Xbox Game Pass Analysis

Le Duy Anh Ngo

My Github: AnhNgo2803

Learn more: Insight Dashboard, Github Repository

Data Source: Enhanced_Gamepass_Games_v2

Excutive summary

- This data analysis project examines the Xbox Game Pass dataset ('Enhanced_Gamepass_Games_v2' from Kaggle), comprising 455 games, to uncover factors influencing game popularity, completion rates, and ratings. The objective is to deliver actionable insights for game developers, enabling them to enhance user experiences and optimize gameplay strategies on the platform.
- Utilizing Python libraries such as Pandas, NumPy, Seaborn, Matplotlib, and Scikit-learn, the analysis began with comprehensive data preparation: inspecting structures, handling missing values (using mode for categorical and mean for numerical data), converting formats (e.g., time ranges to numerical averages), and dropping non-essential columns. Exploratory data analysis focused on genre preferences, revealing Sandbox, Compilation, and MMO as the most popular genres based on average gamers (247,504; 181,648; and 133,438 respectively). Visualization techniques, including bar charts and bubble plots, highlighted two success models: High-Engagement (long playtimes in MMO and Compilation, fostering deep immersion and community) and Mass-Appeal (short, creative experiences in Sandbox, attracting broad audiences).
- Further, K-means clustering on difficulty ratio and playtime classified games into four distinct types: Casual (easy-short), Epic (hard-very long), Challenge (very hard-short), and Mainstream (hard-long). Cross-tabulation and stacked bar charts showed Casual dominating with diverse genres like Action and Adventure, while Epic offers untapped potential with high gamer averages but limited variety. Key insights indicate that popularity stems from engagement depth or accessibility rather than completion rates. Recommendations include prioritizing multiplayer features and narrative depth for loyal communities (e.g., MMO/RPG), while emphasizing creativity and freedom for mass appeal (e.g., Sandbox). This approach can guide Xbox developers in resource allocation, genre diversification, and user-centric design, ultimately boosting retention and acquisition.
- Skills demonstrated: Data cleaning, statistical analysis, visualization, unsupervised machine learning, and insight derivation for business impact.

Problem statement

The goal of this project is to analyze the dataset for uncovering factors that affect the popularity, completion rate, and ratings of games. As a result, provide useful insights for game developers to optimize gaming experience.

Data preparation

Data inspection

```
# import essential library
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import matplotlib.patches as mpatches
```

```
# call the dataset
game = pd.read_csv('Enhanced_Gamepass_Games_v2.csv')
```

```
# briefly inspect structure
game.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 455 entries, 0 to 454
Data columns (total 15 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   GAME              455 non-null    object
 1   RATIO             455 non-null    object
 2   GAMERS            455 non-null    object
 3   COMP %            455 non-null    float64
 4   TIME              421 non-null    object
 5   RATING            452 non-null    float64
 6   ADDED             454 non-null    object
 7   True_Achievement  455 non-null    int64
 8   Game_Score        455 non-null    int64
 9   GENRES            453 non-null    object
 10  Main_Genre        453 non-null    object
 11  second_genre      452 non-null    object
 12  third_genre       350 non-null    object
 13  fourth_genre      5 non-null      object
 14  name_genre_match  453 non-null    object
dtypes: float64(2), int64(2), object(11)
memory usage: 53.4+ KB
```

```
# inspect data in first rows of data set
game.head()
```

	GAME	RATIO	GAMERS	COMP %	TIME	RATING
0	Mass Effect Legendary Edition	1.87	84,143	4.1	100-120 hours	4.8
1	The Elder Scrolls V: Skyrim Special Edition	1.97	213,257	8.0	80-100 hours	4.7
2	Mass Effect 2	1.34	221,178	9.6	50-60 hours	4.7
3	Stardew Valley	3.04	51,530	1.0	150-200 hours	4.7
4	It Takes Two	1.68	71,981	15.6	12-15 hours	4.7

Data manipulation

Through data structure and its corresponding values, there are variables that need to be adjusted to correct formats.

Deal with categorical data

```
# change data type to categorical data
game_new = game.copy() # create a copy of original data

game_new[['GAME','Main_Genre','second_genre','third_genre','fourth_genre','name_genre_match']
    ['GAME','Main_Genre','second_genre','third_genre','fourth_genre','name_genre_match']].as

game_new[['GAME','Main_Genre','second_genre','third_genre','fourth_genre','name_genre_match']
```

```
GAME                category
Main_Genre          category
second_genre        category
third_genre         category
fourth_genre        category
name_genre_match    category
dtype: object
```

```
# since the ratio column realize '-' in values, I inspect and remove it before changing data
game_new.groupby('RATIO')['RATIO'].agg('count').head()
```

```
RATIO
-       3
1.04    1
```

```
1.06    1
1.1     1
1.13    1
Name: RATIO, dtype: int64
```

```
# RATIO colum
game_new['RATIO'] = game_new['RATIO'].replace('-',np.nan).astype('float64')

# GAMERS column
game_new['GAMERS'] = game_new['GAMERS'].str.replace(',','').astype('int64')

# check result
game_new[['GAMERS','RATIO']].dtypes
```

```
GAMERS      int64
RATIO     float64
dtype: object
```

