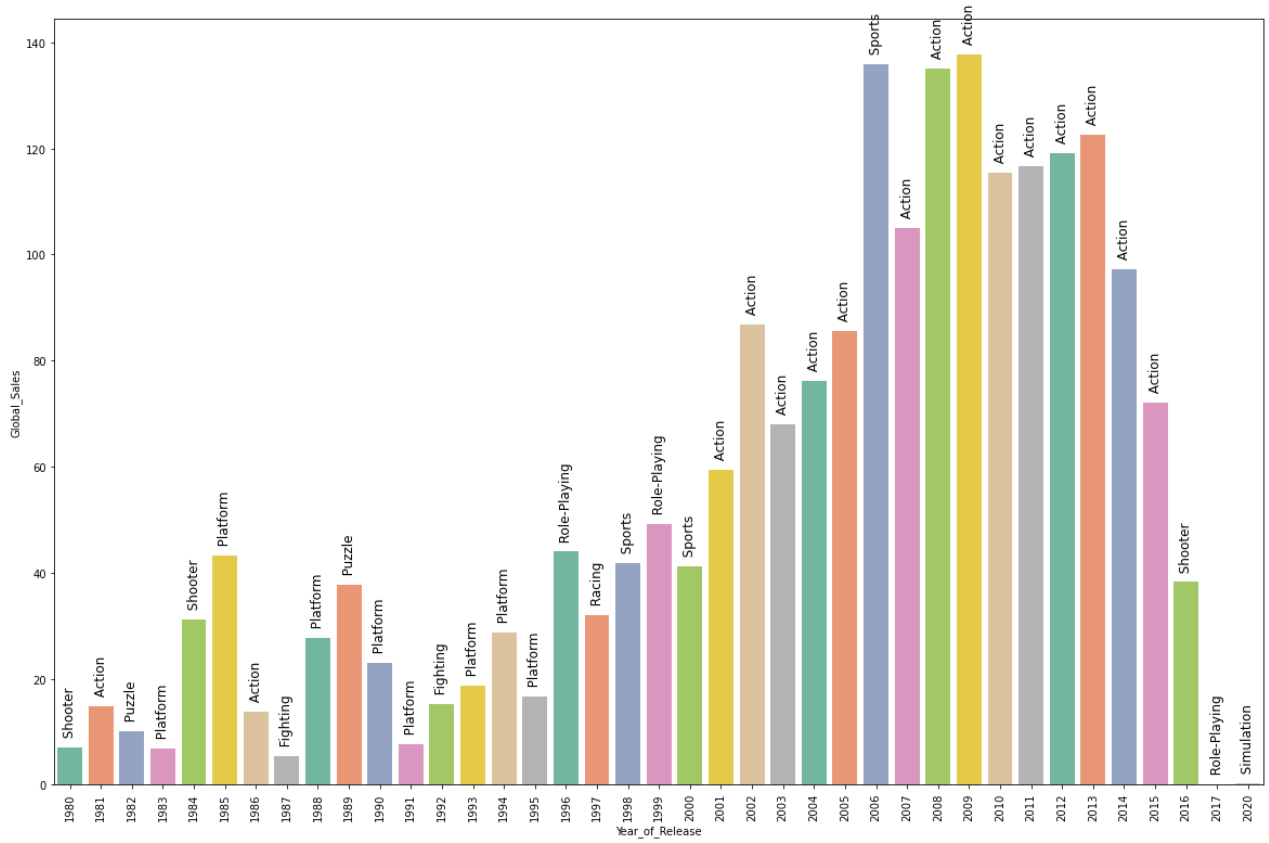
In [23]:
```
year_genre_grp = df.groupby(['Year_of_Release', 'Genre'])['Global_Sales'].sum().reset_i
yearly_max = year_genre_grp.groupby(['Year_of_Release'])['Global_Sales'].transform(max)
best_yearly_genre = year_genre_grp[yearly_max].reset_index(drop=True)
best_yearly_genre.head(10)
```

Out[23]:

	Year_of_Release	Genre	Global_Sales
0	1980	Shooter	7.07000
1	1981	Action	14.84000
2	1982	Puzzle	10.03000
3	1983	Platform	6.93000

	Year_of_Release	Genre	Global_Sales
4	1984	Shooter	31.10000
5	1985	Platform	43.17000
6	1986	Action	13.74000
7	1987	Fighting	5.42000
8	1988	Platform	27.73000
9	1989	Puzzle	37.75000

In [24]:
```
genre = best_yearly_genre['Genre']
plt.figure(figsize=(20, 13))
g = sns.barplot(x='Year_of_Release', y='Global_Sales', palette="Set2",data=best_yearly_
index = 0
for max_global in best_yearly_genre['Global_Sales']:
    g.text(index, max_global + 1, str("  "+genre[index]),size=12,ha="center",rotation=
    index += 1
plt.xticks(rotation=90)
plt.show()
```



From Platforms to Action : The 90s set the ball rolling for all of the games that we play today, no matter what the genre, and it'll likely be remembered for that reason. Platformers drove the industry in the 90s with games like Super Mario World and Super Mario 64—two of the highest selling games of the decade. The defining consoles for this generation included the PlayStation, Sega Saturn, and Nintendo 64, one of the last cartridge based consoles in production. Nintendo's marquee franchise was as popular as ever, and Mario wasn't all they offered—throw in franchises like Metroid, Kirby, Donkey Kong, Yoshi, and Banjo Kazooie.

Alongside these technological changes, gaming itself evolved and grew. The first-person shooter, real-time strategy, and survival horror genres were all popularized and defined during this time.

Furthermore, as we can observe from the above graph the end of 1990's and early 2000's saw a spike in the popularity of the Action genre. We believe this was largely due to the "3D Revolution" where action games made the transition from 2D and pseudo-3D graphics to real-time 3D polygon graphics. On personal computers, the first-person shooter (FPS) genre was popularized by Doom; it is also considered, despite not using 3D polygons, a major leap forward for three-dimensional environments in action games.

Let's dive deeper into the unit sales breakdown of the top 5 video game genres

In [25]:
```
top_5_genre=df.groupby("Genre")["Global_Sales"].sum().sort_values(ascending=False) [:5]
top_5_genre.index
```

Out[25]:
```
Index(['Action', 'Sports', 'Shooter', 'Role-Playing', 'Platform'], dtype='object', name
='Genre')
```

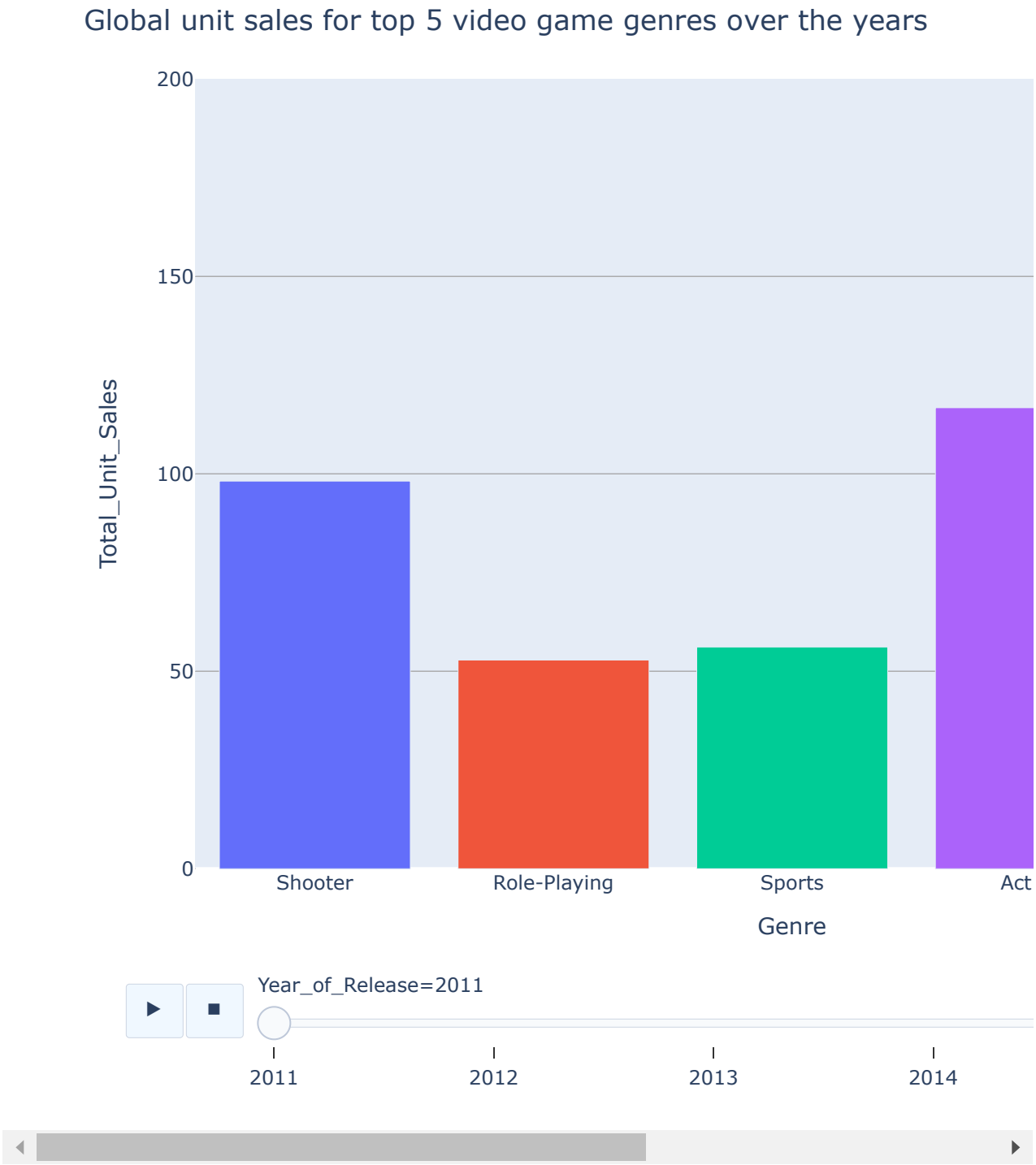
In [26]:
```
genre_sales = df.loc[:,["Year_of_Release","Genre",'Global_Sales']]
genre_sales['Total_Unit_Sales'] = genre_sales.groupby([genre_sales.Genre,genre_sales.Ye
genre_sales.drop('Global_Sales', axis=1, inplace=True)
genre_sales = genre_sales.drop_duplicates()
genre_sales = genre_sales[(genre_sales['Year_of_Release']>=2011) & (genre_sales['Year_o
genre_sales = genre_sales.sort_values("Year_of_Release",ascending = False)
genre_sales = genre_sales.loc[genre_sales['Genre'].isin(top_5_genre.index)]
genre_sales = genre_sales.sort_values("Year_of_Release")
genre_sales.head(15)
```

