Board Game Recommender

Sia Zach Tjunchern 03/03/2023

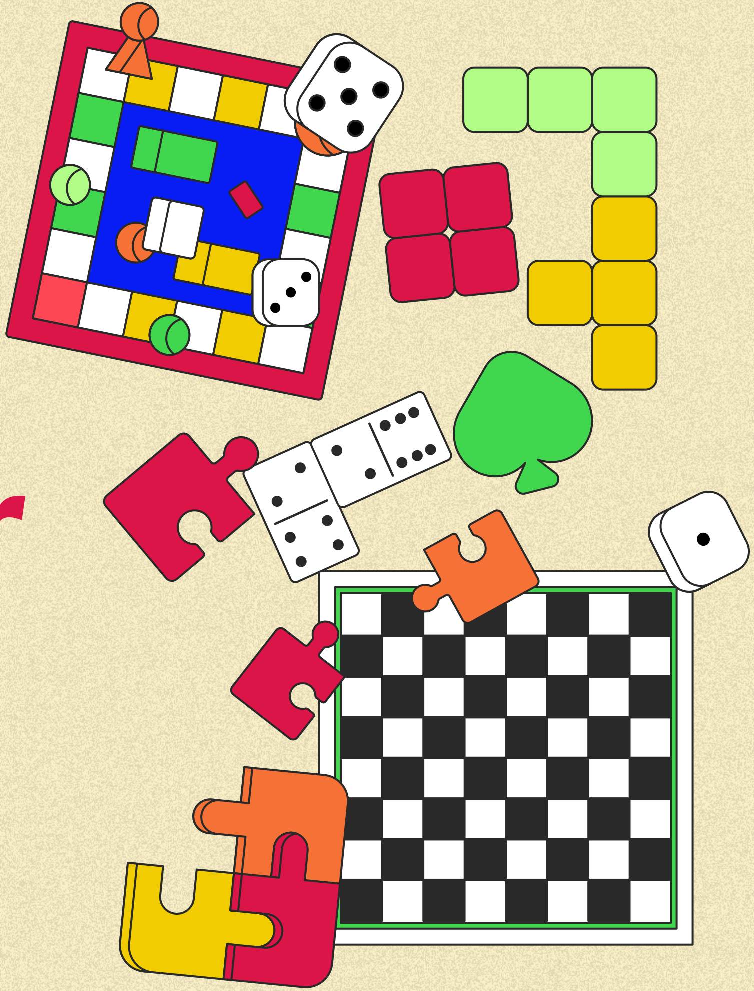
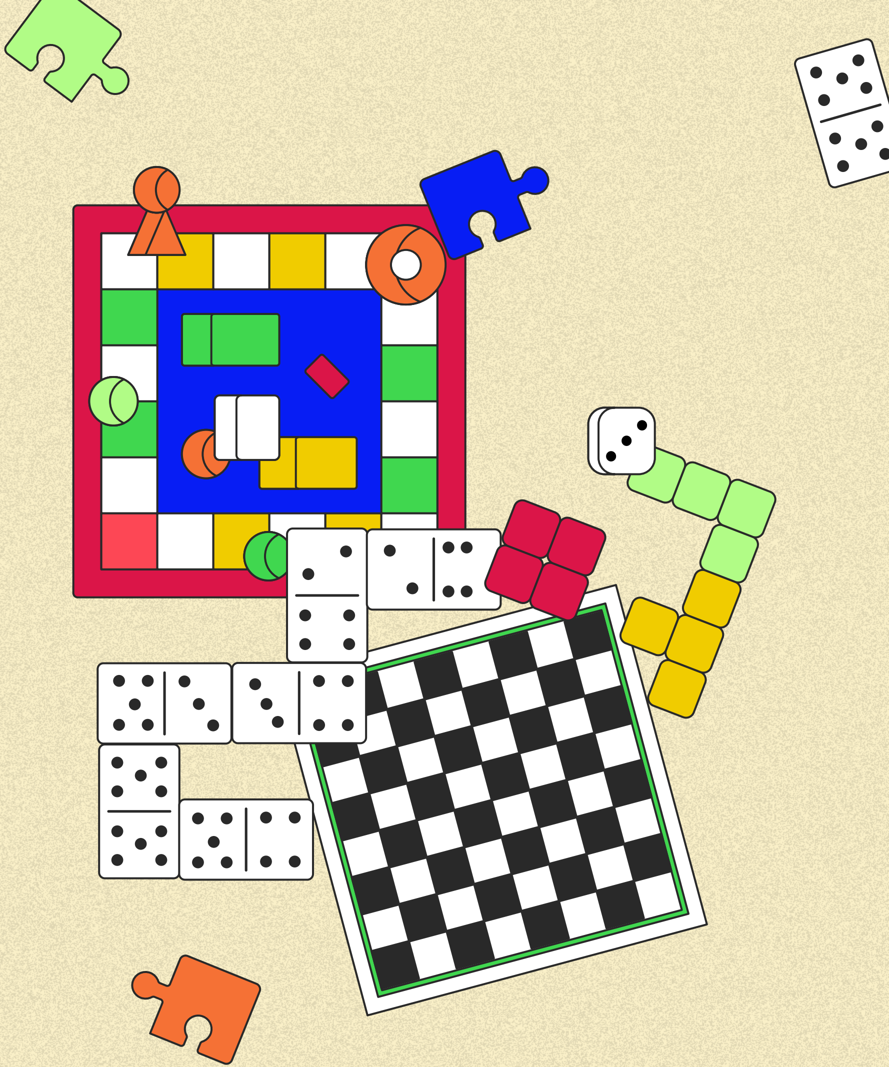


