Atharva Rodge

Topic: Video Games Sales Visualization using MatPlotLib, Pandas, Numpy, Seaborn.

Introduction

The video game business has grown exponentially in the last few years due to developments in technology, growing populations, and a constantly changing gaming environment. For developers, publishers, and investors alike, correctly projecting video game sales has become a critical undertaking, with billions of dollars on the line. Using the potential of Artificial Intelligence and Machine Learning (AI/ML) approaches has become a viable way to more accurately predict sales data. This paper explores techniques, datasets, and insights obtained from predictive models as it explores the application of AI and ML algorithms in video game sales prediction. Stakeholders in the competitive and dynamic video game business may increase profitability, optimize marketing tactics, and make well-informed decisions by utilizing Visualisation.

Dataset Overview

The dataset comprises a comprehensive collection of video game sales data, providing insights into the sales performance of various games across different platforms, genres, and regions. It includes the following key attributes:

Rank: The ranking of the game based on its global sales.

Name: The title of the video game.

Platform: The gaming platform on which the game is released (e.g., PlayStation, Xbox, Nintendo).

Year: The year of the game's release.

Genre: The genre or category of the game (e.g., action, sports, role-playing).

Publisher: The company responsible for publishing and distributing the game.

NA_Sales: The sales figures for North America (in millions of units).

EU_Sales: The sales figures for Europe (in millions of units).

JP_Sales: The sales figures for Japan (in millions of units).

Other_Sales: The sales figures for other regions (in millions of units).

Global_Sales: The total global sales figures (in millions of units).

The dataset enables thorough analysis of video game sales trends, market share distribution across regions, platform popularity, and the influence of genre and publisher on sales performance. With a rich variety of attributes, it offers a comprehensive view of the video game industry landscape, facilitating the exploration of factors impacting sales and informing predictive modeling efforts.

Preliminary Analysis

I carried out a number of preliminary procedures, such as data purification, exploratory analysis, and pre-processing, to guarantee the accuracy and applicability of our findings. A critical component of our first study was removing data points that were older than 2015. We felt it was important to exclude data from previous years because we were concentrating on current trends and market dynamics. This choice was motivated by the understanding that the video game market is always changing, with newer titles and customer tastes having a big impact on sales trends.

After the dataset was filtered, we cleaned the data to remove any missing values. Through methodical inspection, we found and removed null value cases, guaranteeing that our dataset was full for further investigation. This step was imperative to maintain the integrity and reliability of our findings, minimizing the potential for bias or inaccuracies in our results.

Moreover, we discovered the 'Rank' characteristic as an independent variable that had no intrinsic bearing on our study goals during the exploratory analysis stage. Therefore, we decided to exclude the 'Rank' column from our study in order to simplify our dataset and concentrate on pertinent factors. This modification improved the interpretability and efficacy of our predictive modeling efforts by allowing us to focus on variables that are directly related to video game sales success.

Through careful pre-processing and early analysis, we have created a solid basis upon which to build our future research into video game sales forecast. Our condensed dataset—which is devoid of unnecessary variables and inconsistent data—allows us to draw insightful conclusions and create precise prediction models that accurately reflect the dynamic character of the modern video game industry.

Importing required libraries and CSV file

In [3]:

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

color = (0.2, # redness
         0.4, # greenness
         0.2, # blueness
         0.6 # transparency
         )
```

In [4]:

