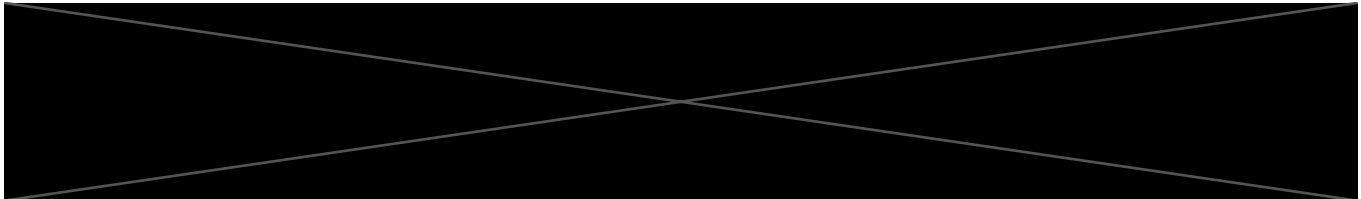


WESTYN HILLIARD



1.1 1. Load the dataset as a Pandas data frame.

```
[17]: import pandas as pd

# Load the dataset
file_path = "Video_Games_Sales.csv"
df = pd.read_csv(file_path)
```

1.2 2. Display the first ten rows of the data.

```
[18]: # Display the first ten rows of the data
print("First ten rows of the data:")
print(df.head(10))
```

First ten rows of the data:

	Name	Platform	Year_of_Release	Genre \
0	Wii Sports	Wii	2006.0	Sports
1	Super Mario Bros.	NES	1985.0	Platform

2	Mario Kart Wii	Wii	2008.0	Racing
3	Wii Sports Resort	Wii	2009.0	Sports
4	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing
5	Tetris	GB	1989.0	Puzzle
6	New Super Mario Bros.	DS	2006.0	Platform
7	Wii Play	Wii	2006.0	Misc
8	New Super Mario Bros. Wii	Wii	2009.0	Platform
9	Duck Hunt	NES	1984.0	Shooter

	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	\
0	Nintendo	41.36	28.96	3.77	8.45	82.53	
1	Nintendo	29.08	3.58	6.81	0.77	40.24	
2	Nintendo	15.68	12.76	3.79	3.29	35.52	
3	Nintendo	15.61	10.93	3.28	2.95	32.77	
4	Nintendo	11.27	8.89	10.22	1.00	31.37	
5	Nintendo	23.20	2.26	4.22	0.58	30.26	
6	Nintendo	11.28	9.14	6.50	2.88	29.80	
7	Nintendo	13.96	9.18	2.93	2.84	28.92	
8	Nintendo	14.44	6.94	4.70	2.24	28.32	
9	Nintendo	26.93	0.63	0.28	0.47	28.31	

	Critic_Score	Critic_Count	User_Score	User_Count	Developer	Rating
0	76.0	51.0	8	322.0	Nintendo	E
1	NaN	NaN	NaN	NaN	NaN	NaN
2	82.0	73.0	8.3	709.0	Nintendo	E
3	80.0	73.0	8	192.0	Nintendo	E
4	NaN	NaN	NaN	NaN	NaN	NaN
5	NaN	NaN	NaN	NaN	NaN	NaN
6	89.0	65.0	8.5	431.0	Nintendo	E
7	58.0	41.0	6.6	129.0	Nintendo	E
8	87.0	80.0	8.4	594.0	Nintendo	E
9	NaN	NaN	NaN	NaN	NaN	NaN

1.3 3. Find the dimensions (number of rows and columns) in the data frame.
What do these two numbers represent in the context of the data?

```
[19]: # Find the dimensions of the data frame
dimensions = df.shape
print("\nDimensions of the data frame (rows, columns):", dimensions)

# Explanation of the dimensions
print(f"\nThe data frame has {dimensions[0]} rows and {dimensions[1]} columns. "
      "The rows represent individual video games, and the columns represent_
↳different attributes of these games, such as title, platform, genre, sales,
↳and scores.")
```

Dimensions of the data frame (rows, columns): (16719, 16)

The data frame has 16719 rows and 16 columns. The rows represent individual video games, and the columns represent different attributes of these games, such as title, platform, genre, sales, and scores.