Out[26]:

	Year_of_Release	Genre	Total_Unit_Sales
29	2011	Shooter	98.17000
75	2011	Role-Playing	52.85000
122	2011	Sports	56.12000
118	2011	Action	116.76000
53	2011	Platform	27.71000
83	2012	Role-Playing	46.91000
34	2012	Shooter	71.80000
62	2012	Platform	18.37000
81	2012	Action	119.10000
473	2012	Sports	30.42000
126	2013	Sports	41.23000
260	2013	Platform	24.59000
16	2013	Action	122.57000
33	2013	Role-Playing	44.42000
60	2013	Shooter	62.05000

```
fig=px.bar(genre_sales,x='Genre', y="Total_Unit_Sales", width=1000, height=700, animati
```

In [27]:
```
            animation_group="Genre", color="Genre", hover_name="Genre",range_y=[0,200])
fig.update_layout(title_text="Global unit sales for top 5 video game genres over the ye
    paper_bgcolor="aliceblue",
)
fig.show()
```

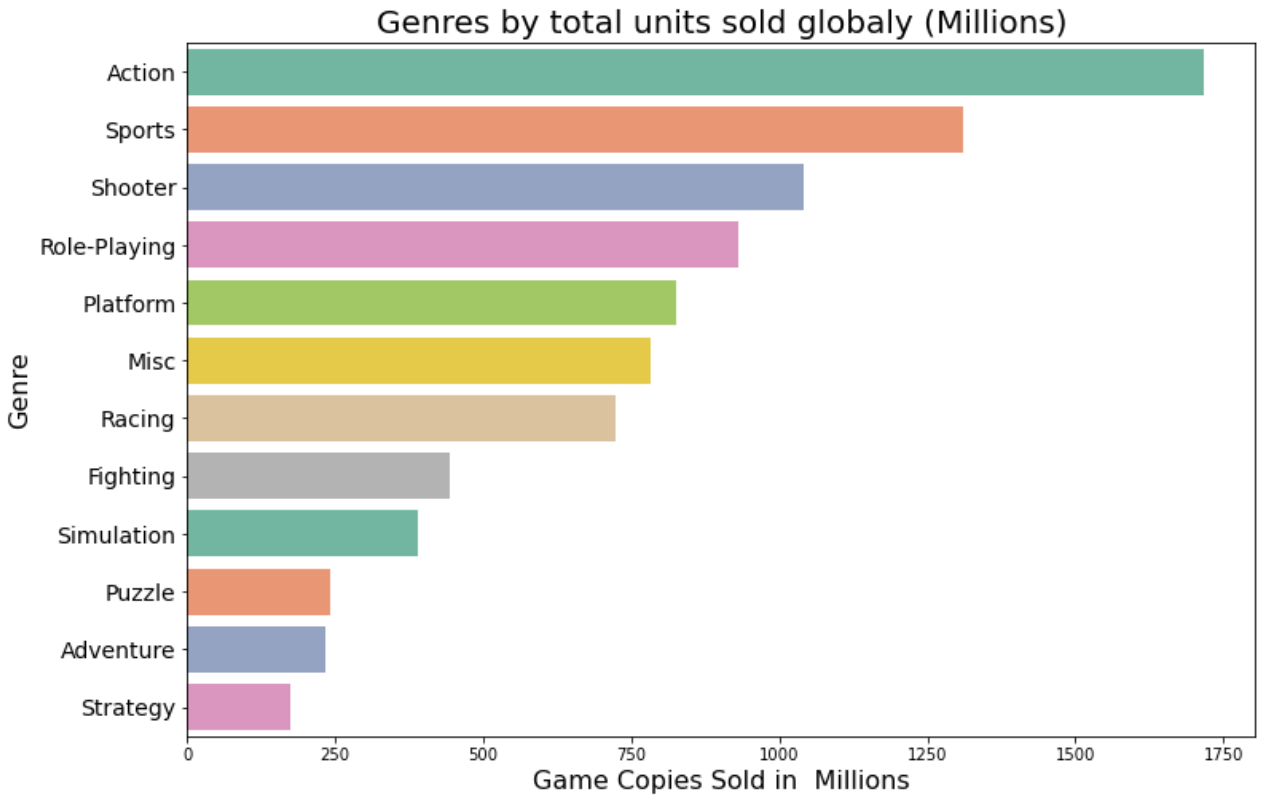


Popular gaming genre's : While observing the trends in video game unit sales globally, we noticed that the last five year trends indicate that the top five video game genres are Action, Shooting, Sports, Platform, role-playing- with Action and Shooting games still having a steady hold over the market.

With Doom being a pioneer in the areas of immersive 3D graphics and networked multiplayer gaming, it greatly helped in popularizing the first-person shooter genre. Moreover with games like Call of Duty (2003), Halo (2001) and Fortnite (2017) the action-shooting genre is largely popular among masses.

Total video game units sold globally per genre

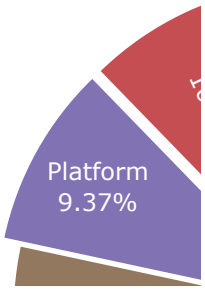
In [28]:
```
units = df.groupby(['Genre']).sum()['Global_Sales']
units = pd.DataFrame(units.sort_values(ascending=False))
genres = units.index
units.columns = ['Copies_Sold']

colors = sns.color_palette('Set2', len(units))
plt.figure(figsize=(12,8))
ax = sns.barplot(y = genres , x = 'Copies_Sold', data=units, orient='h', palette=colors
ax.set_xlabel(xlabel='Game Copies Sold in  Millions', fontsize=16)
ax.set_ylabel(ylabel='Genre', fontsize=16)
ax.set_title(label='Genres by total units sold globaly (Millions)', fontsize=20)
ax.set_yticklabels(labels = genres, fontsize=14)
plt.show();
```



Visualizing the % wise market share of each gaming genre:

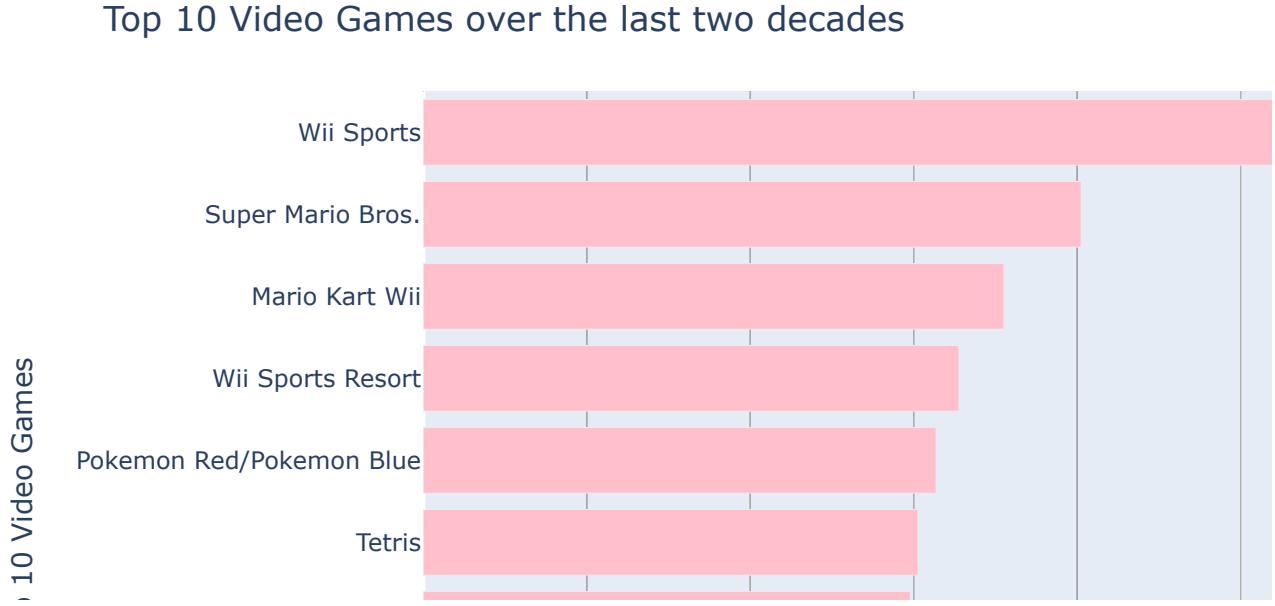
In [29]:
```
genre = df.loc[:,['Genre','Global_Sales']]
genre['total_sales'] = genre.groupby('Genre')['Global_Sales'].transform('sum')
genre.drop('Global_Sales', axis=1, inplace=True)
genre = genre.drop_duplicates()
fig = px.pie(genre, names='Genre', values='total_sales', template='seaborn')
fig.update_traces(rotation=90, pull=[0.2,0.06,0.06,0.06,0.06], textinfo="percent+label"
fig.update_layout(title="Genre Wise Game Sales",title_x=0.5)
fig.show()
```

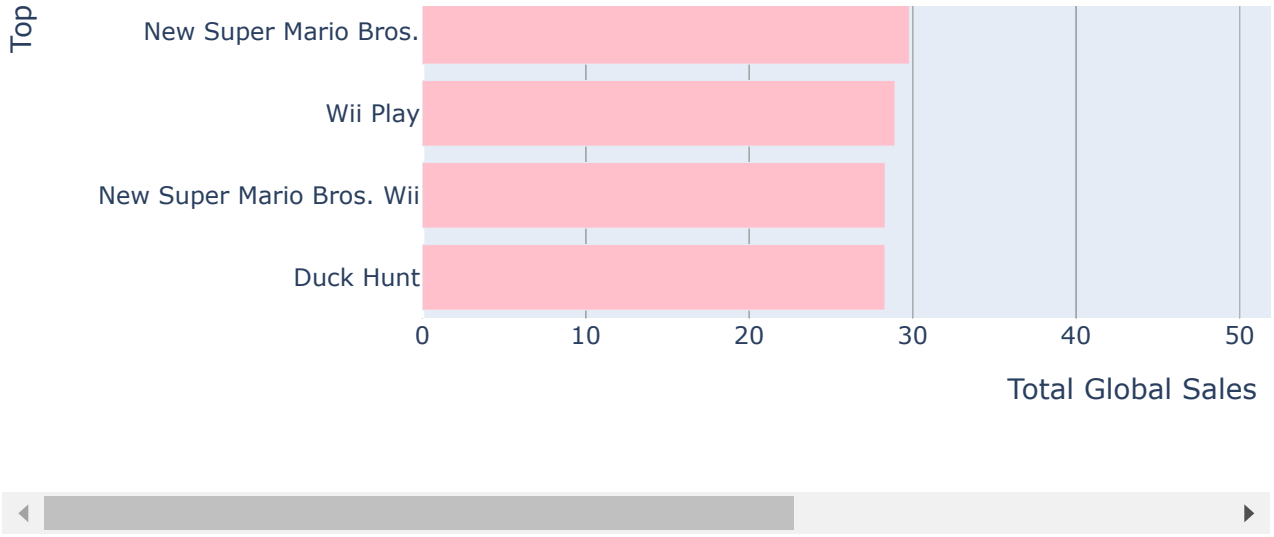
Ger



Global Unit Sales of the Top 10 video games

In [30]:
```
Top10 = df.sort_values("Global_Sales").tail(10)[[ 'Name', 'Platform', 'Year_of_Release'
a=Top10[['Name','Global_Sales']]
import plotly.express as px
fig = px.bar(a, x="Global_Sales", y="Name",
                orientation='h', width=1000, height=600,color_discrete_sequence=["pink"
fig.update_layout(title="Top 10 Video Games over the last two decades",height=550, pape
    xaxis_title="Total Global Sales")
```

