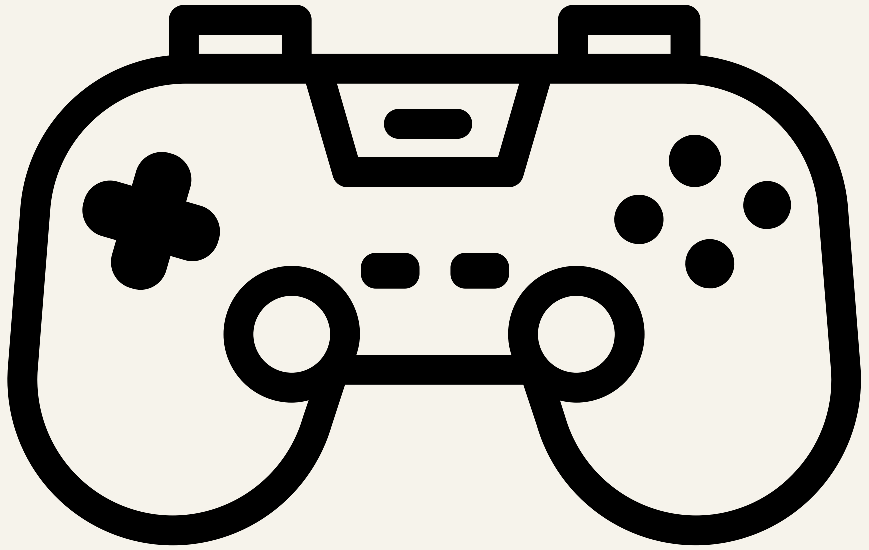
ANALYZING VIDEO GAME SALES DATA



GROUP MEMBERS

Fadiyah alanazi	2201001182
Ghazlan alanazi	2201000995
Elham khatim	2201001444
Ebtisam falih	2201002692
Reem alqahtani	2201003207
Latifah alanazi	2201002712

OUTLINE

- Introduction
- opjective
- goal
- dataset ovrview
- data exploration
- data cleaning & preprocessing
- data visualzation
- ML almplementation
- Conclusion
- Thank You

INTRODUCTION

The video game industry has seen remarkable growth over the years, becoming a multi-billion dollar global market. The availability of data on video game sales provides a unique opportunity to gain insights into player preferences, market trends, and the evolution of the industry The world of video games is a vast and dynamic industry that has captivated millions of players and generated billions of dollars in revenue over the years. With the advancement of technology and the emergence of various gaming platforms, understanding the trends and patterns within the video game market has become increasingly important.

INTRODUCTION

In this project, we delve into the fascinating realm of video game sales data. We utilize the "Video Game Sales" dataset obtained from Kaggle, which contains a wealth of information about video game titles, their platforms, genres, release years, and sales figures in different regions. By analyzing this dataset, we aim to gain valuable insights into the world of video games, uncover trends,

OBJECTIVES

Objective 1

Visualize key trends and patterns in video game sales data.

Objective 2

Extract insights into the most popular platforms, genres, and sales trends.

GOAL

Throughout this project, we will employ various data analysis and visualization techniques to extract meaningful information from the dataset. We will also perform data cleaning and preprocessing to ensure the quality and reliability of our analyses.

DATASET OVERVIEW

Source: Kaggle's "Video Game Sales" dataset
- Number of Rows: [16598]
- Number of Columns: [11]

So, let's embark on this journey to uncover the secrets hidden within the world of video game sales data!

DATA EXPLORATION

First of all, let's get familiarized with our data content. So, We displayed the first top records of our dataset.

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

DATA EXPLORATION

As shown below, the number of records is 16598 and the number of columns is 11.

