Board Games Data Set

Christina Ha

Introduction

Description

About this dataset

This dataset contains data collected on board games from the BoardGameGeek (BGG) website in February 2021. BGG is the largest online collection of board game data which consists of data on more than 100,000 total games (ranked and unranked).

The voluntary online community contributes to the site with reviews, ratings, images, videos, session reports and live discussion forums on the expanding database of board games.

This data set contains all ranked games (~20,000) as of the date of collection from the BGG database. Unranked games are ignored as they have not been rated by enough BGG users (a game should receive at least 30 votes to be eligible for ranking).

- Source: https://www.kaggle.com/andrewmvd/board-games?select=bgg_dataset.csv

Background

```
[2] df = pd.read_csv("/content/drive/MyDrive/Data Science Bootcamp/Project 2/bgg_dataset.csv", sep=';')
    df.head()
```

	ID	Name	Year Published	Min Players	Max Players	Play Time	Min Age	Users Rated	Rating Average	BGG Rank	Complexity Average	Owned Users	Mechanics	Domains
0	174430.0	Gloomhaven	2017.0	1	4	120	14	42055	8,79	1	3,86	68323.0	Action Queue, Action Retrieval, Campaign / Bat...	Strategy Games, Thematic Games
1	161936.0	Pandemic Legacy: Season 1	2015.0	2	4	60	13	41643	8,61	2	2,84	65294.0	Action Points, Cooperative Game, Hand Manageme...	Strategy Games, Thematic Games
2	224517.0	Brass: Birmingham	2018.0	2	4	120	14	19217	8,66	3	3,91	28785.0	Hand Management, Income, Loans, Market, Networ...	Strategy Games
3	167791.0	Terraforming Mars	2016.0	1	5	120	12	64864	8,43	4	3,24	87099.0	Card Drafting, Drafting, End Game Bonuses, Han...	Strategy Games
4	233078.0	Twilight Imperium: Fourth Edition	2017.0	3	6	480	14	13468	8,70	5	4,22	16831.0	Action Drafting, Area Majority / Influence, Ar...	Strategy Games, Thematic Games

- 14 features
- 20,343 rows
- Target feature will be the Rating Average of a board game (Regression problem)

Cleaning the Data

- Delete unnecessary columns
  - Decided to keep all columns
- Delete duplicated rows
  - N/A, there were no duplicated rows
- Rename columns to make them more “Python”
  - Example: df.rename(column={“Year Published”: “Year_Published”}
- Correct inconsistencies
  - Replacing commas with periods in rating values
  - Example: 4,67 rating became 4.67
- Correct datatype
  - Turning rating and year values into int/floats

Null Values

```
df.isnull().sum()

#There are missing values in ID, Year_Published, Owned_Users, Mechanics, Domains
```

```
ID                     16
Name                    0
Year_Published          1
Min_Players             0
Max_Players             0
Play_Time               0
Min_Age                 0
Users_Rated             0
Rating_Average          0
BGG_Rank                0
Complexity_Average      0
Owned_Users            23
Mechanics            1598
Domains             10159
dtype: int64
```

- There were missing values in the columns:
  - ID, Year_Published, Owned_Users, Mechanics, and Domains

Null ID Values

```
df.isnull().sum()

#There are missing values in ID, Year_Published, Owned_Users, Mechanics, Domains
```

```
ID                    16
Name                   0
Year_Published         1
Min_Players            0
Max_Players            0
Play_Time              0
Min_Age                0
Users_Rated            0
Rating_Average         0
BGG_Rank               0
Complexity_Average     0
Owned_Users           23
Mechanics           1598
Domains            10159
dtype: int64
```

Missing ID values:

- Replaced missing values with 0 to keep the corresponding data
- Reasoning: The ordinal value of the ID is not important and want to keep the other data points in the row

