Steam Games Data Analysis

2025

Exploratory Data Analysis (EDA) on Steam Games Dataset - Name: Rahul P G - Domain: Data Analytics / Data Science - Tools: Python, Pandas, NumPy, Matplotlib, Seaborn, SciPy

INTRODUCTION

The project is based on Exploratory Data Analysis on the Steam Games Dataset.
The main objective of this project is to understand game pricing, popularity, ratings, and player engagement using statistical analysis and data visualization.
I performed data cleaning, handled missing values and outliers, and then applied descriptive and inferential statistics to extract meaningful insights.
This project helped me understand real-world data behavior and improve my data analysis skills.

OBJECTIVES OF THE PROJECT

- To perform Exploratory Data Analysis on the Steam Games Dataset
- To clean and preprocess real-world gaming data
- To analyze pricing, popularity, ratings, and platform support
- To apply statistical methods for deeper insights
- To visualize data distributions and relationships

INTRODUCTION

The Steam Games Dataset contains structured information about video games available on the Steam digital distribution platform. The dataset includes a wide range of attributes related to games, such as pricing details, release information, age requirements, platform compatibility, genre classification, downloadable content (DLC) count, user recommendation metrics, and Metacritic critic ratings.

This dataset represents real-world gaming data collected from the Steam store and associated sources. It reflects the diversity of games available on the platform, ranging from free-to-play titles to premium-priced games, and from independent releases to large commercial productions. The presence of missing values, categorical variables, and skewed numerical distributions makes the dataset suitable for exploratory data analysis and statistical examination.

DATASET DESCRIPTION

Total records: 13,298 games
Unique titles: 13,284
Data types include numerical, categorical, and boolean features
Dataset covers free-to-play and paid games across multiple platforms

TOOLS AND TECHNOLOGIES USED

Python
Pandas
NumPy
Matplotlib
Seaborn
SciPy
Jupyter Notebook

DATA PREPROCESSING

Initial preprocessing involved loading the dataset, inspecting its structure, and identifying inconsistencies. Duplicate records were removed to avoid biased analysis. Column names were standardized for clarity, and missing values were inspected across all features.

```
import pandas as pd

df = pd.read_csv("steam_data.csv")

# Remove duplicate records
df = df.drop_duplicates()

# Check remaining duplicates
df.duplicated().sum()
```

HANDLING MISSING VALUES

Missing game names were filled with a placeholder value. Release dates were converted to datetime format, and release years were extracted. Missing release years were retained as unknown values for interpretability during EDA. Domain knowledge was applied to handle missing or invalid values in rating-related columns.

```
# Handle missing game names
df["GameName"] = df["GameName"].fillna("Unknown")

# Convert ReleaseDate to datetime
df["ReleaseDate"] = pd.to_datetime(df["ReleaseDate"], errors="coerce")

# Extract ReleaseYear
df["ReleaseYear"] = df["ReleaseDate"].dt.year

# Handle missing release years
df["ReleaseYear"] = df["ReleaseYear"].fillna("Unknown")
```

It converts values like:
"2020-05-10"
"10/05/2020"
"May 10, 2020"
into datetime objects.
So instead of text (string), the column becomes date-type data.

Original value	Result
"2018-11-20"	11/20/2018
"not a date"	NaT
""	NaT
null	NaT

FEATURE ENGINEERING

Feature engineering was applied to improve data readability. Age requirements were categorized into meaningful groups. Boolean variables such as free-to-play status and controller support were converted into descriptive categorical values. Genre indicators were transformed for better interpretability.

```
# Categorize age requirements
df["RequiredAge"] = df["RequiredAge"].apply(
   lambda x: "All Ages" if x == 0 else ("Teen (13+)" if x <= 17 else "Adult (18+)")
)

# Convert boolean fields to readable categories
df["IsFree"] = df["IsFree"].replace({True: "Free", False: "Paid"})
df["ControllerSupport"] = df["ControllerSupport"].replace(
   {True: "Provided", False: "Not Provided"}
)
```

```
df.head()
```

✓ 0.0s

	GameName	ReleaseDate	RequiredAge
0	Counter-Strike	2000-11-01	All Ages
1	Team Fortress Classic	1999-04-01	All Ages
2	Day of Defeat	2003-05-01	All Ages
3	Deathmatch Classic	2001-06-01	All Ages
4	Half-Life: Opposing Force	1999-11-01	All Ages