	Rank	Name	Platform	Year	Genre	Publisher	Na_sales	Eu_sales	Jp_sales	Other_sales	Global_sales
0	1	Wii Sports	Wii	2006.0	Sports	Nintendo	41.49	29.02	3.77	8.46	82.74
1	2	Super Mario Bros.	NES	1985.0	Platform	Nintendo	29.08	3.58	6.81	0.77	40.24
2	3	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.85	12.88	3.79	3.31	35.82
3	4	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.75	11.01	3.28	2.96	33.00
4	5	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	1.00	31.37
...	...	...	...	...	...	...	...	...	...	...	...
16593	16596	Woody Woodpecker in Crazy Castle 5	GBA	2002.0	Platform	Kemco	0.01	0.00	0.00	0.00	0.01
16594	16597	Men in Black II: Alien Escape	GC	2003.0	Shooter	Infogrames	0.01	0.00	0.00	0.00	0.01
16595	16598	SCORE International Baja 1000: The Official Game	PS2	2008.0	Racing	Activision	0.00	0.00	0.00	0.00	0.01
16596	16599	Know How 2	DS	2010.0	Puzzle	7G//AMES	0.00	0.01	0.00	0.00	0.01
16597	16600	Spirits & Spells	GBA	2003.0	Platform	Wanadoo	0.01	0.00	0.00	0.00	0.01

16598 rows × 11 columns

DATA EXPLORATION

Now, let's know more about the columns and their data types in addition to viewing statistical factors about each column. , The columns' data types seem to be logical and we have some erorr

```
In [18]:  df.dtypes

Out[18]:  Rank              int64
          Name             object
          Platform         object
          Year            float64
          Genre            object
          Publisher        object
          Na_sales        float64
          Eu_sales        float64
          Jp_sales        float64
          Other_sales     float64
          Global_sales    float64
          dtype: object
```

DATA EXPLORATION

The describe method is used to view basic information about each column. It shows The values count The values' average ,standard deviation, minimum value ,maximum value.

In [65]:
```
df.describe()
```

Out[65]:

	Rank	Year	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
count	16598.000000	16327.000000	16598.000000	16598.000000	16598.000000	16598.000000	16598.000000
mean	8300.605254	2006.406443	0.264667	0.146652	0.077782	0.048063	0.537441
std	4791.853933	5.828981	0.816683	0.505351	0.309291	0.188588	1.555028
min	1.000000	1980.000000	0.000000	0.000000	0.000000	0.000000	0.010000
25%	4151.250000	2003.000000	0.000000	0.000000	0.000000	0.000000	0.060000
50%	8300.500000	2007.000000	0.080000	0.020000	0.000000	0.010000	0.170000
75%	12449.750000	2010.000000	0.240000	0.110000	0.040000	0.040000	0.470000
max	16600.000000	2020.000000	41.490000	29.020000	10.220000	10.570000	82.740000

DATA CLEANING & PREPROCESSING

Check the missing values in each column We checked the missing values in each column n 2 different ways by counting the missing values and plotting them. our dataset have some erorr and we can handle with erorrs.

```
In [72]:  # Data Cleaning
          # Handle missing values
          df.isnull().sum()

Out[72]:  Rank              0
          Name              0
          Platform          0
          Year            271
          Genre             0
          Publisher        58
          NA_Sales          0
          EU_Sales          0
          JP_Sales          0
          Other_Sales       0
          Global_Sales      0
          dtype: int64
```

```
In [19]:  plt.figure(figsize=(10, 6))
          sns.heatmap(df.isnull(), cmap='viridis', cbar=False)
          plt.title('Missing Value Heatmap')
          plt.show()
```

DATA CLEANING & PREPROCESSING

Dropping rows with missing values using df.dropna(). And check after dropping
-Print clumnus with missing values and cunting of missing values:

Drop the missing values

```
In [73]: print(df.shape)
         df = df.dropna()
         print(df.shape)

(16598, 11)
(16291, 11)

In [75]: df.isnull().sum()

Out[75]: Rank            0
         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 [48]: # Print columns with missing values and the count of missing values
         print("Columns with Missing Values:")
         print(missing_values[missing_values > 0])

Columns with Missing Values:
Series([], dtype: int64)
```

DATA CLEANING & PREPROCESSING

We choose to handle missing values by dropping rows with missing values using data.dropna(inplace=True) and we verify that there are no more missing value.