Null Year Published Values

```
df.isnull().sum()

#There are missing values in ID, Year_Published, Owned_Users, Mechanics, Domains
```

```
ID                    16
Name                   0
Year_Published         1
Min_Players            0
Max_Players            0
Play_Time              0
Min_Age                0
Users_Rated            0
Rating_Average         0
BGG_Rank               0
Complexity_Average     0
Owned_Users           23
Mechanics           1598
Domains            10159
dtype: int64
```

Missing Year_Published values:

- Dropped the row with the missing value
- Reasoning: There was only 1 missing value, which is an extremely small subset of the data

Null Owned Users Values

```
df.isnull().sum()

#There are missing values in ID, Year_Published, Owned_Users, Mechanics, Domains
```

```
ID                    16
Name                   0
Year_Published         1
Min_Players            0
Max_Players            0
Play_Time              0
Min_Age                0
Users_Rated            0
Rating_Average         0
BGG_Rank               0
Complexity_Average     0
Owned_Users           23
Mechanics           1598
Domains            10159
dtype: int64
```

Missing Owned_Users values:

- Found the average value of Owned_Users / Users_Rated (1.674…)
- Multiplied this average value by the number of Users_Rated in the row with missing Owned_Users values to fill in the missing value
- Reasoning: Looking at the dataset, there appeared to be a trend between the number of users who owned a game and the number of users who rated the game

Null Mechanics and Domains Values

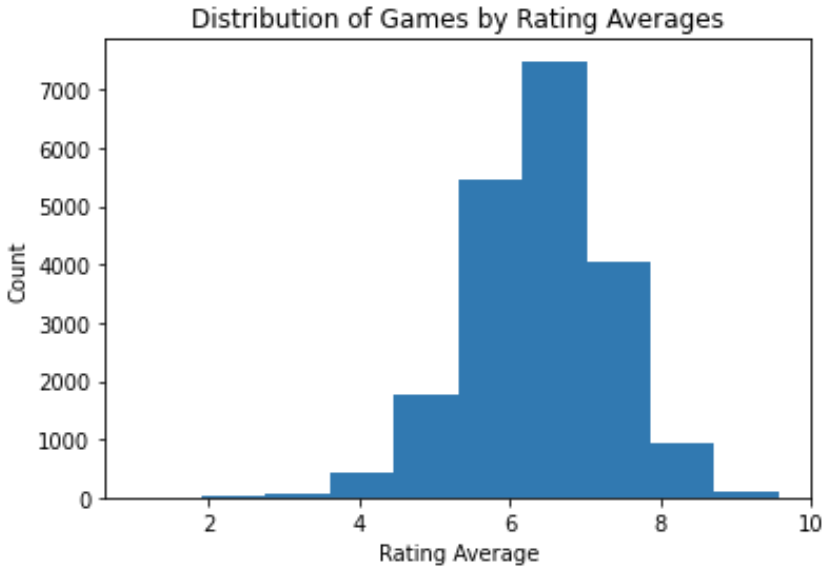
```
df.isnull().sum()

#There are missing values in ID, Year_Published, Owned_Users, Mechanics, Domains
```