1.4 4. Find the top five games by critic score.

```
[20]: # Find the top five games by critic score
top_critic_score_games = df.nlargest(5, 'Critic_Score')[['Name',
↪ 'Critic_Score']]
print("\nTop five games by critic score:")
print(top_critic_score_games)
```

Top five games by critic score:

	Name	Critic_Score
51	Grand Theft Auto IV	98.0
57	Grand Theft Auto IV	98.0
227	Tony Hawk's Pro Skater 2	98.0
5350	SoulCalibur	98.0
16	Grand Theft Auto V	97.0

1.5 5. Find the number of video games in the data frame in each genre.

```
[21]: # Find the number of video games in each genre
genre_counts = df['Genre'].value_counts()
print("\nNumber of video games in each genre:")
print(genre_counts)
```

Number of video games in each genre:

Genre	
Action	3370
Sports	2348
Misc	1750
Role-Playing	1500
Shooter	1323
Adventure	1303
Racing	1249
Platform	888
Simulation	874
Fighting	849
Strategy	683
Puzzle	580

Name: count, dtype: int64

1.6 6. Find the first five games in the data frame on the SNES platform.

```
[22]: # Find the first five games on the SNES platform
snes_games = df[df['Platform'] == 'SNES'].head(5)
print("\nFirst five games on the SNES platform:")
print(snes_games[['Name', 'Platform']])
```

First five games on the SNES platform:

	Name	Platform
18	Super Mario World	SNES
56	Super Mario All-Stars	SNES
71	Donkey Kong Country	SNES
76	Super Mario Kart	SNES
137	Street Fighter II: The World Warrior	SNES

1.7 7. Find the five publishers with the highest total global sales. Note: You will need to calculate the total global sales for each publisher to do this.

```
[23]: # Find the five publishers with the highest total global sales
publisher_sales = df.groupby('Publisher')['Global_Sales'].sum().nlargest(5)
print("\nFive publishers with the highest total global sales:")
print(publisher_sales)
```

Five publishers with the highest total global sales:

Publisher	
Nintendo	1788.81
Electronic Arts	1116.96
Activision	731.16
Sony Computer Entertainment	606.48
Ubisoft	471.61

Name: Global_Sales, dtype: float64

1.8 8. Create a new column in the data frame that calculates the percentage of global sales from North America. Display the first five rows of the new data frame.

```
[24]: # Create a new column for percentage of global sales from North America
df['NA_Sales_Percentage'] = (df['NA_Sales'] / df['Global_Sales']) * 100

# Display the first five rows of the new data frame
print("\nFirst five rows with the new column for NA sales percentage:")
print(df[['Name', 'NA_Sales', 'Global_Sales', 'NA_Sales_Percentage']].head(5))
```

First five rows with the new column for NA sales percentage:

	Name	NA_Sales	Global_Sales	NA_Sales_Percentage
0	Wii Sports	41.36	82.53	50.115110

1	Super Mario Bros.	29.08	40.24	72.266402
2	Mario Kart Wii	15.68	35.52	44.144144
3	Wii Sports Resort	15.61	32.77	47.635032
4	Pokemon Red/Pokemon Blue	11.27	31.37	35.926044

1.9 9. Find the number NaN entries (missing data values) in each column..

```
[25]: # Find the number of NaN entries in each column
nan_counts = df.isna().sum()
print("\nNumber of NaN entries in each column:")
print(nan_counts)
```

Number of NaN entries in each column:

```
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
NA_Sales_Percentage 0
dtype: int64
```

1.10 10. Try to calculate the median user score of all the video games. You will likely run into an error because some of the user score entries are a non-numerical string that cannot be converted to a float. Find and replace this string with NaN and then calculate the median. Then, replace all NaN entries in the user score column with the median value.

```
[26]: # Replace non-numerical user scores with NaN
df['User_Score'] = pd.to_numeric(df['User_Score'], errors='coerce')

# Calculate the median user score
median_user_score = df['User_Score'].median()
print("\nMedian user score after handling non-numerical values:",
      median_user_score)

# Replace NaN entries in the user score column with the median value
```

```
df['User_Score'].fillna(median_user_score, inplace=True)

# Verify the replacement
print("\nUser scores after replacing NaN values with the median value:")
print(df['User_Score'].head(10))
```

Median user score after handling non-numerical values: 7.5

User scores after replacing NaN values with the median value:

0	8.0
1	7.5
2	8.3
3	8.0
4	7.5
5	7.5
6	8.5
7	6.6
8	8.4
9	7.5

Name: User_Score, dtype: float64

[]: