MUKESH PATEL SCHOOL OF TECHNOLOGY MANAGEMENT AND ENGINEERING

(Affiliated to NMIMS Deemed to be University, Mumbai)

Data Extraction and Processing

Project Report

on

"Video Game Sales and Ratings Analysis"

Submitted by:

Group 6		
Name	**Roll No.**	**SAP ID**
Parthiv Sheth	C041	70322000014
Ishwari Birje	C047	70322000030
Chahel Gupta	C049	70322000047
Aneri Patel	C056	70322000099
Thanush Raju Kaneshan	C059	70322000109
Semester/Year: VII/IV		
Division: B		

B. Tech Integrated Program

Department of Computer Science Engineering

MPSTME, Mumbai

2023-2024

About the Dataset

Dataset Title: Video Game Sales with Ratings

URL for Dataset Download:
https://www.kaggle.com/datasets/rush4ratio/video-game-sales-with-ratings/data

The "Video Game Sales with Ratings" dataset is a comprehensive collection of information related to video games. It includes details about various aspects of each video game, sales figures in different regions, ratings from both critics and users, information about the development of the games, and their ESRB ratings. Here's a brief description of the dataset's attributes:

Attribute Name	Description
Name	The title of the video game.
Platform	The gaming platform on which the game was released.
Year_of_Release	The year in which the game was released.
Genre	The genre or category to which the game belongs.
Publisher	The company that published the game.
NA_Sales	Sales figures for North America (in millions of units).
EU_Sales	Sales figures for Europe (in millions of units).
JP_Sales	Sales figures for Japan (in millions of units).
Other_Sales	Sales figures for regions other than NA, EU, and JP (in millions of units).
Global_Sales	Total global sales figures for the game (in millions of copies).
Critic_score	An aggregate score compiled by Metacritic staff based on reviews from critics.
Critic_count	The number of critics used in coming up with the Critic_score. (out of 100)
User_score	A score provided by subscribers of Metacritic.
User_count	The number of users who provided the user_score.(out of 10)
Developer	The party responsible for creating the game (the game developer).
Rating	The Entertainment Software Rating Board (ESRB) rating, indicating the game's content and suitability for different age groups (e.g., E for Everyone, T for Teen, M for Mature).

This dataset is valuable for exploring and analysing trends in the video game industry, understanding how various factors influence game sales and ratings, and gaining insights into the relationships between critic and user reviews and sales performance. It can be used for data exploration, preprocessing, and visualization to uncover valuable information about the video game market.

DATA EXPLORATION

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

- ***import numpy as np:*** *Imports the NumPy library as 'np' for numerical and mathematical operations in Python.*
- ***import pandas as pd****: Imports the Pandas library as 'pd' for data manipulation and analysis, particularly for working with tabular data.*
- ***import seaborn as sns:*** *Imports the Seaborn library as 'sns' for creating aesthetically pleasing statistical data visualizations in Python.*
- ***import matplotlib.pyplot as plt:*** *Imports the Matplotlib library's submodule 'pyplot' as 'plt' for creating customizable, high-quality data visualizations like plots and charts.*

```
df=pd.read_csv("/content/Video_Games_Sales_as_at_22_Dec_2016.csv")
df
```

	Name	Platform	Year_of_Release	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_
0	Wii Sports	Wii	2006.0	Sports	Nintendo	41.36	28.96	3.77	8.45	
1	Super Mario Bros.	NES	1985.0	Platform	Nintendo	29.08	3.58	6.81	0.77	
2	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.68	12.76	3.79	3.29	
3	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.61	10.93	3.28	2.95	
4	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	1.00	
...	...	...	...	...	...	...	...	...	...	
16714	Samurai Warriors: Sanada Maru	PS3	2016.0	Action	Tecmo Koei	0.00	0.00	0.01	0.00	
16715	LMA Manager 2007	X360	2006.0	Sports	Codemasters	0.00	0.01	0.00	0.00	
16716	Haitaka no Psychedelica	PSV	2016.0	Adventure	Idea Factory	0.00	0.00	0.01	0.00	
16717	Spirits & Spells	GBA	2003.0	Platform	Wanadoo	0.01	0.00	0.00	0.00	
16718	Winning Post 8 2016	PSV	2016.0	Simulation	Tecmo Koei	0.00	0.00	0.01	0.00	

16719 rows × 16 columns

Reads the CSV file and stores its data in a Pandas DataFrame called 'df' for further data analysis in Python.

```
df.shape
```

```
(16719, 16)
```

Returns a tuple representing the dimensions (number of rows and columns) of the DataFrame 'df.'

```
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
```

Provides a summary of information about the DataFrame, including data types, non-null values, and memory usage.

```
df.head(10)
```

	Name	Platform	Year_of_Release	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global
0	Wii Sports	Wii	2006.0	Sports	Nintendo	41.36	28.96	3.77	8.45	
1	Super Mario Bros.	NES	1985.0	Platform	Nintendo	29.08	3.58	6.81	0.77	
2	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.68	12.76	3.79	3.29	
3	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.61	10.93	3.28	2.95	
4	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	1.00	
5	Tetris	GB	1989.0	Puzzle	Nintendo	23.20	2.26	4.22	0.58	
6	New Super Mario Bros.	DS	2006.0	Platform	Nintendo	11.28	9.14	6.50	2.88	
7	Wii Play	Wii	2006.0	Misc	Nintendo	13.96	9.18	2.93	2.84	
8	New Super Mario Bros. Wii	Wii	2009.0	Platform	Nintendo	14.44	6.94	4.70	2.24	
9	Duck Hunt	NES	1984.0	Shooter	Nintendo	26.93	0.63	0.28	0.47	