Deal with Time data (from categorical to number)

```
# inspect the values in time column
game_new.groupby('TIME')['TIME'].value_counts(dropna= False) # Na included
```

```
TIME
0-0.5 hours       1
0.5-1 hour        1
1-2 hours        17
10-12 hours      12
100-120 hours     9
1000+ hours       5
12-15 hours      15
120-150 hours     3
15-20 hours      31
150-200 hours    19
2-3 hours        11
20-25 hours      36
200-300 hours     7
25-30 hours      30
3-4 hours        11
30-35 hours      10
300-500 hours     3
```

```
35-40 hours       4
4-5 hours         7
40-50 hours      39
5-6 hours         5
50-60 hours      24
500-750 hours     1
6-8 hours        20
60-80 hours      42
8-10 hours       27
80-100 hours     31
Name: count, dtype: int64
```

For easier for analysis later, I added a column that take the mean of time frames in time column, except the "1000+ hours" data, which will be maintained as 1000.

```
# replace the words "hours" in new column
game_new['time_num'] = game_new['TIME'].str.replace(r' hours|hour','', regex=True)

# define a function
def time_convert(value):
    value = str(value)
    if "+" in value:
       x = value.replace("+","") # remove "+" for "1000+" cases
       return np.int64(x) # change data type
    else:
        x = value.split("-") # from "40-60" to ["40","60"]
        num = list(map(float,x)) # map func float to each value and change to list
        return np.mean(num) # calculate the mean

# apply function for each row in new column
game_new['time_num'] = game_new['time_num'].apply(time_convert)

# compare 2 columns (categorical vs numeric data)
print(game_new[['TIME','time_num']])
```

```
              TIME  time_num
0    100-120 hours     110.0
1     80-100 hours      90.0
2      50-60 hours      55.0
3    150-200 hours     175.0
4      12-15 hours      13.5
..             ...       ...
```

```
450    15-20 hours      17.5
451            NaN       NaN
452            NaN       NaN
453            NaN       NaN
454            NaN       NaN

[455 rows x 2 columns]
```

Deal with Na values

```
# inspect Na in each column
x = game_new.isnull().sum().sort_values(ascending=False)

print(x)
```

```
fourth_genre        450
third_genre         105
TIME                 34
time_num             34
RATIO                 3
RATING                3
second_genre          3
GENRES                2
Main_Genre            2
name_genre_match      2
ADDED                 1
GAME                  0
GAMERS                0
COMP %                0
True_Achievement      0
Game_Score            0
dtype: int64
```

Even though fourth and third genre column have the most na values among other variables, I did not process those variables, instead, I chose to eliminate these colomns since it's not highly essential for the analysis target.

```
# remove fourth and third genre column
game_new = game_new.drop(['fourth_genre','third_genre'], axis=1) # eliminate assigned coloum
```

Handle Na values

- With categorical data such as `TIME`, `second_genre`, `GENRES`, `Main_Genre`, `name_genre_match`, and `ADDED` (Time data), I used *mode* method to fill those Na values
- With continuos data such as `RATIO`, `RATING`, and `time_num`, I used *mean* method to fill those values.

```
# fill Na values of categorical and text data
cate_cols = ['TIME','second_genre','GENRES','Main_Genre', 'name_genre_match','ADDED']

# create for loop for automatically repeat process
for col in cate_cols:
    mode_val = game_new[col].mode()[0]
    game_new[col] = game_new[col].fillna(mode_val)
```

```
# fill Na values of numeric data
num_cols = ['RATIO','RATING','time_num']

# create for loop for automatically repeat process
for col in num_cols:
    mean_val = np.mean(game_new[col])
    game_new[col] = game_new[col].fillna(mean_val)
```

```
# check Na again
x = game_new.isnull().sum().sort_values(ascending=False)

print(x)
```

```
GAME                0
RATIO               0
GAMERS              0
COMP %              0
TIME                0
RATING              0
ADDED               0
True_Achievement    0
Game_Score          0
GENRES              0
Main_Genre          0
second_genre        0
name_genre_match    0
time_num            0
dtype: int64
```

