Predicting average ratings for board games

A case study in futility

The name of the game

- BoardGameGeek.com

- Created in 2000

- Over 80,000 board games
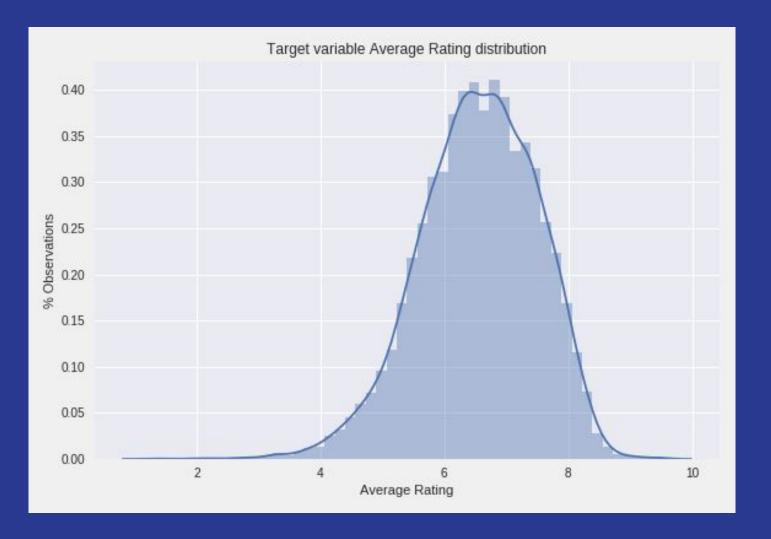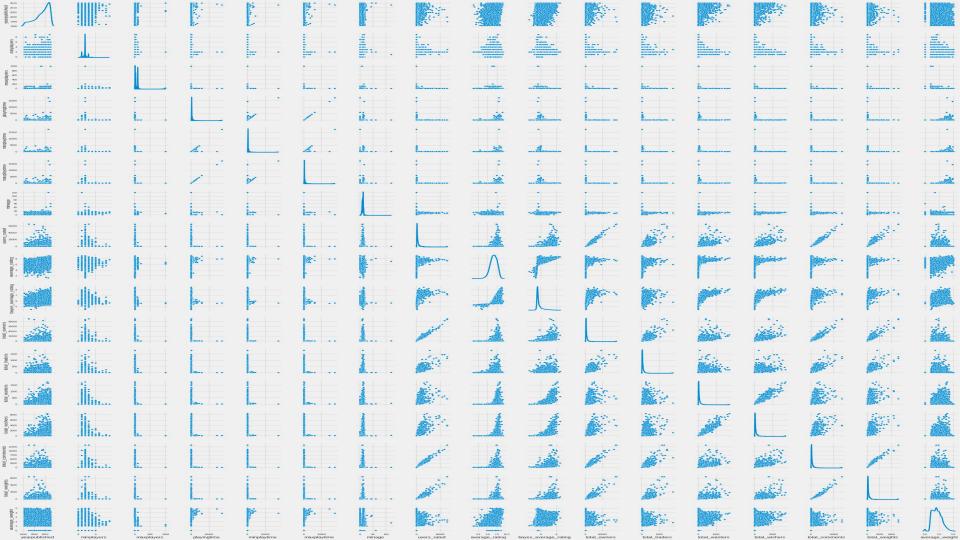
- Strong member community

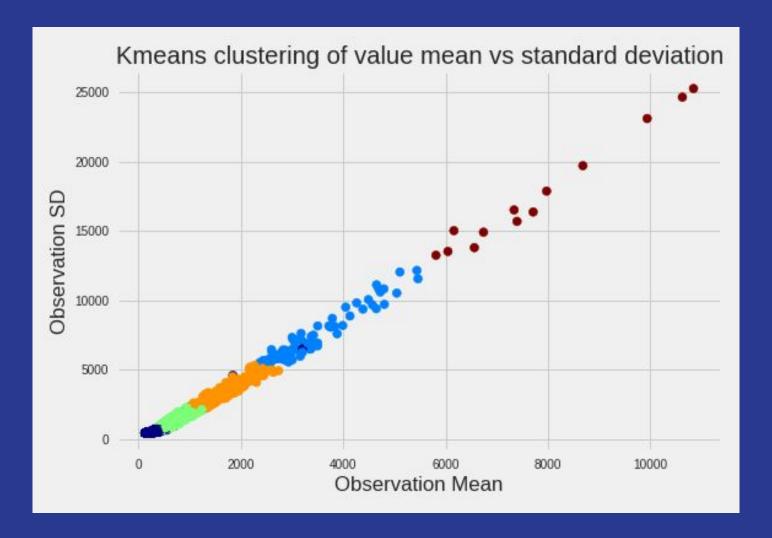# 1. How to play

I used Scrapy and Beautiful Soup to pull down my data into csv form before pulling it into pandas. .

➔ **Cleaning**
"Empty" games

➔ **Simplify**
Non-numerical features

➔ **Explore...**

Target variable Average Rating distribution

Kmeans clustering of value mean vs standard deviation

2. Correlations

```
yearpublished            0.348639
minplayers              -0.169357
maxplayers              -0.024367
playingtime              0.118681
minplaytime              0.114625
maxplaytime              0.118681
minage                   0.141881
users_rated              0.153301
average_rating           1.000000
bayes_average_rating     0.516445
total_owners             0.206832
total_traders            0.081275
total_wanters            0.318231
total_wishers            0.257790
total_comments           0.152957
total_weights            0.133433
average_weight           0.510265
Name: average_rating, dtype: float64
```
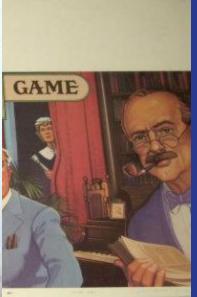
Which features actually matter? *Not many.*

Tip

Put the robber on the tile of someone whom you love to watch suffer!

3. Regression

I removed all collinear features before modeling my training data.

➜ **Cross-valididated L1**
100 sample α

➜ **Cross-validated L2**

➜ **Optimized CV L1**
500 sample α

➜ **Lasso**
Trained on the entire sample using the optimized α

HOW NEAT IS THAT?

THAT'S PRETTY NEAT

3. Scoring

➜ **MSE**
0.459

➜ **RMSE**
0.679

➜ **Target σ**
0.953

Fixed model complexity

Cross-validation error

Training error

Mean Error

Data size

Ratings by publishing year compared to model

Ratings by publishing year vs model predictions

Win or lose, it's how you play the game

- Garbage in, garbage out

- Target and features are redundant

- New features that describe the game

- Detecting trends using clustering