DATA CONSISTENCY CHECKS

Logical consistency checks were applied to ensure that free games had zero price values, correcting any inconsistencies between pricing information and free-to-play indicators. Additional validation checks were performed to identify and resolve contradictory records, such as paid games with zero prices. Duplicate entries were examined and handled where necessary to prevent skewed analysis. Missing or invalid pricing values were reviewed to maintain data integrity. These steps ensured strong alignment between pricing, availability status, and overall dataset reliability, leading to more accurate exploratory analysis and downstream modeling.

```
df.loc[df['IsFree'] == True, 'PriceFinal'] = 0
```

✓ 0.0s

```
df[(df['IsFree'] == True) & (df['PriceFinal'] > 0)]
```

✓ 0.0s

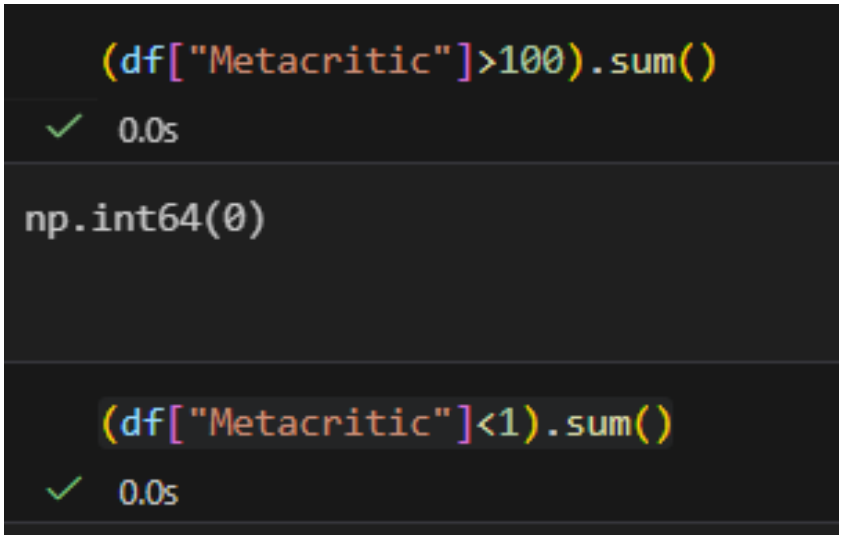
```
df[(df['IsFree'] == True) & (df['PriceFinal'] > 0)]
```

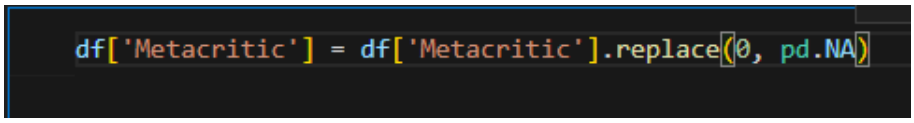
✓ 0.0s

	GameName	ReleaseDate	RequiredAge	PriceFinal	DLCCount	Metacritic	RecommendationCount	IsFree
12	Half-Life: Source	2004-06-01	0	9.99	0	0	2547	True
16	Half-Life Deathmatch: Source	2006-05-01	0	9.99	0	0	864	True
65	The Ship	2006-07-11	0	9.99	0	76	5026	True
69	Gumboy: Crazy Adventures	2006-12-19	0	4.99	0	69	0	True
258	Bone: Out from Boneville	2008-06-17	0	9.99	0	68	0	True
259	Bone: The Great Cow Race	2008-06-17	0	9.99	0	76	0	True
283	DOOM 3	2007-08-03	17	4.99	0	87	1645	True
285	Master Levels for DOOM II	2007-08-03	0	14.99	0	0	181	True
570	Penumbra: Black Plague	2009-01-23	17	9.99	0	78	930	True
573	Penumbra: Overture	2009-03-06	17	9.99	0	73	1096	True

HANDLING INVALID VALUES

Metacritic ratings have a valid range of 1 to 100. A value of zero does not represent a real rating and was treated as missing. Instead of using statistical methods, domain knowledge was applied to correctly classify such values as unknown.





outliers were handled without using the IQR method.
Instead, invalid values were identified using domain knowledge.
The Metacritic score of 0 does not represent an actual rating, so it was treated as an unknown value and replaced with NaN.

We use it to represent missing or unknown values in numeric data.