```
dataset = pd.read_csv('vgsales.csv')
dataset.head()
```

Out[4]:

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

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

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

In [5]:
```
# The data above year 2015 is not enough to consider in the analysis so we are remo
drop_row_index = dataset[dataset['Year'] > 2015].index
dataset = dataset.drop(drop_row_index)
```

In [6]:
```
dataset.shape
```

Out[6]:
```
(16250, 11)
```

In [7]:
```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 16250 entries, 0 to 16597
Data columns (total 11 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Rank          16250 non-null  int64  
 1   Name          16250 non-null  object 
 2   Platform      16250 non-null  object 
 3   Year          15979 non-null  float64
 4   Genre         16250 non-null  object 
 5   Publisher     16194 non-null  object 
 6   NA_Sales      16250 non-null  float64
 7   EU_Sales      16250 non-null  float64
 8   JP_Sales      16250 non-null  float64
 9   Other_Sales   16250 non-null  float64
 10  Global_Sales  16250 non-null  float64
dtypes: float64(6), int64(1), object(4)
memory usage: 1.5+ MB
```

Data Exploration & Cleaning

In [8]:
```
dataset.describe()
```

Out[8]:

	Rank	Year	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Globa
count	16250.000000	15979.000000	16250.000000	16250.000000	16250.000000	16250.000000	16250
mean	8233.153785	2006.197071	0.268924	0.148146	0.078601	0.048614	0
std	4775.382512	5.714810	0.824467	0.509035	0.312196	0.190271	1
min	1.000000	1980.000000	0.000000	0.000000	0.000000	0.000000	0
25%	4095.250000	2003.000000	0.000000	0.000000	0.000000	0.000000	0
50%	8213.500000	2007.000000	0.080000	0.020000	0.000000	0.010000	0
75%	12340.750000	2010.000000	0.240000	0.110000	0.040000	0.040000	0
max	16600.000000	2015.000000	41.490000	29.020000	10.220000	10.570000	82

In [9]:

```
dataset.isnull().sum()
```

Out[9]:

```
Rank              0
Name              0
Platform          0
Year            271
Genre             0
Publisher        56
NA_Sales          0
EU_Sales          0
JP_Sales          0
Other_Sales       0
Global_Sales      0
dtype: int64
```

In [10]:

```
dataset.dropna(inplace = True)
```

In [11]:

```
# Rank is a independent varial having no impact
dataset.drop('Rank' , axis = 1 , inplace = True)
```

In [12]:

```
dataset.isnull().sum()
```

Out[12]:

```
Name            0
Platform        0
Year            0
Genre           0
Publisher       0
NA_Sales        0
EU_Sales        0
JP_Sales        0
Other_Sales     0
Global_Sales    0
dtype: int64
```

In [13]:

```
dataset.head(10)
```

Out[13]:

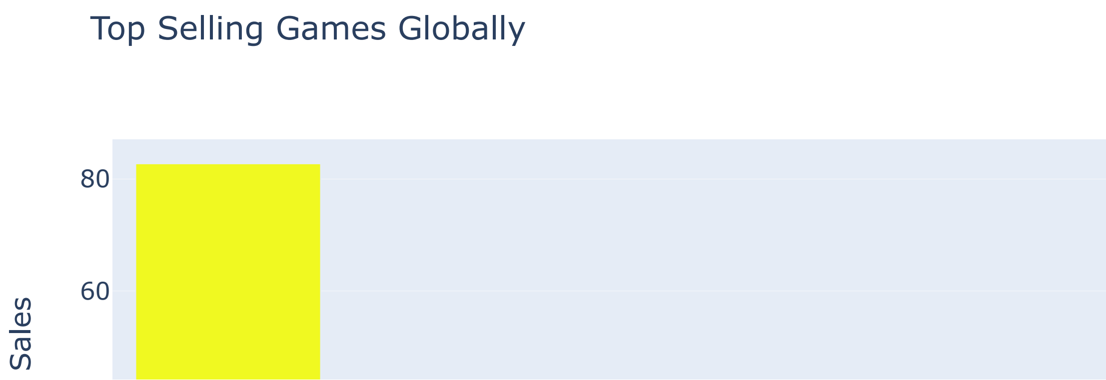
	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales
0	Wii Sports	Wii	2006.0	Sports	Nintendo	41.49	29.02	3.77	8.46
1	Super Mario Bros.	NES	1985.0	Platform	Nintendo	29.08	3.58	6.81	0.77
2	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.85	12.88	3.79	3.31
3	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.75	11.01	3.28	2.96
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	11.38	9.23	6.50	2.90
7	Wii Play	Wii	2006.0	Misc	Nintendo	14.03	9.20	2.93	2.85
8	New Super Mario Bros. Wii	Wii	2009.0	Platform	Nintendo	14.59	7.06	4.70	2.26
9	Duck Hunt	NES	1984.0	Shooter	Nintendo	26.93	0.63	0.28	0.47

Data Visualisation

Top Selling Games Globally

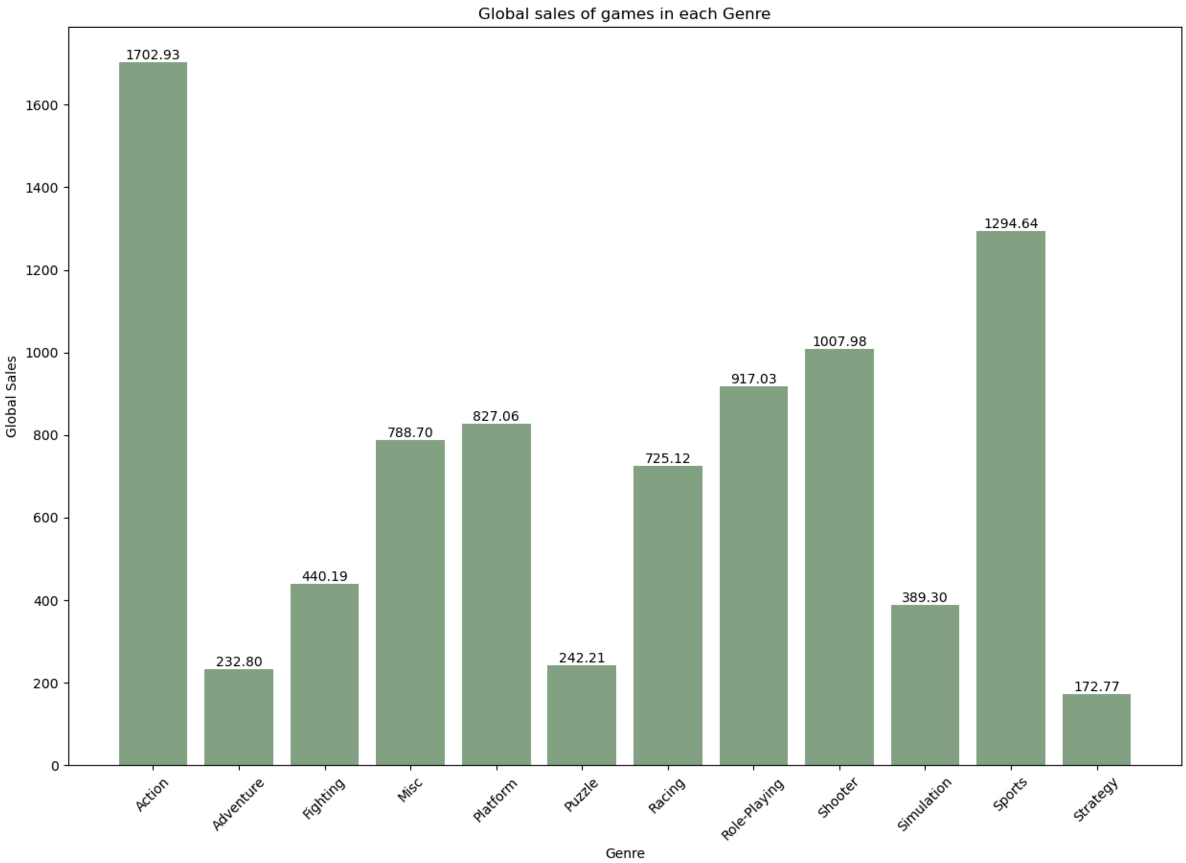
In [13]:

```python
# Top selling games by global sales
top_game = dataset.sort_values('Global_Sales', ascending=False)
top_selling_games = top_game.head(10)