In [79]:
```python
# Handle missing values (you can choose your preferred method)
# Example: Drop rows with missing values
df.dropna(inplace=True)
```

Finally, we verify that there are no more missing values.

In [80]:
```python
# Verify that there are no more missing values
missing_values_after_handling = df.isnull().sum()
print("\nColumns with Missing Values after Handling:")
print(missing_values_after_handling[missing_values_after_handling > 0])
```

```
Columns with Missing Values after Handling:
Series([], dtype: int64)
```

DATA CLEANING & PREPROCESSING

Check the duplicates

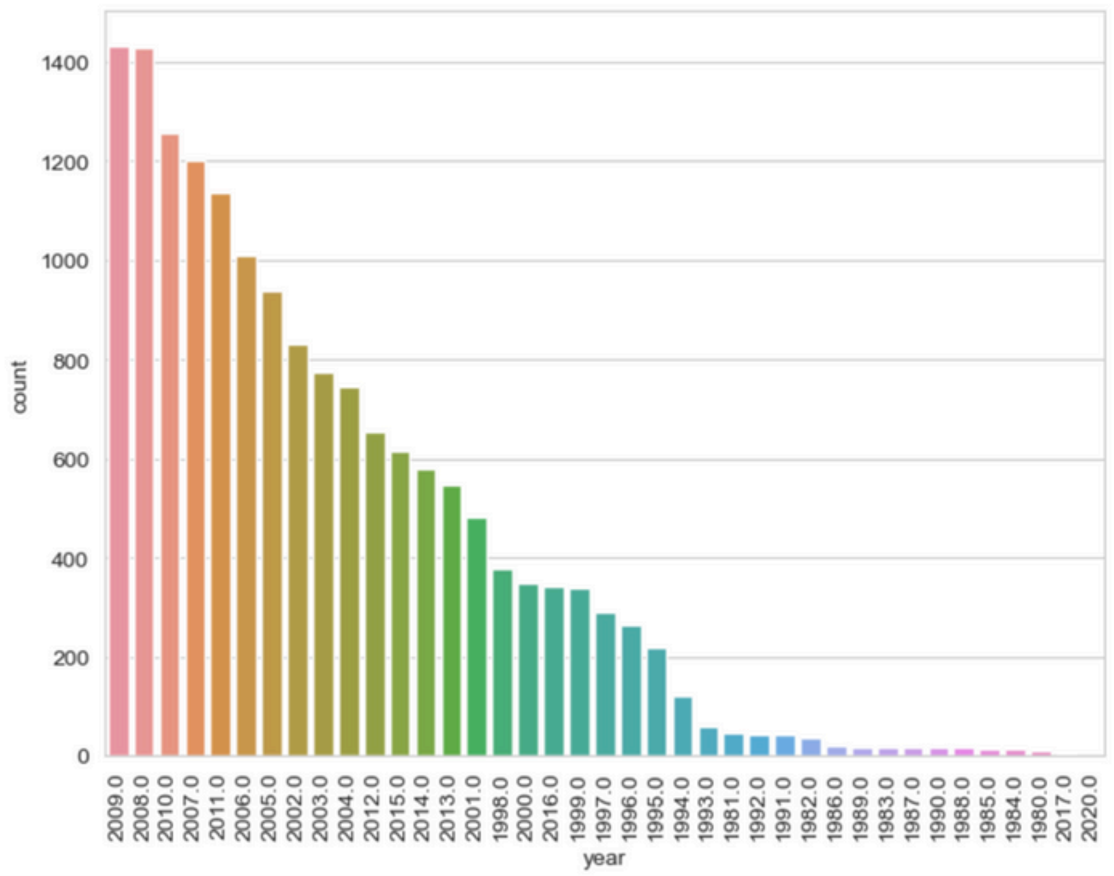
As shown above, we don't have duplicates .

```
In [82]:  df.columns = df.columns.str.strip().str.lower()

In [22]:  duplicates = df.duplicated().sum()

In [23]:  print("Duplicate Rows:")
          print(duplicates)

          Duplicate Rows:
          0
```