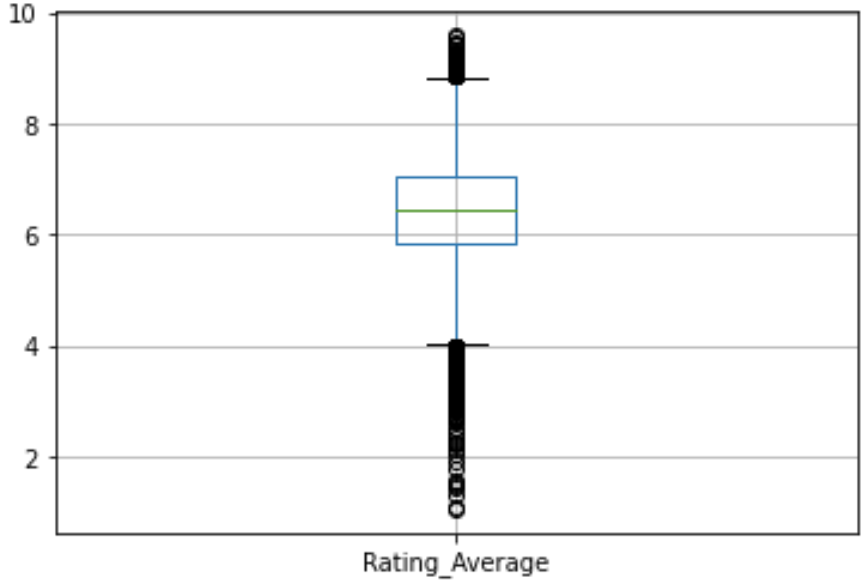
```
ID                   16
Name                  0
Year_Published        1
Min_Players           0
Max_Players           0
Play_Time             0
Min_Age               0
Users_Rated           0
Rating_Average        0
BGG_Rank              0
Complexity_Average    0
Owned_Users          23
Mechanics          1598
Domains           10159
dtype: int64
```

Missing Mechanics/Domains values:

- Filled in 'Undefined' for the missing values
- Reasoning: There were too many missing values to just drop from the dataset

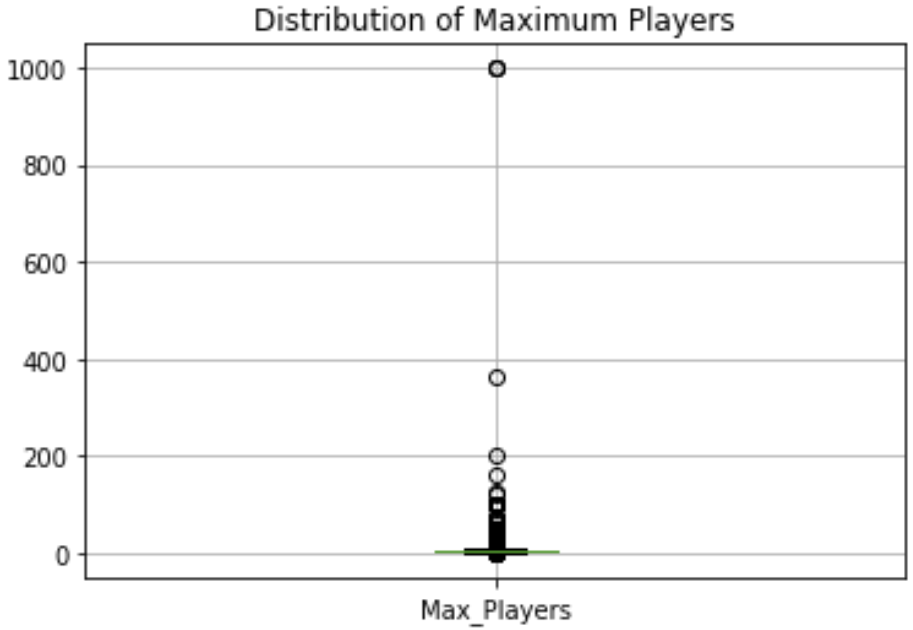
Exploratory Visuals

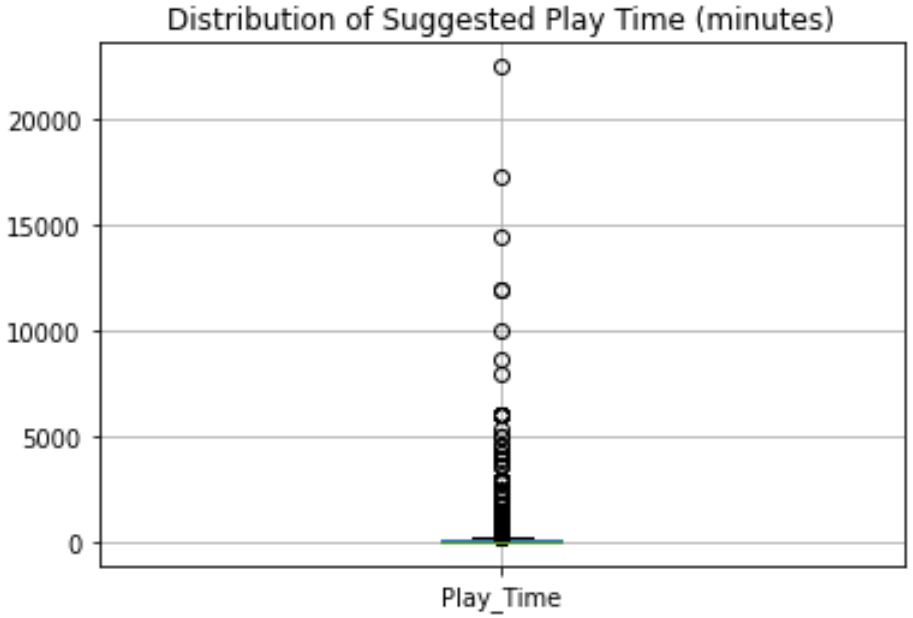




- Histogram and Boxplot for Target Feature (Rating_Average)
- Mean: 6.4
- Min: 1.05
- Max: 9.58
- Outliers: acceptable, within range