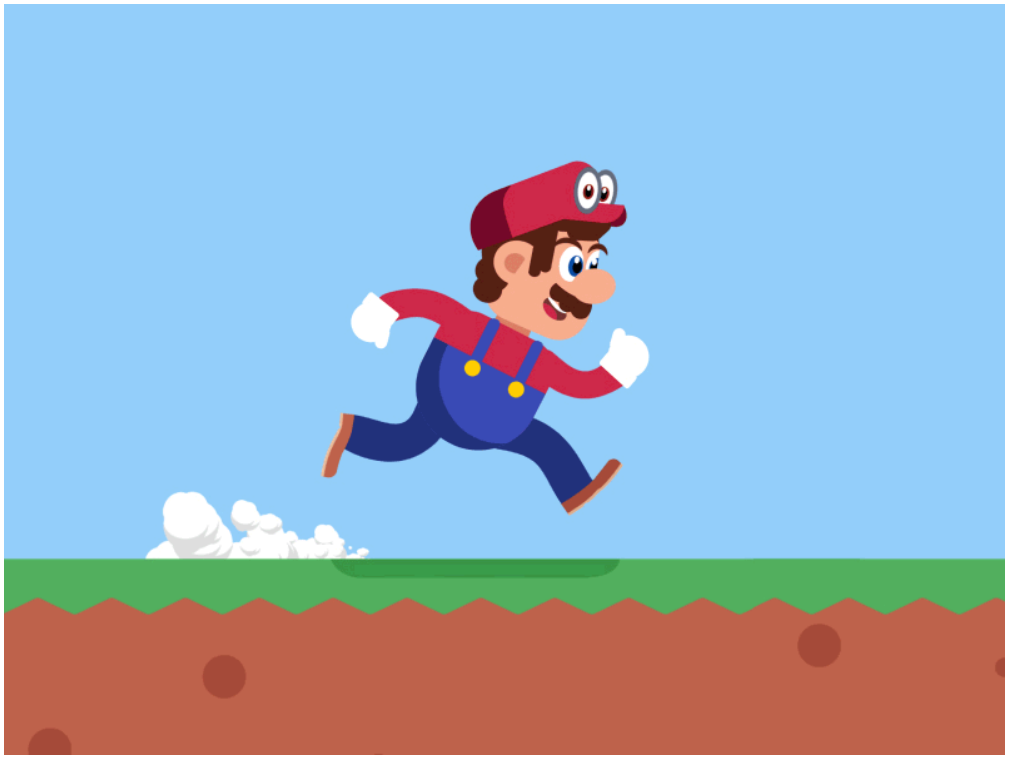




Popular Video Game Alert :

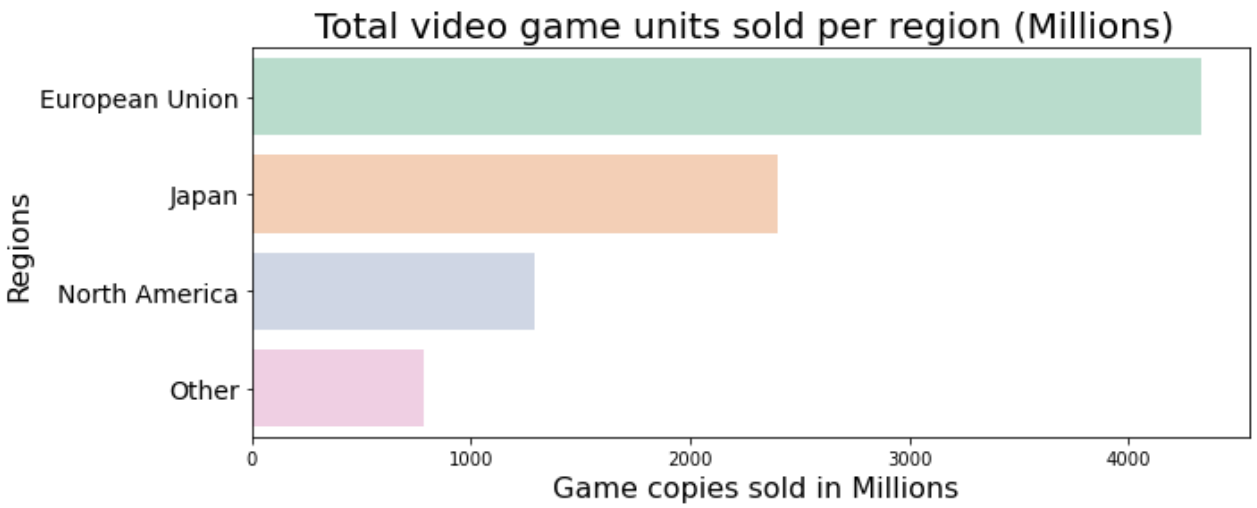
While looking at the top-games across all regions we observe that Wii sports takes the first place especially with the sales in the North America. Wii Sports is a sports video game developed and published by Nintendo for the Wii video game console. The game was released in North America along with the Wii on November 19, 2006, and was released in Japan, Australia, and Europe the following month.The game is a collection of five sports simulations, designed to demonstrate the motion-sensing capabilities of the Wii Remote. The five sports included are tennis, baseball, bowling, golf and boxing.

Second place is Super Mario Bros. It is a platform video game developed and published by Nintendo. The successor to the 1983 arcade game, Mario Bros., it was released in Japan in 1985 for the Famicom, and in North America and Europe for the Nintendo Entertainment System (NES) in 1985 and 1987 respectively. Players control Mario, or his brother Luigi in the multiplayer mode, as they travel the Mushroom Kingdom to rescue Princess Toadstool from the antagonist, Bowser.



Total video game units sold per region :

In [31]:
```
data = df.sum()
data = pd.DataFrame([data['EU_Sales'], data['JP_Sales'], data['NA_Sales'], data['Other_
regions = ['European Union', 'Japan', 'North America', 'Other']
data.index = regions
data.columns = ['Units']
data = data.sort_values(by='Units', ascending=False)
plt.figure(figsize=(10,4))
ax = sns.barplot(y = regions , x = 'Units', palette="Pastel2",data=data, orient='h')
ax.set_xlabel(xlabel=' Game copies sold in Millions', fontsize=16)
ax.set_ylabel(ylabel='Regions', fontsize=16)
ax.set_title(label='Total video game units sold per region (Millions) ', fontsize=20)
ax.set_yticklabels(labels = regions, fontsize=14)
plt.show();
```



Sales across geographical regions : When viewing overall video game sales, it is important to understand how specific geographical regions may have different trends. Different countries have significant cultural differences, so this seemingly might translate into the types of video games people play. To get an insight into the regional sales, we plot the total number of units sold across the four regions: namely, the European Union, Japan, North America and Others.

Europe potentially comes across as a big market for games.

Top genre releases over years

In [32]:
```
genres_over_years = df[['Genre', 'Year_of_Release']]
top_genres_over_years = genres_over_years.groupby(['Year_of_Release', 'Genre']).size().
top_genres_over_years
```

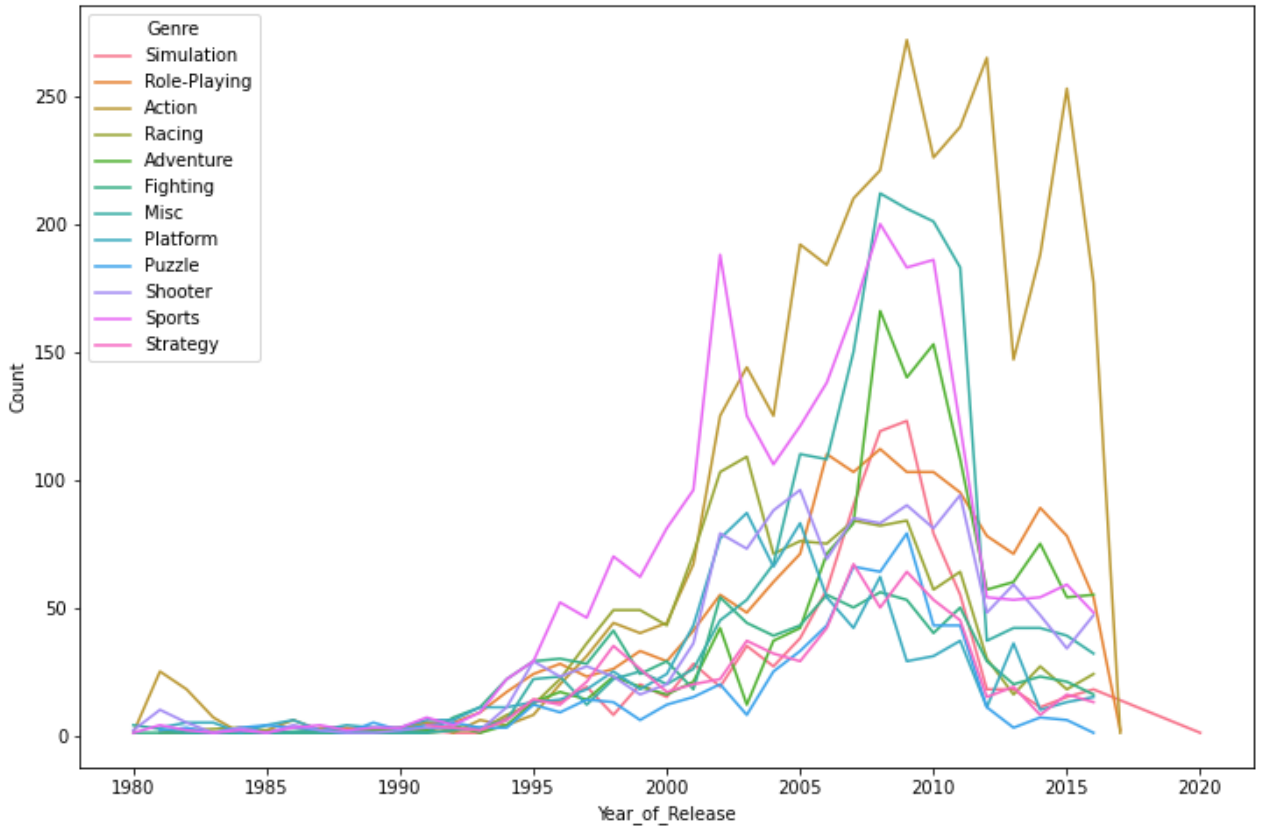
Out[32]:

	Year_of_Release	Genre	Count
389	2020	Simulation	1
388	2017	Role-Playing	2
387	2017	Action	1
381	2016	Racing	24
375	2016	Action	177
...	...	...	...
3	1980	Shooter	2
2	1980	Misc	4
1	1980	Fighting	1
4	1980	Sports	1
0	1980	Action	1

390 rows × 3 columns

In [33]:
```
plt.figure(figsize=(12,8))
sns.lineplot(data=top_genres_over_years, x="Year_of_Release", y="Count", hue='Genre', s
```

Out[33]:
```
<AxesSubplot:xlabel='Year_of_Release', ylabel='Count'>
```



Comparing genre with unit sales per year, we see a spike in the action genre starting in the early 2000s. This is likely explainable by the fact that action-based games typically involve significant movement within the game environment, often in the form of shooting or other computationally-expensive actions. Early consoles, which lacked the extensive graphics and 3-D visualization of modern consoles, were not capable of handling the processing needed for action games, which had to wait until the late 90s to take foot. A number of titles in the action genre have developed large followings in the past two decades, including titles such as Halo, Counterstrike, Destiny, or Fortnite, and have ballooned into a sub-industry on their own in which players can earn six-figure incomes by beating other top players in the world.

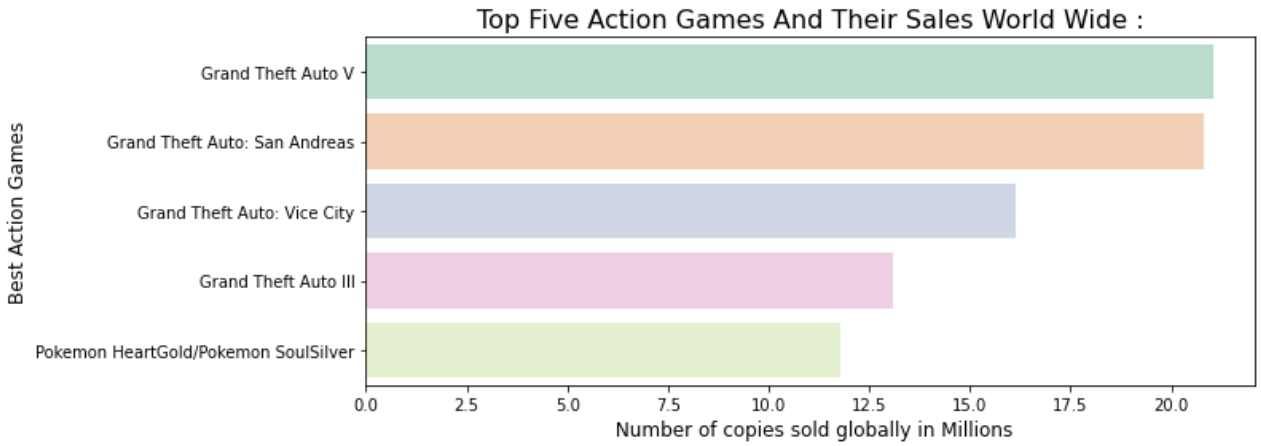
References : To corroborate our results,we checked out IMDB and saw that most of the top games post 2000 were heavily focussed on Action. **Click picture to navigate:**

Which are the Top Five Best Action Games?

In [34]:
```
top_five_action_games = df[df.Genre == "Action"][["Name","Global_Sales"]]
top_five_action_games = top_five_action_games.sort_values(by = "Global_Sales",ascending
top_five_action_games = top_five_action_games.drop_duplicates(["Name"]).head(5)
plt.figure(figsize=(10,4))
ax=sns.barplot(x=top_five_action_games["Global_Sales"],y=top_five_action_games["Name"],
ax.set_xlabel(xlabel=' Number of copies sold globally in Millions', fontsize=12)
ax.set_ylabel(ylabel='Best Action Games', fontsize=12)
ax.set_title(label="Top Five Action Games And Their Sales World Wide :", fontsize=16)
top_five_action_games
```

Out[34]:

	Name	Global_Sales
16	Grand Theft Auto V	21.04000
17	Grand Theft Auto: San Andreas	20.81000
24	Grand Theft Auto: Vice City	16.15000
38	Grand Theft Auto III	13.10000
46	Pokemon HeartGold/Pokemon SoulSilver	11.77000



The Grand Theft Auto franchise is the highest rated in the entire games industry. Unsurprisingly, Grand Theft Auto V (GTA V) is the crown jewel in the series and is the most played version of GTA

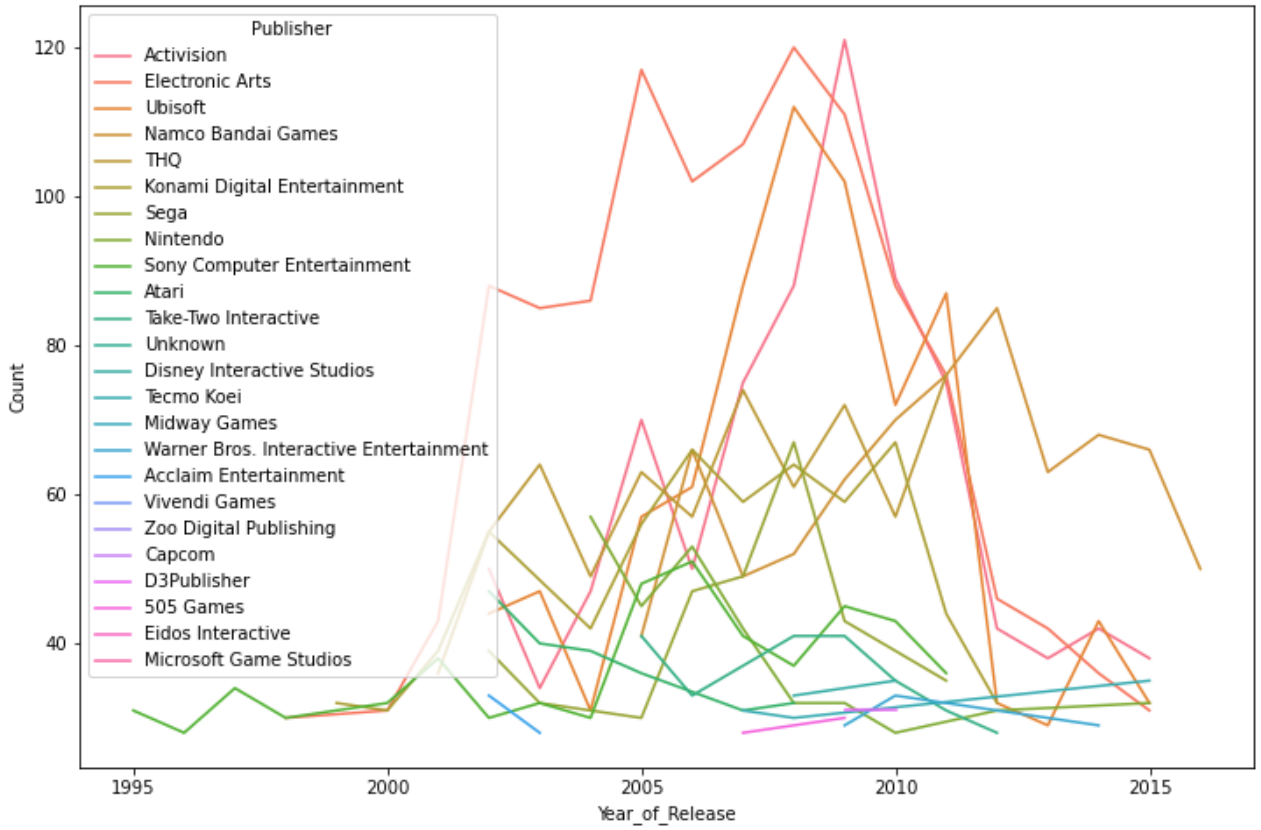
rated by 63% of its fans. With fantastic gameplay and memorable storylines and characters, our analysis finds GTA 5 to be the best selling video game in the action genre.

In [35]:
```
#Wordcloud to visualize the top games in the dataset
text = list(set(df['Name']))
plt.rcParams['figure.figsize'] = (10,10)
wordcloud = WordCloud(max_font_size=30, max_words=50,background_color="black").generate
plt.imshow(wordcloud,interpolation="bilinear")
plt.axis("off")
plt.show()
```

A glance at the game publishers over years :

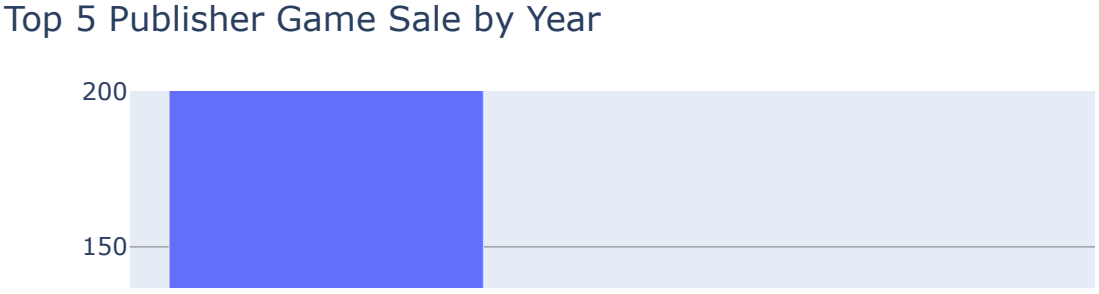
In [36]:
```
publishers_over_years = df[['Publisher', 'Year_of_Release']]
top_publishers_over_years = publishers_over_years.groupby(['Year_of_Release', 'Publishe
plt.figure(figsize=(12,8))
sns.lineplot(data=top_publishers_over_years, x="Year_of_Release", y="Count", hue='Publi
```

Out[36]:
```
<AxesSubplot:xlabel='Year_of_Release', ylabel='Count'>
```



Video Game Publishers: A video game publisher is a company that publishes video games that have been developed either internally by the publisher or externally by a video game developer. They often finance the development, sometimes by paying a video game developer (the publisher calls this external development) and sometimes by paying an internal staff of developers called a studio. The large video game publishers also distribute the games they publish, while some smaller publishers instead hire distribution companies (or larger video game publishers) to distribute the games they publish.