Option	Problem
`0`	Looks like a real score (wrong)
`"Unknown"` (string)	Breaks numeric calculations & plotting
`np.nan`	Treated as missing and still keeps the column numeric ✓

OUTLIER ANALYSIS

Outliers can arise from data entry errors, measurement issues, or natural variations. In this project, different strategies were used depending on the feature and its meaning. Not all extreme values were removed, as many represented valid real-world cases.

```
df.describe()
```

✓ 0.0s

	PriceFinal	DLCCount	RecommendationCount
count	13298.000000	13298.000000	1.329800e+04
mean	8.733259	0.721838	1.166077e+03
std	13.603069	8.755862	1.519003e+04
min	0.000000	0.000000	0.000000e+00
25%	1.990000	0.000000	0.000000e+00
50%	4.990000	0.000000	0.000000e+00
75%	9.990000	0.000000	2.407500e+02
max	449.990000	630.000000	1.427633e+06

OUTLIER HANDLING – RECOMMENDATION COUNT

```
Q1 = df['RecommendationCount'].quantile(0.25)
Q3 = df['RecommendationCount'].quantile(0.75)
IQR = Q3 - Q1

lower = Q1 - 1.5 * IQR
upper = Q3 + 1.5 * IQR

df['RecommendationCount'] = df['RecommendationCount'].clip(lower, upper)
```

✓ 0.0s Python

```
(df['RecommendationCount'] > 100000).sum()
```

✓ 0.0s Python

```
np.int64(0)
```

Values greater than upper → set to upper
Values less than lower → set to lower

DESCRIPTIVE STATISTICAL ANALYSIS

Descriptive statistics such as mean, median, standard deviation, and range were used to summarize numerical features. The analysis showed that most games are priced affordably, while a few highly popular games receive a large number of recommendations.

summarizing data
describing relationships
understanding patterns
It does not make decisions or predictions.

CORRELATION ANALYSIS

Pearson correlation analysis was conducted to measure linear relationships between numerical variables.

```
#Pearson Correlation (Price vs Metacritic)
df[['PriceFinal', 'Metacritic']].corr(method='pearson')
```

	PriceFinal	Metacritic
PriceFinal	1.000000	0.177564
Metacritic	0.177564	1.000000

The correlation between price and Metacritic rating is 0.18, indicating a very weak relationship. Therefore, a game's rating does not strongly affect its price on Steam.

Category	Number of Games	Probability Formula	Probability
Free Games	1,044	1044 / 13356	7.82%
Paid Games	12312	12312 / 13356	92.18%
Total Games	13,356	—	100%

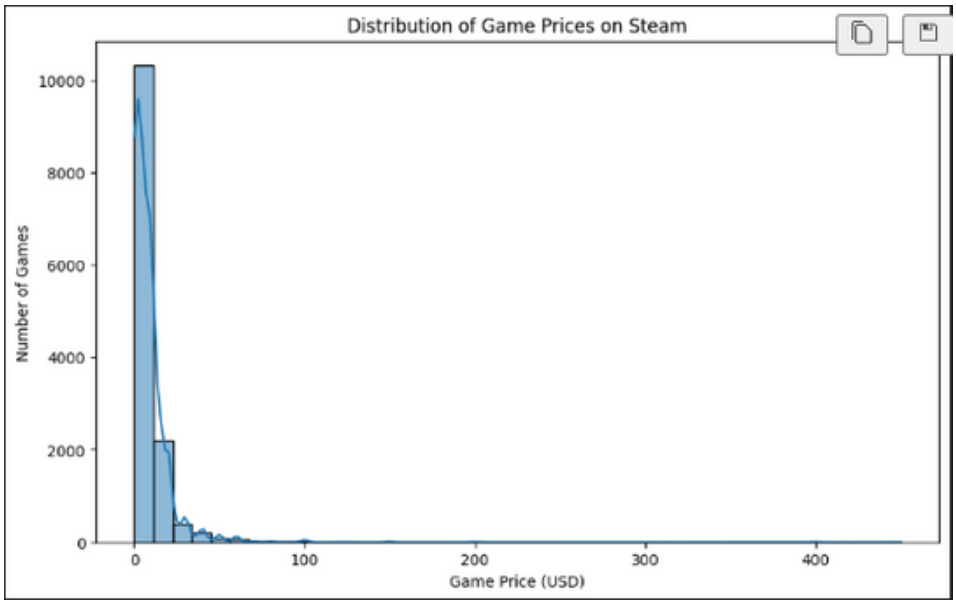
UNIVARIATE ANALYSIS

(Analyzing ONE variable at a time)

