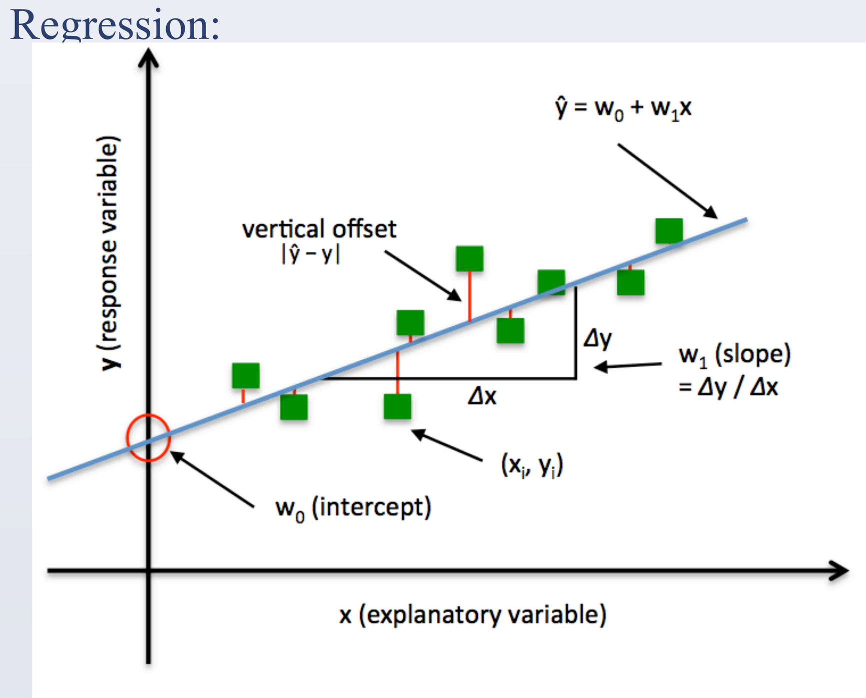
Linear Regression vs Random Forest Regression in Predictions of Board Game Reviews

Daniel Lecona

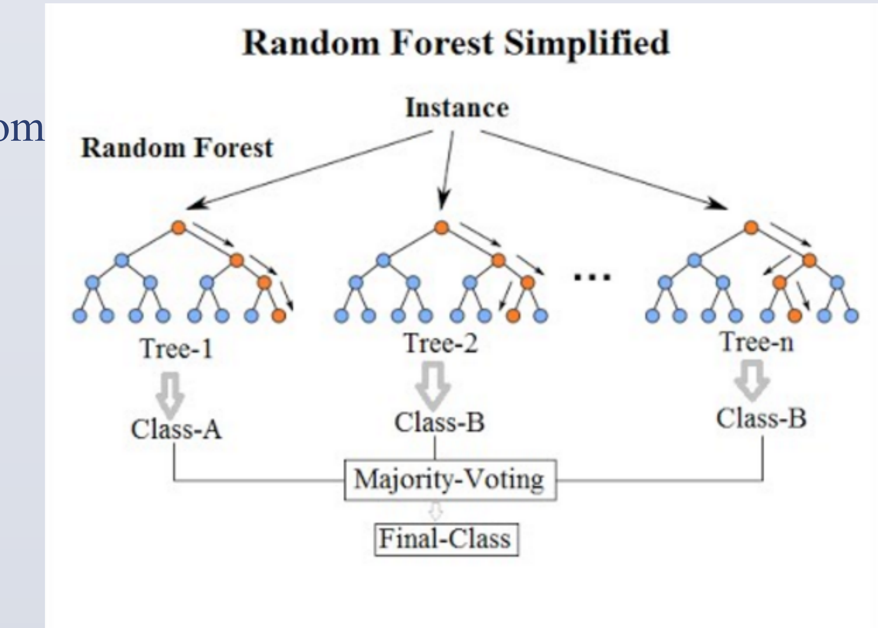
Abstract

Linear regression is used for Statistical Data Analysis, which was done within these methods. However, Random Forest Regression was also used and gave interesting results when implemented. The two models are very different, because linear regression is a simple linear prediction between x & y, whereas Random Forest produces a significant amount of decision trees to decide on a prediction. Using these methods, I was able to determine the best regression type for predicting review ratings of board games.

Linear Regression:



Random



References

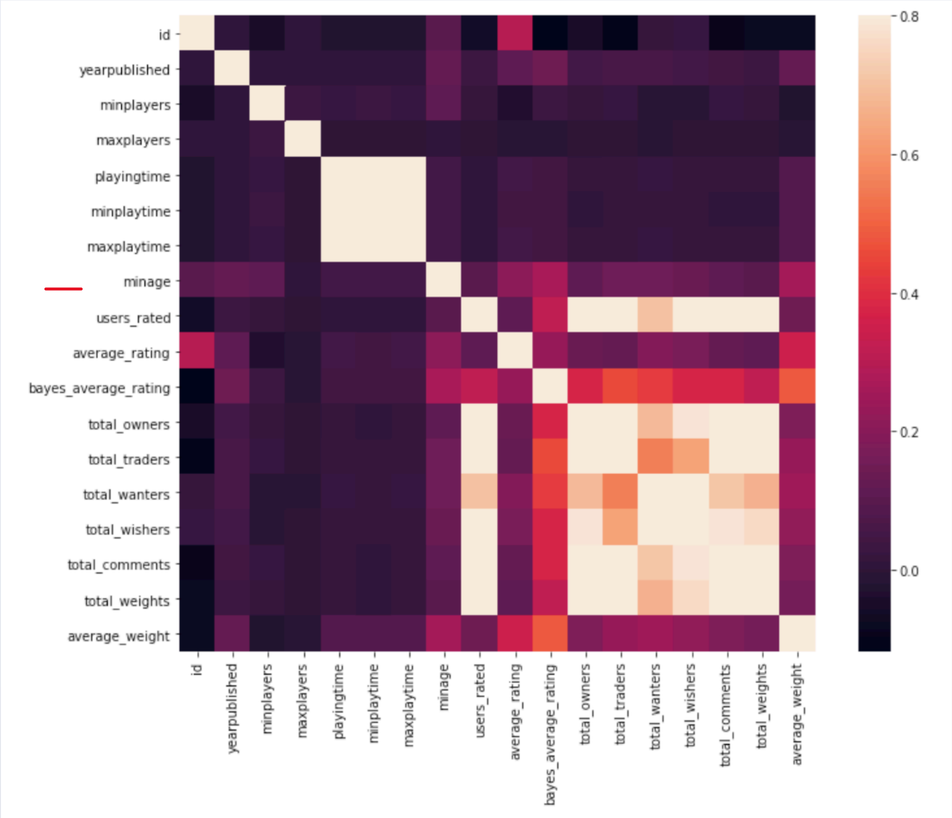
Srivastava, Tavish. "Tuning the Parameters of Your Random Forest Model." *Analyticsvidhya*, 9 June 2015, www.analyticsvidhya.com/blog/2015/06/tuning-random-forest-model/

https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

Introduction

Board games play a significant role in people's lives, but everyone always wants the best of the best. This incentive made reviews much more relevant to millennials because they always look at the rating before deciding to purchase a certain item.

In this study, the model will be able to predict the rating of a board game on a scale of one to ten. It does this by analyzing many weights, but most importantly, the average rating and the complexity of the game. There are many more, but these were the most important because according to the following correlation matrix.



Methods

I first started by analyzing my data and seeing what was given to me. I made sure to look at the columns (weights) that were provided and the amount of data also. After that, linear regression seemed like the obvious method to use for a problem like this. However, when calculating the mean squared error the resulting error was 2.07, which was higher than I wanted and expected.

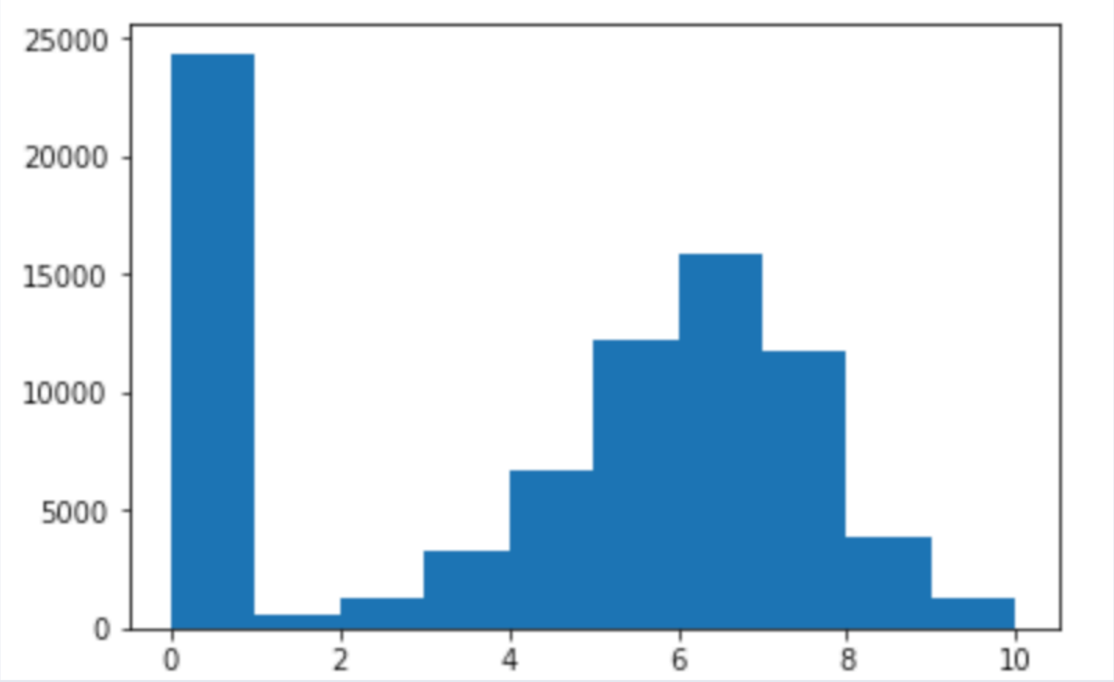
After researching other methods, random forest regression became a solid choice for the dataset.

After implementing, random forest regression gave surprising results, as it outdid the Linear Regression model by a score of about 0.5. the r squared error calculated with the random forest regression model was about 1.64. However, the accuracy was still a bit low, ranging from 0.3-0.7. The model wouldn't seem to increase from that range until I researched more in depth about the parameters and read a few articles/

Data

At the start, the data consisted of 81312 reviews and 20 weights.