In [37]:
```
top_5_publishers = ['Nintendo', 'Sony Computer Entertainment','Activision','Ubisoft','E
perc = df.loc[:,["Year_of_Release","Publisher",'Global_Sales']]
perc['total_sales'] = perc.groupby([perc.Publisher,perc.Year_of_Release])['Global_Sales
perc.drop('Global_Sales', axis=1, inplace=True)
perc = perc.drop_duplicates()
perc = perc[(perc['Year_of_Release']>=2006)]
perc = perc.sort_values("Year_of_Release",ascending = False)
perc = perc.loc[perc['Publisher'].isin(top_5_publishers)]
perc = perc.sort_values("Year_of_Release")
fig=px.bar(perc,x='Publisher', y="total_sales", animation_frame="Year_of_Release",
           animation_group="Publisher", color="Publisher", hover_name="Publisher",range
fig.update_layout(title_text="Top 5 Publisher Game Sale by Year", xaxis_domain=[0.05, 1
fig.show()
```

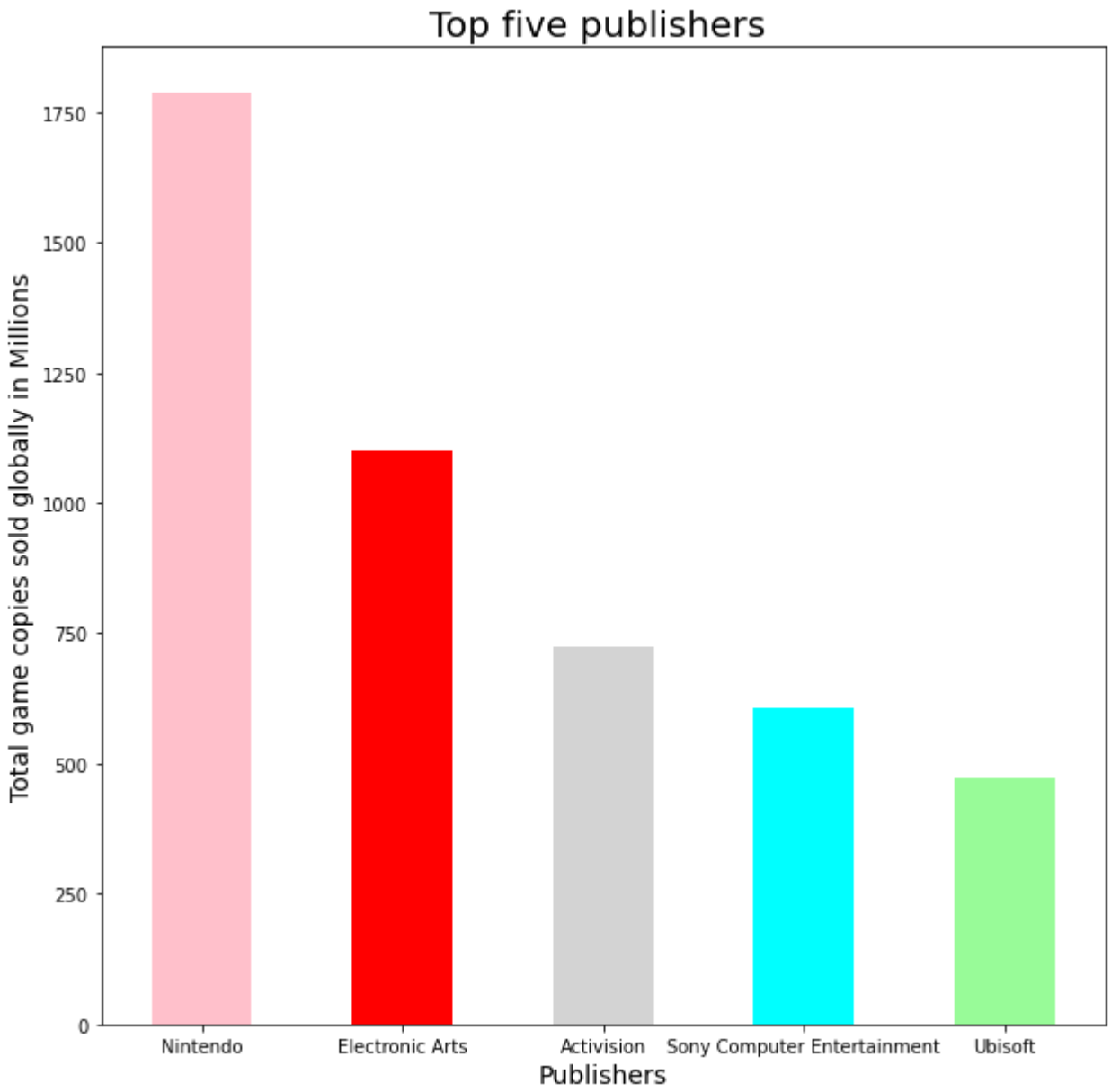


Nintendo for the Win!

Nintendo Co., Ltd. is a Japanese multinational consumer electronics and video game company headquartered in Kyoto. Nintendo is one of the world's largest video game companies by market capitalization, creating some of the best-known and top-selling video game franchises, such as Mario, The Legend of Zelda, and Pokémon.

In [38]:
```
top_5_publishers_globaly=df.groupby("Publisher")["Global_Sales"].sum().sort_values(asce
ax=top_5_publishers_globaly.plot(kind="bar",color=["Pink","red","lightgray","cyan","pal
ax.set_xlabel(xlabel='Publishers', fontsize=14)
ax.set_ylabel(ylabel='Total game copies sold globally in Millions', fontsize=14)
ax.set_title(label='Top five publishers ', fontsize=20)
plt.xticks(rotation=360)
```

Out[38]:
```
(array([0, 1, 2, 3, 4]),
 [Text(0, 0, 'Nintendo'),
  Text(1, 0, 'Electronic Arts'),
  Text(2, 0, 'Activision'),
  Text(3, 0, 'Sony Computer Entertainment'),
  Text(4, 0, 'Ubisoft')])
```

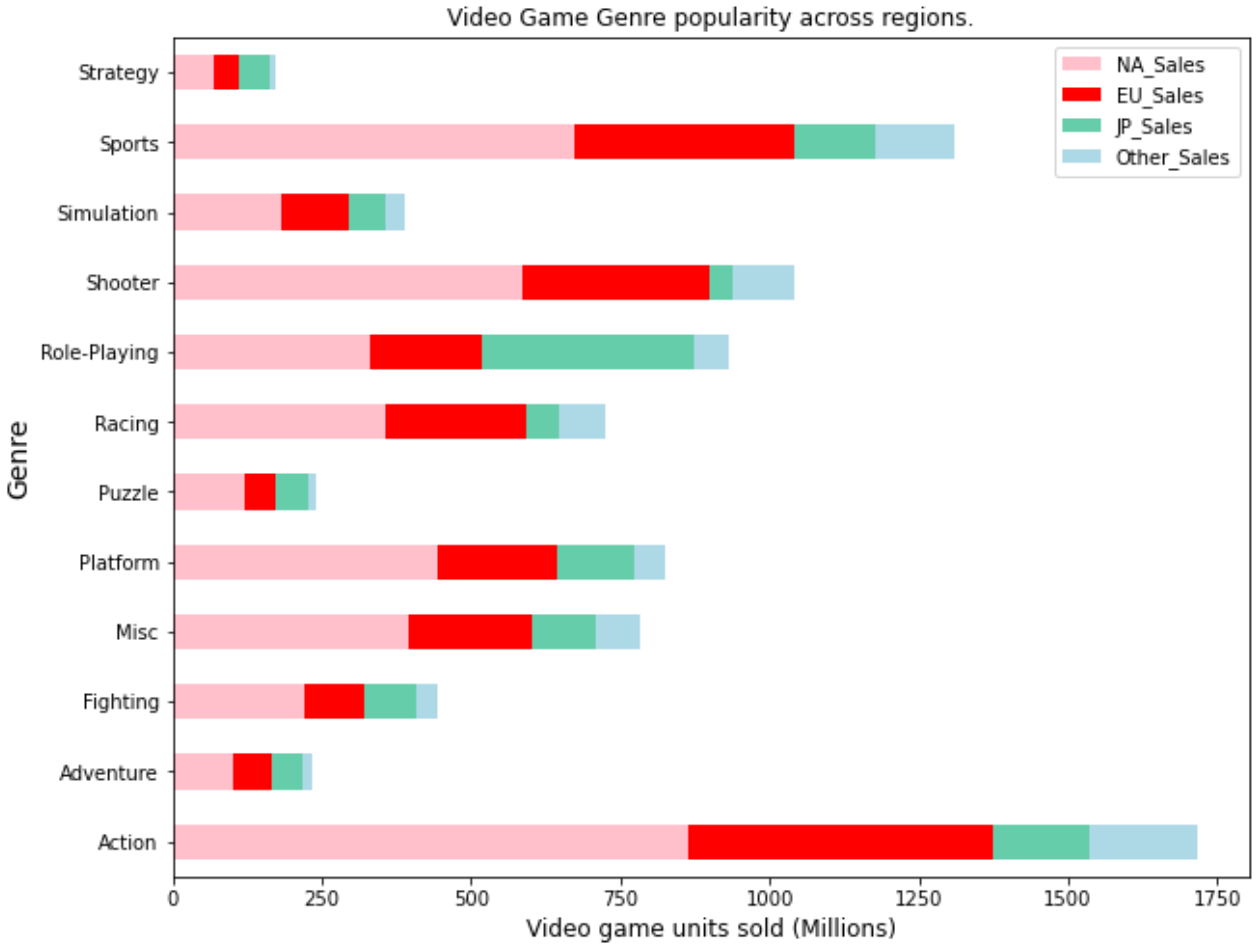


Nintendo has overwhelmingly dominated the market since the early 1980s and has maintained a grip on being the global sales, with Electronic Arts topping them only a handful of years in the 2000s and 2010s. It is interesting to ponder why this is so. Nintendo established themselves early, producing some of the most-played titles in the 1980s, such as Super Mario Brothers or Duck Hunt, and continued this output in the 1990s and 2000s with improved consoles and entirely new concepts, such as the Wii.