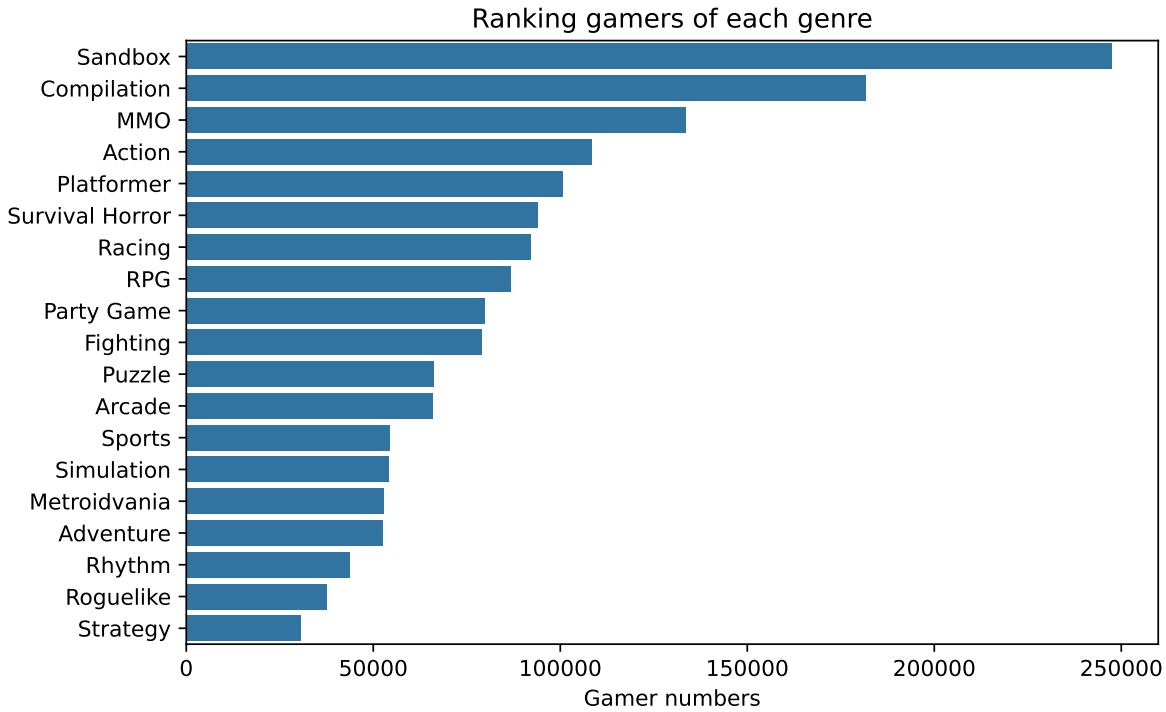
Genre analysis

In this section, I want to determine what are genres that gamers prefer most compared to the others. From that, finding features of those popular games to have a transparent model about a successful game, which enhances developing strategies for game developers in Xbox pass product.

```
game_new.info() # recall the structure and columns
```

```
genre_ana = game_new.groupby('Main_Genre', observed=True).agg(
    gamer = pd.NamedAgg(column = "GAMERS", aggfunc= "mean"),
    rating = pd.NamedAgg(column = "RATING", aggfunc= "mean"),
    comp_ratio = pd.NamedAgg(column = "COMP %", aggfunc= "mean"),
    play_time = pd.NamedAgg(column= "time_num", aggfunc = "mean")
).round(2).sort_values(by='gamer', ascending=False)
genre_ana
```

```
# visualize the number of gamers of each genre
plt.figure(figsize=(8,5))
f1 = sns.barplot(genre_ana, x="gamer", y="Main_Genre", orient="y", order = genre_ana.index)
f1.set_title("Ranking gamers of each genre")
f1.set_ylabel("")
f1.set_xlabel("Gamer numbers")
plt.show()
```



Initial analysis

Overall, it can be seen that Sandbox, Compilation and MMO are 3 genres played most by gamers, while Rhythm, Roguelike, and Strategy are least chosen. Specially, when comparing rating of the top 3 games and the least game, there are insignificant difference among those games. However, the main difference lies in the completion ratio and play time. Specifically, the most genre played (sandbox) have the high completion ratio with short play time, which is similar to least games played. In contrast, the following genres (Compilation and MMO) have the low completion percentage with long play time.

Thus, I dived into these differences among top ranked genres by using bubble chart, which shows the relationship between gamers, completion ratio, and playing duration.

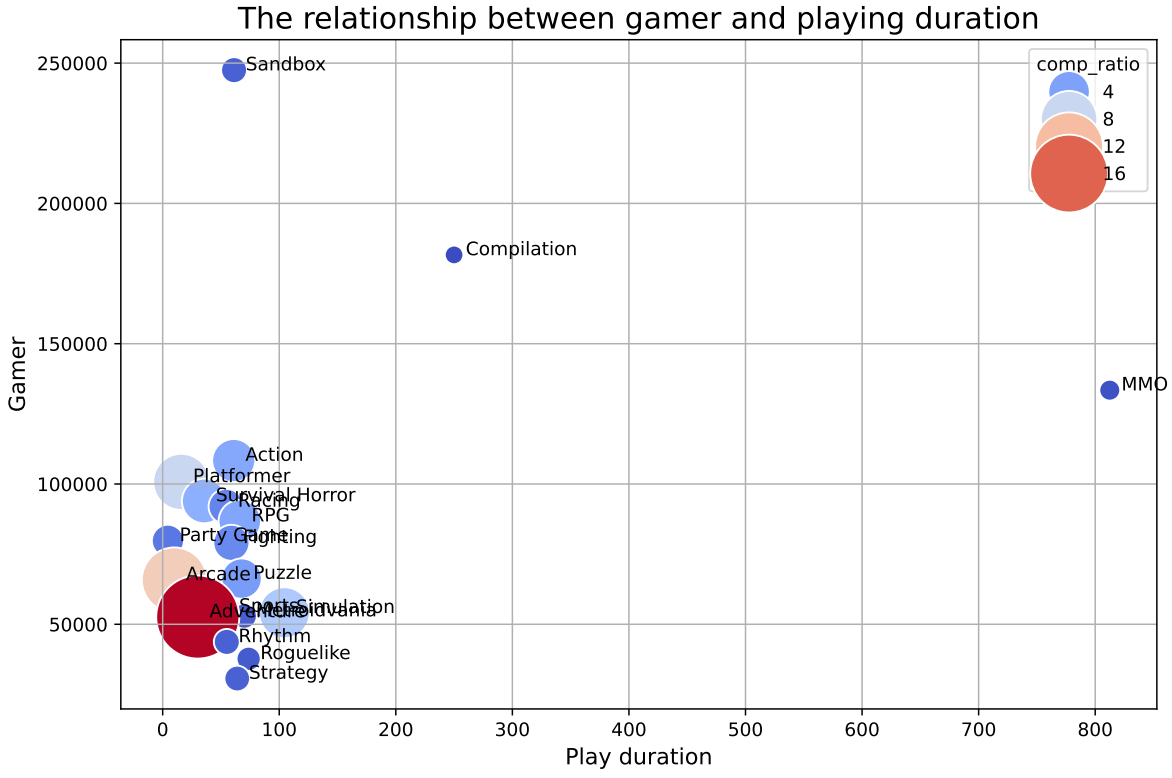
```
# create a bubble chart
plt.figure(figsize=(9,6))
f2 = sns.scatterplot(data = genre_ana,
                     y = "gamer",
                     x = "play_time",
                     size = "comp_ratio",
                     hue="comp_ratio",
                     sizes=(100,2000),
```

```
                    palette="coolwarm")

# set the legend for each genre in chart
for genre in genre_ana.index:
    plt.text(x = genre_ana.loc[genre,'play_time'] + 10,
             y = genre_ana.loc[genre, 'gamer'] - 100,
             s = genre,
             fontdict= dict(color="black", size=10))

# set title and legend
plt.title("The relationship between gamer and playing duration", fontsize=16)
plt.xlabel("Play duration", fontsize = 12)
plt.ylabel("Gamer", fontsize = 12)

plt.grid(True)
plt.tight_layout()
plt.show()
```



Insight

Our analysis reveals that popular game genres on Xbox Game Pass achieve success through two distinct models: **building a high-engagement community** and **attracting a mass-appeal audience.**

- The High-Engagement Model (MMO & Compilation): These genres have large number of players with long gaming duration. In other words, with these genres, gamers tend to invest more time in experiencing and exploring games. Moreover, with MMO, a genre whose game focuses on online multiple player, it also strengthens to the fact that people love player connection experience, which makes them spend more time in gaming. Meanwhile, Compilation genre concentrates on experiencing a game with extended versions or stories, which arouses the curiorsity of gamers to make them spend more time in games.
- The Mass-Appeal Model (Sandbox): This genre appriciates the creativity of players instead of building complex stories inside. In contrast to long average playtime as MMO and Compilation, Sandbox though has a short playing duration, it still obtains advocates from gamers thanks for its emphasize on freedom, creativity, and accessible gameplay. This suggests that games prioritizing freedom and creativity are highly effective in appealing to a wider, more general audience compared to other gameplay styles.

Conclusion & Recommendation: Notably, **completion ratio** is not a primary driver of popularity for these top-tier genres. Success is determined by the mode of engagement—either deep retention or broad appeal. Consequently, Xbox game developers should focus on crafting innovative gameplay experiences and investing in multiplayer enhancements. This dual approach can help sustain a loyal gaming community while also attracting new players with fresh game offerings.