DATA VISUALIZATION

A bar chart showing global sales by platform

```
plt.figure(figsize=(8, 6))
sns.countplot(x="year", data=df, order = df.groupby(by=['year'])['name'].count().sort_values(ascending=False).index)
plt.xticks(rotation=90)
plt.show()
```

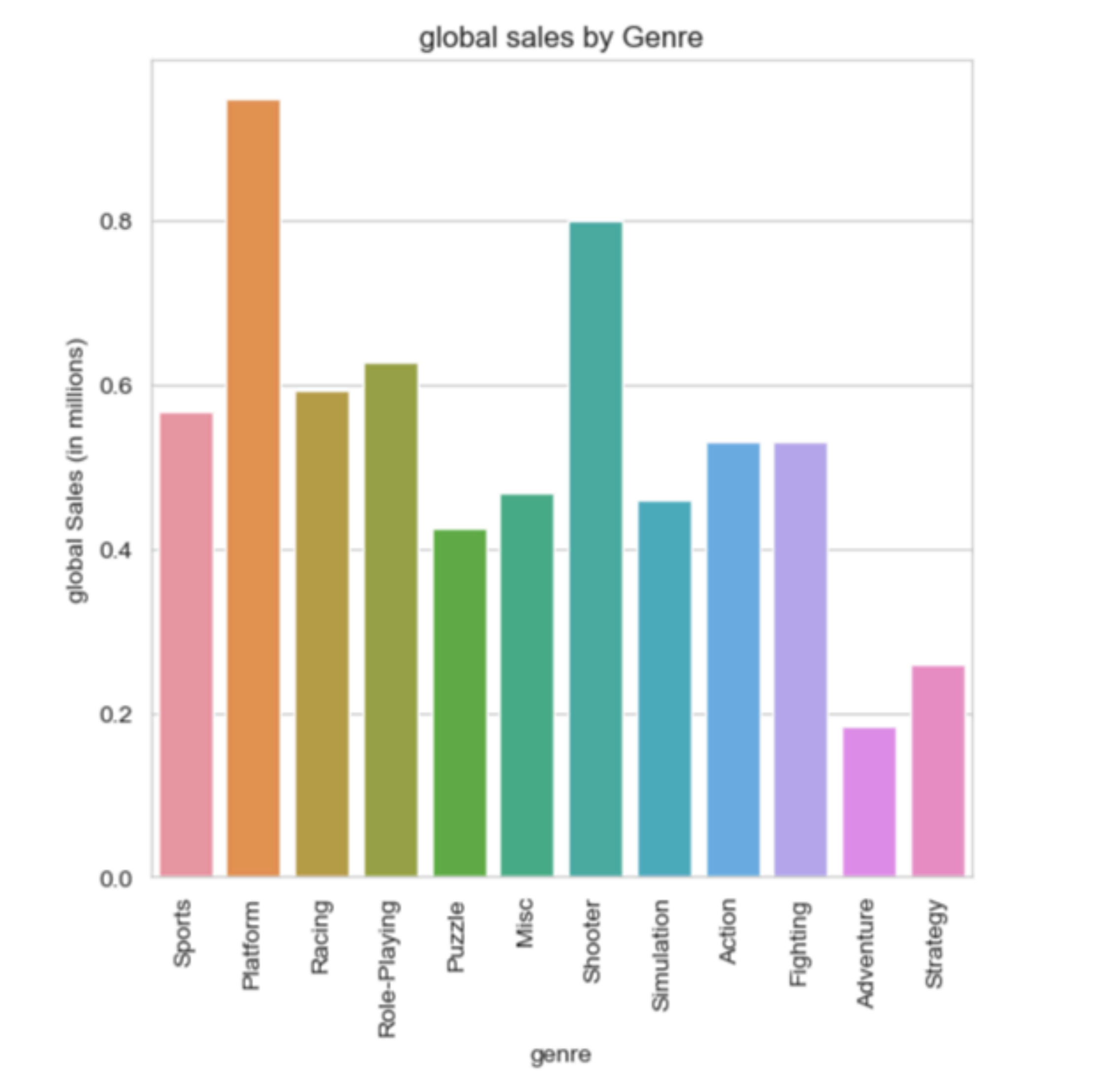


This line displays the plot on the screen.

The resulting figure is a bar plot (countplot) showing the number of video game releases per year. The