Displays the first 10 rows of the DataFrame 'df' for a quick overview of the data.

```
df.tail(5)
```

	Name	Platform	Year_of_Release	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Glob
16714	Samurai Warriors: Sanada Maru	PS3	2016.0	Action	Tecmo Koei	0.00	0.00	0.01	0.0	
16715	LMA Manager 2007	X360	2006.0	Sports	Codemasters	0.00	0.01	0.00	0.0	
16716	Haitaka no Psychedelica	PSV	2016.0	Adventure	Idea Factory	0.00	0.00	0.01	0.0	
16717	Spirits & Spells	GBA	2003.0	Platform	Wanadoo	0.01	0.00	0.00	0.0	
16718	Winning Post 8 2016	PSV	2016.0	Simulation	Tecmo Koei	0.00	0.00	0.01	0.0	

Shows the last 5 rows of the DataFrame 'df' for a glimpse of the data's end.

```
df.columns
```

```
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')
```

Returns a list of column names in the DataFrame 'df.'

```
df['Name'].unique()
```

```
array(['Wii Sports', 'Super Mario Bros.', 'Mario Kart Wii', ...,
       'Woody Woodpecker in Crazy Castle 5', 'LMA Manager 2007',
       'Haitaka no Psychedelica'], dtype=object)
```

```
[143] df['Platform'].unique()
```

```
array(['Wii', 'NES', 'GB', 'DS', 'X360', 'PS3', 'PS2', 'SNES', 'GBA',
       'PS4', '3DS', 'N64', 'PS', 'XB', 'PC', '2600', 'PSP', 'XOne',
       'WiiU', 'GC', 'GEN', 'DC', 'PSV', 'SAT', 'SCD', 'WS', 'NG', 'TG16',
       '3DO', 'GG', 'PCFX'], dtype=object)
```

```
[144] df['Genre'].unique()
```

```
array(['Sports', 'Platform', 'Racing', 'Role-Playing', 'Puzzle', 'Misc',
       'Shooter', 'Simulation', 'Action', 'Fighting', 'Adventure',
       'Strategy', nan], dtype=object)
```

```
[145] df['Year_of_Release'].unique()
```

```
array([2006., 1985., 2008., 2009., 1996., 1989., 1984., 2005., 1999.,
       2007., 2010., 2013., 2004., 1990., 1988., 2002., 2001., 2011.,
       1998., 2015., 2012., 2014., 1992., 1997., 1993., 1994., 1982.,
       2016., 2003., 1986., 2000.,   nan, 1995., 1991., 1981., 1987.,
       1980., 1983., 2020., 2017.])
```

Returns an array of unique values from the 'Name', 'Platform', 'Genre', 'Year_of_Release' columns in the DataFrame 'df.'

```
df.isnull().sum()
```

```
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
```

This function lets us see the sum of null values that are there for each column.

```
df.dtypes
```

```
Name                object
Platform            object
Year_of_Release    float64
Genre               object
Publisher           object
NA_Sales           float64
EU_Sales           float64
JP_Sales           float64
Other_Sales        float64
Global_Sales       float64
Critic_Score       float64
Critic_Count       float64
User_Score          object
User_Count         float64
Developer           object
Rating              object
dtype: object
```

It gives the data type of the data in each column.

```
df.describe()
```

	Year_of_Release	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	Critic_Score	Critic_Count	User_Count
count	16450.000000	16719.000000	16719.000000	16719.000000	16719.000000	16719.000000	8137.000000	8137.000000	7590.000000
mean	2006.487356	0.263330	0.145025	0.077602	0.047332	0.533543	68.967679	26.360821	162.229908
std	5.878995	0.813514	0.503283	0.308818	0.186710	1.547935	13.938165	18.980495	561.282326
min	1980.000000	0.000000	0.000000	0.000000	0.000000	0.010000	13.000000	3.000000	4.000000
25%	2003.000000	0.000000	0.000000	0.000000	0.000000	0.060000	60.000000	12.000000	10.000000
50%	2007.000000	0.080000	0.020000	0.000000	0.010000	0.170000	71.000000	21.000000	24.000000
75%	2010.000000	0.240000	0.110000	0.040000	0.030000	0.470000	79.000000	36.000000	81.000000
max	2020.000000	41.360000	28.960000	10.220000	10.570000	82.530000	98.000000	113.000000	10665.000000

This gives us the count, mean, std, min, max, 25% value, 50% value, 75% value for all the numeric values in the dataframe.

DATA PRE-PROCESSING

We have created a copy of the original data, so changes applied to the original data won't affect the new dataset.

```
[149] df_cleaned = df.copy()
```

```
df_cleaned.isnull().sum()
```

```
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
```

As we can see there are multiple columns with NULL values, to deal with each of the columns we will apply different methods.