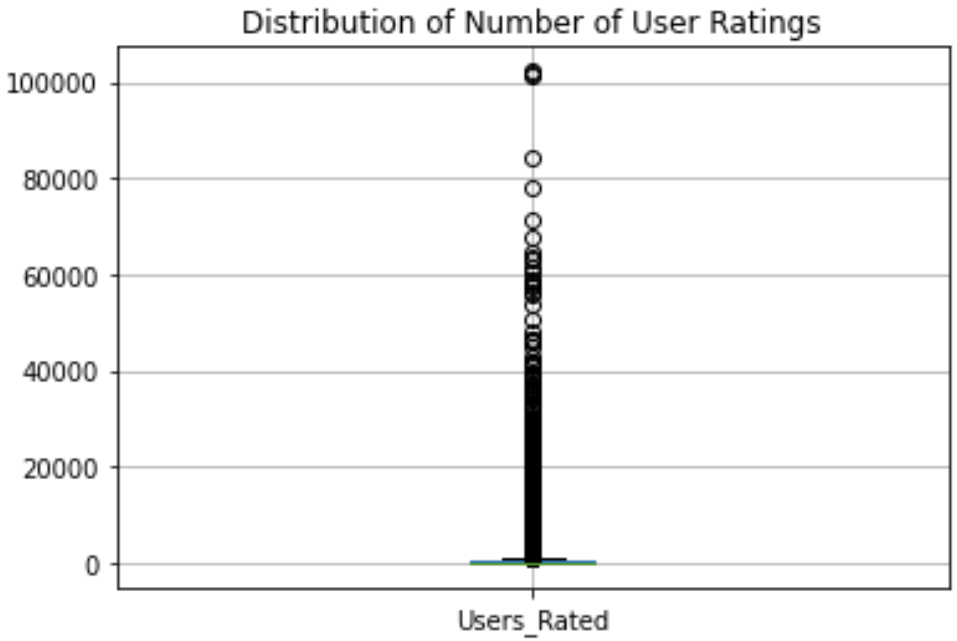
Exploratory Visuals





- Identified and dropped outliers where appropriate

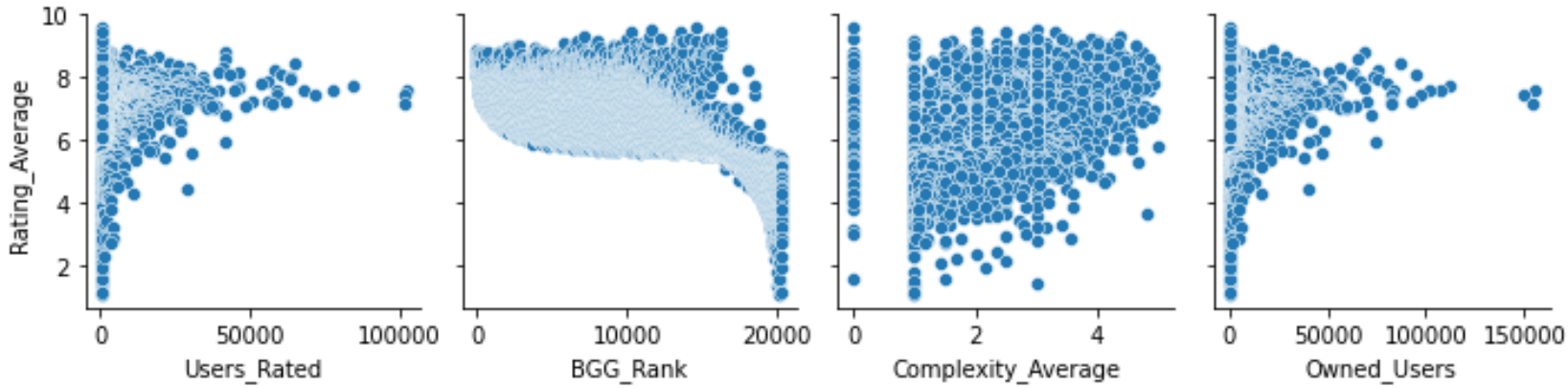
Exploratory Visuals



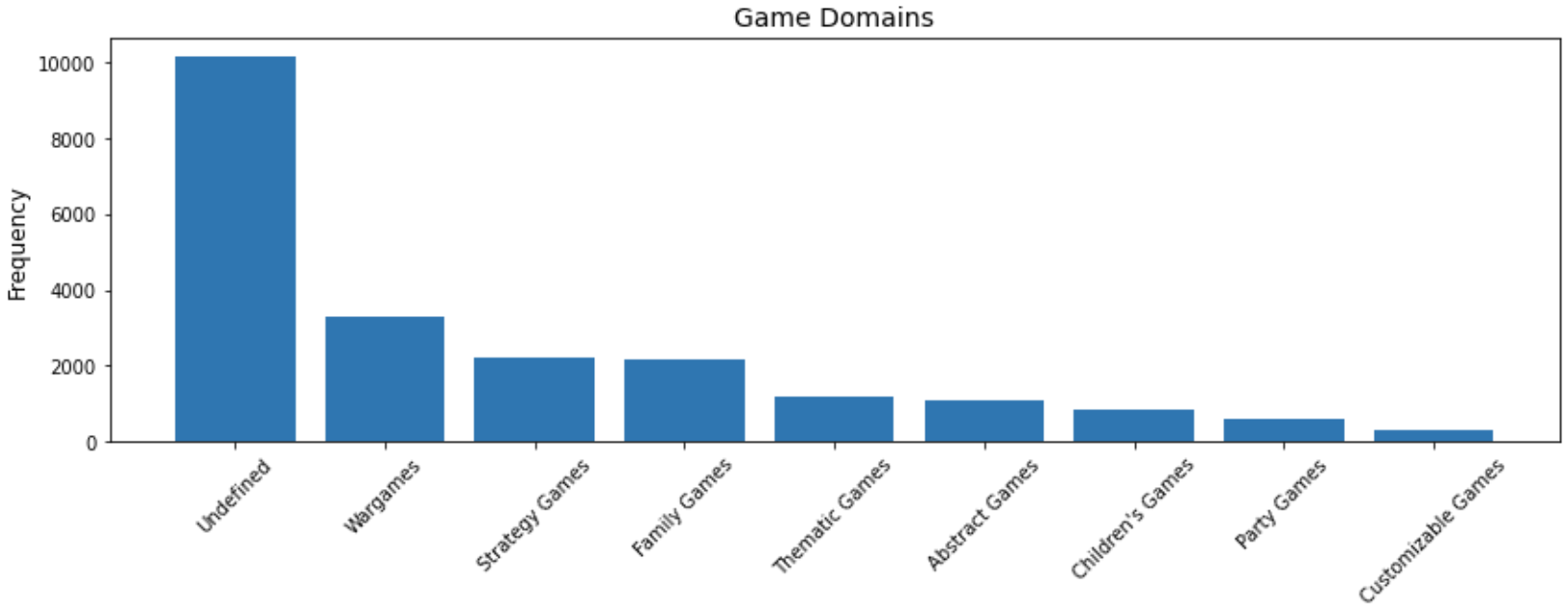
- Identified outliers and kept where appropriate

Correlations

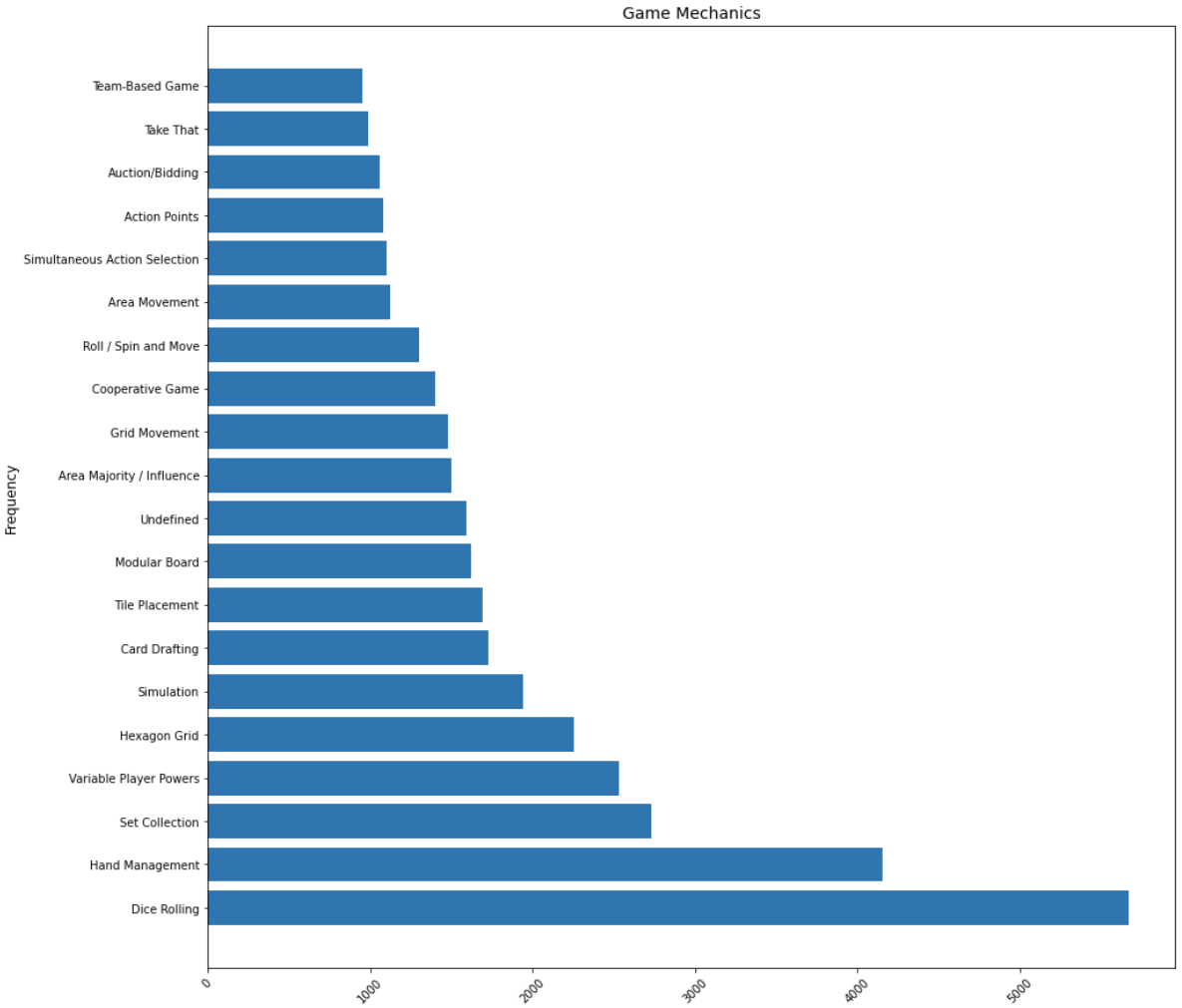
```
sns.pairplot(df,
             x_vars = ['Users_Rated', 'BGG_Rank', 'Complexity_Average', 'Owned_Users'],
             y_vars = ['Rating_Average']);
```



Exploratory Visuals



Exploratory Visuals



Unique Challenges

- Domain and Mechanic features

```
df['Domains'].unique()
```

```
array(['Strategy Games, Thematic Games', 'Strategy Games',
       'Thematic Games', 'Strategy Games, Wargames',
       'Thematic Games, Wargames', 'Family Games, Strategy Games',
       'Customizable Games, Thematic Games',
       'Abstract Games, Family Games', 'Customizable Games',
       'Family Games', 'Party Games', 'Customizable Games, Wargames',
       'Wargames', 'Party Games, Thematic Games', 'Abstract Games',
       'Customizable Games, Strategy Games',
       'Family Games, Thematic Games', 'Family Games, Party Games',
       'Abstract Games, Strategy Games', "Children's Games, Family Games",
       'Undefined', 'Party Games, Strategy Games', "Children's Games",
       "Children's Games, Party Games",
       'Abstract Games, Customizable Games',
       'Family Games, Strategy Games, Thematic Games',
       'Family Games, Party Games, Thematic Games',
       'Strategy Games, Thematic Games, Wargames',
       'Abstract Games, Party Games', "Abstract Games, Children's Games",
       'Family Games, Wargames', 'Family Games, Thematic Games, Wargames',
       'Abstract Games, Wargames',
       "Children's Games, Family Games, Party Games",
       'Party Games, Wargames', "Children's Games, Wargames",
       'Customizable Games, Thematic Games, Wargames',
       "Abstract Games, Children's Games, Wargames",
       'Abstract Games, Strategy Games, Thematic Games',
       'Abstract Games, Thematic Games'], dtype=object)
```

Unique Challenges

```
#Turn the strings in the Domains column into lists by splitting the strings by commas and strip white space
df['Domains'] = [x.strip(' ').split(',') for x in df['Domains']]
```

Mechanics	Domains
Action Queue, Action Retrieval, Campaign / Bat...	[Strategy Games, Thematic Games]
Action Points, Cooperative Game, Hand Manageme...	[Strategy Games, Thematic Games]
Hand Management, Income, Loans, Market, Networ...	[Strategy Games]

Unique Challenges

```
#create function to find value counts of list items
#source: https://towardsdatascience.com/dealing-with-list-values-in-pandas-dataframes-a177e534f173

def to_1D(series):
 return pd.Series([x.strip() for _list in series for x in _list])
```

```
#find value counts of types of Domains
to_1D(df['Domains']).value_counts()
```

```
Undefined             10158
Wargames               3315
Strategy Games         2205
Family Games           2173
Thematic Games         1174
Abstract Games         1070
Children's Games        849
Party Games             605
Customizable Games      297
dtype: int64
```