DATA VISUALIZATION TECHNIQUES

Programme Title: HDIP Data Analytics Cohort: HDIP Feb 22 FT/PT
Module Title(s): Data Visualization Techniques
Assignment Type: Individual
Weighting(s): 50%
Assignment Title: CA1_DVis_HDip_Lvl8
Lecturer(s): David McQuaid
Issue Date: 19/9/2023
Submission Deadline Date: 20/10/2023 11:55pm

Student: ***Bárbara Abreu Costa 2023099***

GitHub Repository: https://github.com/Babreucosta/CA1_DataVisTech.git

---

In [1]:
```
from IPython.display import Image
image_path = 'image1.jpg'
Image(filename=image_path, width=950)
```

Out[1]:

SCENARIO

You have been retained by a retail company to analyse a dataset based on board games. This analysis will help determine the sales strategy for the company in their upcoming Winter season.

Each answer MUST have a separate visualization that can be easily understood, visually represents the answer, and all data wrangling, analysis, and visualizations must generated using python.

The companies CTO also requires you to rationalize all the decisions that you have made in your report.

This rationalization MUST include your visualization design decisions, how you have engineered the data, feature selection and any other information that you deem relevant.

DATA DICTIONARY

In [2]:
```
from IPython.display import Image
image_path = 'dictionary.jpg'
Image(filename=image_path)
```

Out[2]:

variable	class	description
game_id	text	Unique game identifier
description	text	A paragraph of text describing the game
image	text	URL image of the game
max_players	integer	Maximum recommended players
max_playtime	integer	Maximum recommended playtime (min)
min_age	integer	Minimum recommended age
min_players	integer	Minimum recommended players
min_playtime	integer	Minimum recommended playtime (min)
name	text	Name of the game
playing_time	integer	Average playtime
thumbnail	text	URL thumbnail of the game
year_published	integer	Year game was published
artist	text	Artist for game art
category	text	Categories for the game (separated by commas)
compilation	text	If part of a multi-compilation - name of compilation
designer	text	Game designer
expansion	text	If there is an expansion pack - name of expansion
family	text	Family of game - equivalent to a publisher
mechanic	text	Game mechanic - how game is played, separated by comma
publisher	text	Comoany/person who published the game, separated by comma
average_rating	double	Average rating on Board Games Geek (1-10)
users_rated	double	Number of users that rated the game

BOARD GAMES

A Review of the Board Games Business

> Acoording to market research companies Technavio and Imarc, the global board game market has an estimated value between USS11 billion and USS13.4 billion and is projected to grow by about 7 to 11 percent within the next 5 years.
>
> Over 3.000 new games are released each year (excluding expansion packs), according to the website and online forum 'BoardGameGeek', wich aims to log every game published.
>
> For business in this industry, the winter season which is marked by cozy get-togethers and festive celebrations, gives a wonderful chance to take advantage of the increased demand for

> entertainment. Understanding the diverse categories and mechanics of board games is crucial for devising a successful marketing strategy. A study can be useful to explore the importance of this knowledge, shedding light on how it can be turned to carve out a profitable niche in the bustling winter board games market.
>
> To conduct this analysis I decided to structure an approach starting with an general undertanding of the data. That initial stage will involve loading the dataset, its structure to well undertanding and check if there is any missing values. Where appropriate, I'll also trying to engineer features that can improve the dataset.

Load the dataset

In [3]:
```
import pandas as pd
import numpy as np
```

In [4]:
```
board_games = pd.read_csv('board_games.csv')
board_games.head()
```

Out[4]:

	game_id	description	image	max_players	max_playtime	min_age	min_players	n
0	1	Die Macher is a game about seven sequential po...	//cf.geekdo-images.com/images/pic159509.jpg	5	240	14	3	
1	2	Dragonmaster is a trick-taking card game based...	//cf.geekdo-images.com/images/pic184174.jpg	4	30	12	3	
2	3	Part of the Knizia tile-laying trilogy, Samura...	//cf.geekdo-images.com/images/pic3211873.jpg	4	60	10	2	
3	4	When you see the triangular box and the luxuri...	//cf.geekdo-images.com/images/pic285299.jpg	4	60	12	2	
4	5	In Acquire, each player strategically invests ...	//cf.geekdo-images.com/images/pic342163.jpg	6	90	12	3	

5 rows × 22 columns

Understand its structure

In [5]:
```
board_games.shape
```

Out[5]:
```
(10532, 22)
```

In [6]:
```
board_games.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10532 entries, 0 to 10531
Data columns (total 22 columns):
 #   Column          Non-Null Count  Dtype
```

```
---  ------           --------------  -----
 0   game_id          10532 non-null  int64
 1   description      10532 non-null  object
 2   image            10531 non-null  object
 3   max_players      10532 non-null  int64
 4   max_playtime     10532 non-null  int64
 5   min_age          10532 non-null  int64
 6   min_players      10532 non-null  int64
 7   min_playtime     10532 non-null  int64
 8   name             10532 non-null  object
 9   playing_time     10532 non-null  int64
 10  thumbnail        10531 non-null  object
 11  year_published   10532 non-null  int64
 12  artist           7759 non-null   object
 13  category         10438 non-null  object
 14  compilation      410 non-null    object
 15  designer         10406 non-null  object
 16  expansion        2752 non-null   object
 17  family           7724 non-null   object
 18  mechanic         9582 non-null   object
 19  publisher        10529 non-null  object
 20  average_rating   10532 non-null  float64
 21  users_rated      10532 non-null  int64
dtypes: float64(1), int64(9), object(12)
memory usage: 1.8+ MB
```

In [7]:
```
board_games.describe().T
```

Out[7]:

	count	mean	std	min	25%	50%	75%	
game_id	10532.0	62059.203095	66223.716828	1.00000	5444.500000	28822.500000	126409.500000	21672
max_players	10532.0	5.657330	18.884403	0.00000	4.000000	4.000000	6.000000	99
max_playtime	10532.0	91.341436	659.754400	0.00000	30.000000	45.000000	90.000000	6000
min_age	10532.0	9.714964	3.451226	0.00000	8.000000	10.000000	12.000000	4
min_players	10532.0	2.070547	0.664394	0.00000	2.000000	2.000000	2.000000	
min_playtime	10532.0	80.882738	637.873893	0.00000	25.000000	45.000000	90.000000	6000
playing_time	10532.0	91.341436	659.754400	0.00000	30.000000	45.000000	90.000000	6000
year_published	10532.0	2003.070832	12.278296	1950.00000	1998.000000	2007.000000	2012.000000	201
average_rating	10532.0	6.370856	0.850364	1.38421	5.829585	6.392965	6.942675	
users_rated	10532.0	870.081466	2880.214998	50.00000	85.000000	176.000000	518.000000	6765

In [8]:
```
board_games.describe(include=object)
```

Out[8]:

	description	image	name	thumbnail	artist	
count	10532	10531	10532	10531	7759	
unique	10528	10527	10357	10527	4641	
top	How could that have happened? Black Stories ar...	//cf.geekdo-images.com/images/pic2410035.png	Robin Hood	//cf.geekdo-images.com/images/pic2410035_t.png	Franz Vohwinkel	Warg
freq	3	2	5	2	166	

Check for missing values

In [9]:
```
board_games.isnull().sum()
```

Out[9]:
```
game_id               0
description           0
image                 1
max_players           0
max_playtime          0
min_age               0
min_players           0
min_playtime          0
name                  0
playing_time          0
thumbnail             1
year_published        0
artist             2773
category             94
compilation       10122
designer            126
expansion          7780
family             2808
mechanic            950
publisher             3
average_rating        0
users_rated           0
dtype: int64
```

Handling missing values

> This is a critical step in data preprocessing that significantly impacts the quality and reliability of any analytical process because missing data can introduce biases, distort statistical analyses and lead to inaccurate or unreliable results.
>
> When data is not missing at random, but rather systematically, failing to address these gaps can introduce biases into analyses. This can lead to skewed insights and misrepresentations of the true underlying patterns in the data (Rubin, 1996).
>
> As a handling strategy I decided to create a new category "Not Mentioned" and input it in all the missing values for these categorical columns below

In [10]:
```
board_games['artist'] = board_games['artist'].fillna("Not Mentioned")
board_games['category'] = board_games['category'].fillna("Not Mentioned")
board_games['compilation'] = board_games['compilation'].fillna("Not Mentioned")
board_games['designer'] = board_games['designer'].fillna("Not Mentioned")
board_games['expansion'] = board_games['expansion'].fillna("Not Mentioned")
board_games['family'] = board_games['family'].fillna("Not Mentioned")
board_games['mechanic'] = board_games['mechanic'].fillna("Not Mentioned")
board_games['publisher'] = board_games['publisher'].fillna("Not Mentioned")
```

In [11]:
```
board_games.isnull().sum()
```

Out[11]:
```
game_id           0
description       0
image             1
max_players       0
max_playtime      0
min_age           0
min_players       0
min_playtime      0
name              0
playing_time      0
```

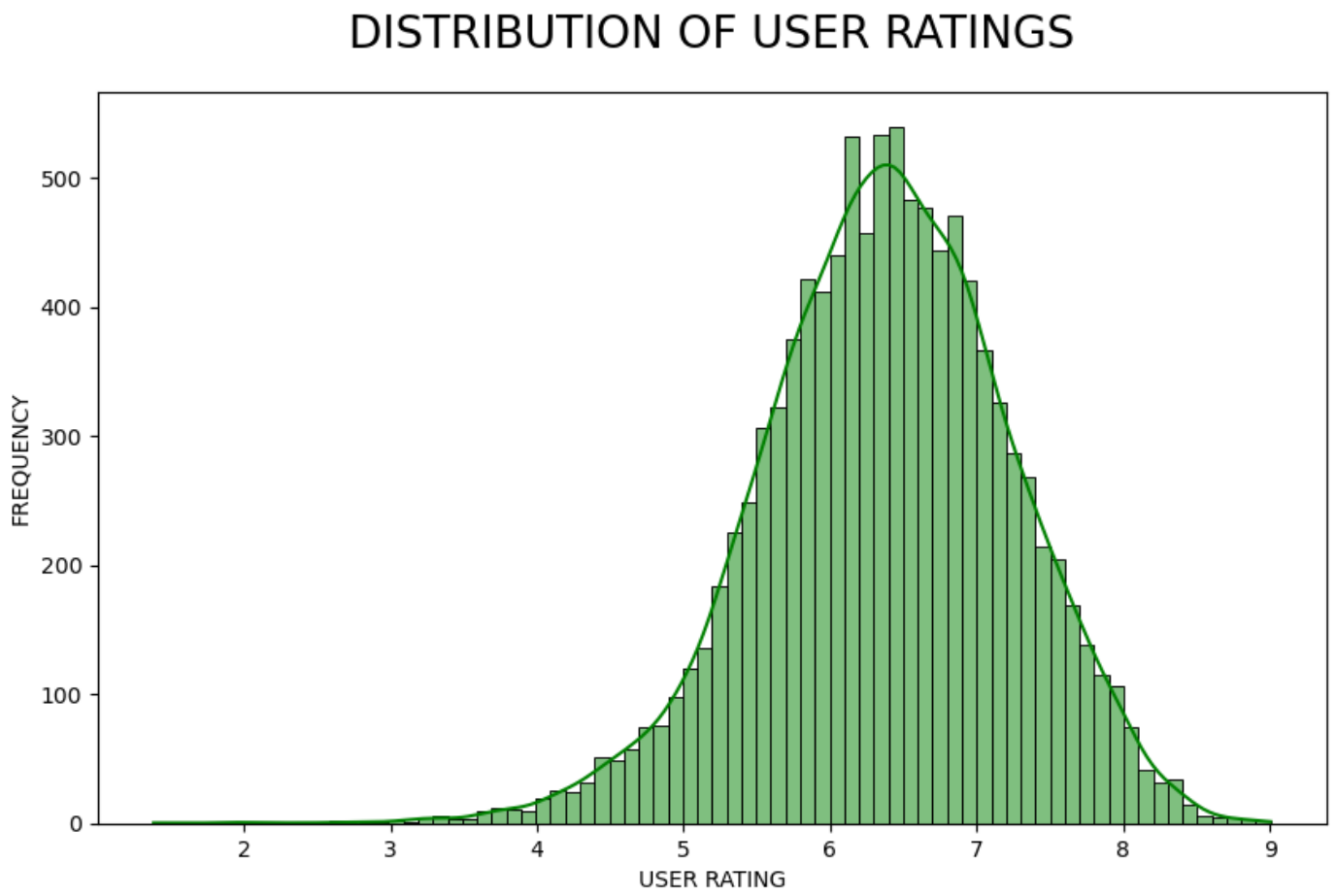
```
thumbnail          1
year_published     0
artist             0
category           0
compilation        0
designer           0
expansion          0
family             0
mechanic           0
publisher          0
average_rating     0
users_rated        0
dtype: int64
```

Exploratory Data Analysis (EDA):

- Understand the distribution of key variables.
- Identify any trends or patterns in the data.
- Visualize summary statistics.

In [12]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [13]:
```
plt.figure(figsize=(10, 6))
sns.histplot(board_games['average_rating'], kde=True, color='green')
plt.title('DISTRIBUTION OF USER RATINGS', pad=20, fontsize=20)
plt.xlabel('USER RATING')
plt.ylabel('FREQUENCY')
plt.show()
```



Distribuition of user ratings

By analysing the distribuition of user ratings we can indentify trends indicating which games are well-liked and which may need to be improved, so it will be crucial in providing us with insights into how diffent board games are received by the gaming community.

This information can give to the industry experts a competitve edge and enables them to make datadriven decision that appeal to a wide range of gaming audiances.

So, first of all I decided of utilizing a histogram with a kernel density estimate overlay that gives the dsitribution of user ratings a clear and plain visual depiction. This graphical method successfully conveys the range of ratings and makes it simple

Analysing the plot above - where 1 might represent a very low rating and 10 might represent a perfect rating - is possible to see the highest bars are between 6 and 7 that indicate that the majority of board games in the dataset have received user ratings within this range.

This distribution could imply there is a cluster of games that are considered good or satisfactory by most users, however there may be a lack of games that are considered exceptional or subpar according to the ratings. This information could be valuable for game developers, publishers, and distributors as it provides insights into the general sentiment of the gaming community towards the board games in question. They may use this information to refine existing games or make decisions regarding future game development and marketing strategies.

Correlation

In [14]:
```
board_games.corr()
```

```
C:\Users\barba\AppData\Local\Temp\ipykernel_8408\1236302191.py:1: FutureWarning: The def
ault value of numeric_only in DataFrame.corr is deprecated. In a future version, it will
default to False. Select only valid columns or specify the value of numeric_only to sile
nce this warning.
  board_games.corr()
```

Out[14]:

	game_id	max_players	max_playtime	min_age	min_players	min_playtime	playing_time	year_p
game_id	1.000000	0.017927	-0.037511	0.046359	-0.047049	-0.044301	-0.037511	
max_players	0.017927	1.000000	-0.003646	-0.008389	0.063518	-0.004041	-0.003646	
max_playtime	-0.037511	-0.003646	1.000000	0.041610	0.060724	0.975336	1.000000	-
min_age	0.046359	-0.008389	0.041610	1.000000	0.054614	0.037476	0.041610	-
min_players	-0.047049	0.063518	0.060724	0.054614	1.000000	0.067274	0.060724	
min_playtime	-0.044301	-0.004041	0.975336	0.037476	0.067274	1.000000	0.975336	-
playing_time	-0.037511	-0.003646	1.000000	0.041610	0.060724	0.975336	1.000000	-
year_published	0.704302	0.017946	-0.051373	-0.008630	0.026440	-0.055729	-0.051373	
average_rating	0.345837	-0.026564	0.056439	0.167772	-0.117876	0.037745	0.056439	
users_rated	-0.006245	-0.003682	-0.004342	0.073209	0.001996	-0.007752	-0.004342	

Correlation analysis allows data scientists to determine the strength and direction of associations, helping to identify potential predictor variables for modeling tasks (Hastie et al., 2021). According to James et al. (2013), understanding the correlations between variables is

crucial in feature engineering, as it helps in identifying redundant or highly correlated features. Notably, a positive correlation of 0.308 exists between "year_published" and "average_rating", suggesting that games published in later years may tend to receive higher average ratings. Furthermore, a positive correlation of 0.228 is observed between "average_rating" and "users_rated", implying that games with higher average ratings tend to garner more user ratings. It is important to emphasize that while correlations offer valuable insights into associations between variables, they do not imply causation.

PART 1

• **What are the top 5 "average rated" games?**

In [15]:
```
top_rated_games = board_games.sort_values(by='average_rating', ascending=False).head(5)
top_rated_games[['name', 'average_rating']]
```

Out[15]:

	name	average_rating
8348	Small World Designer Edition	9.00392
6392	Kingdom Death: Monster	8.93184
9964	Terra Mystica: Big Box	8.84862
8526	Last Chance for Victory	8.84603
9675	The Greatest Day: Sword, Juno, and Gold Beaches	8.83081

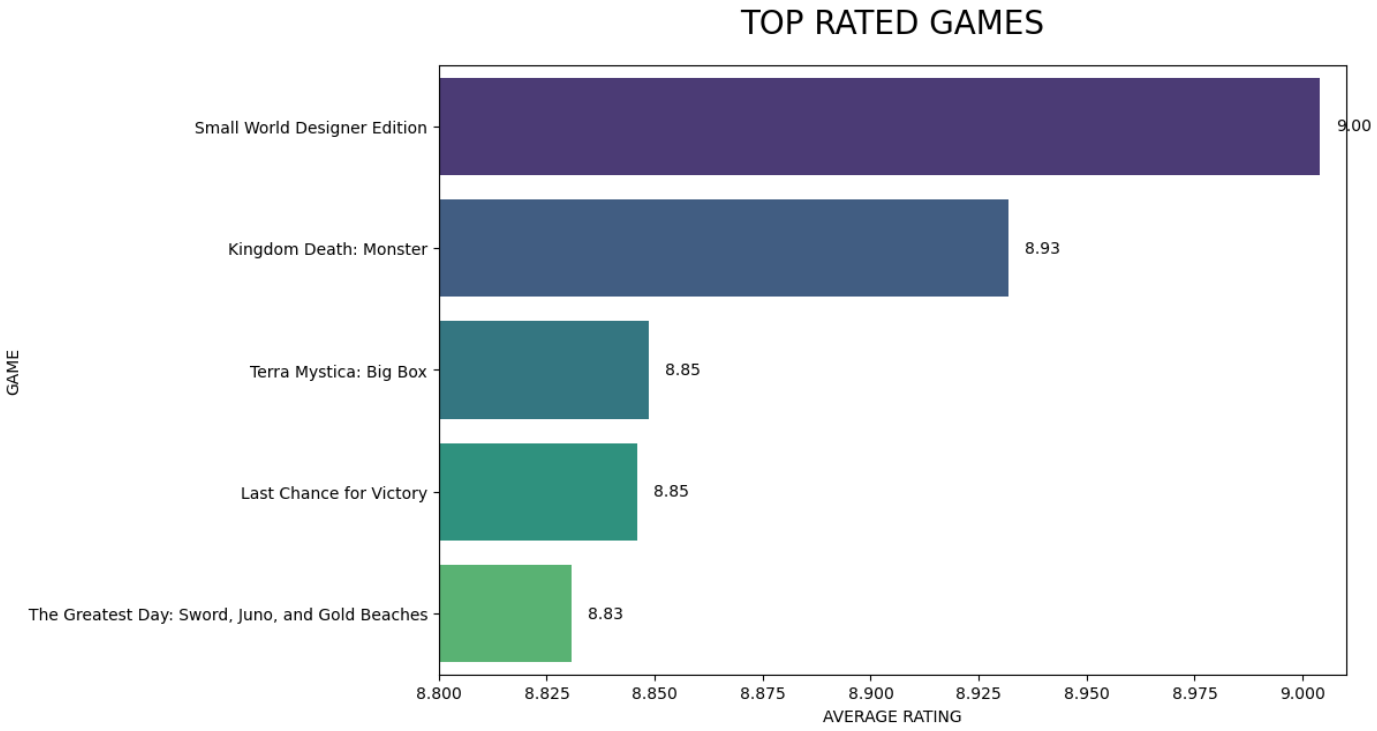
Above we have a table with the names and average rating of the top 5 games

To find the games with the highest average ratings I decided to order the dataset by 'average_rating' in descending order. Python provides an efficient way to accomplish this task using the .sort_values() method in the pandas library. The names and ratings are displayed toghether to give a clear and concise piscture of the best-rated game.

Through this table, stakeholders will be able to essily see which game have the highest average ratings, which can be helpful when deciding which titles to emphasize in the sales plan.

In [16]:
```
plt.figure(figsize=(12,6))
sns.set_palette("viridis")
ax = sns.barplot(x="average_rating", y= "name", data=top_rated_games, orient="h")

for p in ax.patches:
    ax.annotate(f'{p.get_width():.2f}', (p.get_width(),p.get_y()+ p.get_height() / 2), h
    plt.tight_layout()

plt.ylabel('GAME')
plt.xlabel('AVERAGE RATING')
plt.title('TOP RATED GAMES', pad=20, fontsize=20)

# Set the x-axis limits
```

```
plt.xlim(8.8, 9.01)

plt.show()
```



> To represent the top 5 “average rated” games in a graphic I chose the bar plot - firt of all I tryied a vertical plot which wasn't too good to visualize the diferences, so I changed to the horizontal one.
>
> Using the 'viridis' color palette for a visually pleasing gradient effect, ensuring the colors are visually appealing and distinguishable. Game names are displayed on the vertical axis, with corresponding average ratings on the horizontal axis, allowing easy comparison and understanding.
>
> Even though according to Cleveland and McGill's seminal work on graphical perception, bar plots are particularly effective for comparing values across categories, what we can see here is that the differences are not very large, so I decided to add the rates at the top of the bars, increase x parameter and also inlcude values at the top of each bar to highlight it even more.

- **Is there a correlation between “users_rated” and “max_playtime”?**

> To compute the correlation, I used the Pearson correlation coefficient, which measures the strength and direction of a linear relationship between two continuous variables. In Python, this was achieved using the corr() function from the pandas library. It ranges from -1 to +1, where:
>
> ```
> -1 indicates a perfect negative linear relationship;
> +1 indicates a perfect positive linear relationship and
>  0 indicates no linear relationship.
> ```

> Moreover, the Pearson correlation is particularly important because it is sensitive to linear relationships, making it a useful tool for assessing connections between variables (Freedman, Pisani, & Purves, 2007).

In [17]:
```
correlation = board_games['users_rated'].corr(board_games['max_playtime'])
print(correlation)
```

```
-0.004341647333776705
```

In [18]:
```
correlation = board_games[['users_rated','max_playtime']]
```

In [19]:
```
correlation.head()
```

Out[19]:

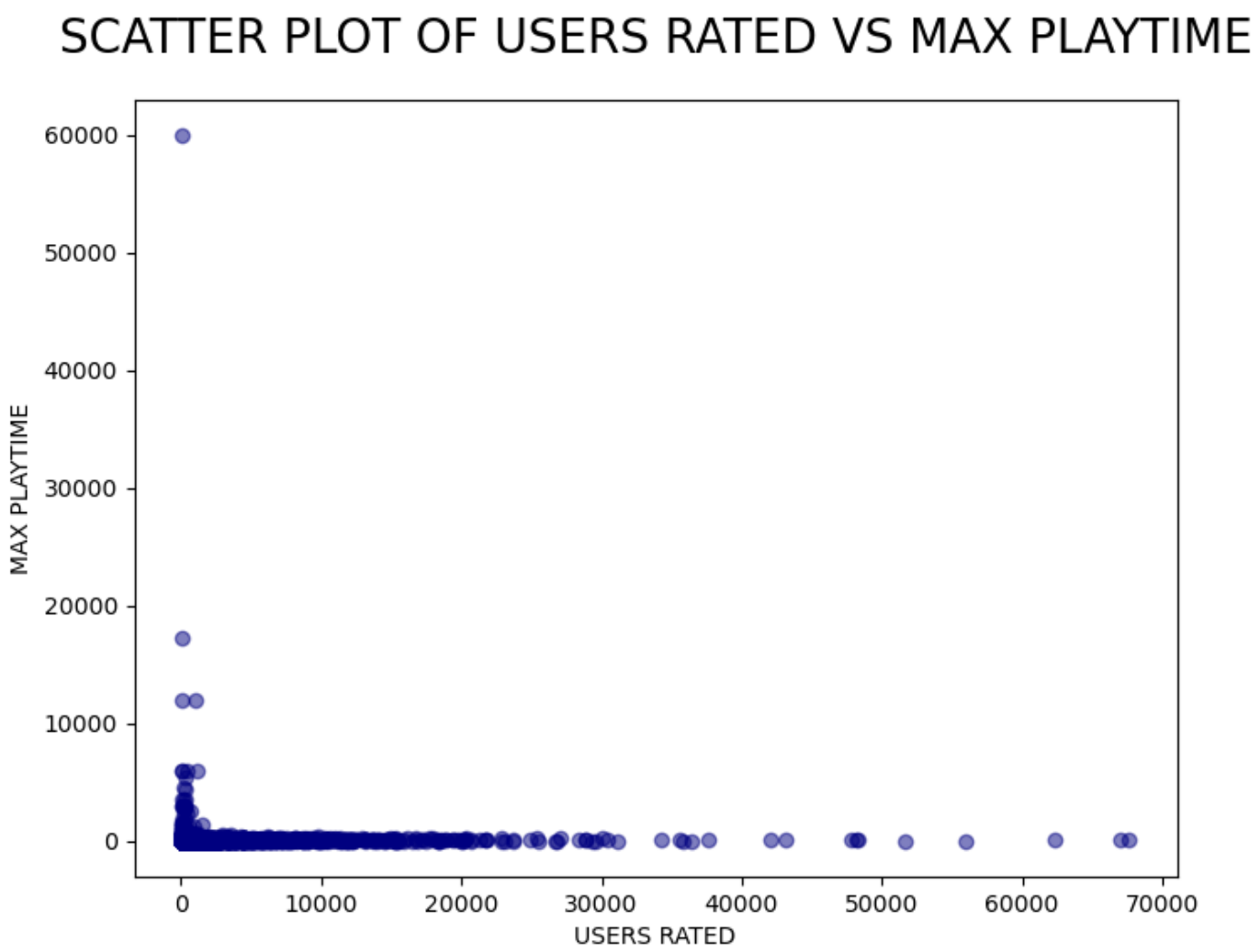
	users_rated	max_playtime
0	4498	240
1	478	30
2	12019	60
3	314	60
4	15195	90

Correlation result

> The resultante correlation coefficient which is aroud -0.0043 indicates a weak, nearly insignificant, linear association between these variables. This suggests that the popularity of a game, as indicated by the number of ratings, is not significantly influenced by how long the game takes to play. Consequently it could be more advantageous for the retail organization to concentrate on others aspects.

Visualization

In [20]:
```
plt.figure(figsize=(8, 6))
plt.scatter(board_games['users_rated'], board_games['max_playtime'], alpha=0.5, c='navy'
plt.title('SCATTER PLOT OF USERS RATED VS MAX PLAYTIME',  pad=20, fontsize=20)
plt.xlabel('USERS RATED')
plt.ylabel('MAX PLAYTIME')
plt.show()
```



> According to Healy and Moody (2014), scatter plots are particularly useful for revealing relationships between variables and are widely used in exploratory data analysis (EDA) to gain initial insights into the data. Based on that the scatter plot was chosen for its effectiveness in visually representing the relationship between two numerical variables.

• **What is the distribution of game categories? (You cannot use a bar chart)**

> To achieve an understanding of the distribution of game categories I decided to employ .counts function to tally the occurrences of each unique category within the 'category' column. This function efficiently counts the number of instances of each category, providing a concise summary. Subsequently, I construct a DataFrame named category_counts_df to organize and present this information in a structured format. This DataFrame contains two columns: 'Category' to hold the names of the different game categories, and 'Count' to store the corresponding frequency of each category.

In [21]:
```
category_counts = board_games['category'].value_counts()
category_counts_df = pd.DataFrame({'Category': category_counts.index, 'Count': category_
print(category_counts_df)
```

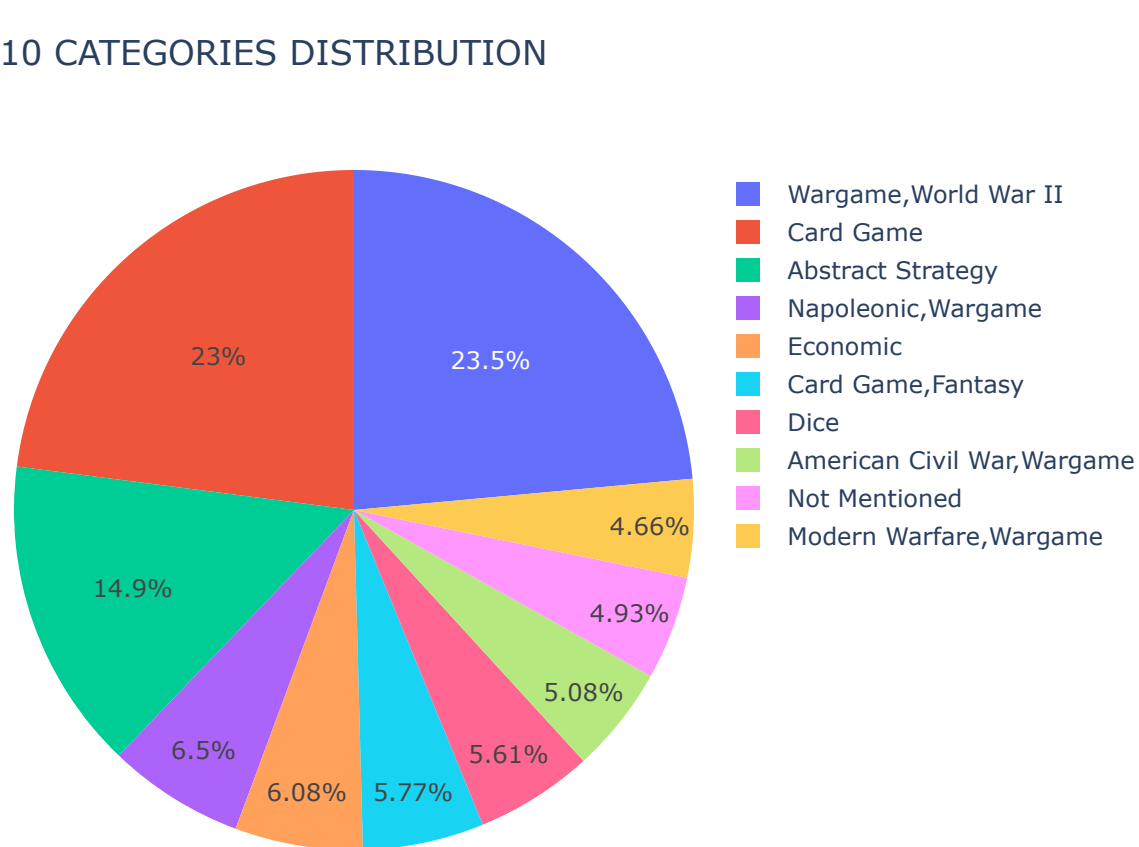
```
                                  Category  Count
0                    Wargame,World War II    449
1                               Card Game    438
```

```
2                                      Abstract Strategy    284
3                                    Napoleonic,Wargame    124
4                                              Economic    116
...                                                 ...    ...
3856                          Book,Fantasy,Miniatures        1
3857  Adventure,Card Game,Fantasy,Humor,Movies / TV ...      1
3858                Card Game,Deduction,Print & Play         1
3859  Card Game,Collectible Components,Comic Book / ...      1
3860  Bluffing,Horror,Maze,Movies / TV / Radio theme...      1

[3861 rows x 2 columns]
```

Visualization I

> To represent the distribution I decided to taken the top 10 categories.
>
> Opting to visualize only the top 10 categories is a strategic choice with several benefits. This approach will focus on the dataset's most influential parts, providing insights into key areas that are likely to significantly impact the company's sales strategy. Also, it reduces complexity, ensuring that the visualization remains clear and interpretable without becoming cluttered. Presenting a large number of categories may overwhelm stakeholders and dilute the impact of the visualization. A pie chart, for example, can become visually cluttered if it contains too many categories. Limiting the chart to the top 10 ensures that the visualization remains aesthetically pleasing and easy to interpret.

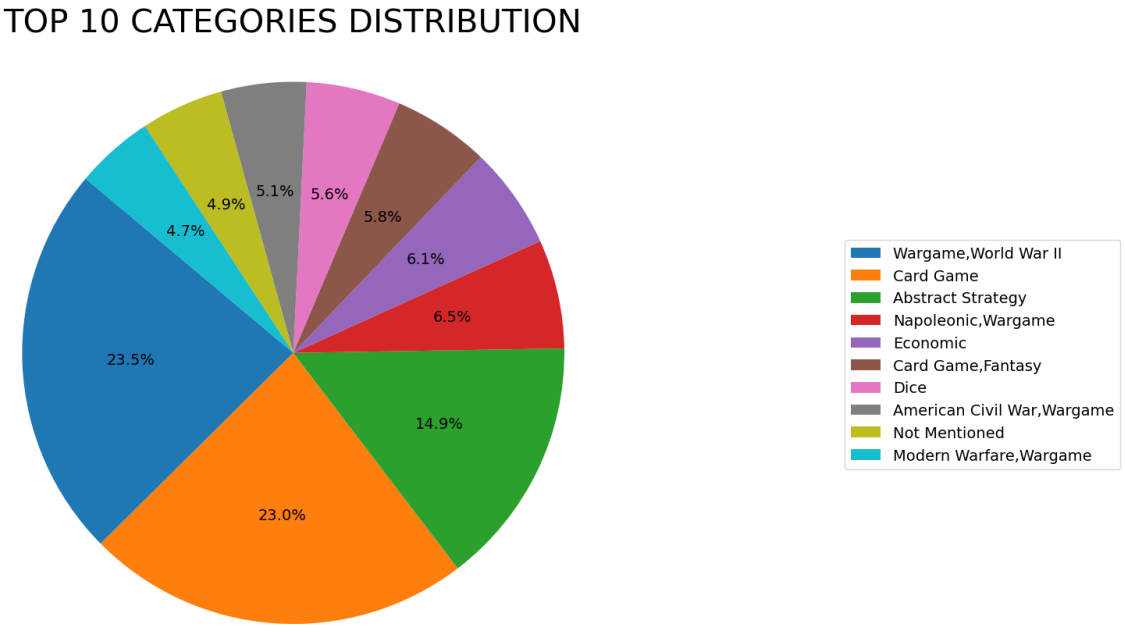
In [22]:
```
import pandas as pd
import plotly.express as px

top_10_categories = category_counts_df.head(10)

fig = px.pie(top_10_categories, names='Category', values='Count', title='TOP 10 CATEGORI
fig.show()
```



> **NOTE:** Doesn't appear on GitHub, so I did a similar one but not interative.

Visualization II

In [23]:
```
import matplotlib.pyplot as plt

# Assuming you have a DataFrame 'top_10_categories' with 'Category' and 'Count' columns
top_10_categories = category_counts_df.head(10)

# Define custom colors to match the Plotly Express colors
custom_colors = ['#1f77b4', '#ff7f0e', '#2ca02c', '#d62728', '#9467bd', '#8c564b', '#e37

# Increase the title distance (pad) and adjust the figure size
plt.figure(figsize=(18, 10))
plt.title('TOP 10 CATEGORIES DISTRIBUTION', pad=20, fontsize=30)  # Increase the pad val

# Create a pie chart with custom colors, remove labels, and adjust percentage size
wedges, texts, autotexts = plt.pie(top_10_categories['Count'], autopct='%1.1f%%', starta
                                   labels=['' for _ in top_10_categories['Category']])

# Adjust the percentage size inside the pie
for autotext in autotexts:
    autotext.set_fontsize(14)  # Adjust fontsize as needed

# Create a legend box beside the pie plot
plt.legend(wedges, top_10_categories['Category'], loc='center left', bbox_to_anchor=(1,

plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
plt.show()
```



Visualization III

> I also decided to use another data visualization technique called Word Cloud that represents

textual data in a visual format. Their simplicity and intuitive nature make them easily digestible for audiences, with larger, bolder words indicating greater frequency or importance (Brown, 2018). Since it is a gaming industry, I thought it would be similar to a puzzle or a word search game of sorts that could immediately capture the audience's attention. I should point out, however, that WordClouds do not always provide accurate numerical information, making them less suitable for presentations that require exact values (Jones, 2021). But since they would already be familiar with the content, this wouldn't be a problem.

In [24]:
```
from wordcloud import WordCloud
import matplotlib.pyplot as plt
```

In [25]:
```
# Combine all category strings into a single string
all_categories = ','.join(board_games['category'].dropna())

# Create a word cloud object
wordcloud = WordCloud(width=800, height=400, background_color='white').generate(all_cate

# Display the word cloud
plt.figure(figsize=(10, 6))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title('DISTRIBUTION OF GAME CATEGORIES - WORD CLOUD', pad=20, fontsize=20)
plt.show()
```

• Do older games (1992 and earlier) have a higher MEDIAN "average rating" than newer games (after 1992)?

> To determine if older games (released in 1992 and earlier) have a higher median "average rating" than newer games I will fisrt split the data into those two groups.

In [26]:
```
older_games = board_games[board_games['year_published'] <= 1992]
```

```
newer_games = board_games[board_games['year_published'] > 1992]
```

> Now I will find the median "average rating" for each group.

In [27]:
```
median_rating_older = older_games['average_rating'].median()
median_rating_newer = newer_games['average_rating'].median()
```

> And finally I can print and compare the Medians

In [28]:
```
print(median_rating_older, median_rating_newer)
```

```
6.08812 6.462915
```

> By comparing the median average ratings of older and newer games, we can gain insights into whether there is a trend in ratings based on the release year. This analysis can help inform decisions about which types of games may be more popular or well-received by the audience.
>
> As it shows in the result the median rating for older games (released in 1992 and earlier) is approximately 6.08812, compared the median rating for newer games (released after 1992) is approximately 6.462915. So that suggest in general newer games have a litlle higher median rating than older games.
>
> It's crucial to remember that the difference between the median is quite small (under 0.4) - indicating that there MAY NOT be much of a substatial diffference in the overall quality or popularity among them. In the bussiness point of view I would say based on the generally positive ratings both have that there is potetial for sale.

Visualization

In [29]:
```
games_age = board_games[['name', 'year_published', 'average_rating']]
```

In [30]:
```
games_age.head()
```

Out[30]:

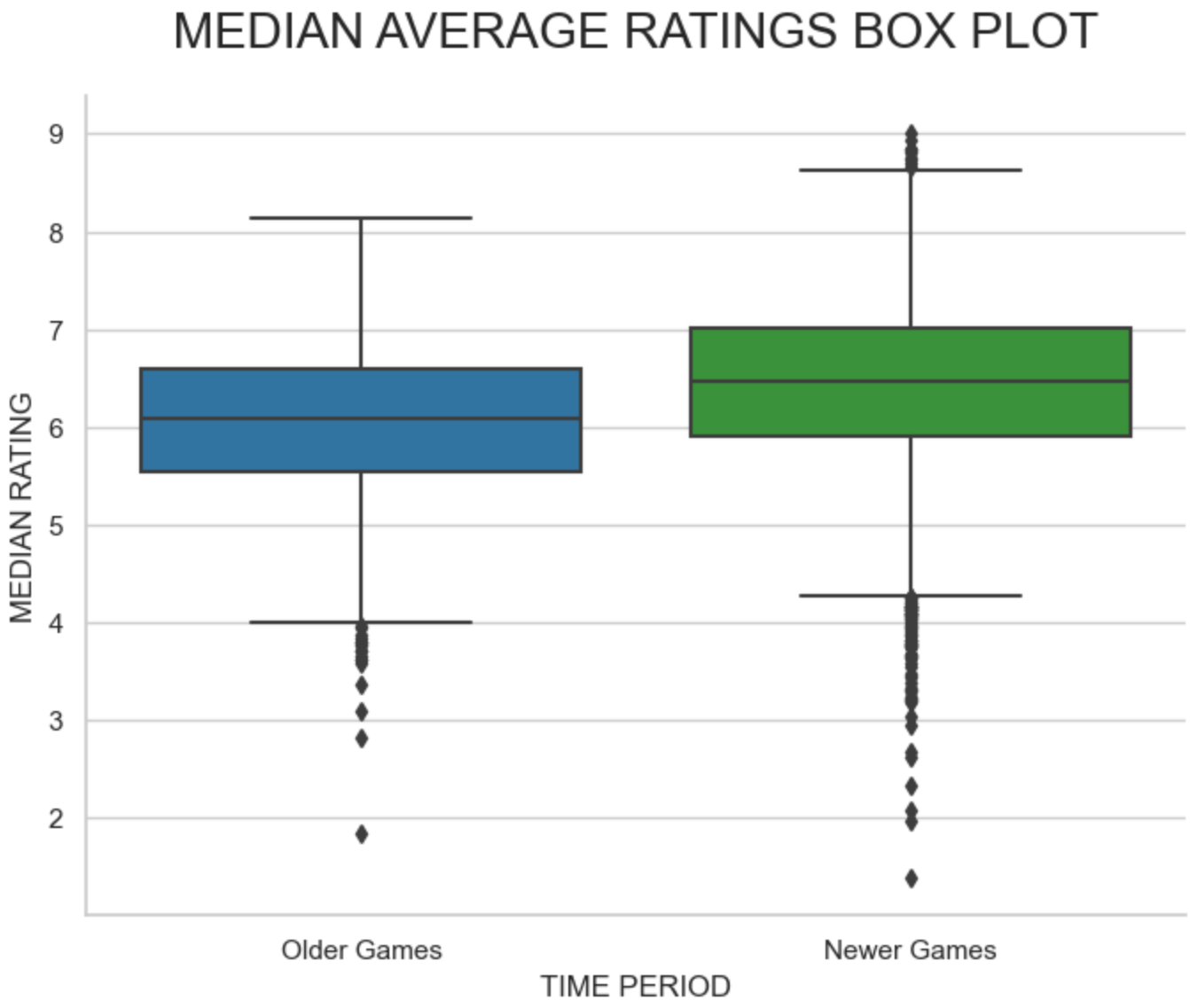
	name	year_published	average_rating
0	Die Macher	1986	7.66508
1	Dragonmaster	1981	6.60815
2	Samurai	1998	7.44119
3	Tal der Könige	1992	6.60675
4	Acquire	1964	7.35830

In [31]:
```
games_age.loc[board_games['year_published'] <= 1992, 'category'] = 'Older Games'
games_age.loc[board_games['year_published'] > 1992, 'category'] = 'Newer Games'
```

```
C:\Users\barba\AppData\Local\Temp\ipykernel_8408\3510818633.py:1: SettingWithCopyWarnin
g:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
```

In [32]:
```python
def create_comparison_boxplot(dataframe, x_column, y_column, title):

    # Set the Seaborn style to mimic The Economist's style
    sns.set(style="whitegrid")
    sns.set_palette(["#1f77b4", "#2ca02c"])

    #Creating a boxplot
    plt.figure(figsize=(8, 6))
    ax = sns.boxplot(x=x_column, y=y_column, data=dataframe)
    sns.despine(right=True, top=True)

    #Adding labels and title
    plt.xlabel('TIME PERIOD')
    plt.ylabel('MEDIAN RATING')
    plt.title('MEDIAN AVERAGE RATINGS BOX PLOT', pad=20, fontsize=20)
    plt.show()

create_comparison_boxplot(games_age, 'category', 'average_rating', 'GAMES AGE')
```



> This graphical representation provides a clear depiction of the central tendencies of each group, allowing for easy discernment of any differences. The results displayed in the box plot indicate that, on average, newer games (6.21) tend to receive slightly higher ratings compared

> to older games (5.95). These median values are represented by the middle line within each box, signifying the 50th percentile of the data when sorted. For both groups, this line reflects the median average rating. Notably, it's evident from the graph that the middle line (median) for older games (left box) is lower than that for newer games (right box), further highlighting the difference in average ratings between the two categories. This observation aligns with initial indications of distinct player preferences between older and newer games.

• **What are the 5 most common “mechanics” in the dataset?**

> As we know from the previews output "mechanic" has 950 missing values. So I have decided replace NA values as "Not Mentioned".

In [33]:
```
board_games["mechanic"] = board_games["mechanic"].fillna("Not Mentioned")
```

> That collumn also have other particularites as a list of string values, values separeted by "," and "/". To exclued these I will be cleaning and forming a tokenized value.

In [34]:
```
import re

def Clean(Text):
    sms = re.sub('[^a-zA-Z], ,', ' ', str(Text))
    sms = sms.replace("/",",")
    sms = sms.split()
    sms = ' '.join(sms)
    return sms
board_games["mechanic"] = board_games["mechanic"].apply(Clean)

print("First 5 values of Mechanics after cleaning text:")
print(board_games["mechanic"][:5], "\n")
```

```
First 5 values of Mechanics after cleaning text:
0    Area Control , Area Influence,Auction,Bidding,...
1                                        Trick-taking
2    Area Control , Area Influence,Hand Management,...
3    Action Point Allowance System,Area Control , A...
4         Hand Management,Stock Holding,Tile Placement
Name: mechanic, dtype: object
```