Name & Genre only have 2 missing values we will drop those rows.

```
for i, row in df_cleaned.iterrows():
    if pd.isnull(row['Name']) or pd.isnull(row['Genre']) or pd.isnull(row['Year_of_Release']):
        df_cleaned.drop(i, inplace=True)
```

Since the same games are released on different platforms, we check if there are entries with duplicate names and same platform.

```
df_cleaned[['Name']].duplicated().sum()
```

```
5019
```

```
df_cleaned[['Name','Platform']].duplicated(keep=False).sum()
```

```
6
```

```
df_cleaned[df_cleaned[['Name','Platform']].duplicated(keep=False)]
```

	Name	Platform	Year_of_Release	Genre	Publisher	NA_Sales
604	Madden NFL 13	PS3	2012-01-01	Sports	Electronic Arts	2.11
1190	Need for Speed: Most Wanted	X360	2012-01-01	Racing	Electronic Arts	0.62
1591	Need for Speed: Most Wanted	X360	2005-01-01	Racing	Electronic Arts	1.00
5973	Need for Speed: Most Wanted	PC	2005-01-01	Racing	Electronic Arts	0.02
11716	Need for Speed: Most Wanted	PC	2012-01-01	Racing	Electronic Arts	0.00
16233	Madden NFL 13	PS3	2012-01-01	Sports	Electronic Arts	0.00

It shows that the game "Need for Speed: Most Wanted" released in both 2005 and 2012 on both X360 and PC platforms, so there is no need to deduplicate them.

We will remove the duplicate row that may be added by mistake.

```
df_cleaned = df_cleaned.drop(16233)
```

```
df_cleaned['User_Score'].unique()
```

```
array(['8', nan, '8.3', '8.5', '6.6', '8.4', '8.6', '7.7', '6.3', '7.4',
       '8.2', '9', '7.9', '8.1', '8.7', '7.1', '3.4', '5.3', '4.8', '3.2',
       '8.9', '6.4', '7.8', '7.5', '2.6', '7.2', '9.2', '7', '7.3', '4.3',
       '7.6', '5.7', '5', '9.1', '6.5', 'tbd', '8.8', '6.9', '9.4', '6.8',
       '6.1', '6.7', '5.4', '4', '4.9', '4.5', '9.3', '6.2', '4.2', '6',
       '3.7', '4.1', '5.8', '5.6', '5.5', '4.4', '4.6', '5.9', '3.9',
       '3.1', '2.9', '5.2', '3.3', '4.7', '5.1', '3.5', '2.5', '1.9', '3',
       '2.7', '2.2', '2', '9.5', '2.1', '3.6', '2.8', '1.8', '3.8', '0',
       '1.6', '9.6', '2.4', '1.7', '1.1', '0.3', '1.5', '0.7', '1.2',
       '2.3', '0.5', '1.3', '0.2', '0.6', '1.4', '0.9', '1', '9.7'],
      dtype=object)
```

```
[172] df_cleaned['Critic_Score'].unique()
```

```
array([76., 82., 80., 89., 58., 87., 91., 61., 97., 95., 77., 88., 83.,
       94., 93., 85., 86., 98., 96., 90., 84., 73., 74., 78., 92., 71.,
       72., 68., 62., 49., 67., 81., 66., 56., 48., 45., 79., 70., 59.,
       64., 75., 60., 63., 42., 69., 32., 50., 25., 44., 55., 47., 57.,
       29., 65., 51., 54., 20., 53., 37., 30., 38., 33., 27., 52., 43.,
       35., 40., 46., 28., 39., 34., 31.,  0., 16., 41., 36., 24., 18.,
       17., 11., 26.,  3., 19., 23.,  7., 13.,  2., 21., 14.,  9., 12.,
        6.])
```

User_Score ranges from 0 to 10 and Critic_Score ranges from 1 to 100.
But User_Score has null values & tbd values (tbd values are converted to NaN values), additionally it is of object type, so we will convert that to float.

```
for i, row in df_cleaned.iterrows():
    if row['User_Score'] == "tbd":
        df_cleaned.loc[i, 'User_Score'] = np.nan
df_cleaned['User_Score'] = df_cleaned['User_Score'].astype(float)
```

If both the User_Score and Critic_Score are null then they are useless for analysis as they have not been played/bought by anyone so we will remove them.
But if the User_Score exists and Critic_Score doesn't exist we will convert the Critic_Score to the User_Score. (User_Score → 1 - 10 and Critic_Score → 1 - 100)
After which we will multiply by 10 to scale it to the same range.

```
for i, row in df_cleaned.iterrows():
    if pd.isnull(row['User_Score']):
        if pd.isnull(row['Critic_Score']):
            df_cleaned.drop(i, inplace=True)
        else:
            df_cleaned.loc[i, 'User_Score'] = df_cleaned.loc[i, 'Critic_Score']
    else:
        df_cleaned.loc[i, 'User_Score'] = df_cleaned.loc[i, 'User_Score'] * 10
        if pd.isnull(row['Critic_Score']):
            df_cleaned.loc[i, 'Critic_Score'] = df_cleaned.loc[i, 'User_Score']
```

We will fill the null values of User_Count and Critic_Count with mean values for better analysis.

```
for i, row in df_cleaned.iterrows():
    if pd.isnull(row['User_Count']):
        df_cleaned.loc[i, 'User_Count'] = df_cleaned['User_Count'].mean()
    if pd.isnull(row['Critic_Count']):
        df_cleaned.loc[i, 'Critic_Count'] = df_cleaned['Critic_Count'].mean()
```