Table of contents

01

Board game recommender

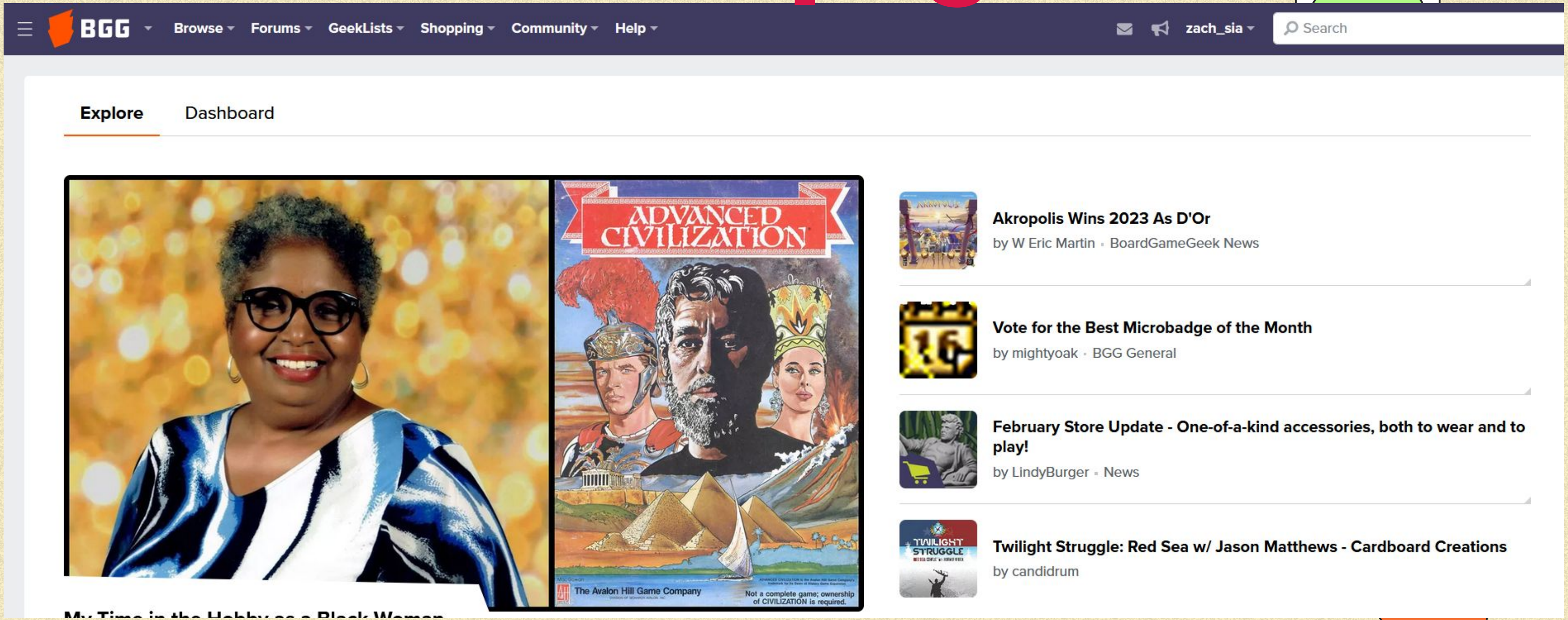
Who are **BoardGameGeek.com**?



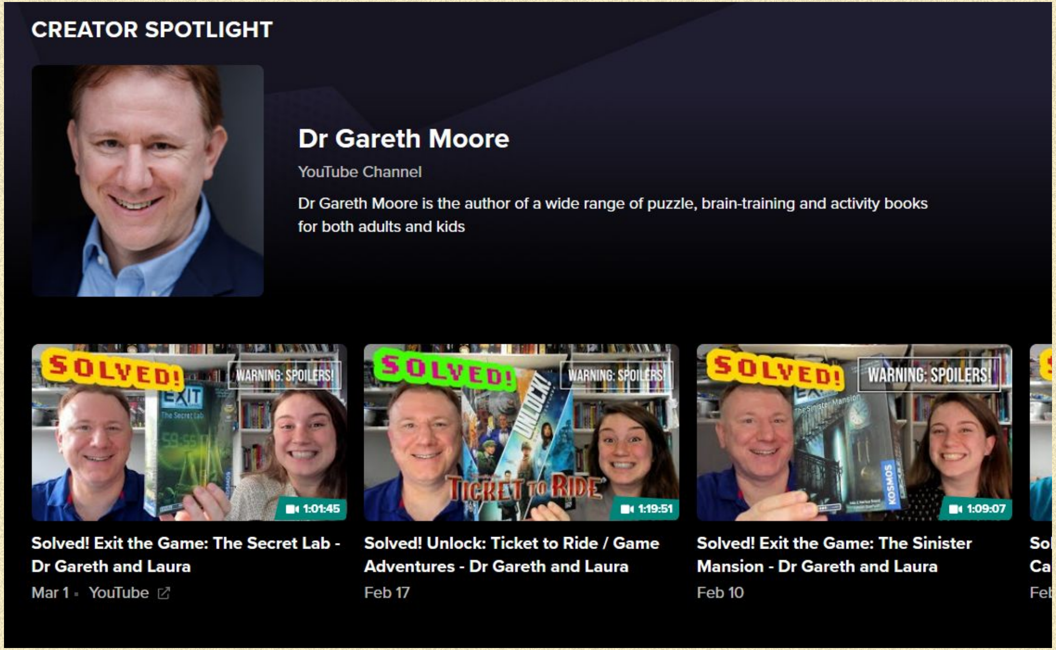
Problem Statement

Our project aims to generate game recommendations based on user reviews from BoardGameGeeks.com, focusing on modern games published from 2017 to 2021 to attract new market share and add value to existing users.

Homepage

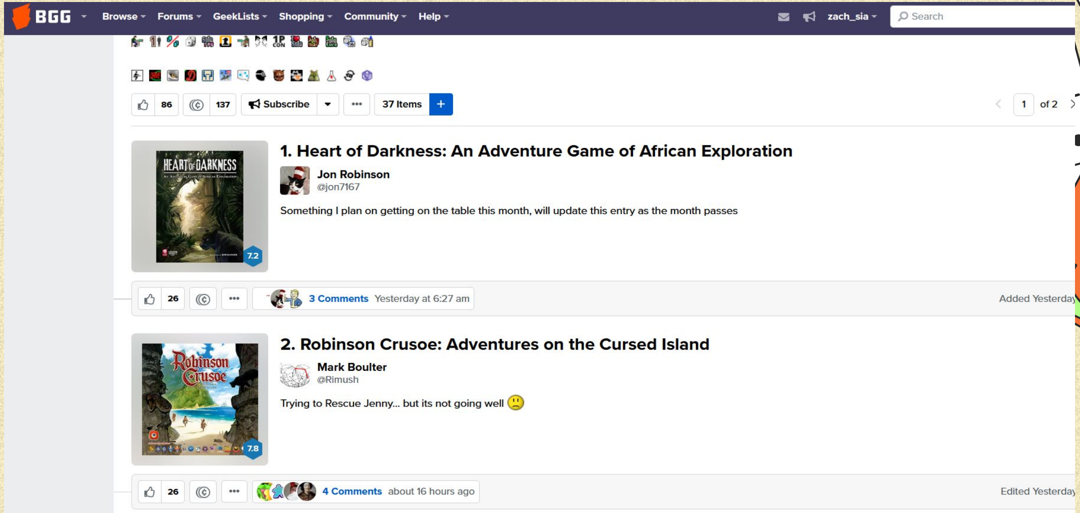

BGG
Browse
Forums
GeekLists
Shopping
Community
Help
zach_sia
Search
Explore
Dashboard
ADVANCED CIVILIZATION
The Avalon Hill Game Company
Not a complete game; ownership of CIVILIZATION is required.
Akropolis Wins 2023 As D'Or
by W Eric Martin · BoardGameGeek News
Vote for the Best Microbadge of the Month
by mightyoak · BGG General
February Store Update - One-of-a-kind accessories, both to wear and to play!
by LindyBurger · News
Twilight Struggle: Red Sea w/ Jason Matthews - Cardboard Creations
by candidrum