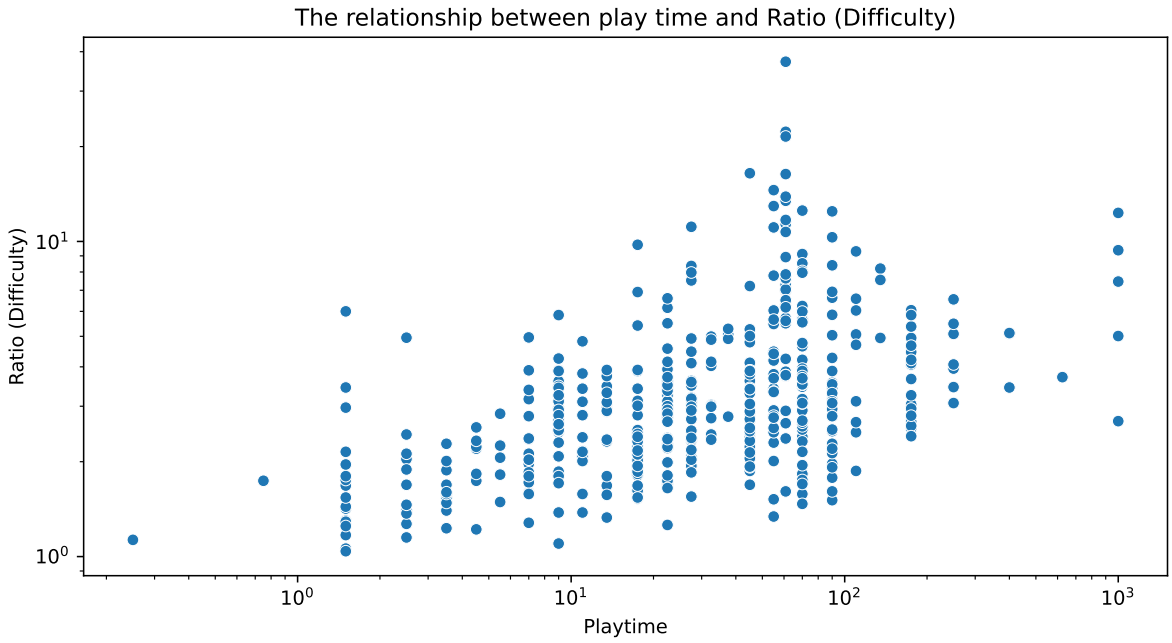
Difficulty and Playtime Analysis

In this part, I want to focus on the gaming experience of gamers, particularly in the achievement difficulty of genres compared to true score from players. Therefore, I used `RATIO` and `time_num` columns to inspect this problem.

```
# recall the data information
game_new.info()
```

```
plt.figure(figsize=(10,5))
sns.scatterplot(data = game_new, y="RATIO", x="time_num")
plt.title("The relationship between play time and Ratio (Difficulty)")
plt.xlabel("Playtime")
```

```
plt.ylabel("Ratio (Difficulty)")
plt.xscale('log')
plt.yscale('log')
plt.show()
```



Brief decription

In the code above, since values are distributed mostly around low play time and ratio, with a few of long play time and high ratio data. Thus, I transfered two variables to logarit to expand data in low range and narrow the data in high range.

Apply K-means clusterring for better classification

From the scatter plot, it can be seen that games may be divided into clusters with different features, instead of having a linear relationship. However, common observation is objective, which is consequently essential to have an unsupervised machine learning, k-means clustering. The final target is to determine whether the model can classify these clusters as innitally predicted or not.

```
# import model
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
```

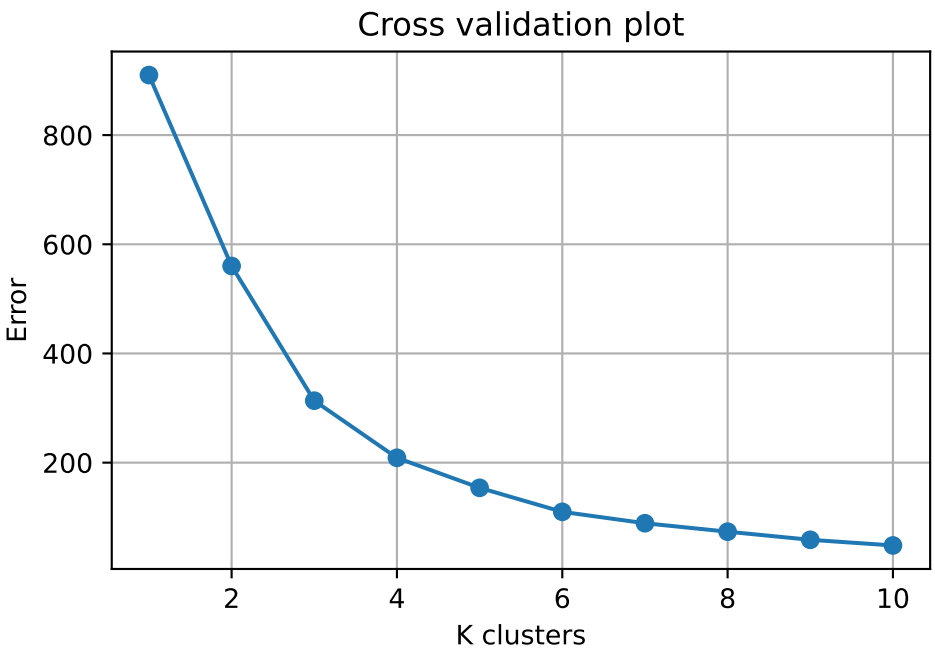
```
# extract and standardize variables for model
km_va = game_new[['RATIO','time_num']]

scaler = StandardScaler()
km_scaled = pd.DataFrame(
    scaler.fit_transform(km_va), # data scaled (numpy array)
    columns = km_va.columns
)
```

```
# Run k-means model with cross validation
inertia = []
K = range(1,11)

for k in K:
    k_mdl = KMeans(n_clusters=k, n_init=20, max_iter= 100, random_state= 42)
    k_mdl.fit(km_scaled)
    inertia.append(k_mdl.inertia_)

# Cross validation plot
plt.plot(K,inertia, marker="o")
plt.xlabel("K clusters")
plt.ylabel("Error")
plt.title("Cross validation plot")
plt.grid(True)
plt.show()
```