The rating here is not gameplay scores, is about whether children under 18 can play it, it is of no use, and the Publisher is different for different regions for almost every game which is unrequired data, so we drop these columns.

```
df_cleaned = df_cleaned.drop(columns = ['Publisher','Rating'])
```

```
df_cleaned.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 8551 entries, 0 to 16709
Data columns (total 14 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Name             8551 non-null   object
 1   Platform         8551 non-null   object
 2   Year_of_Release  8551 non-null   datetime64[ns]
 3   Genre            8551 non-null   object
 4   NA_Sales         8551 non-null   float64
 5   EU_Sales         8551 non-null   float64
 6   JP_Sales         8551 non-null   float64
 7   Other_Sales      8551 non-null   float64
 8   Global_Sales     8551 non-null   float64
 9   Critic_Score     8551 non-null   float64
 10  Critic_Count     8551 non-null   float64
 11  User_Score       8551 non-null   float64
 12  User_Count       8551 non-null   float64
 13  Developer        8538 non-null   object
dtypes: datetime64[ns](1), float64(9), object(4)
memory usage: 1002.1+ KB
```

```
df_cleaned.isnull().sum()
```

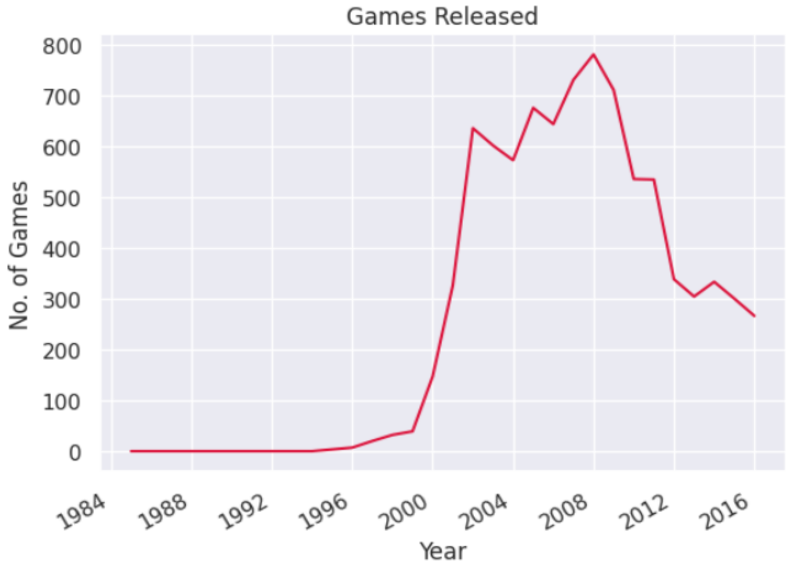
```
Name                0
Platform            0
Year_of_Release     0
Genre               0
NA_Sales            0
EU_Sales            0
JP_Sales            0
Other_Sales         0
Global_Sales        0
Critic_Score        0
Critic_Count        0
User_Score          0
User_Count          0
Developer          13
dtype: int64
```

Now the cleaned data has no null values, except in Developer column, but since the developer is not known we have to ignore it.

DATA VISUALIZATION

Year with maximum number of games released.

```
plt.title("Games Released")
df_cleaned['Year_of_Release'] = df_cleaned['Year_of_Release'].astype(int)
df_cleaned['Year_of_Release'] = pd.to_datetime(df_cleaned["Year_of_Release"].astype(str), format="%Y")
total_games = df_cleaned.groupby('Year_of_Release')['Name'].count()
total_games.plot(kind = 'line', color = 'crimson', xlabel ='Year', ylabel = 'No. of Games')
plt.show()
```