1) Histogram: Price Distribution

```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_csv("steam_data_cleaned.csv")

plt.figure(figsize=(10,6))
sns.histplot(df['PriceFinal'], bins=40, kde=True)
plt.xlabel('Game Price (USD)')
plt.ylabel('Number of Games')
plt.title('Distribution of Game Prices on Steam')
plt.show()
```

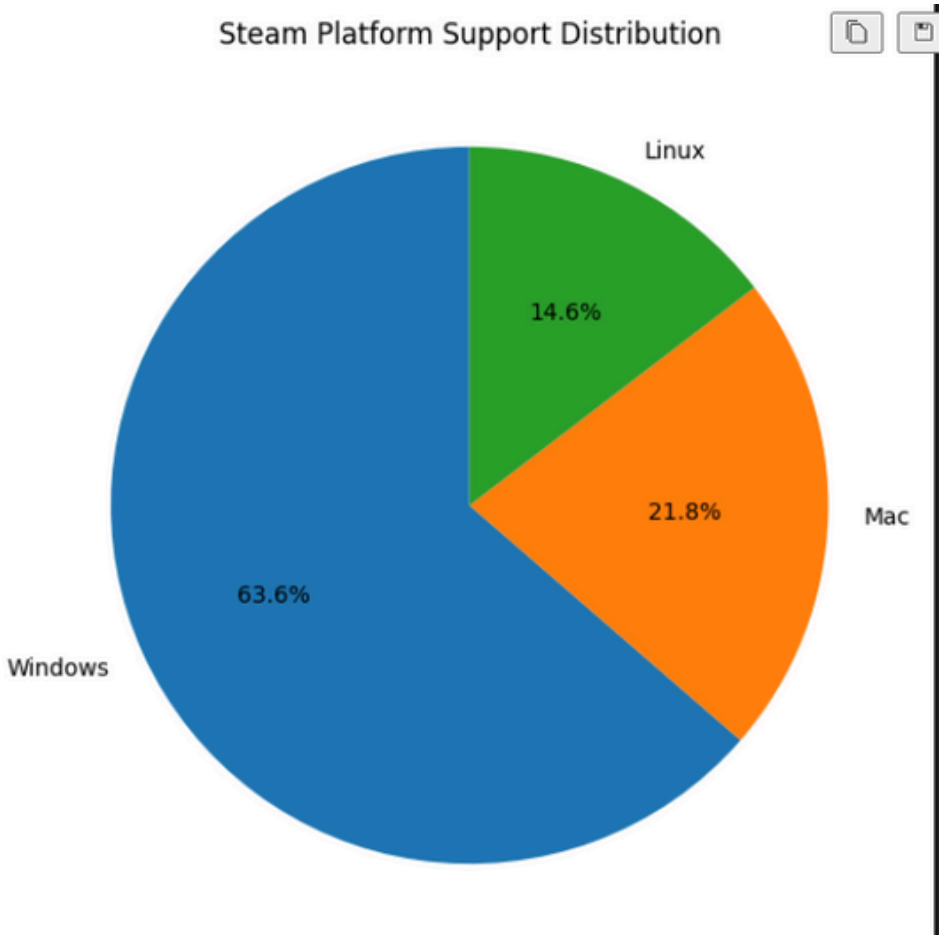
Python



The histogram shows that most Steam games are either free or priced at lower values. As the price increases, the number of games decreases sharply, indicating that affordable games dominate the platform.

```python
#univariate
platform_counts = df[['PlatformWindows', 'PlatformMac', 'PlatformLinux']].sum

plt.figure(figsize=(7,7))
plt.pie(platform_counts, labels=['Windows', 'Mac', 'Linux'], autopct='%1.1f%%
plt.title('Steam Platform Support Distribution')
plt.show()
```

Python



Most Steam games support Windows, while macOS and Linux have significantly lower support. This indicates that game developers primarily target the Windows platform.