Count the frequency of each game mechanic

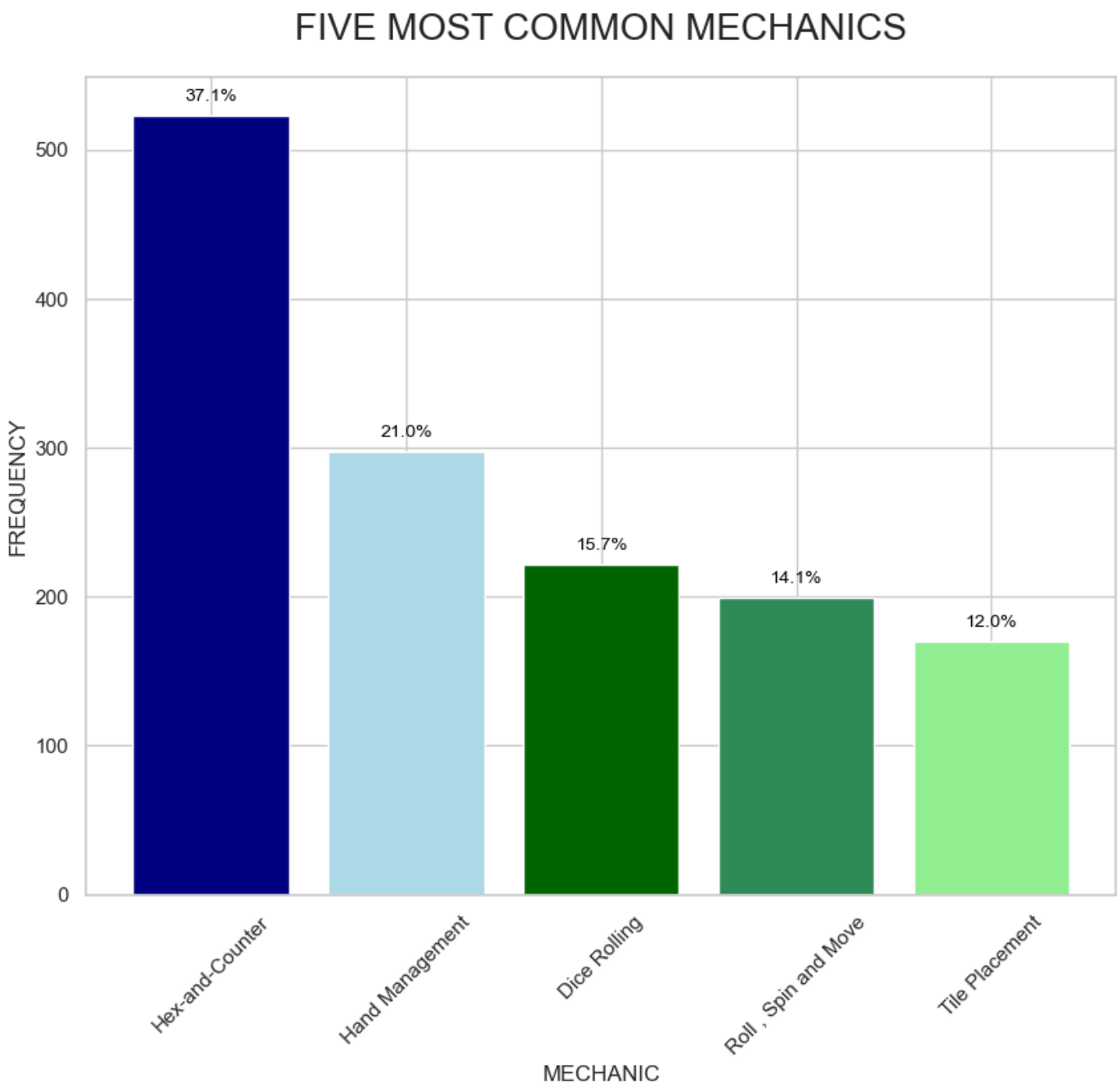
In [35]:
```
common_mechanics = board_games['mechanic'].value_counts().head(6)
common_mechanics = common_mechanics[common_mechanics.index != "Not Mentioned"]

print("The 5 most common “mechanics” in the dataset are:")
common_mechanics
```

Out[35]:
```
The 5 most common “mechanics” in the dataset are:
Hex-and-Counter          523
Hand Management          297
Dice Rolling             222
Roll , Spin and Move     199
Tile Placement           170
Name: mechanic, dtype: int64
```

> This information will may help stakeholders understand which game mechanics are most commonly associated with the board games. This knowledge can be valuable for making decision related to game selection and marketing strategies.

Visualization

In [36]:
```
import matplotlib.pyplot as plt

# Count the frequency of each game mechanic
common_mechanics = board_games['mechanic'].value_counts().head(6)

# Excluding "Not Mentioned"
common_mechanics = common_mechanics[common_mechanics.index != "Not Mentioned"]

# Define custom colors
colors = ['navy', 'lightblue', 'darkgreen', 'seagreen', 'lightgreen']

# Calculate percentages
total_count = common_mechanics.sum()
percentages = (common_mechanics / total_count) * 100


# Create a vertical bar chart with different colors and add percentages
plt.figure(figsize=(10, 8))
bars = plt.bar(common_mechanics.index, common_mechanics.values, color=colors)

for bar, percentage in zip(bars, percentages):
    plt.text(bar.get_x() + bar.get_width()/2 - 0.0, bar.get_height() + 10, f'{percentage

plt.xlabel('MECHANIC')
plt.ylabel('FREQUENCY')
plt.title('FIVE MOST COMMON MECHANICS', pad=20, fontsize=20)
plt.xticks(rotation=45)  # Rotate x-axis labels for better visibility
plt.show()
```



> The horizontal bar chart clearly shows how the various game mechanics are distributed, with hex-and-counter being the most common mechanism, appearing in a total of 37.1% occurrences. With 21% occurrences, Hand Management comes in second with a sizable margin, and Tile Placement takes fifth with 12%. It was intentional to choose a horizontal bar chart with a white grid background because it successfully communicates this information with visual clarity and aesthetic appeal, facilitating simple understanding.
>
> This serves as a valuable tool for quickly finding the parts of the data that occur most frequently while also providing a clear depiction of the distribution across categories.
>
> I chose to rotate the legend and include the percentage of each bar to again even more the aesthetic attractiveness and communication.

PART 2

• You must answer a "Statistically Relevant" question, OF YOUR OWN CHOOSING, using the dataset, that has not been asked in Part 1. This must have a logical basis that enhances the information and insight gained in the scenario.

QUESTION: *Does the "average rating" of games vary significantly between different "categories"?*

> For this analysis, first, I will prepare the data focus on the top 10 categories.

In [37]:
```
# Assuming 'board_games' is your DataFrame with columns 'average_rating' and 'category'
top_10_categories = board_games['category'].value_counts().head(10).index
top_10_category_games = board_games[board_games['category'].isin(top_10_categories)]
```

> Having a list of the top 10 categories, now I will apply the ANOVA test

In [38]:
```
from scipy.stats import f_oneway

# Perform one-way ANOVA
f_statistic, p_value = f_oneway(
    *[top_10_category_games[top_10_category_games['category'] == category]['average_rati
)

if p_value < 0.05:
    print("There is a significant difference in average ratings between different catego
else:
    print("There is no significant difference in average ratings between different categ
```

```
There is a significant difference in average ratings between different categories.
```