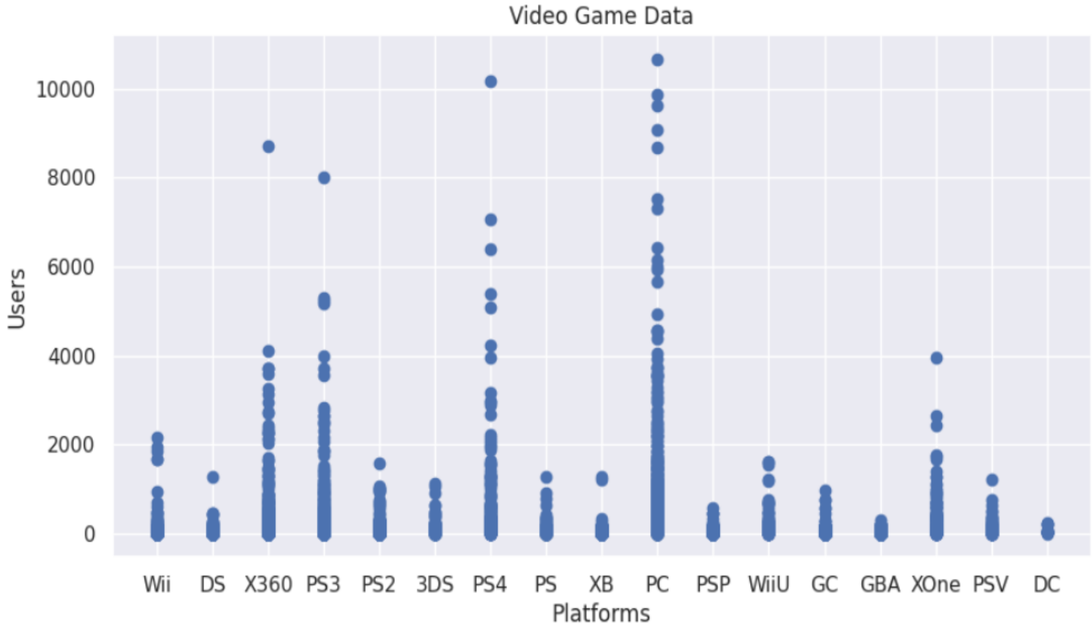


From the graph, till year 1996, the gaming industry was sluggish. But there after some activity started, and after 2000, it shoots high. The gaming industry grew too quickly. Around year 2008 it was at the peak. 750+ games were released in that year. Thereafter again declining with eventually getting down to approximately 37% of the peak in year 2016.

Which Platform was Used the Most?

```
plt.figure(figsize=(10, 5))
plt.scatter(df_cleaned['Platform'],df_cleaned['User_Count'])
plt.title("Video Game Data")
plt.xlabel("Platforms")
plt.ylabel("Users")
plt.show
```

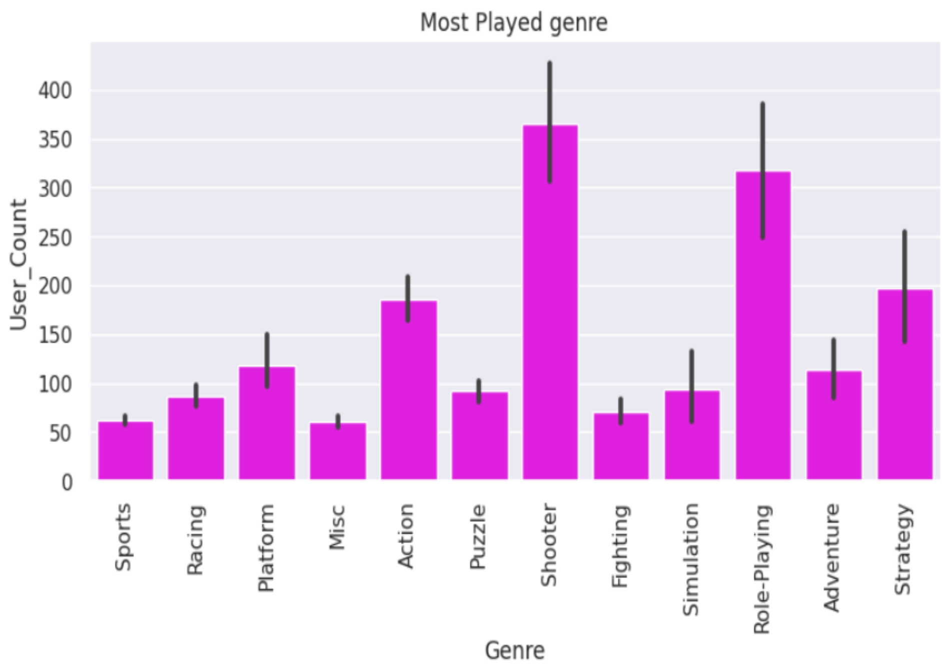
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



The plot shows that the most popular platform for videogames is undoubtedly PC. However, X360, PS3 and PS4 have also been quite widely used.

What was the most played game genre?

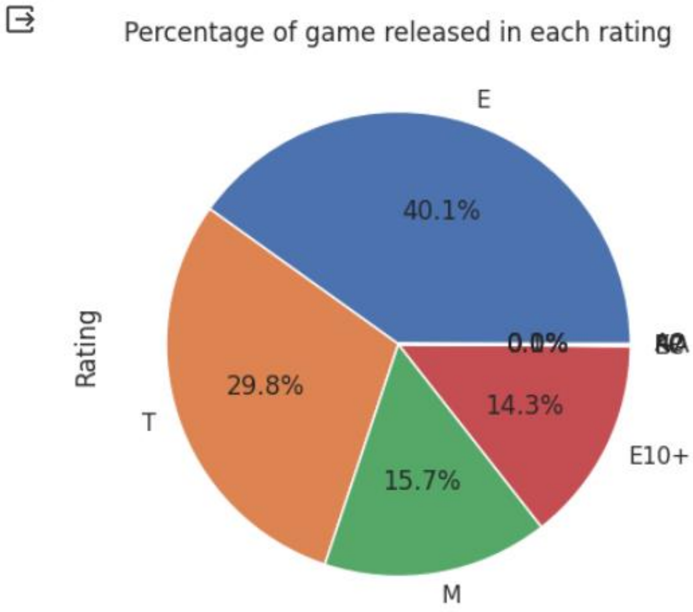
```
cat_plot=sns.catplot(x='Genre', y='User_Count', color = 'magenta', kind='bar', data=df_cleaned,height=4, aspect=2)
cat_plot.set(title='Most Played genre')
plt.xticks(rotation=90)
plt.show()
```



Shooting games are the most popular genre played, with 350+ users, followed by role-playing which has a user count of 300+. Action is 50% less popular than shooting. Surprisingly sports are the least popular having only 50+ users.

Which were the most common rating among the games released?

```python
counts =df['Rating'].value_counts()
plt.figure(figsize=(5,5))
counts.plot(kind='pie', autopct='%1.1f%%')
plt.title('Percentage of game released in each rating')
plt.show()
```

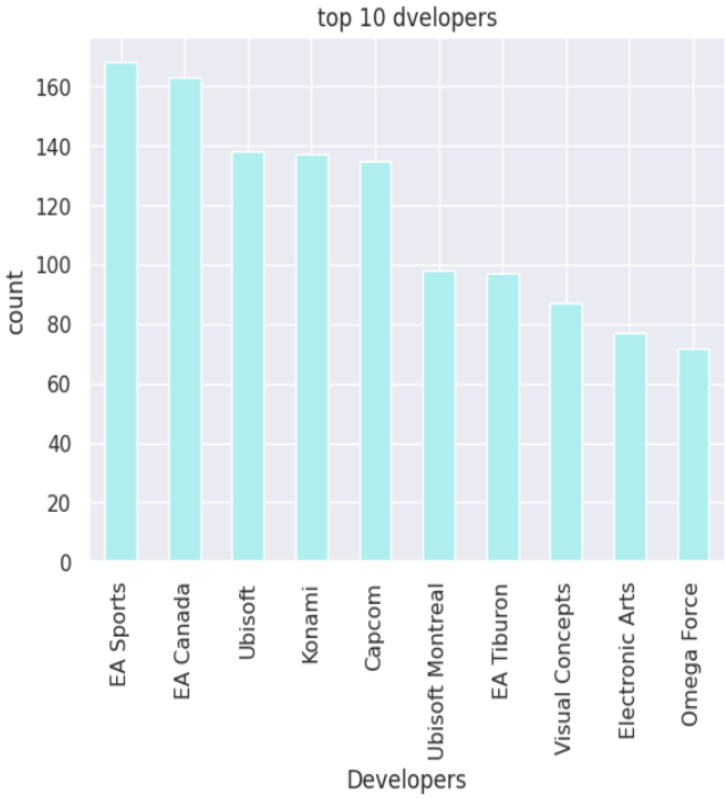


The highest Percentage of game released under each rating was E- 40.1%, followed by T 29.8%, M-15.7% and E10 with minor difference 14.3%.

Top 10 Developers

```
value_counts = df_cleaned['Developer'].value_counts()
top_10_values = value_counts.head(10)
bargraph = top_10_values.plot.bar(x = 'Developer',xlabel='Developers',ylabel='count', color = 'paleturquoise')
plt.title("top 10 dvelopers")
```

```
Text(0.5, 1.0, 'top 10 dvelopers')
```

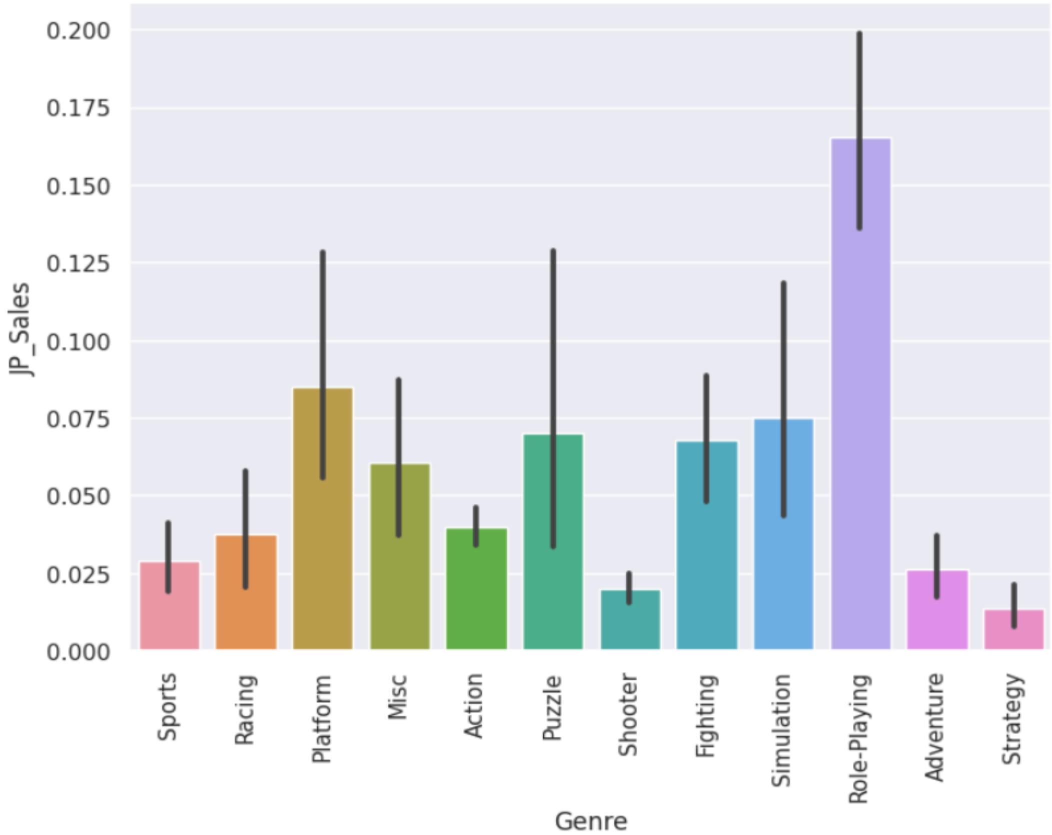


EA Sports tops this chart by a count of 170, followed by EA Canada. Konami and Ubisoft are battling closely for the third position with 130+ count. Omega Force is the lowest, who developed less than 50% of the toppers. Ubisoft Montreal and EA Tiburon were in a close neck to neck competition, with approx. 90 counts.