BIVARIATE ANALYSIS

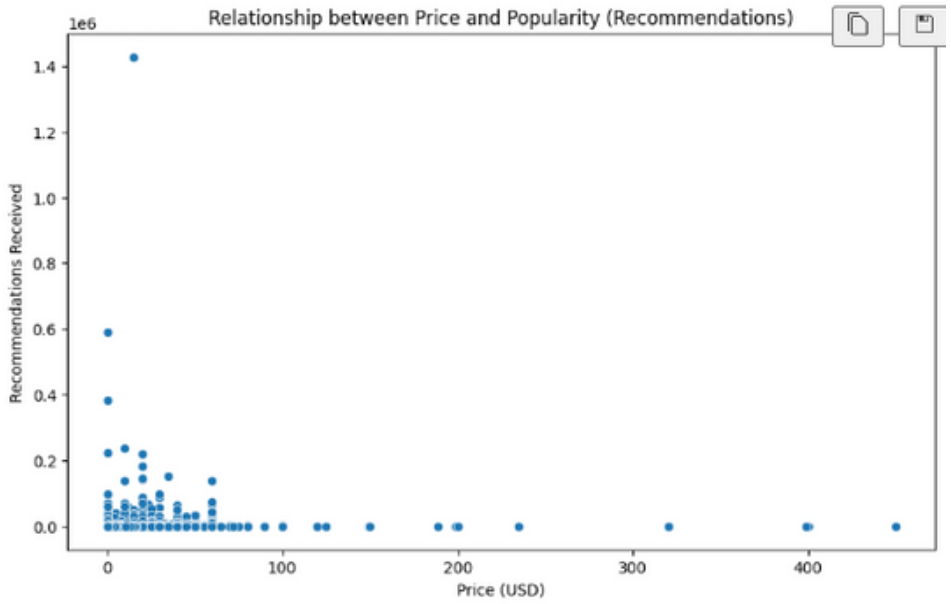
(Analyzing TWO variables together)

Scatter Plot: Price vs Recommendations

Bivariate analysis examines the relationship between two variables, helping identify patterns or associations between them.

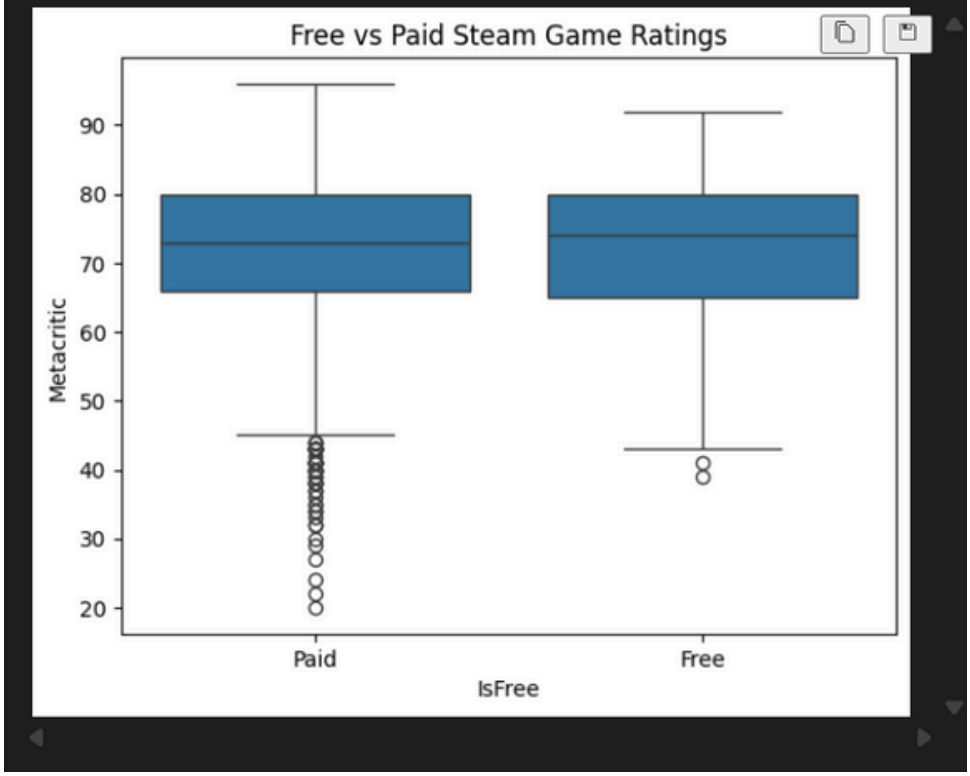
```python
plt.figure(figsize=(10,6))
sns.scatterplot(x='PriceFinal', y='RecommendationCount', data=df)
plt.xlabel('Price (USD)')
plt.ylabel('Recommendations Received')
plt.title('Relationship between Price and Popularity (Recommendations)')
plt.show()
```