Popular video game genres across regions

In [39]:
```
region_sales=df.copy()
region_sales=region_sales.groupby(['Genre']).sum()
```

```
fig, ax = plt.subplots()
region_sales[["NA_Sales", "EU_Sales","JP_Sales","Other_Sales"]].plot.barh(
        stacked=True,
        ax=ax,
        figsize=(10,8),
        linewidth=2,
        color=['pink','red','mediumaquamarine','lightblue'],
        title ='Video Game Genre popularity across regions.')
ax.set_xlabel("Video game units sold (Millions)", fontsize=12)
ax.set_ylabel("Genre", fontsize=14)
ax.get_xaxis().set_major_formatter(tick.FuncFormatter(lambda x, p: format(int(x))))
```



Through our analysis we have been able to narrow down regionally popular gaming genre's. Let's take a look at the top few:

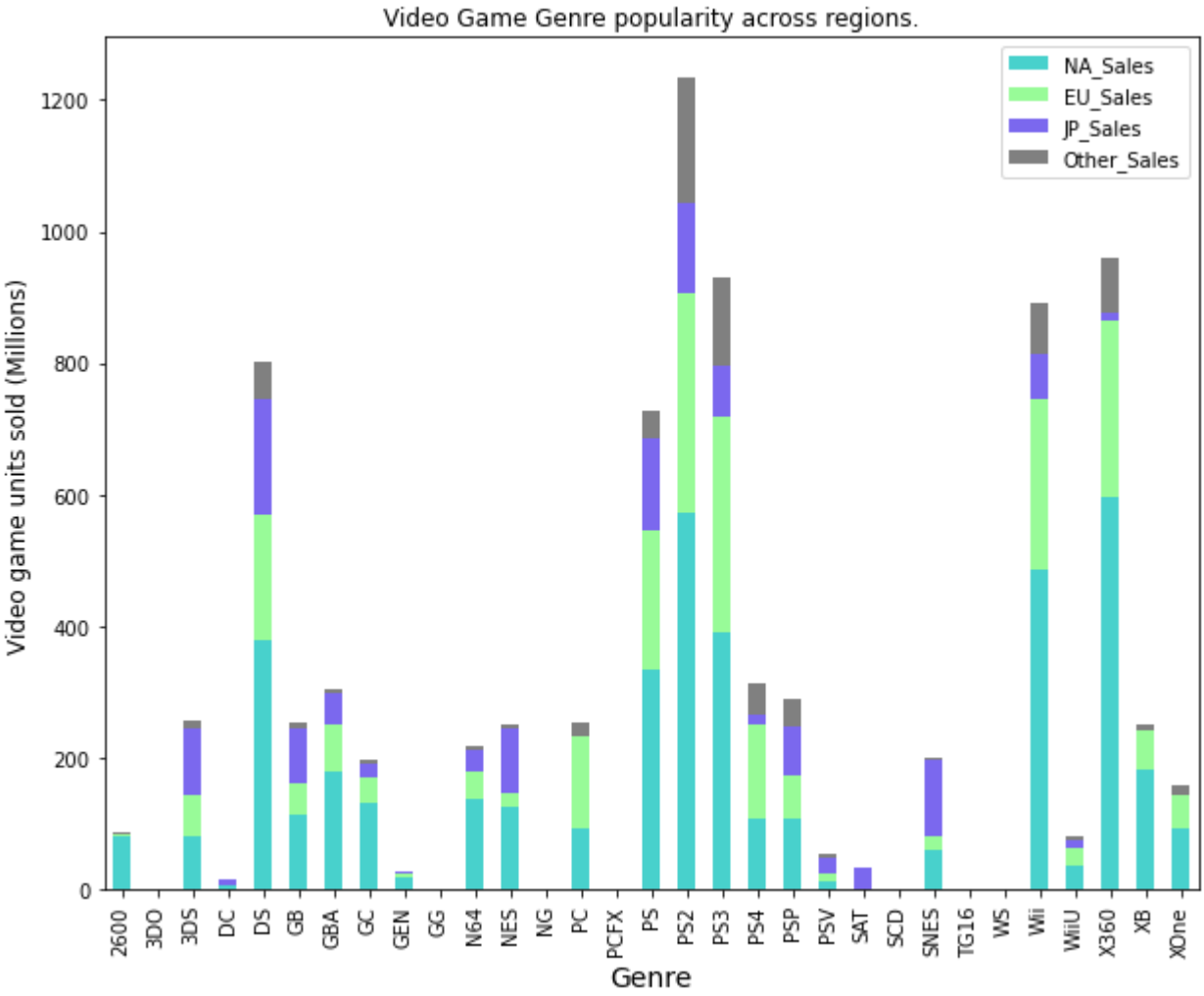
1. **North America** : Action,Shooting and Sports
2. **Europe** : Action, Sports
3. **Japan** : Role-Playing and Action
4. **Others** : Sports and Action

Popular gaming platforms across regions:

In [40]:
```
platform_sales=df.copy()
platform_sales=platform_sales.groupby(['Platform']).sum()
fig, ax = plt.subplots()
platform_sales[["NA_Sales", "EU_Sales","JP_Sales","Other_Sales"]].plot.bar(
        stacked=True,
        ax=ax,
```

```
        figsize=(10,8),
        linewidth=2,
        color=['mediumturquoise','palegreen','mediumslateblue','gray'],
        title ='Video Game Genre popularity across regions.')
ax.set_ylabel("Video game units sold (Millions)", fontsize=12)
ax.set_xlabel("Genre", fontsize=14)
```

Out[40]: Text(0.5, 0, 'Genre')



Through our analysis we have been able to narrow down regionally popular gaming platforms's. Let's take a look at the top few:

1. **North America** : PS2, X360
2. **Europe** : PS2,PS3, X360
3. **Japan** : DS,PS2,PS3
4. **Others** : PS2

Addressing our **first usecase**, we recommend the publishers to narrow down regionally popular gaming platforms and use them to bring their particular game to market. Different countries have significant cultural differences, so this seemingly might translate to the types of video games they most often play. For instance, we observed that Japanese-made consoles are also most popular in Japan, with the DS and Playstation being made by Nintendo and Sony, both Japanese companies.

Similarly, the XBox 360 is most popular in the U.S., which seems to indicate that consoles created by companies within a particular country may sell better in the same country in which they are based. Japan also differs from other global regions in terms of the most popular games and genre. While action games dominate in the U.S., Europe, and the rest of the world, role-based games dominate in Japan, and the Japanese are the only region with Pokemon in their top three most popular titles. Pokemon belongs to the long-standing Japanese culture of anime, which, while very popular in the rest of the world as well, is not as deeply entrenched in popular culture as it is in Japan.

Therefore we recommend publishers to be mindful of the choosing the appropriate platform regionally to target the local customer base.

Geographical view of video game sales

In [41]:
```
world = gpd.read_file(gpd.datasets.get_path('naturalearth_lowres'))
```

In [42]:
```
map=df[["NA_Sales","EU_Sales","JP_Sales"]]
map.rename({'NA_Sales': 'North America', 'EU_Sales': 'Europe','JP_Sales':'Asia'}, axis=
map=map.sum()
df3=map.to_frame().reset_index()
df3.rename(columns = {'index':'continent',0:'Sales'}, inplace = True)
df3
```

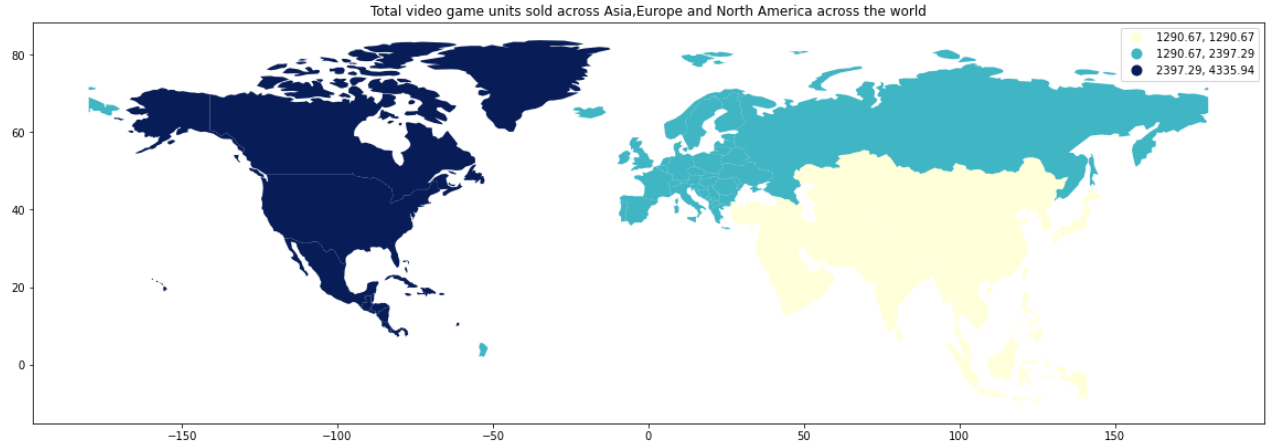
```
C:\Users\Avantika\anaconda3\lib\site-packages\pandas\core\frame.py:4441: SettingWithCopy
Warning:


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/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
```

Out[42]:

	continent	Sales
0	North America	4335.94000
1	Europe	2397.29000
2	Asia	1290.67000

In [43]:
```
for_plotting = world.merge(df3, how='inner', on= 'continent')
fig, ax = plt.subplots(figsize=(20, 15))
plt.title("Total video game units sold across Asia,Europe and North America across the
ax = for_plotting.dropna().plot(ax=ax,column='Sales', cmap =
                                'YlGnBu', figsize=(15,9),
                                 scheme='quantiles',legend =
                                  True);
```

```
C:\Users\Avantika\anaconda3\lib\site-packages\mapclassify\classifiers.py:234: UserWarnin
g:

Warning: Not enough unique values in array to form k classes

C:\Users\Avantika\anaconda3\lib\site-packages\mapclassify\classifiers.py:237: UserWarnin
g:

Warning: setting k to 3
```



When viewing overall video game sales, it is important to understand how specific geographical regions may have different trends. Different countries have significant cultural differences, so this seemingly might translate to the types of video games they most often play. The majority of video game producers are centered in two regions of the world: Japan and the US. Consequently, they also constitute two of the largest consumers of video games, with Europe constituting another significant region.

The Console WARS.

The epic battles fought (and still being fought) amongst the biggest and baddest players in the console gaming industry.

In [44]:
```
top_5_platform=platform_sales.sort_values(by="Global_Sales",ascending=False)[:5]
top_5_gaming_platforms=top_5_platform.index
top_5_platform.reset_index(inplace=True)
```

In [45]:
```
fig = px.pie(top_5_platform, names='Platform', values='Global_Sales', template='seaborn
fig.update_traces(rotation=90, pull=[0.2,0.06,0.06,0.06,0.06], textinfo="percent+label"
fig.update_layout(title="Platform wise game sales",title_x=0.5)
fig.show()
```

The above graph shows the market share of the different gaming platforms. Sony PS2 is leading with 25.6 % of the total global sales.