years are displayed on the x-axis, and the count of game releases is shown on the y-axis. The bars are ordered in descending order of game releases, which means the year with the most game releases will appear first on the left. Rotating the x-axis labels by 90 degrees helps prevent label overlap and makes it easier to read the years

DATA VISUALIZATION

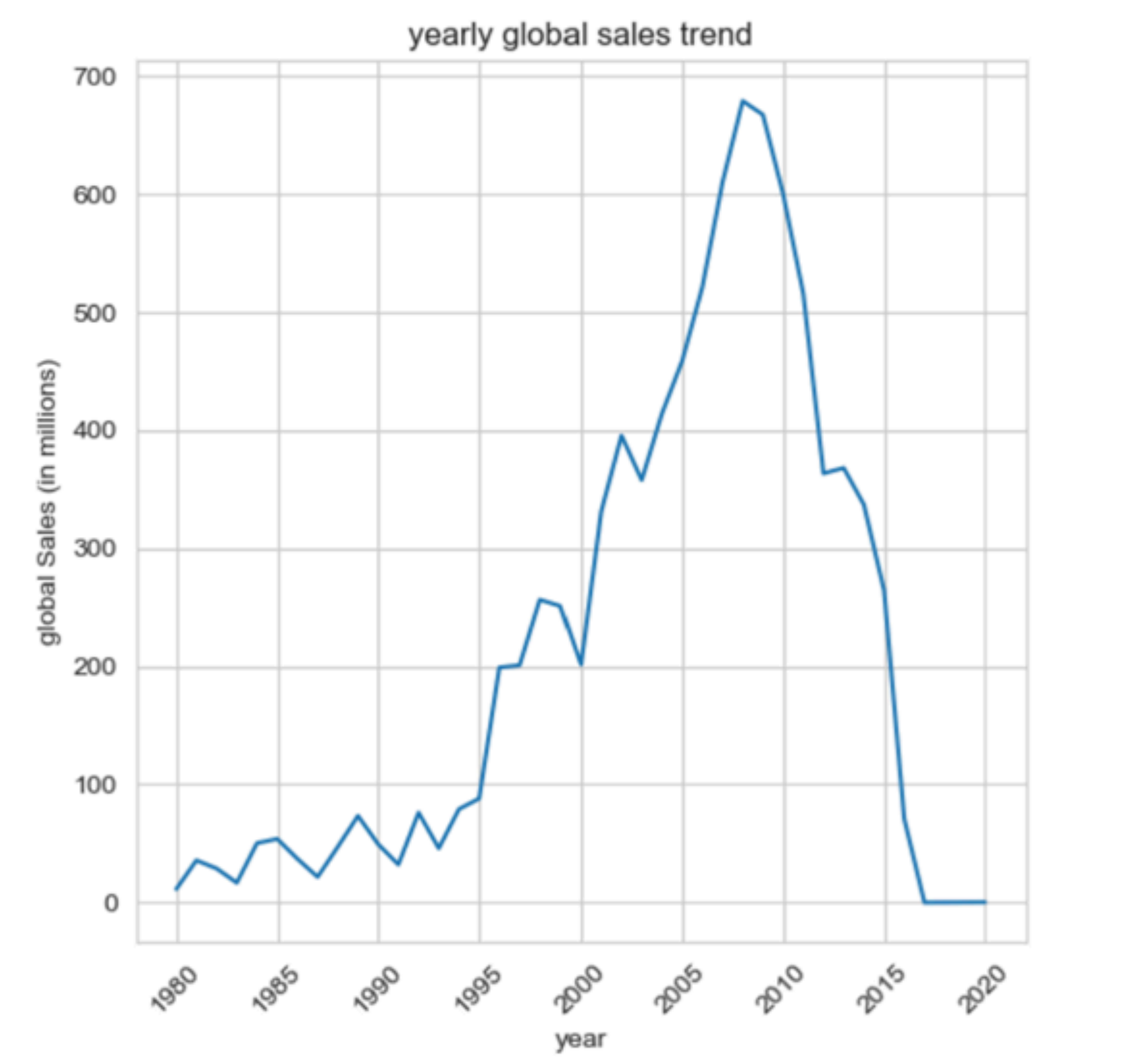
A bar chart showing global sales by genre.