Python



There is no strong relationship between price and popularity. Some cheap or free games receive huge recommendations.

```
sns.boxplot(x='IsFree', y='Metacritic', data=df[df['Metacritic']>0])
plt.title('Free vs Paid Steam Game Ratings')
plt.show()
```



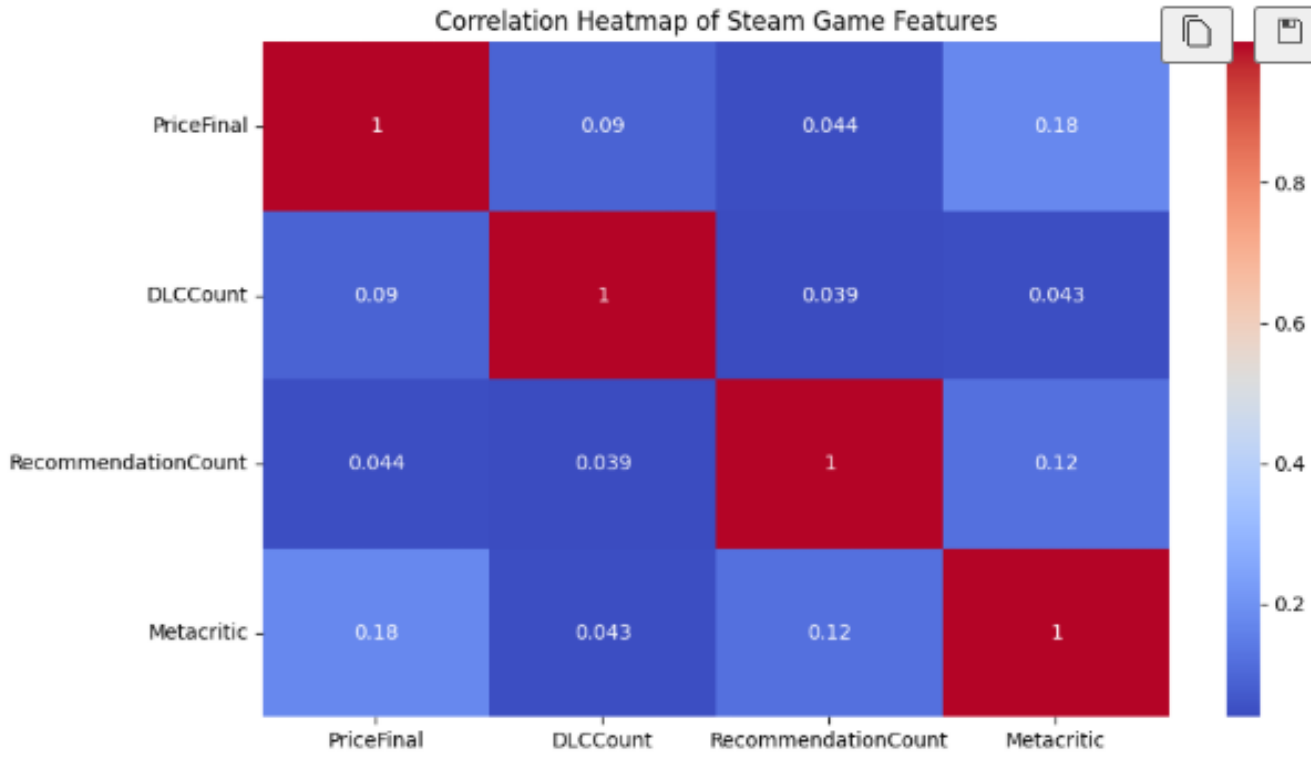
Median ratings for Free and Paid games are similar. Paying does not guarantee better ratings.

MULTIVARIATE ANALYSIS

(Analyzing MORE than two variables)

Heatmap: Correlation Between Numerical Features

```
plt.figure(figsize=(10,6))
sns.heatmap(df[['PriceFinal','DLCCount','RecommendationCount','Metacritic']].
plt.title('Correlation Heatmap of Steam Game Features')
plt.show()
```



Heatmaps help visualize relationships. We found weak correlations between price and other features.

Inferential Statistics

IInferential statistics helps us make conclusions or decisions beyond the data we directly observe.

Hypothesis	Meaning
H_0 (Null)	Paid and Free games have **similar Metacritic ratings**
H_1 (Alternative)	Paid and Free games have **different ratings**

Null Hypothesis (H_o)
"There is no significant difference in Metacritic scores between free and paid games."
Alternative Hypothesis (H_1)
"There is a significant difference in Metacritic scores between free and paid games."