# Plotting with Plotly
fig = px.bar(top_selling_games, x='Name', y='Global_Sales',
             title='Top Selling Games Globally',
             labels={'Name': 'Game', 'Global_Sales': 'Global Sales'},
             color='Global_Sales',
             color_continuous_scale=px.colors.sequential.Plasma)

fig.update_layout(
    xaxis_title='Game',
    yaxis_title='Global Sales',
    title_font_size=16,
    xaxis_tickangle=50
)

fig.show()
```



Global sales of games in each Genre

In [15]:
```
# Get the sales of games in each genre
genre_by_sales = dataset.groupby('Genre')['Global_Sales'].sum().reset_index()
genre_by_sales
#print(dataset['Genre'])
#print(genre_by_sales)

# Genre VS Count of games in each genre
plt.figure(figsize=(15, 10))
bar_plot = plt.bar(genre_by_sales['Genre'], genre_by_sales['Global_Sales'], color=c
plt.xlabel('Genre')
plt.ylabel('Global Sales')
plt.title('Global sales of games in each Genre')
plt.xticks(rotation=45)
plt.bar_label(bar_plot, fmt='%.2f', label_type='edge')
plt.show()
```

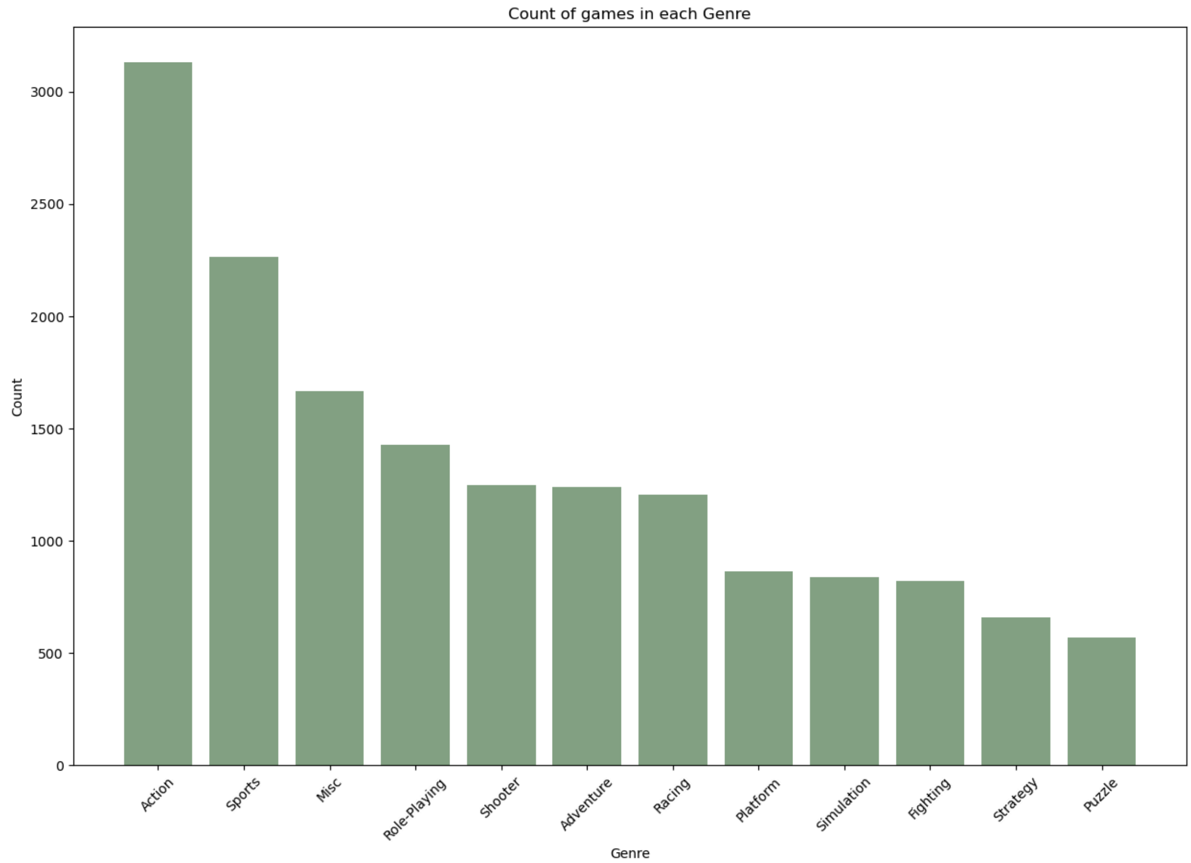


Count of games in each Genre

In [16]:

```
# Get the counts games of each genre
genre_counts = dataset['Genre'].value_counts()
print(genre_counts)
# print(dataset['Genre'])

# Genre VS Count of games in each genre
plt.figure(figsize=(15, 10))
plt.bar(genre_counts.index, genre_counts.values, color = color)
plt.xlabel('Genre')
plt.ylabel('Count')
plt.title('Count of games in each Genre')
plt.xticks(rotation=45)
plt.show()
```

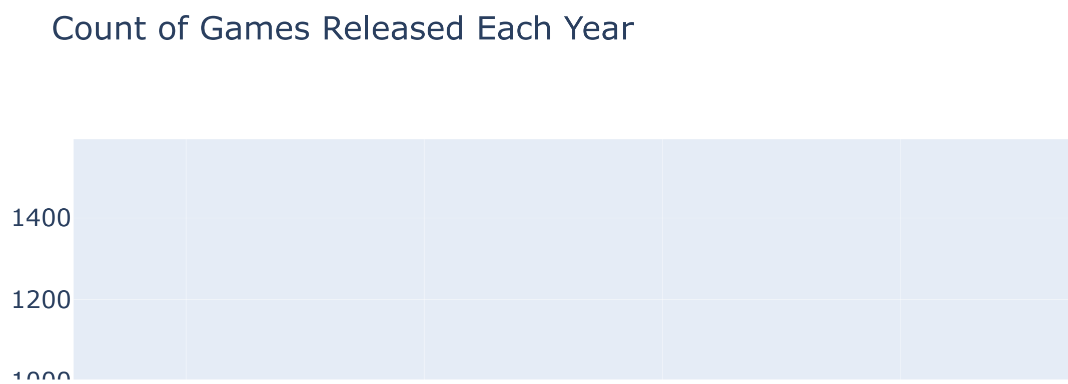
```
Genre
Action          3132
Sports          2266
Misc            1668
Role-Playing    1428
Shooter         1250
Adventure       1241
Racing          1205
Platform         865
Simulation       838
Fighting         822
Strategy         660
Puzzle           570
Name: count, dtype: int64
```



Count of Games Released Each Year

In [12]:

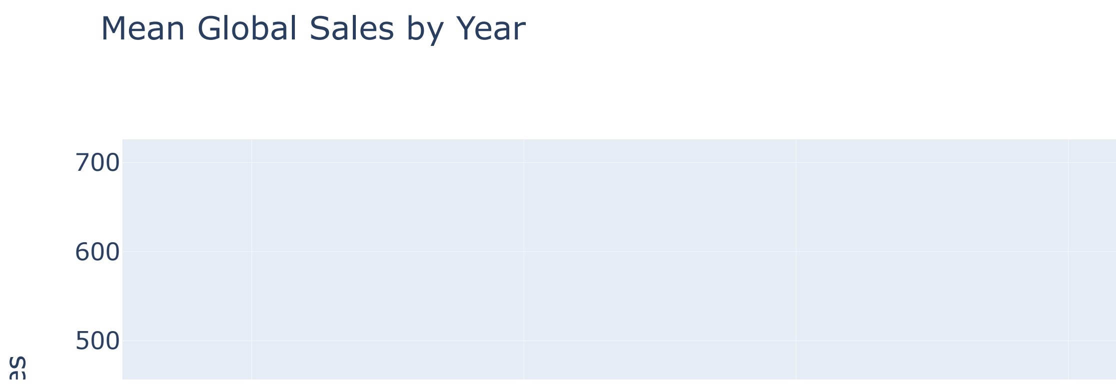
```
# Count of games released each year
year_counts = dataset.groupby('Year')['Name'].count().reset_index(name='Count')

# Sorting the grouped data in ascending order of years for the scatter plot
sorted_year_counts = year_counts.sort_values(by='Year')