This line displays the plot on the screen.
The resulting figure is a bar chart that shows global sales by genre. Each genre is represented on the x-axis, and the global sales (in millions) are displayed on the y-axis. The bars represent the sales for each genre. The title, axis labels, and rotated x-axis labels make the chart more informative and readable.

DATA VISUALIZATION

A line plot showing yearly global sales trends.



This line displays the plot on the screen.
The resulting figure is a line plot that shows the yearly global sales trend. Each point on the line represents the total global sales for a specific year, and the line connects these points to illustrate the trend over time. The title, axis labels, and rotated x-axis labels make the chart more informative and readable. This type of plot is useful for visualizing how global sales have changed over the years.

ML ALGORITHM IMPLEMENTATION

Implementing a machine learning algorithm typically involves several steps, including data preprocessing, model selection, training, evaluation, and prediction. Below, I'll provide a simplified example of implementing a machine learning algorithm using Python's scikit-learn library. We'll use a basic algorithm like random forest Regression for illustration.

•chooce feature

```
X = df[['na_sales', 'year' , 'rank']]#choose feature
y = df['global_sales']
```

ML ALGORITHM IMPLEMENTATION

- **Split the data into training and testing sets**

```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=10)
```

ML ALGORITHM IMPLEMENTATION

- initializing and training a Random Forest Regressor mode

```
# Initialize and train the Random Forest Regressor model
model = RandomForestRegressor(random_state=42)
model.fit(X_train, y_train)
```

ML ALGORITHM IMPLEMENTATION

- Once the model is trained, it's ready to make predictions on new data. In your case, it will predict global sales based on the features you provided during training.

```
# Make predictions on the test set
y_pred = model.predict(X_test)
```

ML ALGORITHM IMPLEMENTATION

- You can use various evaluation metrics, such as mean squared error (MSE), R-squared (R2), or others, to quantify how well the model's predictions align with the actual values and to gauge the model's predictive accuracy. These metrics will help you assess the model's performance and determine if it's suitable for your regression task.

```
# Evaluate the model's performance
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

ML ALGORITHM IMPLEMENTATION

- By comparing these metrics to baseline or other models, you can assess how well your Random Forest Regressor model performs and determine whether it meets your predictive needs.

```
print(f'Mean Squared Error: {mse:.2f}')
print(f'R-squared (R2) Score: {r2:.2f}')
# Now, you can use the trained model to make predictions on new data.
```

```
Mean Squared Error: 0.63
R-squared (R2) Score: 0.82
```

CONCLUSION

In this conversation, you learned how to build and use a Random Forest Regressor model for predicting video game global sales based on selected features. Here's a summary of the key steps and concepts .
The steps you followed represent a typical workflow for building and using a machine learning model for regression tasks. By selecting relevant features, training the model, and evaluating its performance, you can gain valuable insights and make predictions for various applications.