Independent t-test to compare the average Metacritic scores of free and paid games.

My dataset contained missing values, so I used nan_policy='omit' to ensure statistical tests were computed correctly without errors.

```python
from scipy.stats import ttest_ind

free_rating = df[df['IsFree']=='Free']['Metacritic']
paid_rating = df[df['IsFree']=='Paid']['Metacritic']

ttest_ind(free_rating, paid_rating, equal_var=False, nan_policy='omit')
```

Python

```
TtestResult(statistic=np.float64(-9.529838382837818), pvalue=np.float64(6.702993
```

Value	Meaning
t-statistic = -9.53	There is a large difference between the two groups
p-value = 6.70×10^{-21} (very small!)	Difference is extremely significant
df = 1395.19	Degrees of freedom

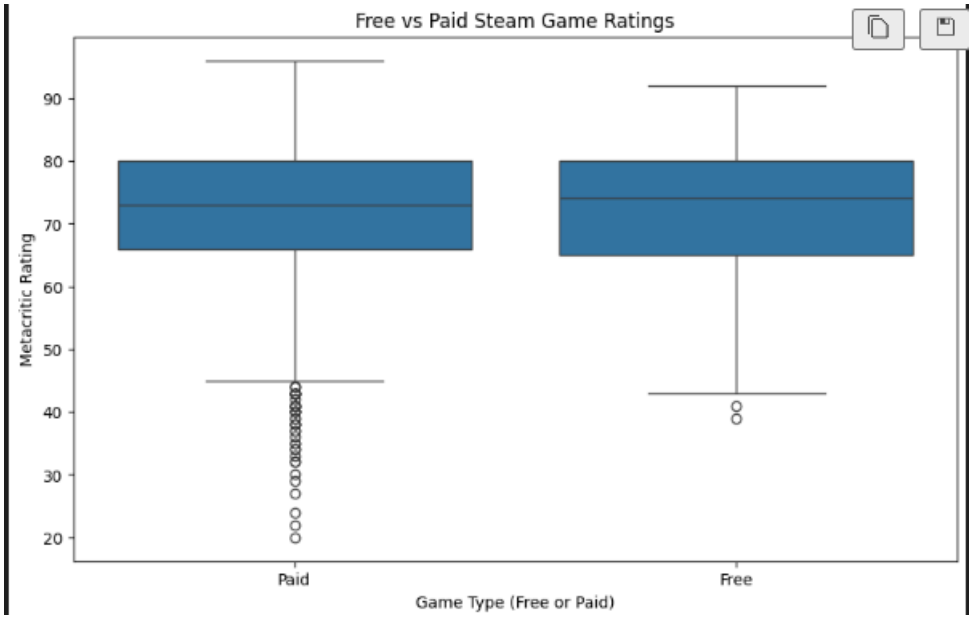
Because p-value $\ll$ 0.05, we Reject the Null Hypothesis.

Paid games and Free games have DIFFERENT METACRITIC ratings.
The negative t-value means Paid games most likely have LOWER average ratings than Free games, or Free games have higher average ratings.

he test result showed whether the difference in Metacritic scores between free and paid games was real or just due to chance

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

df = pd.read_csv("steam_data_cleaned.csv")

# Replace 0 (missing) ratings with N
df['Metacritic'].replace(0, np.nan, inplace=True)

# Keep only rows with valid ratings
df_filtered = df.dropna(subset=['Metacritic'])

plt.figure(figsize=(10,6))
sns.boxplot(x='IsFree', y='Metacritic', data=df_filtered)
plt.xlabel('Game Type (Free or Paid)')
plt.ylabel('Metacritic Rating')
plt.title('Free vs Paid Steam Game Ratings')
plt.show()
```

(parameter) inplace: Literal[True]



The box plot shows that the median rating of Free and Paid games is similar. Even though paid games have more poorly rated outliers, the average rating level is the same.
So, paying for a game does not guarantee better critic reviews.

CONCLUSION AND FUTURE SCOPE

This project successfully demonstrates the application of Exploratory Data Analysis on a real-world gaming dataset. Proper data cleaning, feature engineering, and statistical techniques were used to extract meaningful insights. Future work could involve predictive modeling, sentiment analysis of reviews, or time-based trend analysis.