Optimal km mean cluster

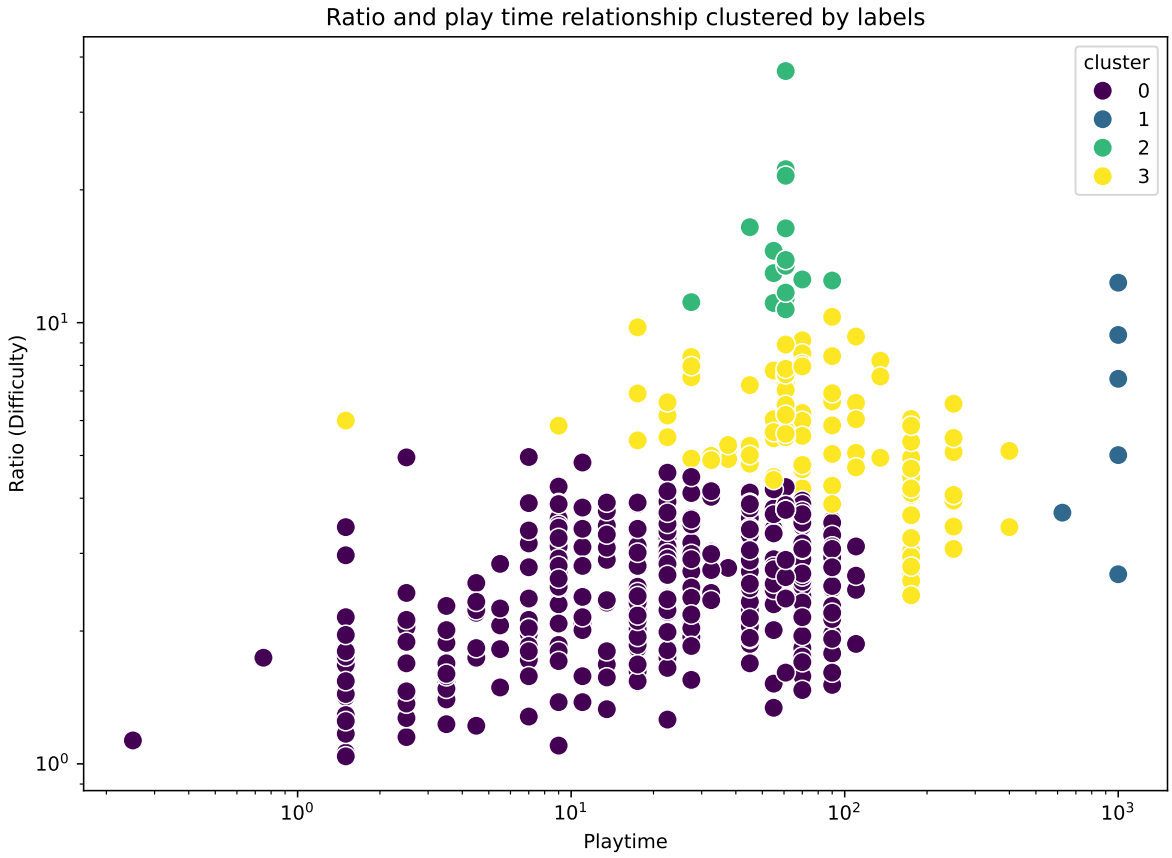
From the plot above, k cluster at point 4 shows the kink of error. Following the "elbow method", I choose 4 as an optimal k cluster for the model.

```
# run again the model
km_mdl = KMeans(n_clusters = 4, n_init=20, max_iter=100, random_state=42)
mdl_fit = km_mdl.fit(km_scaled)
```

```
# add a column containing labels of cluster
km_va['cluster'] = mdl_fit.labels_

# visualize scatter plot using logarit, and coloring data point by cluster label
plt.figure(figsize= (10,7))
sns.scatterplot(data = km_va,
                y= 'RATIO',
                x= 'time_num',
                hue='cluster',
                palette='viridis', s=100)
plt.xscale("log")
plt.yscale("log")
plt.xlabel("Playtime")
plt.ylabel("Ratio (Difficulty)")
```

```
plt.title("Ratio and play time relationship clustered by labels")
plt.show()
```




```
# create the mean of RATIO and time_num by clusters
summary_km = km_va.groupby('cluster').agg(RATIO = pd.NamedAgg(column="RATIO", aggfunc= "mean"
                                time_num = pd.NamedAgg(column="time_num",aggfunc="mean")).round
summary_km
```

cluster	RATIO	time_num
0	2.56	31.29
1	6.76	937.50
2	15.60	59.08
3	5.69	107.01

Table of difficulty (ratio) mean and time_num mean by clusters

Analysis

Based on the scatter plot colored by clusters and summary of RATIO and play time mean by clusters, the data points are divided into 4 clusters with clear features

- Cluster 0 (Easy - short): Casual games, conveniently entertaining for common players.
- Cluster 1 (Hard - very long): Triple A games, require players to invest in playing time for exploring sufficient games.
- Cluster 2 (Very Hard - short): Intelligence concentration, which is suitable for players who does not have much time.
- Cluster 3 (Hard - Long): Long-term games, which make gamers spend long time to play for understanding gameplays.