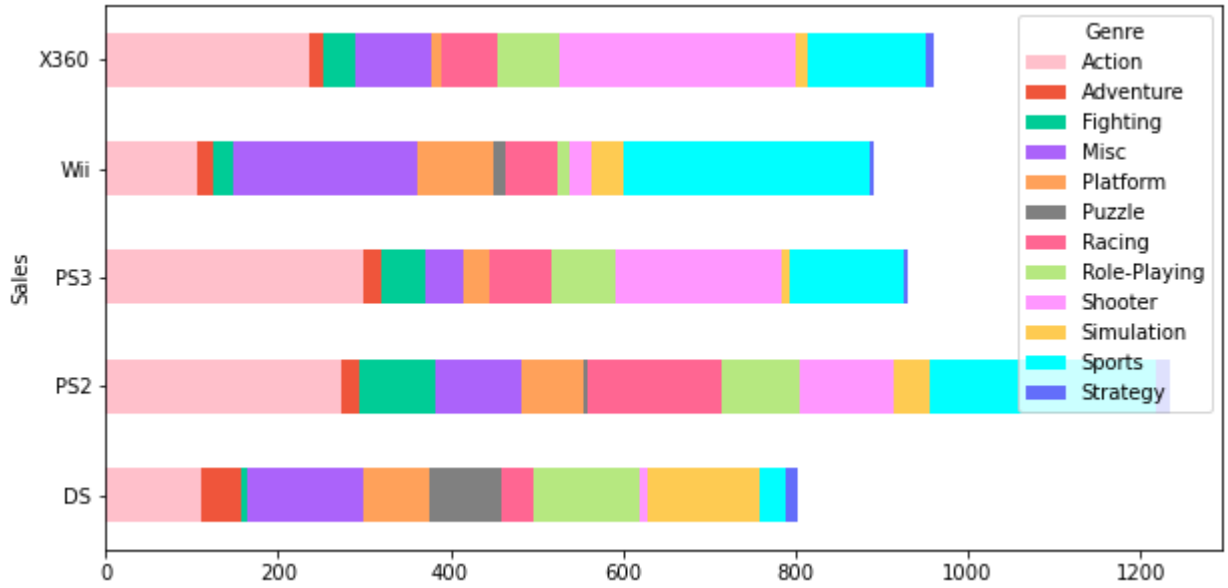
The PlayStation 2 was first released in 2000 in response to Sony's hugely popular original PlayStation. In the 20 years since its release, the PS2 has become the best selling video game console of all time. As of September 2020, it had sold almost 158 million units across the world, including over 55 million units in Europe making it the most successful video game console for the living room by a wide margin. As the following chart shows, Sony has been dominating the gaming landscape for the past two and a half decades, claiming four of the top five spots in the gaming hall of fame.

In order to corroborate our data analysis we used the following external **References:**

1. Statista.com
2. Console Sales

In [46]:
```
top_platforms=df.copy()
top_platforms=top_platforms.loc[(top_platforms['Platform'].isin(top_5_gaming_platforms)
platform_genre = top_platforms.groupby(['Platform','Genre']).Global_Sales.sum()
platform_genre.sort_values(ascending=False,inplace=True)
platform_genre.unstack().plot(kind='barh',stacked=True,color=['pink', '#EF553B', '#00CC
                          grid=False, figsize=(10,5))
plt.ylabel('Sales')
```

Out[46]:
```
Text(0, 0.5, 'Sales')
```



Let's take a deeper look into the top gaming platforms and see which genres are popular on each platform:

While both Sony Playstations PS2 and PS3 supported a lot of action games, Wii focussed on Sports and x360 offered more shooter games.

Overall we've observed that Action, Adventure, Sports, and Shooting are four genres that seem to have been there from the beginning, and persist to this day. They have also seemed to fare well on the top gaming platforms.So for our **second use case** we advise game publishers to focus on these genres if they want to have a low risk portfolio. Furthermore, we've observed that new genre's like role-playing games, puzzles and simulation have also seen significant sales across each of the platforms. Therefore if the game publishers want to diversify their games portfolio, these new genres are definately one's that they should explore.

References: For more information, we recommend exploring the Evolution of Game Genres

Video Games Sales Prediction based on Critic Ratings using Regression

In [47]:
```
path="https://github.com/GitHub847362/2021SpringDataBootcamp/raw/main/Video_Games_Sales_
games=pd.read_csv(path)
games=games[['Platform','Genre','Publisher','Year_of_Release','Critic_Score','Global_Sa
games
```

Out[47]:

	Platform	Genre	Publisher	Year_of_Release	Critic_Score	Global_Sales
0	Wii	Sports	Nintendo	2006.00000	76.00000	82.53000
1	NES	Platform	Nintendo	1985.00000	NaN	40.24000
2	Wii	Racing	Nintendo	2008.00000	82.00000	35.52000
3	Wii	Sports	Nintendo	2009.00000	80.00000	32.77000
4	GB	Role-Playing	Nintendo	1996.00000	NaN	31.37000

	Platform	Genre	Publisher	Year_of_Release	Critic_Score	Global_Sales
...	...	...	...	...	...	...
16714	PS3	Action	Tecmo Koei	2016.00000	NaN	0.01000
16715	X360	Sports	Codemasters	2006.00000	NaN	0.01000
16716	PSV	Adventure	Idea Factory	2016.00000	NaN	0.01000
16717	GBA	Platform	Wanadoo	2003.00000	NaN	0.01000
16718	PSV	Simulation	Tecmo Koei	2016.00000	NaN	0.01000

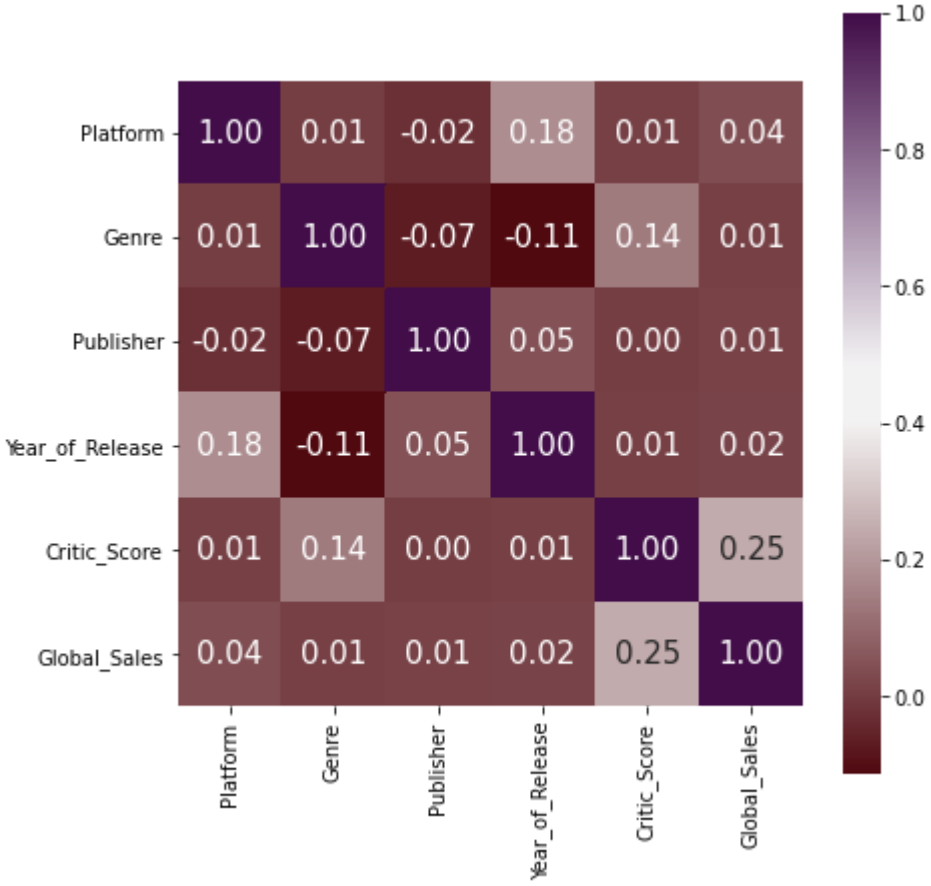
16719 rows × 6 columns

In [48]:
```
games = games.dropna().reset_index(drop=True)
```

In [49]:
```
categorical_labels = ['Platform', 'Genre', 'Publisher']
numerical_lables = ['Year_of_Release','Critic_Score','Global_Sales']
enc = LabelEncoder()
encoded_df = pd.DataFrame(columns=['Platform', 'Genre', 'Publisher','Year_of_Release','

for label in categorical_labels:
    temp_column = games[label]

    encoded_temp_col = enc.fit_transform(temp_column)

    encoded_df[label] = encoded_temp_col

for label in numerical_lables:
    encoded_df[label] = games[label].values
```

In [50]:
```
corr=encoded_df.corr()
corr = (corr)
plt.figure(figsize=(7,7))
cmap = sns.diverging_palette(730, 300, sep=20, as_cmap=True, s=85, l=15, n=20)
sns.heatmap(corr, cbar = True,  square = True, annot=True, fmt= '.2f',cmap=cmap,annot_k
```

Out[50]: <AxesSubplot:>



From the correlation matrix, we observe that the strongest correlations are observed between :

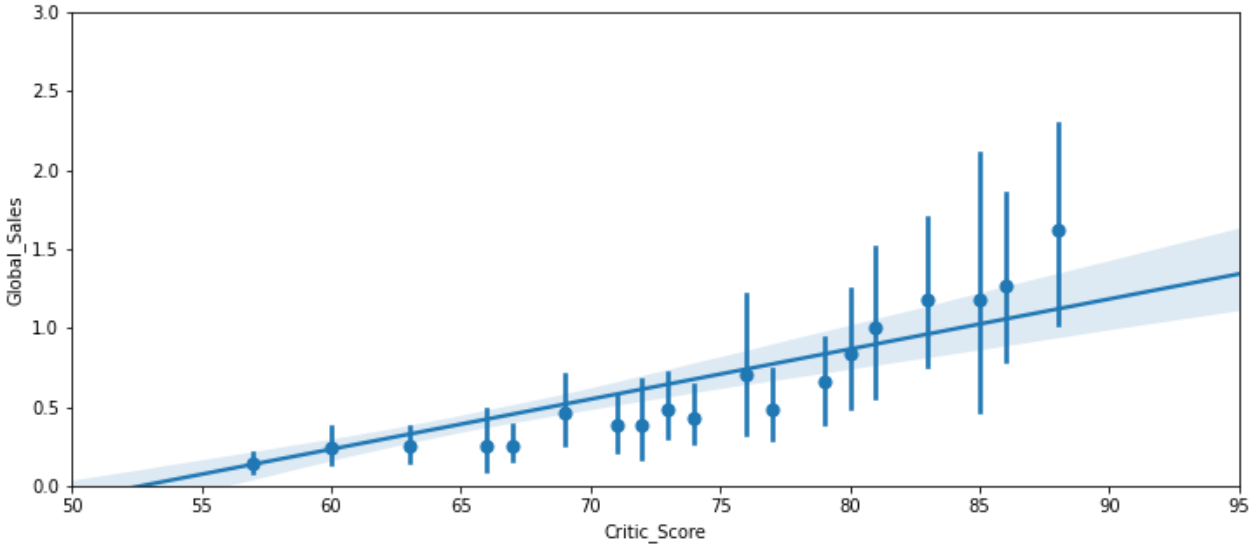
1. The critic score and the Global_Sales
2. The year of release to the platform

We know that it makes sense for the year of release to be correlated to the platforms, as new gaming platforms are introduced periodically.

