Video Game Recommender System

By: Cindy Seth

Problem Statement

Many gamers struggle to discover new video games that align with their preferences, leading to wasted time and money on games they don't enjoy. This project aims to develop a video game recommender system that provides tailored recommendations to users, enhancing their gaming experience. The video game industry continues to expand with a vast number of games available. Many gamers often feel overwhelmed by the amount of game options which can lead to decision fatigue. A successful recommendation system can utilize historical sales data, user reviews, and gameplay metrics to create a personalized game recommendation.

Data Overview

For this project, I utilized the **‘vgchartz-2024.csv’** dataset from Kaggle. The dataset comprises fourteen columns, several of which contained missing data.

- img
- title
- console
- genre
- publisher
- developer
- critic_score
- total_sales
- na_sales
- jp_sales
- pal_sales
- other_sales
- release_date
- last_update

Data Wrangling

To clean up the data:

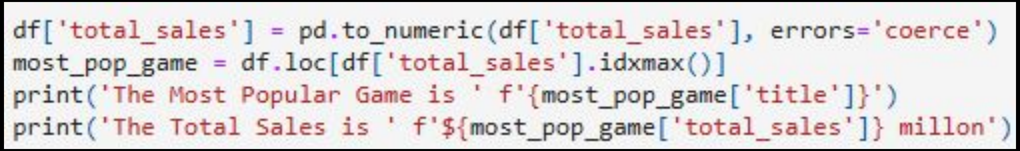
I replaced 'critic_score' with 'rating', I dropped missing values, and dropped any duplicates.

Evaluations Encountered:

- I sorted the video game titles based on their ratings which resulted in "Red Dead Redemption 2" as the top rated game with a 9.7 rating
- I evaluated the rating counts which resulted with "Call of Duty: Advanced Warfare" as the video game with the most rating counts with 4 rating counts
- The top genre was "Sandbox"
- The most genre rating counts was "Shooter" with 46

Exploratory Data Analysis

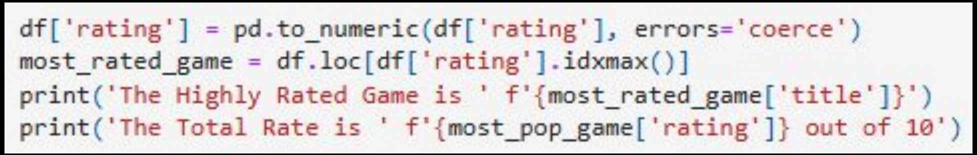
In this phase, we delved into the data to uncover hidden connections. Using the following code, we concluded that the most popular game is “Grand Theft Auto V” with the total sales of 19.39 million dollars.



```
df['total_sales'] = pd.to_numeric(df['total_sales'], errors='coerce')
most_pop_game = df.loc[df['total_sales'].idxmax()]
print('The Most Popular Game is ' f'{most_pop_game['title']}')
print('The Total Sales is ' f'${most_pop_game['total_sales']} millon')
```

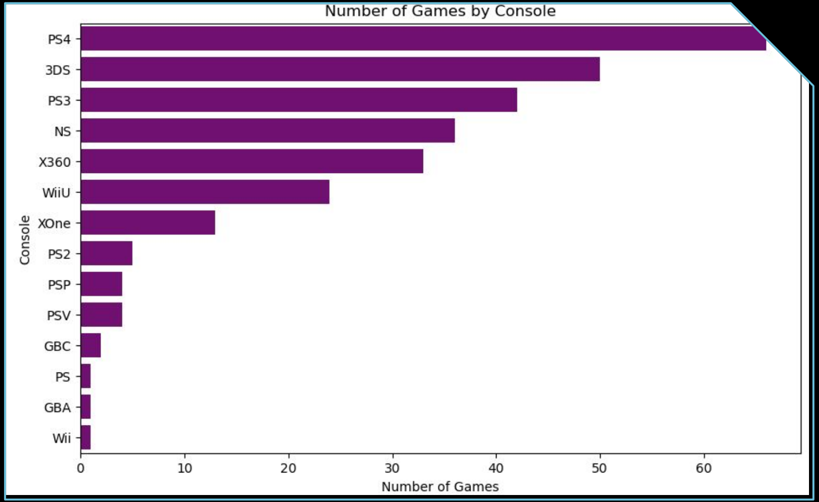
Exploratory Data Analysis

Using the following code, we concluded that the highly rated game is “Red Dead Redemption 2” with the total rating of 9.7 out of 10.



```
df['rating'] = pd.to_numeric(df['rating'], errors='coerce')
most_rated_game = df.loc[df['rating'].idxmax()]
print('The Highly Rated Game is ' f'{most_rated_game['title']}')
print('The Total Rate is ' f'{most_pop_game['rating']} out of 10')
```