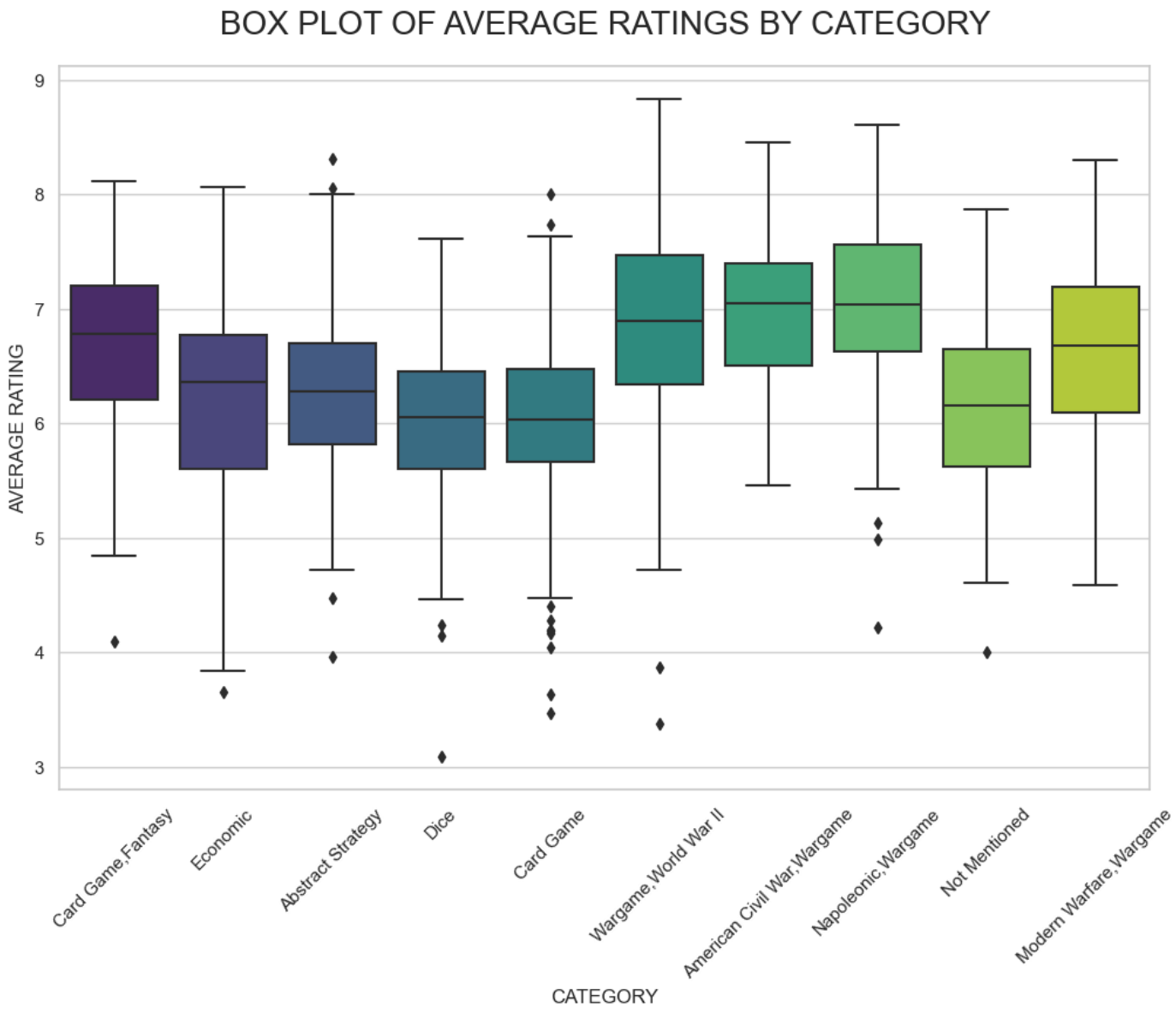
> The result above indicates that the average ratings between various categories of board games are considerably distinct when the p-value is less than 0.05 (assuming a significance level of 5%).
>
> This means that player satisfaction and reception vary significantly depending on the type or genre of the game, which is a crucial finding in regards the game industry
>
> Furthermore, this insight empowers us to tailor the offerings to meet the unique demands of one target audience within a specific category. By aligning those products more closely with the preferences of the users, will be possible to anticipate a positive impact on customer satisfaction and engagement levels.

Visualization

> To visualize my results I decided to use boxplot to compare avarage ratings across the top 10 categories.

In [39]:
```
import seaborn as sns
import matplotlib.pyplot as plt

# Assuming 'top_10_category_games' contains the filtered data for the top 10 categories
```

```
# Create a box plot
plt.figure(figsize=(12, 8))
sns.boxplot(x='category', y='average_rating', data=top_10_category_games, palette='virid
plt.xticks(rotation=45)
plt.title('BOX PLOT OF AVERAGE RATINGS BY CATEGORY', pad=20, fontsize=20)
plt.xlabel('CATEGORY', fontsize= 12)
plt.ylabel('AVERAGE RATING', fontsize= 12)
plt.show()
```



> The generated box plot visually represents the distribution of average ratings across different categories of board games. Each box represents the interquartile range (middle 50% of data) for a specific category, with a line indicating the median rating. Whiskers extend to show the data range, while any points beyond them are considered outliers.

PART 3

Along the project I have alredy explain my decision after each part, but here there is an concise explain overall.

PART 1:

> **Data Engineering:** In order to ensure a robust analysis for the board game dataset, I began by addressing missing data. Several categorical columns contained missing values, which presented a challenge. To mitigate this, I employed a strategy of creating a new category "Not Mentioned" for these variables, ensuring that no information was lost. This approach strikes a balance between maintaining data integrity and allowing for meaningful analysis. Additionally, I performed data wrangling tasks such as filtering out irrelevant columns and selecting specific categories for analysis. Feature engineering was carried out to extract relevant information from existing columns, contributing to a more focused and insightful analysis. Furthermore, I conducted exploratory data analysis (EDA) to gain a deeper understanding of the dataset, which informed subsequent analytical decisions.
>
> **Visualization Choices:** The choice of visualization for each question was carefully tailored to best represent the underlying data and provide clear insights. For instance, in exploring the distribution of user ratings, I opted for a histogram with a kernel density estimate overlay. This choice allows for a visual depiction of the range of ratings, making it straightforward for stakeholders to discern trends in user sentiment. To enhance interpretability, color was selected to evoke positive sentiment, utilizing a palette with green shades. For categorical data, bar charts and pie charts were employed to effectively communicate proportions and distributions, ensuring that stakeholders could readily grasp the relevant information.
>
> **Design Decisions:** Several design considerations were made to optimize the visualizations and ensure they effectively convey information. The choice of font and font size was deliberately selected for readability and coherence across the entire project. Titles were strategically positioned to convey the purpose of each plot effectively. For example, the titles' capitalized and centered format immediately captures the essence of the visualization. Moreover, axis labels were used to provide context and ensure the reader can interpret the visual information accurately. The color palettes chosen were not only aesthetically pleasing but also ensured that data points were easily distinguishable. The general idea was to maintain a standard between all views, keeping the same font, size, and color, which contributes to a cohesive and visually appealing presentation of the data. This consistent design approach facilitates a seamless understanding of the information presented throughout the project.

PART 2:

> **Rationale for Visualization and Question:** The question posed in Part 2, "Does the 'average rating' of games vary significantly between different 'categories'?", is critical in providing valuable insights to stakeholders. To answer this, a one-way ANOVA test was employed. This method is appropriate as it compares means across different categories, allowing us to determine if there are significant differences in average ratings. By using Python and the SciPy library, I was able to perform this statistical analysis efficiently.
>
> The chosen visualization, a box plot, was selected for its effectiveness in displaying the distribution of average ratings across various game categories. Box plots provide a clear representation of central tendency, spread, and potential outliers within each category. The choice of color palette, 'viridis', was made for its perceptually uniform colors, ensuring that the reader can accurately interpret the data without distraction.

> This analysis and visualization add significant value by providing a nuanced understanding of how different game categories may influence user satisfaction. This insight empowers stakeholders to make informed decisions regarding game selection and marketing strategies, ultimately leading to improved customer engagement and satisfaction levels.

OVERALL RATIONALE >

> In summary, the chosen methods and visualizations were carefully selected to address specific aspects of the dataset and enhance the information and insights gained from the analysis. Each decision was made with the goal of providing clear, meaningful, and actionable insights for the hypothetical retail company. The design elements, such as colors, fonts, titles, and sizes, were chosen to optimize visual appeal and communication of the findings.

REFERENCES

Rubin, D. B. (1996). Multiple Imputation after 18+ Years. Journal of the American Statistical Association, 91(434), 473-489.

Cleveland, W. S., & McGill, R. (1984). Graphical perception: Theory, experimentation, and application to the development of graphical methods. Journal of the American Statistical Association, 79(387), 531-554.

Freedman, D., Pisani, R., & Purves, R. (2007). Statistics. W.W. Norton & Company.

Healy, K., & Moody, J. (2014). Data Visualization in Sociology. Annual Review of Sociology, 40, 105-128.

Brown, A. (2018). Visual Communication: From Theory to Practice. Bloomsbury Publishing.

Hastie, T., Tibshirani, R., & Friedman, J. (2021). The Elements of Statistical Learning: Data Mining, Inference, and Prediction (2nd ed.). Springer.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An Introduction to Statistical.

Jones, S. (2021). The Art of Data Science. Cambridge University Press.

Peiser, J (2022) We're in a golden age of board games. It might be here to stay, [Online] Available at: https://www.washingtonpost.com/business/2022/12/24/board-game-popularity/ (accessed October 13, 2023)

Seaborn. (n.d.). Choosing color palettes. Seaborn documentation. https://seaborn.pydata.org/tutorial/color_palettes.html (accessed October 15, 2023)