Game Sales of Different Genres in Japan

```
plt.figure(figsize=(8,6))
plt.xticks(rotation=90)
sns.barplot(data=df_cleaned, x="Genre", y="JP_Sales")
```

```
<Axes: xlabel='Genre', ylabel='JP_Sales'>
```

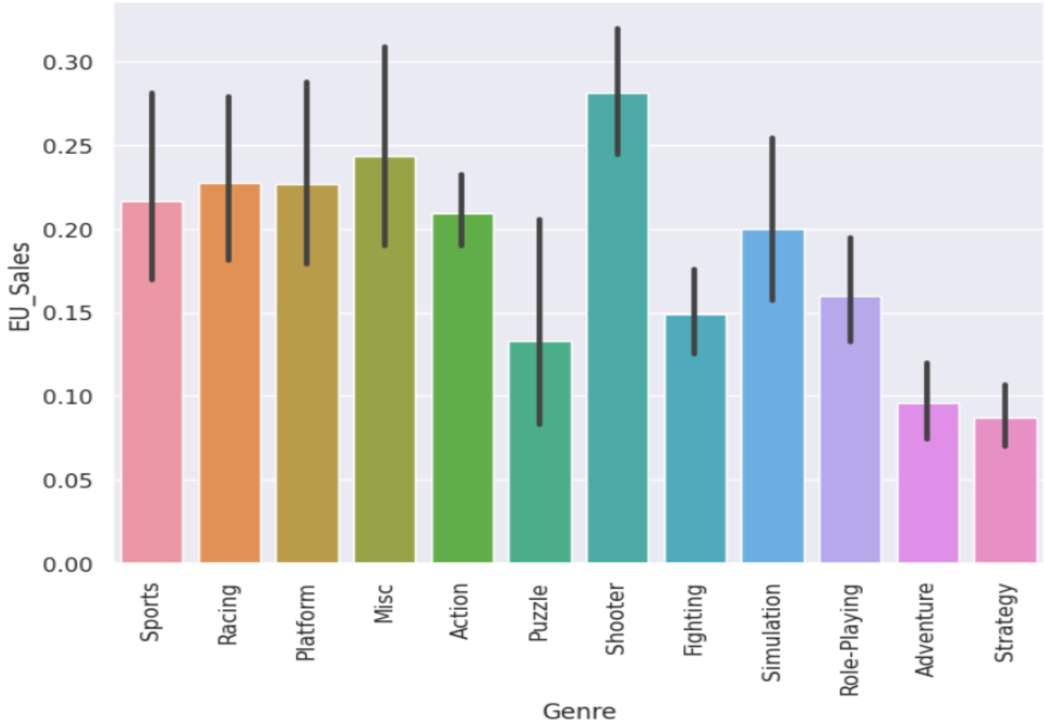


Role-Playing genre tops the chart in Japan, and strategy genre has the least sales. Platform has 50% less sales than Role-playing.

Game Sales of Different Genres in Europe

```
plt.figure(figsize=(8,6))
plt.xticks(rotation=90)
sns.barplot(data=df_cleaned, x="Genre", y="EU_Sales")
```

```
<Axes: xlabel='Genre', ylabel='EU_Sales'>
```