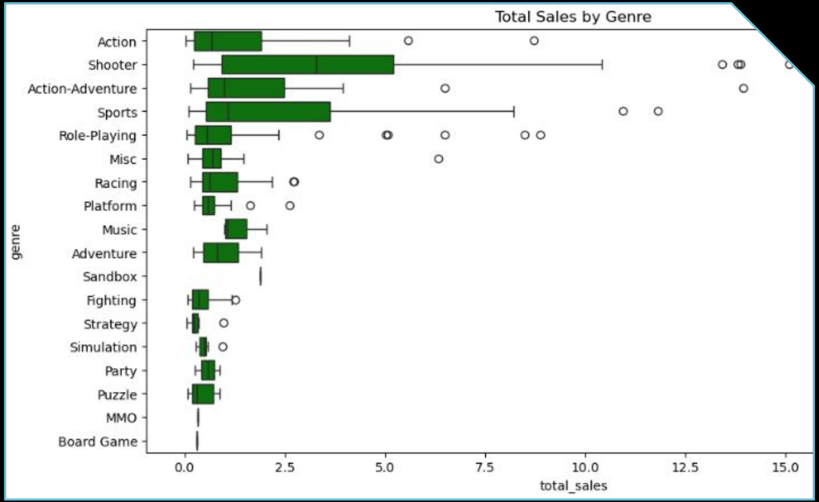
Number of Games by Console



Out of fourteen counts of consoles, the PS4 has most amount of games, followed by 3DS, with Wii as the console with the least amount of games.

Total Sales by Genre

The genre with the most total sales is ‘Shooter’ type games.

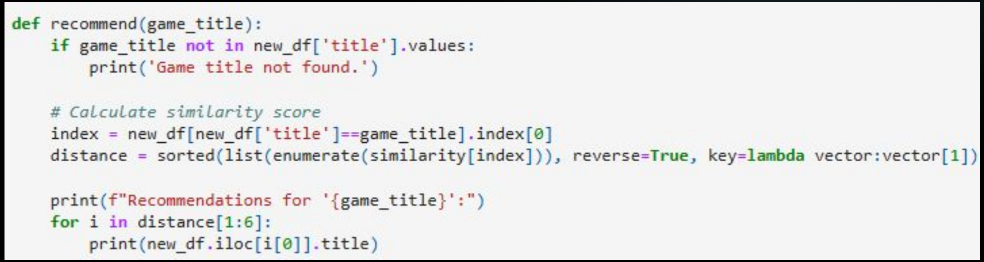


++++

Preprocessing Steps

To create the recommender system, I used the following columns:

- title
- console
- genre
- publisher



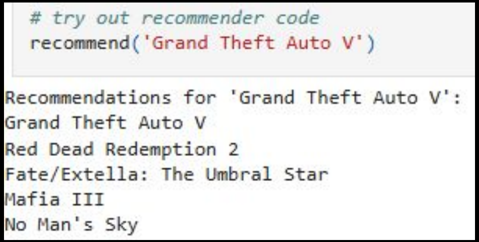
```python
def recommend(game_title):
    if game_title not in new_df['title'].values:
        print('Game title not found.')

    # Calculate similarity score
    index = new_df[new_df['title']==game_title].index[0]
    distance = sorted(list(enumerate(similarity[index])), reverse=True, key=lambda vector:vector[1])

    print(f"Recommendations for '{game_title}':")
    for i in distance[1:6]:
        print(new_df.iloc[i[0]].title)
```

I then created a 'tags' column to combine 'genre' and 'publisher'. Once completed, I applied CountVectorizer to create a matrix for the recommender. I transformed the tags column into a matriX where each row is a document and each column represents the frequency of a word. I applied cosine similarity to measure how similar the documents in the tags column are to each other based on their vectorized representation.

I applied the following code to recommend similar games based on the cosine similarity of their vectorized content to display the top 5 most similar games based on the user input.

Modeling

To determine if the recommender works, I tested it out:



```
# try out recommender code
recommend('Grand Theft Auto V')
```

```
Recommendations for 'Grand Theft Auto V':
Grand Theft Auto V
Red Dead Redemption 2
Fate/Extella: The Umbral Star
Mafia III
No Man's Sky
```

The recommender was able to recommend the top five games most similar to the user input.

++++

Next Step

My next step is to create a streamlit app for the recommender system.

One con of the recommender system is that the user MUST input the exact title, so finding a way to fix that for the app will also be useful.

++++