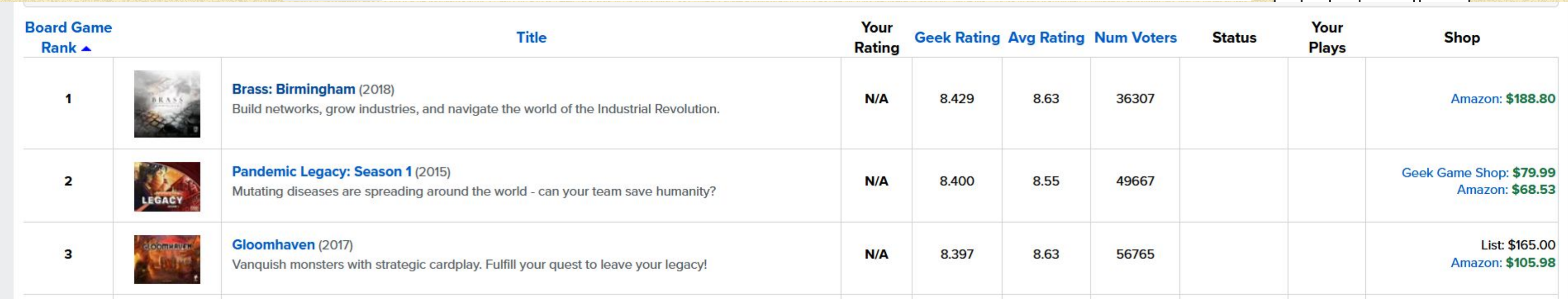
Homepage


CREATOR SPOTLIGHT
Dr Gareth Moore
YouTube Channel
Dr Gareth Moore is the author of a wide range of puzzle, brain-training and activity books for both adults and kids
SOLVED!
WARNING: SPOILERS!
1:01:45
Solved! Exit the Game: The Secret Lab - Dr Gareth and Laura
Mar 1 · YouTube
SOLVED!
WARNING: SPOILERS!
TICKET TO RIDE
1:19:51
Solved! Unlock: Ticket to Ride / Game Adventures - Dr Gareth and Laura
Feb 17
SOLVED!
WARNING: SPOILERS!
1:09:07
Solved! Exit the Game: The Sinister Mansion - Dr Gareth and Laura
Feb 10


GAME SPOTLIGHT – SPONSORED
GENSMAK!
The first pop-culture trivia game that levels the pla
8.2
PHOTOS
VIDEOS
FORUMS
FANS ALSO LIKE

Forum





Board Game Rank		Title	Your Rating	Geek Rating	Avg Rating	Num Voters	Status	Your Plays	Shop
1		Brass: Birmingham (2018) Build networks, grow industries, and navigate the world of the Industrial Revolution.	N/A	8.429	8.63	36307			Amazon: $188.80
2		Pandemic Legacy: Season 1 (2015) Mutating diseases are spreading around the world - can your team save humanity?	N/A	8.400	8.55	49667			Geek Game Shop: $79.99 Amazon: $68.53
3		Gloomhaven (2017) Vanquish monsters with strategic cardplay. Fulfill your quest to leave your legacy!	N/A	8.397	8.63	56765			List: $165.00 Amazon: $105.98

Recap: Who are BoardGameGeeks?

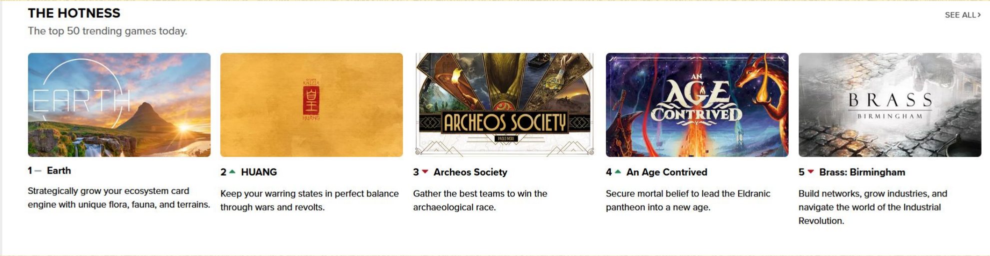
Focus on community

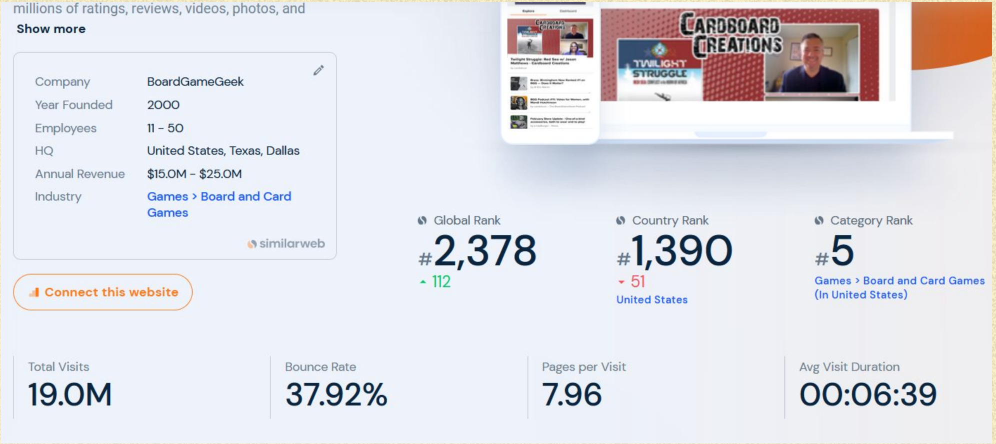
- Community focused
  - Forum discussions
  - Request trades
- Market place
  - Buy
  - Sell

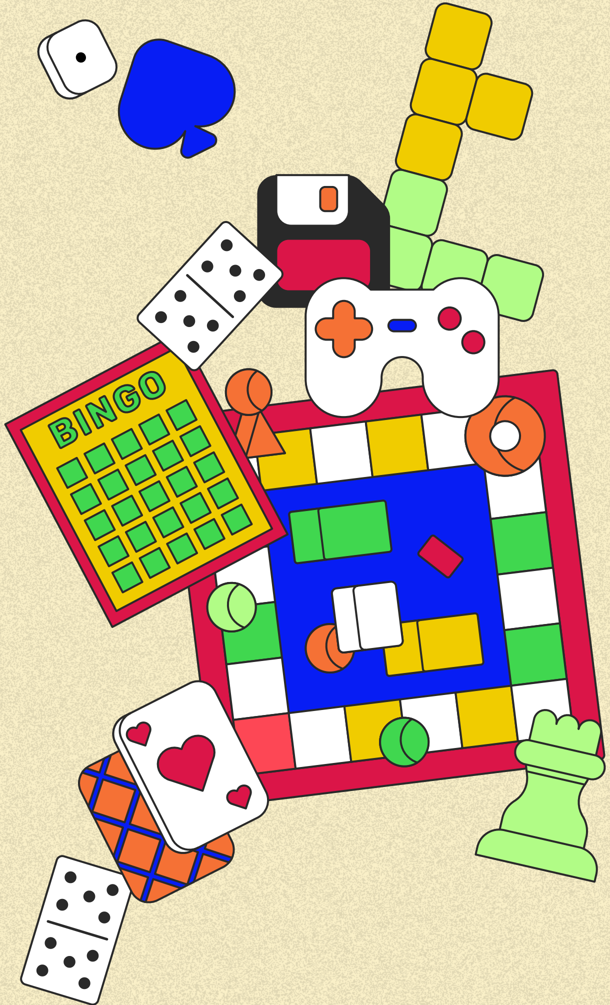
Focus on optimising user engagement

- Multimedia engagement
  - Podcasts
  - Video reviews
  - Blog posts
  - Weekly highlights

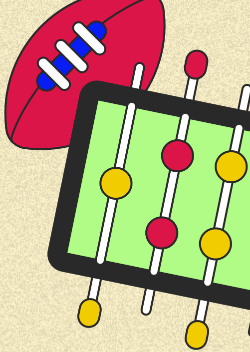
Recommender system natural next step?

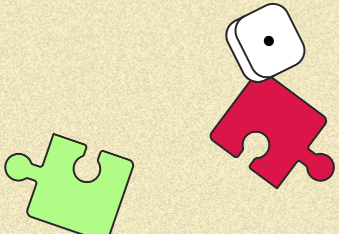




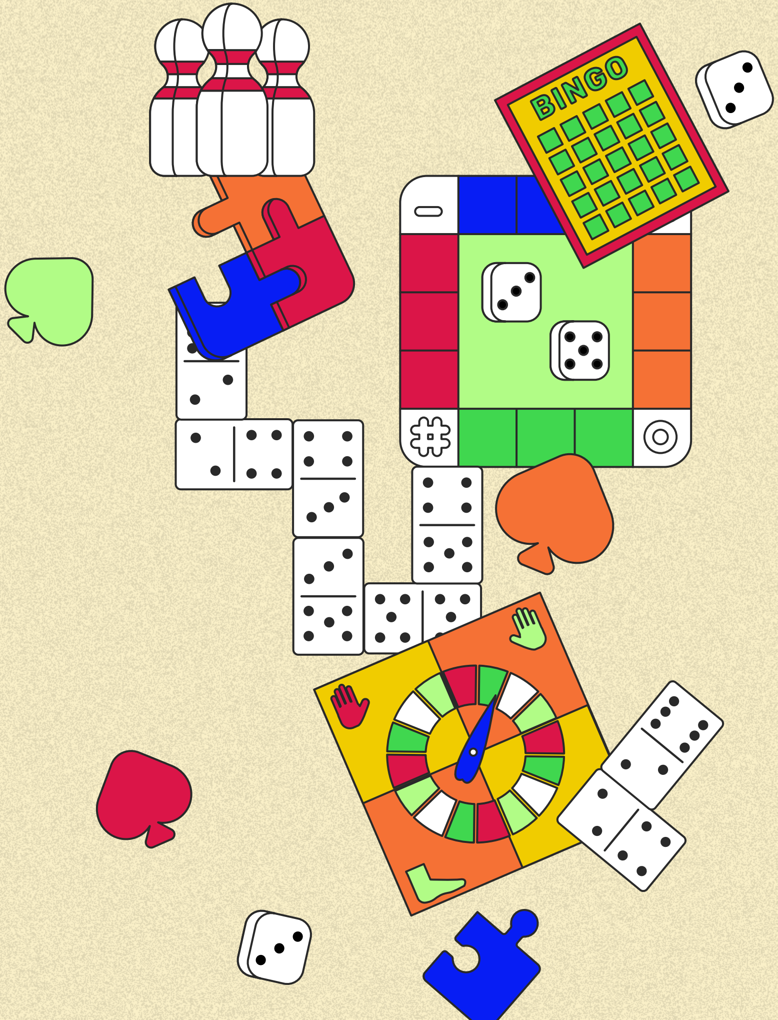


Diving into the data





Our new best friend polars



Games

Description and shape of data

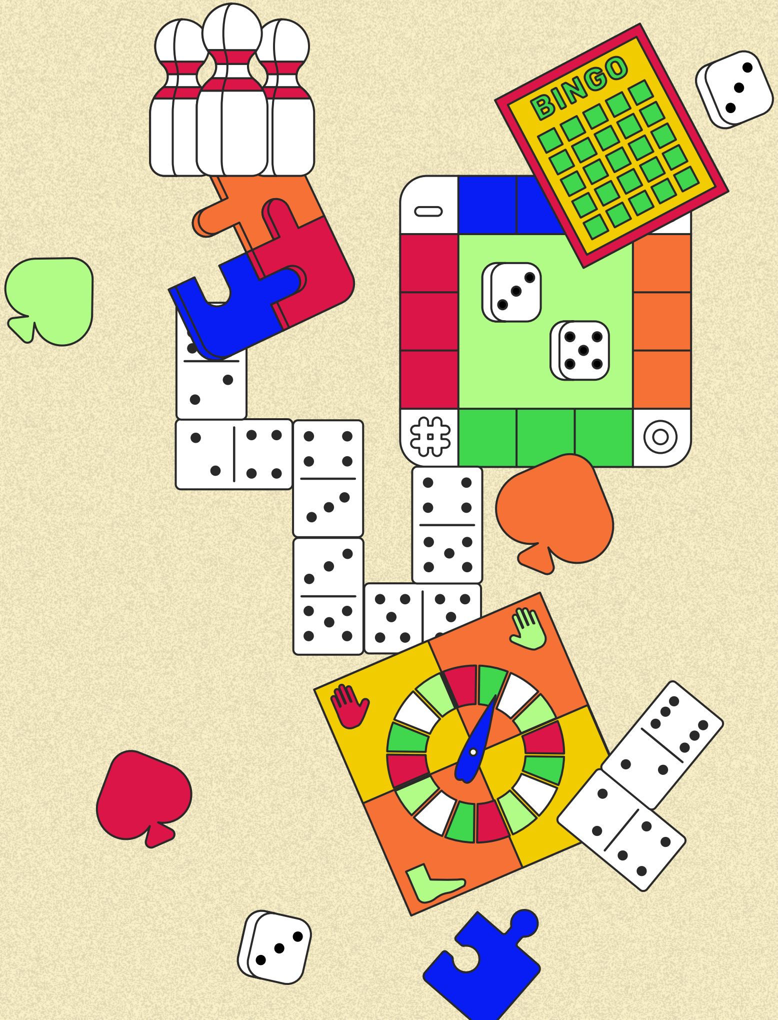
```
dataframe games.csv, shape
(21925, 48)

dataframe games.csv, describe
```

	BGGId	YearPublished	GameWeight	AvgRating	BayesAvgRating	StdDev	MinPlayers	MaxPlayers	ComAgeRec	LanguageEase	...	Rank:partygames	Rank:chi
count	21925.000000	21925.000000	21925.000000	21925.000000	21925.000000	21925.000000	21925.000000	21925.000000	16395.000000	16034.000000	...	21925.000000	
mean	117652.663216	1985.494914	1.982131	6.424922	5.685673	1.516374	2.007343	5.707868	10.004391	216.461819	...	21295.352201	
std	104628.721777	212.486214	0.848983	0.932477	0.365311	0.285578	0.693093	15.014643	3.269157	236.595136	...	3637.139987	
min	1.000000	-3500.000000	0.000000	1.041330	3.574810	0.196023	0.000000	0.000000	2.000000	1.000000	...	1.000000	
25%	12346.000000	2001.000000	1.333300	5.836960	5.510300	1.320720	2.000000	4.000000	8.000000	24.027778	...	21926.000000	
50%	105305.000000	2011.000000	1.968800	6.453950	5.546540	1.476880	2.000000	4.000000	10.000000	138.000000	...	21926.000000	
75%	206169.000000	2017.000000	2.525200	7.052450	5.679890	1.665470	2.000000	6.000000	12.000000	351.000000	...	21926.000000	
max	349161.000000	2021.000000	5.000000	9.914290	8.514880	4.277280	10.000000	999.000000	21.000000	1757.000000	...	21926.000000	

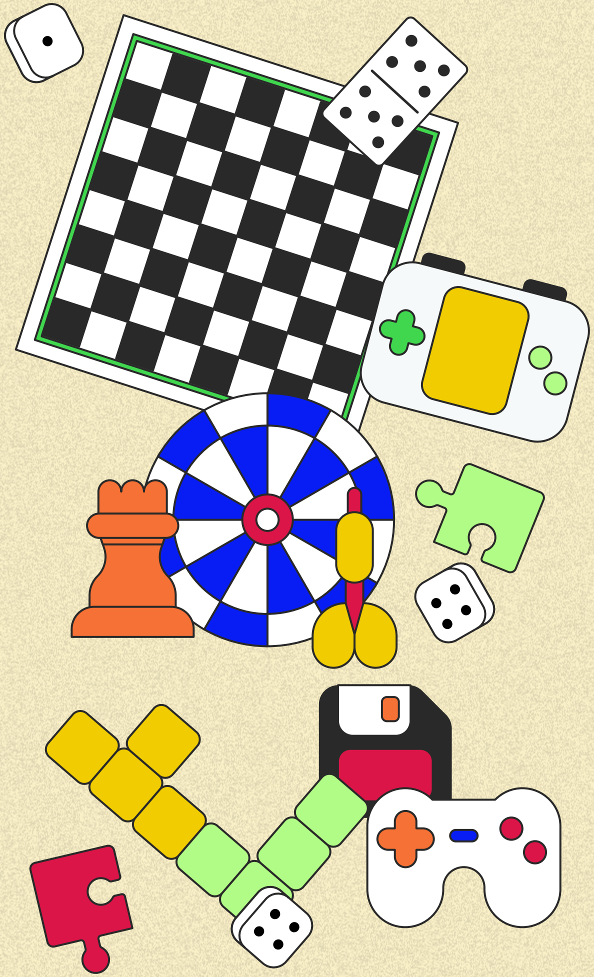
Publish Year and missing data

```
count     21925.000000
mean       1985.494914
std         212.486214
min       -3500.000000
25%        2001.000000
50%        2011.000000
75%        2017.000000
max        2021.000000
Name: YearPublished, dtype: float64
```

```
games.csv
BGGId                      0
Name                       0
Description                1
YearPublished              0
GameWeight                 0
AvgRating                  0
BayesAvgRating             0
StdDev                     0
MinPlayers                 0
MaxPlayers                 0
ComAgeRec               5530
LanguageEase            5891
BestPlayers                0
GoodPlayers                0
NumOwned                   0
NumWant                    0
NumWish                    0
NumWeightVotes             0
MfgPlaytime                0
ComMinPlaytime             0
ComMaxPlaytime             0
MfgAgeRec                  0
NumUserRatings             0
NumComments                0
NumAlternates              0
NumExpansions              0
NumImplementations         0
IsReimplementation         0
Family                 15262
Kickstarted                0
ImagePath                 17
Rank:boardgame             0
Rank:strategygames         0
Rank:abstracts             0
Rank:familygames           0
```



User ratings



User ratings

18,942,215

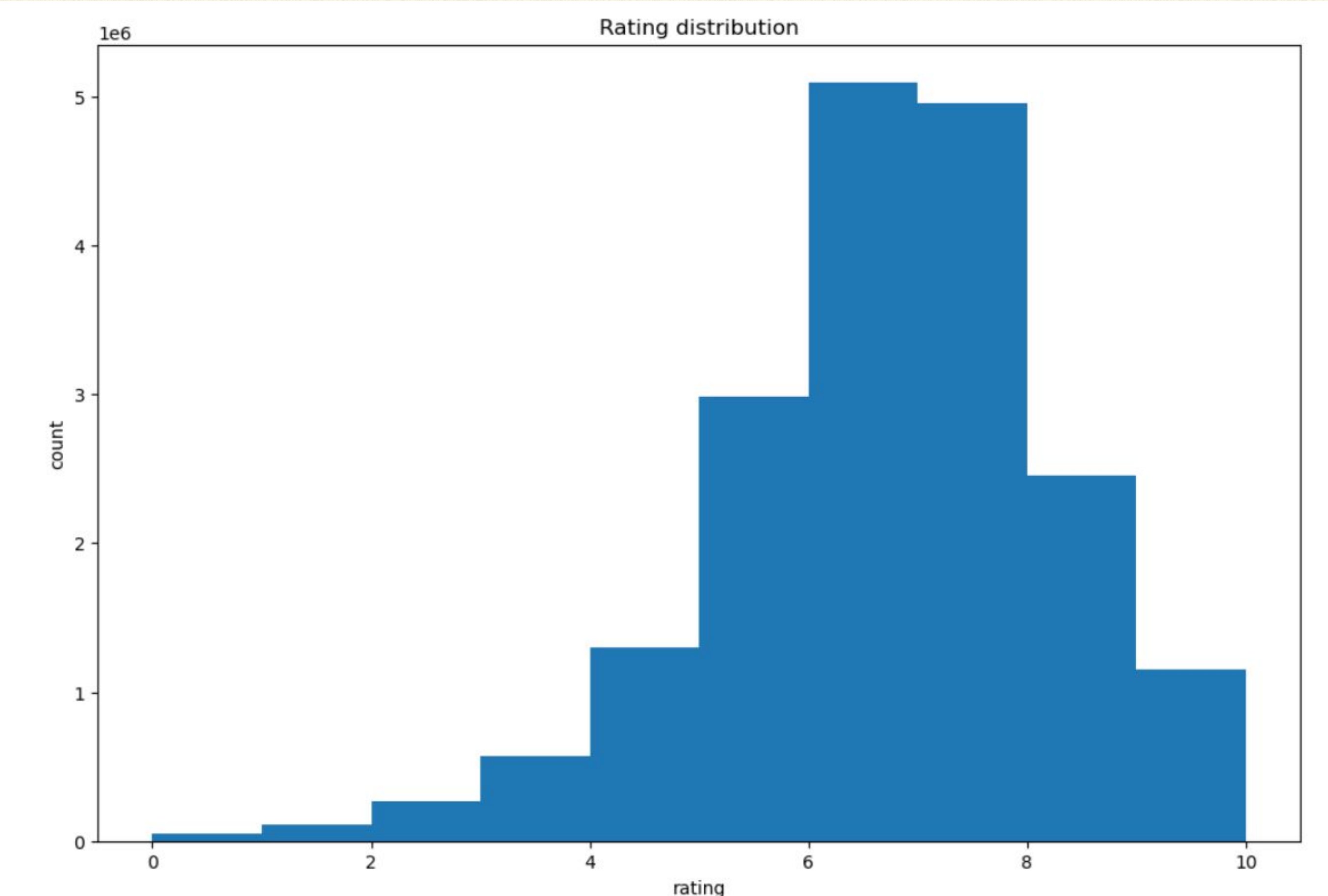
Rows of data across
411,374 unique users
21,925 unique games