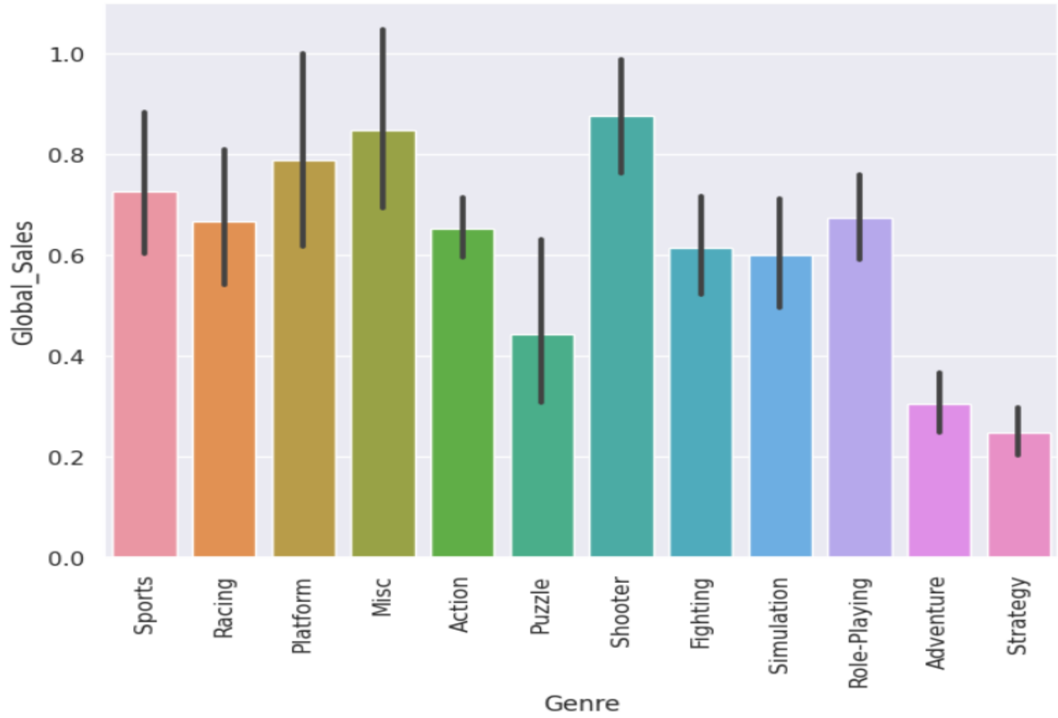


The Shooter genre has the highest sales in EU, followed by misc. Racing and platform-oriented games are on tie with similar sales figures. Simulation also performed well with 30% less sales than shooter. Strategy has the lowest sales in EU.

Game Sales of Different Genres in Worldwide

```
plt.figure(figsize=(8,6))
plt.xticks(rotation=90)
sns.barplot(data=df_cleaned, x="Genre", y="Global_Sales")
```

```
<Axes: xlabel='Genre', ylabel='Global_Sales'>
```

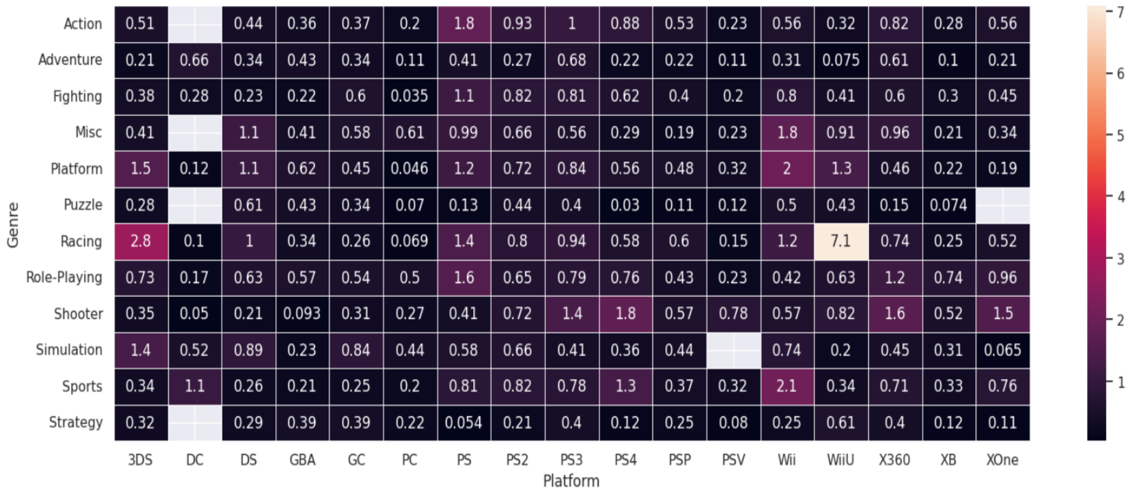


The shooter games topped the world-wide ranking in sales, followed by misc. and platform, and strategy at the lowest. These statistics were like EU sales. Sports and Roleplaying were moderate performers.

Heatmap on Global Sales with Different Genre and Platform

```
sns.set_theme()
data_heat = df_cleaned.pivot_table(index='Genre', columns='Platform', values='Global_Sales')
# Draws a heatmap with the numeric values in each cell
f, ax = plt.subplots(figsize=(18, 6))
sns.heatmap(data_heat, annot=True, linewidths=.5, ax=ax)
```

```
<Axes: xlabel='Platform', ylabel='Genre'>
```



The heatmap shows the activity correlation between Genre and platform. While the dark color shows less magnitude and lighter shade shows high magnitude. The highest correlation magnitude is between Racing games and Wiiu. Racing games are also popular on the 3DS platform. Shooter games have more activity on PS4. Roleplaying games show more activity on PS.

CONCLUSION

Throughout the data analysis process, we used a dataset comprising 16 columns and 16,719 rows about the sales of videogames. Null values, which can disrupt analytical accuracy, were carefully addressed to make sure there were no null values present and removed rows which had very less amount of data. In instances where information was missing in the "User_score" and "Critic_score" columns, we filled the user score value to critic score or vice versa based on the missing data to rectify the empty values, resulting in a more complete dataset. Furthermore, the notation "tbd" was standardized to null values for consistency. To streamline the dataset for visual representation, we made data type conversions for consistency and compatibility. Columns which deemed to have minimal relevance to the visualization objectives, were dropped.

The visualization phase showed the relationships between each column. We then plotted bar plots showing the regional sales dynamics, differentiating preferences among European, Japanese, and Global markets, particularly concerning game genres to see which genre had people prefer the most and least. This comprehensive analysis revealed substantial insights, enriching our understanding of the dataset.

Python code:
https://colab.research.google.com/drive/1xISRqO1bonT9pCmwvVBojGdplgSZFBwg?usp=sharing