Extended problem

From the genre analysis and this result, I have raised a question that even though players in Xbox game pass prefer high-engagement and mass appeal gameplays, what genres do gamers prefer to play within these 4 features?

```
# join the cluster into the game_new df
joined_df = game_new.join(km_va['cluster'])

# add name for clusters
cluster_name = {
    0: 'Casual (easy-short)',
    1: 'Epic (hard-very long)',
    2: 'Challenge (very hard-short)',
    3: 'Mainstream (hard-long)'
}

# map the name with corresponding cluster in df
joined_df['Cluster name'] = joined_df['cluster'].map(cluster_name)
```

```
# create a summary table calculating count of games and sum of gamers by cluster and genre
filter_df = joined_df.groupby(['Cluster name','Main_Genre']).agg(
    current_genres = ('GAME','count'),
    current_player = ('GAMERS','sum')
    ).sort_values(by =['Cluster name','current_genres','current_player'],
                                ascending=[True,False,False])
filter_df
```

```
# sum of genre and player separately by each cluster
total_genre_cluster = filter_df.groupby('Cluster name')['current_genres'].transform('sum')
total_player_cluster = filter_df.groupby('Cluster name')['current_player'].transform('sum')
```

```
# calculate proportion of each genre in each cluster
filter_df['percent genre'] = (filter_df['current_genres']/total_genre_cluster)*100
filter_df['percent player'] = (filter_df['current_player']/total_player_cluster)*100
```

```
filter_df = filter_df.reset_index()
clus_name = filter_df['Cluster name'].unique()
clus_name
```

```
# define a function for plotting cluster data
def rank_plot(cluster):
    clus_df = filter_df[filter_df['Cluster name']== cluster].copy()

    # Prepare data for ploting

    # top 5 (percent genre)
    top5_per_genre = clus_df.nlargest(5, 'percent genre')
    # top 5 (percent player)
    top5_per_player = clus_df.nlargest(5,'percent player')
    # combine two top filtered into a single df
    combine_top5 = pd.concat([top5_per_genre,top5_per_player])
    # remove duplicates row from two top5 tables
    combine_top5 = combine_top5.drop_duplicates(subset= "Main_Genre").sort_values(by = "perce

    # plot bar plot

    labels = combine_top5['Main_Genre'].head(5)
    percent_genre_vals = combine_top5['percent genre'].nlargest(5)
    percent_player_vals = combine_top5['percent player'].nlargest(5)

    # set up axis and plot
    y = np.arange(len(labels))
    height = 0.35

    colors = ["#DD0303","#FA812F","#FAB12F","#dbc732ff","#d1d466ff"] # take color palette

    fig, ax = plt.subplots(figsize=(14,6), constrained_layout=True)
```

```
    for i, label in enumerate(labels):
    # create bars on axis
        bar1 = ax.barh(y[i] - height/2, percent_genre_vals.iloc[i], height, color =colors[i]
        bar2 = ax.barh(y[i] + height/2, percent_player_vals.iloc[i], height, color = colors[
        # Add labels
        ax.bar_label(bar1, padding=3, fmt='%.1f%%',fontsize=14)
        ax.bar_label(bar2, padding=3, fmt='%.1f%%',fontsize=14)

    # set custom legend
    genre_patch = mpatches.Patch(color='gray', alpha=0.4, label='Genre (%)')
    player_patch = mpatches.Patch(color='gray', alpha=1.0, label='Player (%)')
    ax.legend(handles=[genre_patch, player_patch])

    # custom plot design
    ax.set_title(f'{cluster}',fontsize=25, fontweight='bold')
    ax.set_xlabel('Proportion (%)',fontsize=18,fontweight='bold')
    ax.set_yticks(y)
    ax.set_yticklabels(labels, rotation=40, ha="right",fontsize=18,fontweight='bold')
    ax.invert_yaxis()
    ax.grid(True,color="#D3D3D3")
    ax.set_axisbelow(True)

    fig.tight_layout()
    plt.show()
```

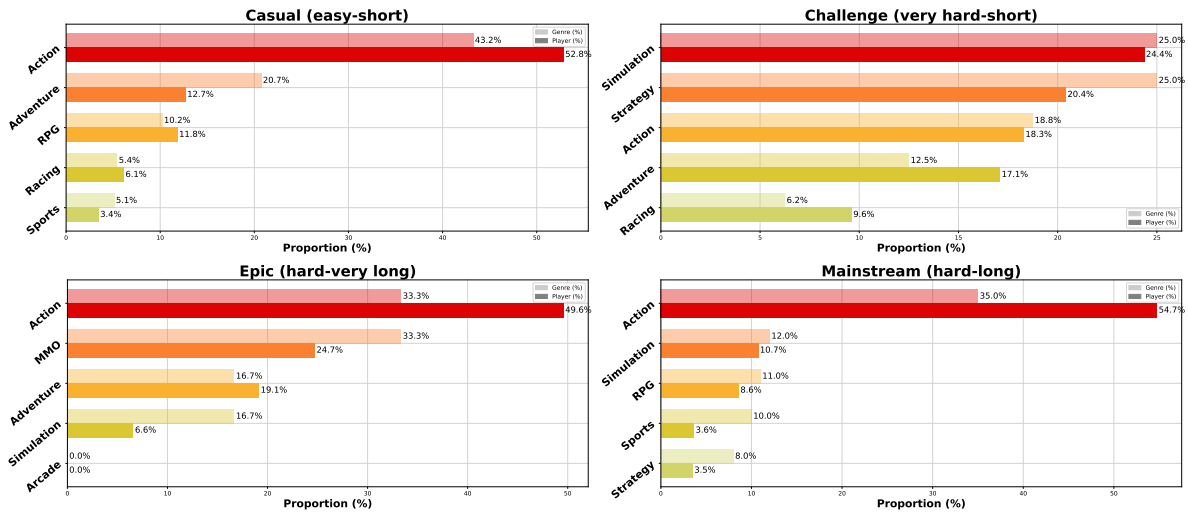
Comparison of games distribution on Xbox with the actual player demand across different gameplay

```
# create a plot of 4 clusters

rank_plot("Casual (easy-short)")
rank_plot("Challenge (very hard-short)")
rank_plot('Epic (hard-very long)')
rank_plot('Mainstream (hard-long)')
```