So instead of exploring that we tend to dive deeper into exploring the relation between the critic scores and the global sales of video games and see if the sales of games can be predicted based of their ratings.

In [51]:
```
fig, ax = plt.subplots(1,1, figsize=(12,5))
sns.regplot(x="Critic_Score", y="Global_Sales", data=encoded_df.loc[encoded_df.Year_of_
            truncate=True, x_bins=20).set(ylim=(0, 3),xlim=(50,95))
```

Out[51]:
```
[(0.0, 3.0), (50.0, 95.0)]
```



We make an intersting observation that the slope gets steeper once the 80's. It seems once a game gets a high critic score, every additional point has a higher impact

In [52]:
```
encoded_df = encoded_df.dropna().reset_index(drop=True)
encoded_df['Hit'] = encoded_df['Global_Sales']
encoded_df.drop('Global_Sales', axis=1, inplace=True)
```

Focus of our regression analysis : We will use regression to predict whether a game would be a hit(i.e. having sales greater than 1 million) or not based of the Critic Scores.

In [53]:
```
def hit(sales):
  return 1 if sales>=1 else 0

encoded_df['Hit'] = encoded_df['Hit'].apply(lambda x: hit(x))
```

In [54]:
```
from pandas import get_dummies
games_predict = pd.get_dummies(encoded_df)
```

In [55]:
```
games_predict
```

Out[55]:

	Platform	Genre	Publisher	Year_of_Release	Critic_Score	Hit
0	12	10	185	2006.00000	76.00000	1
1	12	6	185	2008.00000	82.00000	1
2	12	10	185	2009.00000	80.00000	1
3	2	4	185	2006.00000	89.00000	1
4	12	3	185	2006.00000	58.00000	1
...	...	...	...	...	...	...
7977	5	8	60	2011.00000	61.00000	0
7978	4	5	273	2003.00000	53.00000	0
7979	2	5	153	2008.00000	48.00000	0
7980	5	11	276	2011.00000	60.00000	0
7981	5	1	55	2009.00000	63.00000	0

7982 rows × 6 columns

In [56]:
```
#Step 1 : We begin with selecting features that are positively correlated.
x=encoded_df[['Platform','Genre','Publisher','Year_of_Release','Critic_Score']]
y=encoded_df[['Hit']]
```

In [57]:
```
#Step 2: We split the dataset into the test-train split
X_train,X_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
```

In [58]:
```
#Step 3: Choose a model. We have used logistic regression for prediction purposes.
logreg = LogisticRegression()
```

In [59]:
```
#Step 4: Fit the model using the training dataset.
logreg.fit(X_train,y_train)
```

```
C:\Users\Avantika\anaconda3\lib\site-packages\sklearn\utils\validation.py:72: DataConver
sionWarning:

A column-vector y was passed when a 1d array was expected. Please change the shape of y
to (n_samples, ), for example using ravel().
```

Out[59]:
```
LogisticRegression()
```

In [60]:
```
#Step 5: Use the model to predict the values for the test set
y_pred=logreg.predict(X_test)
```

In [61]:
```
#Step 6: Let's evaluate the accuracy of the model
acc = (metrics.accuracy_score(y_test,y_pred)*100)
print('Accuracy =',round(acc,2),'%')
fpr, tpr, thresholds = roc_curve(y_test, y_pred)
roc_auc = metrics.auc(fpr, tpr)
print("The roc-auc score is:",roc_auc)
```

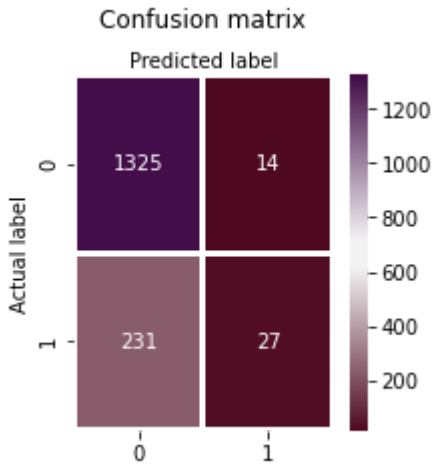
```
Accuracy = 84.66 %
The roc-auc score is: 0.5470977994685378
```

In [62]:
```
#Step 7: Model evaluation using the confusion matrix
cnf_matrix = metrics.confusion_matrix(y_test, y_pred)
print("The confusion matrix is :", cnf_matrix)
```

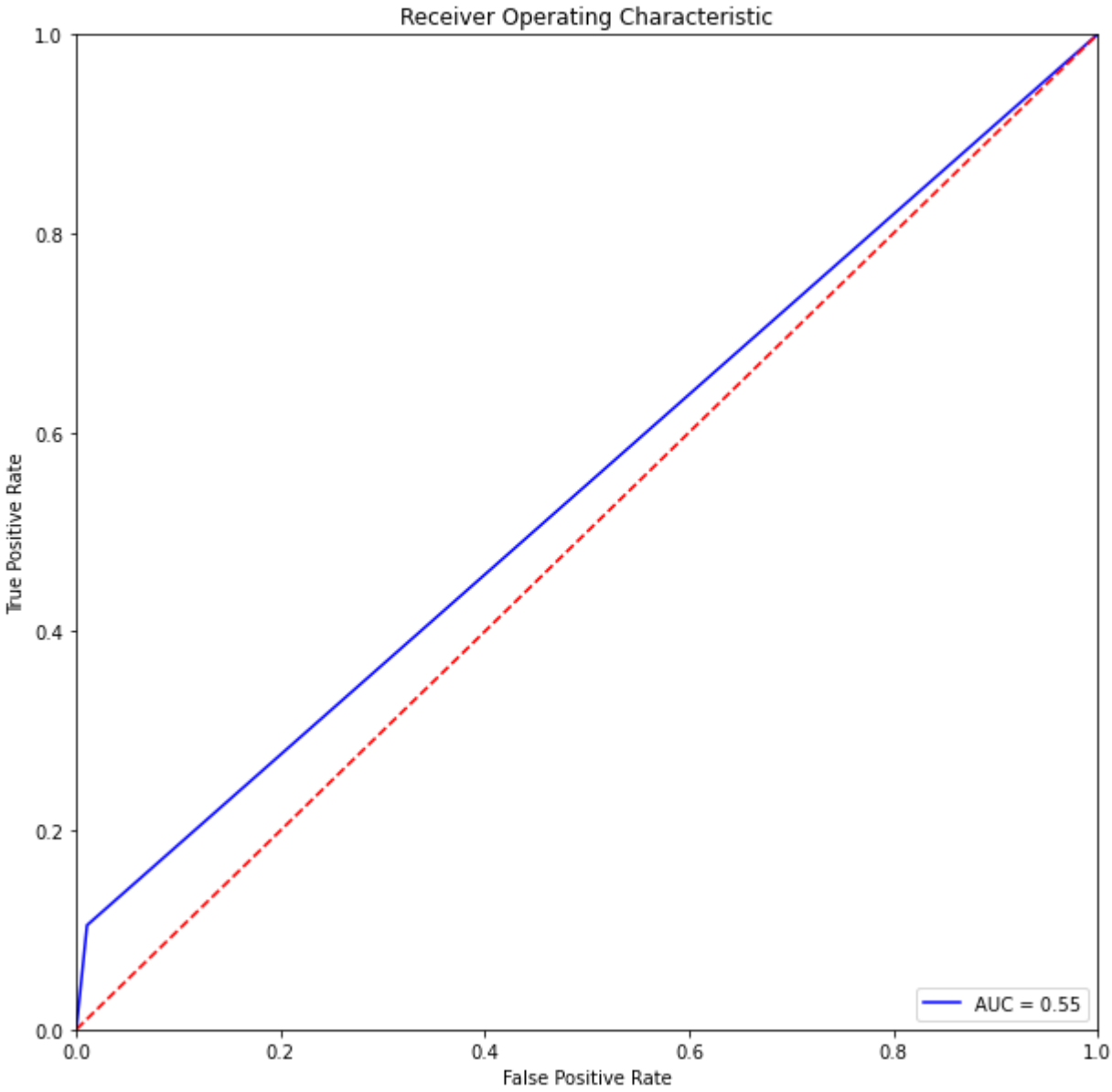
```
The confusion matrix is : [[1325   14]
 [ 231   27]]
```

In [63]:
```
# Step 8: Visualizing the Confusion matrix

class_names=[0,1] # name  of classes
fig, ax = plt.subplots(figsize=(3,3))
tick_marks = np.arange(len(class_names))
plt.xticks(tick_marks, class_names)
plt.yticks(tick_marks, class_names)
# create heatmap
cmap = sns.diverging_palette(720, 300, sep=20, as_cmap=True, s=85, l=15, n=20)
sns.heatmap(pd.DataFrame(cnf_matrix),annot=True, cmap=cmap ,fmt='g',lw=2)
ax.xaxis.set_label_position("top")
plt.tight_layout()
plt.title('Confusion matrix', y=1.1)
plt.ylabel('Actual label')
plt.xlabel('Predicted label')
```

Out[63]: Text(0.5, 194.08, 'Predicted label')



In [64]:
```
# Step 9: Plotting the ROC Curve

plt.title('Receiver Operating Characteristic')

plt.plot(fpr, tpr, 'b', label = 'AUC = %0.2f' % roc_auc)
plt.legend(loc = 'lower right')
plt.plot([0, 1], [0, 1],'r--')
plt.xlim([0, 1])
plt.ylim([0, 1])
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```



References

1. Mobile Gaming Trends

1. Console Wars

1. Gaming Genre