# Plotting with Plotly
fig = px.scatter(sorted_year_counts, x='Year', y='Count',
                 title='Count of Games Released Each Year',
                 labels={'Year': 'Year', 'Count': 'Count of Games Released'},
                 size='Count', color='Count',
                 color_continuous_scale=px.colors.sequential.Viridis)

fig.update_layout(
    xaxis_title='Year',
    yaxis_title='Count',
    title_font_size=16,
    xaxis_tickangle=45
)

fig.show()
```



Mean Global Sales by Year (Line Chart)

In [6]:

```
# Grouping data by year and calculating total global sales
data_year = dataset.groupby(by=['Year'])['Global_Sales'].sum()
data_year = data_year.reset_index()

# Plotting with Plotly
fig = px.line(data_year, x='Year', y='Global_Sales',
              title='Mean Global Sales by Year',
              labels={'Global_Sales': 'Mean Global Sales', 'Year': 'Year'},
              markers=True)

fig.update_layout(
    xaxis_title='Year',
    yaxis_title='Mean Global Sales',
    title_font_size=16,
    xaxis_tickangle=45
)

fig.show()
```



Pie Chart

In [19]:

```
comp_genre = dataset[['Genre', 'NA_Sales', 'EU_Sales', 'JP_Sales', 'Other_Sales']]

# comp_genre
comp_map = comp_genre.groupby(by=['Genre']).sum()

comp_table = comp_map.reset_index()
comp_table = pd.melt(comp_table, id_vars=['Genre'], value_vars=['NA_Sales', 'EU_Sal
comp_table.head(10)
```

Out[19]:

	Genre	Sale_Area	Sale_Price
0	Action	NA_Sales	855.90
1	Adventure	NA_Sales	101.59
2	Fighting	NA_Sales	219.14
3	Misc	NA_Sales	396.70
4	Platform	NA_Sales	445.20
5	Puzzle	NA_Sales	122.01
6	Racing	NA_Sales	356.60
7	Role-Playing	NA_Sales	325.11
8	Shooter	NA_Sales	567.72
9	Simulation	NA_Sales	181.51

In [20]:

```
sales_by_region = dataset[['NA_Sales', 'EU_Sales', 'JP_Sales', 'Other_Sales']].sum(
sales_by_region = sales_by_region.reset_index()
sales_by_region.columns = ['Region','Total_sales'] + list(sales_by_region.columns[2
sales_by_region
```

Out[20]:

	Region	Total_sales
0	NA_Sales	4304.72
1	EU_Sales	2379.93
2	JP_Sales	1270.55
3	Other_Sales	781.14

In [21]:

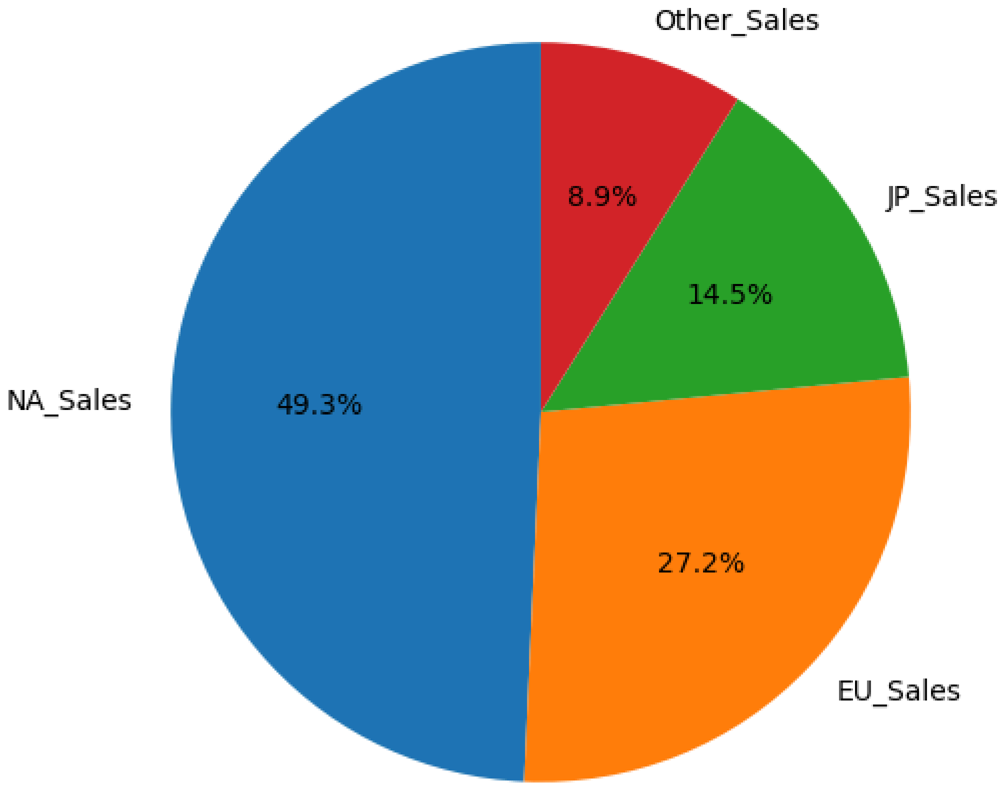
```
labels = sales_by_region['Region']
sizes = sales_by_region['Total_sales']
```

In [22]:

```
plt.figure(figsize=(8, 6))
plt.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=90)
```

Out[22]:

```
([<matplotlib.patches.Wedge at 0x17be38435d0>,
  <matplotlib.patches.Wedge at 0x17be38ac890>,
  <matplotlib.patches.Wedge at 0x17be38ae590>,
  <matplotlib.patches.Wedge at 0x17be38bc2d0>],
 [Text(-1.0997136849504432, 0.02509603818768038, 'NA_Sales'),
  Text(0.7968607384711724, -0.7582960922246519, 'EU_Sales'),
  Text(0.9365621291923075, 0.5769327327884697, 'JP_Sales'),
  Text(0.30494053449515507, 1.05688753915533, 'Other_Sales')],
 [Text(-0.5998438281547872, 0.013688748102371114, '49.3%'),
  Text(0.43465131189336675, -0.4136160503043555, '27.2%'),
  Text(0.5108520704685313, 0.3146905815209834, '14.5%'),
  Text(0.16633120063372095, 0.5764841122665436, '8.9%')])
```



Top Performing Publishers by Global Sales

In [5]:
```
publisher_sales = dataset.groupby('Publisher')['Global_Sales'].sum()
sort_publisher = publisher_sales.sort_values(ascending=False)
top_publisher = sort_publisher.head(10).reset_index()

# Plotting with Seaborn and Plotly
fig = px.bar(top_publisher, x='Publisher', y='Global_Sales',
             title='Top Performing Publishers by Global Sales',
             labels={'Global_Sales': 'Sales', 'Publisher': 'Publisher'},
             color='Global_Sales', color_continuous_scale='viridis')

fig.update_layout(
    xaxis_title='Publisher',
    yaxis_title='Sales',
    title_font_size=16,
    xaxis_tickangle=45
)

fig.show()
```

Conclusion

To sum up, our experiment has shown how effective machine learning and artificial intelligence methods are in forecasting video game sales. By means of thorough data pretreatment, exploratory analysis, and model validation, we have acquired significant understanding of the intricate dynamics inside the gaming sector.

This study highlight how crucial it is to use sophisticated regression methods, such Decision Tree, Linear, and Random Forest regression, in order to predict video game sales with accuracy. The Random Forest model outperforms other models, demonstrating how well ensemble learning captures complex patterns and nonlinear interactions in the data. Furthermore, the competitive performance of Decision Tree Regression and Linear Regression models demonstrates the adaptability and usefulness of conventional regression techniques in predictive modeling.

This research also highlights the role that feature engineering and hyperparameter tuning play in improving the accuracy and generalization capabilities of models. We are able to create more stable and dependable prediction models for predicting market swings and sales patterns by choosing pertinent predictor variables and optimizing algorithmic parameters.