Insight

A deeper comparison of genre game share vs player share across gameplay groups surfaces both opportunities and inefficiencies for Xbox's portfolio. In short, we can see where genres face a



supply gap (player demand > game supply) and where resources may be over-allocated (game supply > player demand).

Overall, Action, Adventure, and Simulation appear across most groups, showing a strong developer focus—though the efficiency of that focus varies by group.

Group: Casual (Easy - Short playtime): Demand remains unmet

In this group, Action and Adventure clearly show a big gap between player demand and game distribution proportion.

- Action: Accounting for 52.8% of players but only 43.2% of games distributed in this genre, which leaves remarkable potential that is not leveraged efficiently.
- Adventure: A larger difference between game supply and real demand compared to Action. It attracts 20.7% of players but only has 12.7% of games. This is also an opportunity for game developers to introduce more Adventure Casual games.
- In contrast, other genres, such as RPG, Racing, and Sports, are distributed reasonably with similar game and player percentages.

Group: Challenge (Very Hard - Short playtime): Balanced, but some waste.

This group demonstrates a productive balance between the proportion of games launched and player preferences. The difference between genre (%) and player (%) is negligible, indicating that developers are doing well in meeting the tastes of players who enjoy challenges. However, there are still genres that are leveraged inefficiently.

- **Simulation and Action:** Have the best balance between games released and real player demand.
- **Strategy**: Resources for Strategy game development are redundant when the number of games introduced exceeds the real demand of users.
- **Adventure and Racing**: They attract 17.1% and 9.6% of players, respectively, but have a low proportion of games launched. The real potential of these genres is not employed productively, which is a chance to retain old players and attract new subscribers.

Group: Epic (Hard - Very long playtime): A potential opportunity is missing

In addition to a balance in Adventure, other genres are losing their development potential.

- **Action:** This genre dominates with nearly 50% of players, yet only accounts for 33.3% of games in this group. In other words, this is a "gold mine" that developers should focus more on, as Epic players show a high attraction to this genre.
- **MMO and Simulation:** MMO and Simulation appear to be overrepresented: their supply (33.3% and 16.7%) far exceeds actual player demand (24.7% and 6.6%). This suggests that resources may currently be allocated inefficiently.

Group: Mainstream (Hard - Long playtime): Optimizing portfolio is essential

Similar to the Epic group, Action in Mainstream has not yet reached its full potential, with 54.7% of players, yet 35% of games on the platform.

- **Sports and Strategy**: These genres show a sign of oversupply, as their game percentage is significantly higher than the share of players.
- **Simulation and RPG**: Maintain a good balance, indicating stability within this segment.

Conclusion

In conclusion, the main factors leveraging the success of gamer attraction on Xbox Game Pass are mass appeal (Sandbox) and high engagement (MMO and Compilation). Furthermore, deeper analysis using machine learning has also contributed significantly by identifying four main game types, ranging from Casual (quick entertainment) to Epic (hardcore, immersive experiences). Therefore, the key elements determining the success of a game do not solely depend on its genre, but also on the combination between genre and the experience it delivers to players (difficulty and playtime). Notably, completion ratio, as revealed through analysis, has shown that it is not a vital factor for popularity.

Recommendation

Based on the analysis, I propose a two-tiered strategic approach for developers to optimize their portfolio on Xbox Game Pass:

1. Strategic Focus Based on Market Goals:

- For Mass-Market Appeal: Developers should prioritize games that emphasize freedom, creativity, and accessibility. Sandbox games are a prime example of this successful model, capable of attracting a broad and diverse player base.
- For Building a Loyal Community: The focus should shift towards titles with profound narratives, long-term content roadmaps, and engaging hardcore gameplay. RPGs and MMOs are ideal for this purpose, fostering high player retention and strong community engagement.

2. Tactical Genre Allocation within Gameplay Clusters:

To maximize player satisfaction, developers must strategically align genre distribution with the distinct player expectations of each gameplay cluster (difficulty and playtime).

- Prioritize High-Demand Genres: In both the Epic (Hard - Very Long) and Mainstream (Hard - Long) clusters, there is a significant, untapped demand for Action games. I strongly recommend prioritizing investment in this genre, as it currently has the highest player engagement but remains critically under-supplied.
- Reallocate Resources from Saturated Genres: The development of additional MMO and Simulation games within the Epic cluster should be carefully reconsidered. Data indicates that these genres are over-supplied relative to player demand. These resources could be more effectively reallocated to the Action genre or used to explore other emerging, high-potential genres.