Missing data, mean ratings and rating counts

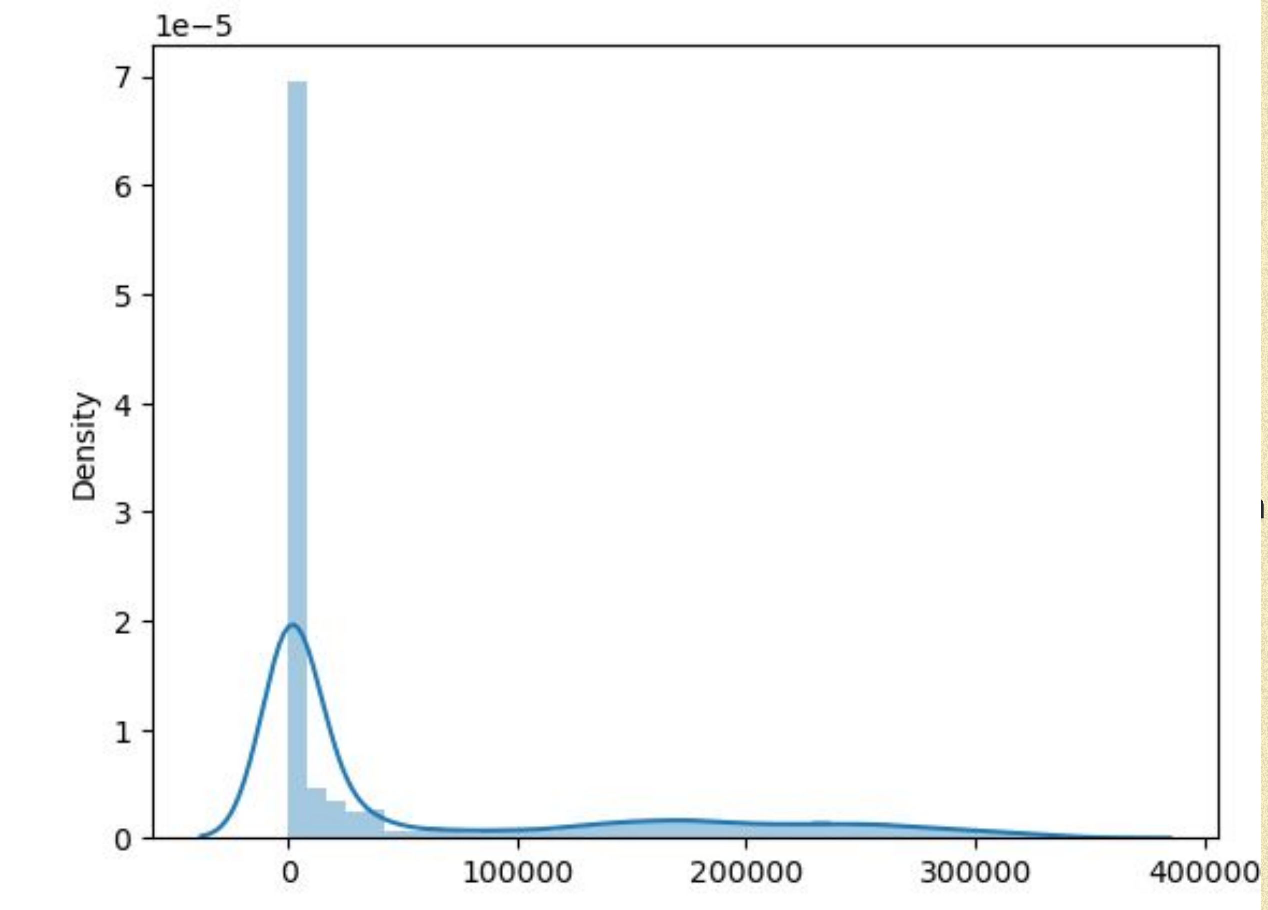
```
user_ratings.csv
BGGId        0
Rating       0
Username    63
dtype: int64
```

	Rating
count	21925.000000
mean	863.955074
std	3627.083866
min	7.000000
25%	57.000000
50%	125.000000
75%	398.000000
max	107760.000000

Distribution of Ratings



IQR of the count of reviews



Filtering our data to our problem statement

Images reveal large amounts of data, so remember: use an image instead of a long text. **Your audience will appreciate it**

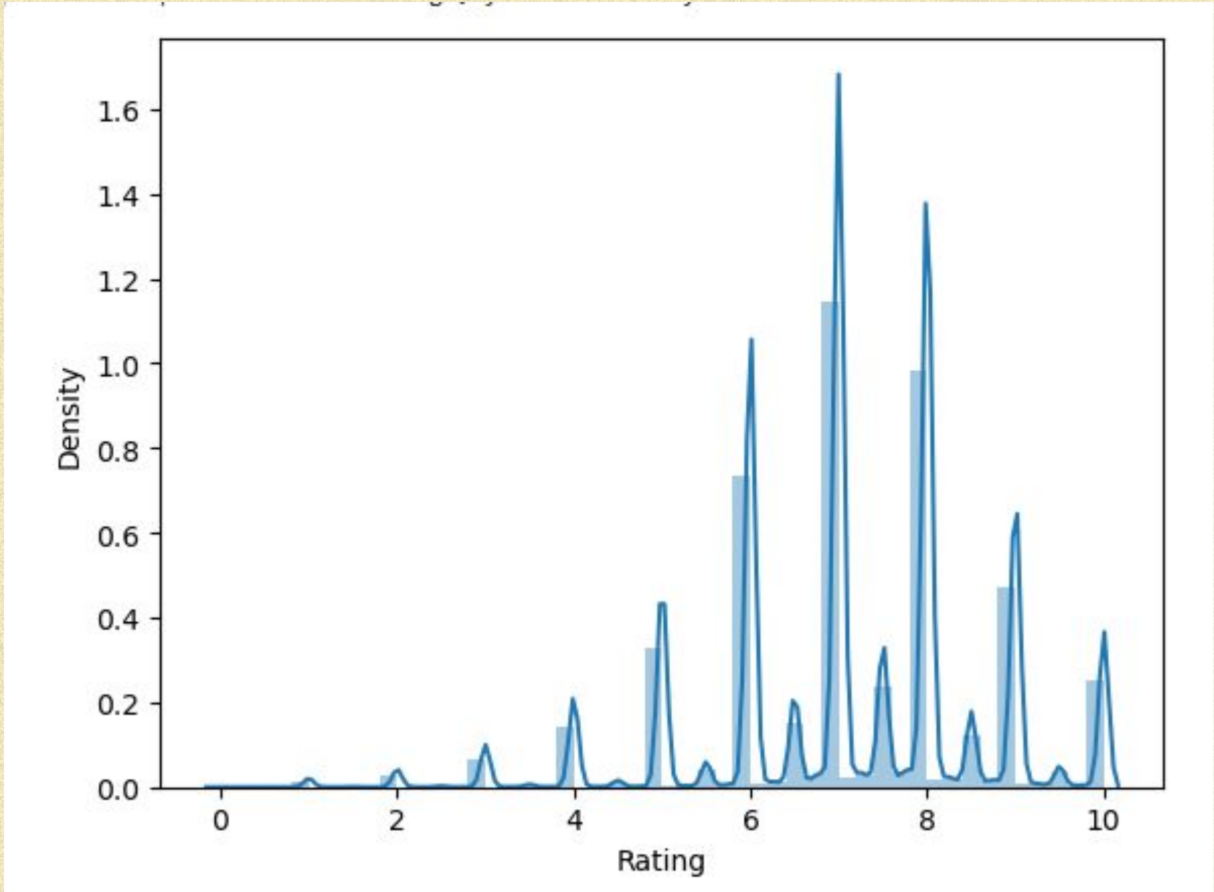
Stratifying our dataset

```
strata = games_csv2.groupby('YearPublished')
# Sample 10% of the data from each stratum
sampled_data = pl.concat([stratum.sample(frac=0.1, seed=42) for _, stratum in strata])
filter_list = sampled_data['BGGId'].to_list()
```

```
df_filter_2 = user_ratings.filter(pl.col('BGGId').is_in(filter_list))
```

```
df_filter_2
```

Distribution of new dataset



Filtering to after 2017 onwards

shape: (3720985, 4)

	BGGId	Rating	Username
i64	i32	f32	str
75	193500	5.0	"Narfbuster"
76	193500	5.0	"Methrin"
77	193500	5.0	"Evabelle"
78	193500	5.0	"ngcx6611"
79	193500	5.0	"bmillerbwm"
80	193500	5.0	"CadizEstocolmo…
81	193500	5.0	"kelvbrown"
82	193500	5.0	"jenf"
83	193500	5.0	"thatthing1999"
84	193500	5.0	"alanB"
85	193500	5.0	"RyanThibault"
86	193500	5.0	"psychomansam"
…	…	…	…
18941829	193422	5.0	"rdunlap1125"
18941830	193422	5.0	"ryansmum2008"
18941831	193422	5.0	"theericbooth"
18941832	193422	5.0	"mljeko"

Limiting to reviews and users to 100

```
:                  1008727
BGGId                 3208
Rating                 659
Username              6150
dtype: int64
```

Strategy for finding best model

Methodology

Evaluate based on metrics

Final model

kfold=5 cross validation

K Nearest Neighbours

KNNBasic, KNNBaseline, KNNZScore, KNNwithMeans.

Baseline Models

Normal_predictor,Baseline.

Matrix Factorization

Slope One,
Co-clustering,NonNegative Matrix Factorization

https://surprise.readthedocs.io/en/stable/prediction_algorithms_package.html

Key discussion points

RMSE

We will be using **RMSE** as our **loss function**

Precision@k

Precision@k as main performance metric that measures the proportion of relevant items among the top k recommended items to a user.

Recall@k

Recall@k as secondary performance metric, proportion of relevant items among all the items that should have been recommended to a user, up to the top k items.

RMSE

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}$$

RMSE

We will be using **RMSE** as our **loss function**

- RMSE is a widely recognized and accepted evaluation metric and is commonly used in machine learning and recommender systems to measure the difference between predicted and actual ratings.
- Punishes larger discrepancies between predictions and true values.

Precision@k

Precision@k

Precision@k as main performance metric that measures the proportion of relevant items among the top k recommended items to a user.

$$\text{Precision@k} = \frac{|\{\text{Recommended items that are relevant}\}|}{|\{\text{Recommended items}\}|}$$

$$\text{Recall@k} = \frac{|\{\text{Recommended items that are relevant}\}|}{|\{\text{Relevant items}\}|}$$

We must decide a k value aka the **number of recommendations** as our top k value and our threshold for relevant items

Deciding threshold and k values

Our dataset to begin with is already skewed more toward the higher end with a 50th percentile of 7.0.
To have more confidence that our model is generalisable via looking at our precision and recall@k, we we set it to 7.5 a slightly higher threshold.

```
]: np.percentile(data_for_model['Rating'],50)

]: 7.0

]: np.percentile(data_for_model['Rating'],60)

]: 7.5
```

Recall@k

Recall@k

Recall@k as secondary performance metric, proportion of relevant items among all the items that should have been recommended to a user, up to the top k items.

$$\text{Precision@k} = \frac{|\{\text{Recommended items that are relevant}\}|}{|\{\text{Recommended items}\}|}$$

$$\text{Recall@k} = \frac{|\{\text{Recommended items that are relevant}\}|}{|\{\text{Relevant items}\}|}$$

We must decide a k value aka the **number of recommendations** as our top k value and our threshold for relevant items

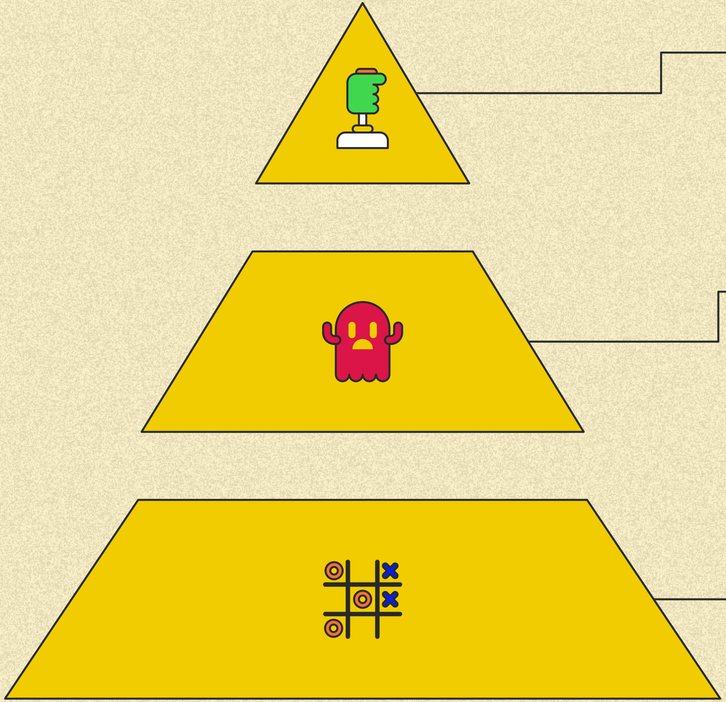
Final results

	precision_at_k	recall_at_k	average_rmse
KNNBasic	0.698465	0.428652	1.071927
SVD	0.698177	0.390089	1.020469
KNNBaseline	0.688521	0.356677	1.016195
Baseline	0.686130	0.367994	1.020571
KNNWithZScore	0.683345	0.367066	1.032019
Slope One	0.679875	0.360517	1.017509
Co-clustering	0.665922	0.333746	1.045657
KNNWithMeans	0.661810	0.334689	1.032379
Normal_predictor	0.465807	0.314209	1.853386
NonNegative Matrix Factorization	0.087494	0.009557	1.784536

Our best model

	precision_at_k	recall_at_k	average_rmse
KNNBasic	0.698465	0.428652	1.071927
SVD	0.698177	0.390089	1.020469
KNNBaseline	0.688521	0.356677	1.016195
Baseline	0.686130	0.367994	1.020571
KNNWithZScore	0.683345	0.367066	1.032019
Slope One	0.679875	0.360517	1.017509
Co-clustering	0.665922	0.333746	1.045657
KNNWithMeans	0.661810	0.334689	1.032379
Normal_predictor	0.465807	0.314209	1.853386
NonNegative Matrix Factorization	0.087494	0.009557	1.784536

Analysis & development



Phase EDA and filtering

We took a look at our data and made the decision to filter it based on our business problem and computational limitations.

Finding our best model

Using key metrics we exhausted all possible models and decided to go with **KNNBasic.**

Deployment

We then deployed our model into streamlit for demonstration.

Analysis & development

Phase EDA and filtering

We took a look at our data and made the decision to filter it based on our business problem and computational limitations.

Finding our best model

Using key metrics we exhausted all possible models and decided to go with **KNNBasic.**

Deployment

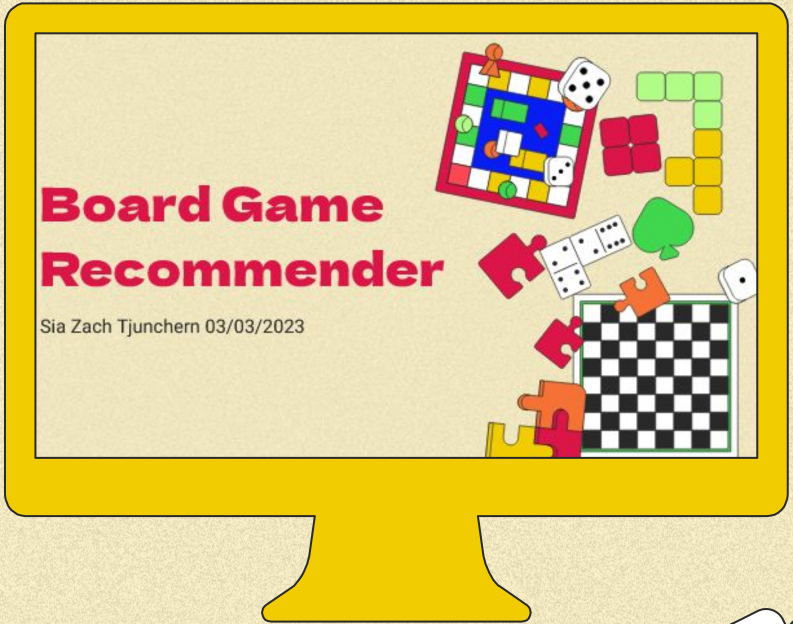
We then deployed our model into streamlit for demonstration.

Take a look

https://siazachtj-capstone-codestreamlit-5es7vw.streamlit.app/

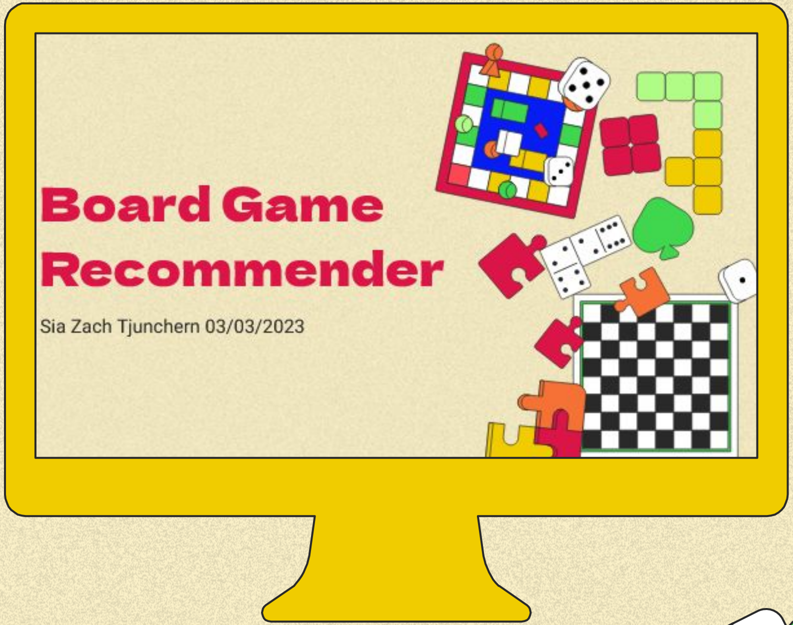
Conclusions

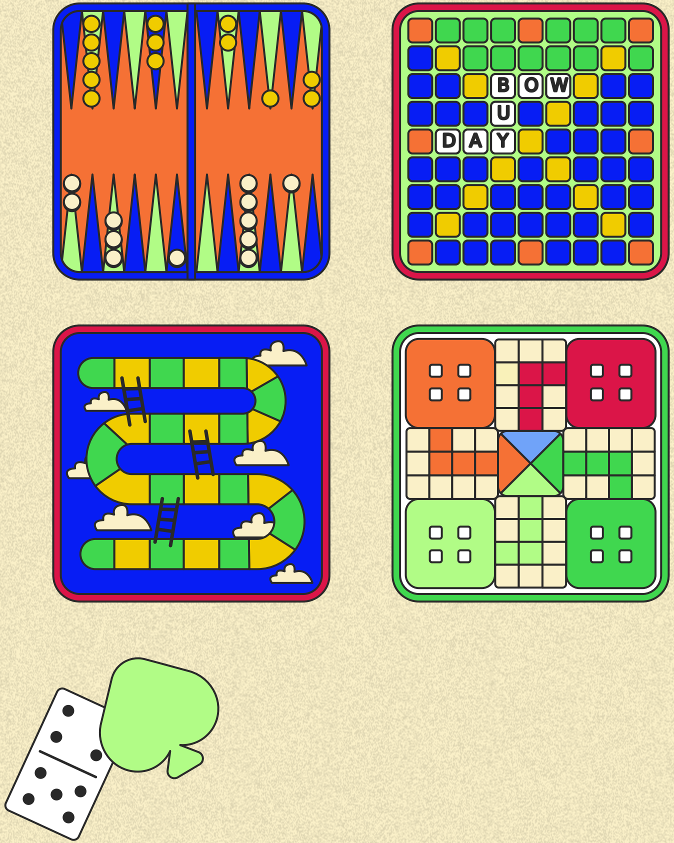
Our collaborative filtering model is aligned with the company’s goals of being community drive, give the vast amount of clearly active users, our model's ability to provide insightful and relevant recommendations will only increase.



Limitations and further work

Our model only scratched the surface of the potential of this dataset with its vast number of reviews and games. There are other tools and packages more in the realm of deep learning that might benefit model performance as a long term strategy.





Thanks for listening!

https://github.com/siazachtj

https://www.linkedin.com/in/zach-sia