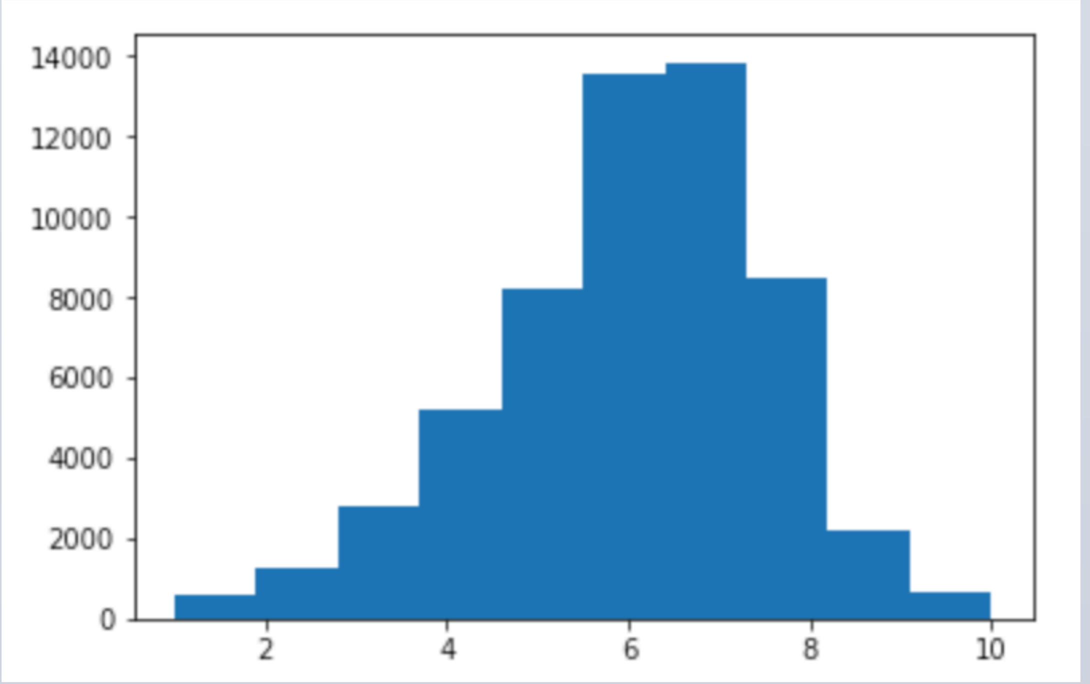
The most logical thing to do was to see how the data looked for the 'average reviews' as that was the most beneficial weight to observe. However, in doing this, I realized that there was a slight bump in my data.



As seen on the diagram above, there seemed to be a lot reviews bunching up around 0, so I decided to look deeper into my data and print out individual reviews until I found the reason why it was so bad.

```
In [307]: print(data[data['average_rating'] == 0].iloc[0])
          id                               318
          type                       boardgame
          name                      Looney Leo
          yearpublished                      0
          minplayers                         0
          maxplayers                         0
          playingtime                        0
          minplaytime                        0
          maxplaytime                        0
          minage                             0
          users_rated                        0
          average_rating                     0
          bayes_average_rating               0
          total_owners                       0
          total_traders                      0
          total_wanters                      0
          total_wishers                      1
          total_comments                     0
          total_weights                      0
          average_weight                     0
          Name: 13048, dtype: object
```

There were board games that were never even published, therefore, never having any reviews, resulting in a score of 0. Filtering out the data that wasn't greater than 0 wasn't difficult and it improved the data significantly!



Conclusion

results

	0
0	8.042688604857169
1	7.979894931999945
2	7.951923988571439
3	7.793821603999955
4	7.924659268571478
5	7.87116211999997
6	7.92145892538099
7	7.816103819999993
8	7.860188800000048
9	7.8727199410000575
10	7.6563734199999764
11	7.7747291649999575
12	7.607266039999963
13	7.6159155550000035
14	7.615500461333329
15	7.564696039036042
16	7.624683219999996
17	7.636785099999958
18	7.454495420000033
19	7.564615159999964
20	7.552969799333293

actual_results

	average_rating
9	8.07933
14	7.99115
15	8.030710000000001
18	7.87047
20	7.98786
27	7.82181
39	7.92585
41	7.8579
42	7.86088
47	7.81642
48	7.82838
51	7.94325
52	7.733910000000001
59	7.75639
61	7.633830000000001
64	7.771610000000001
68	7.8028699999999995
78	7.616689999999999
86	7.51173
94	7.53118
96	7.657769999999999

After adjusting the random state, number of estimators, and oob score, my accuracy shot up to .91! The results shown above have these parameters modified and it can be seen that the predictions are very close to the actual review score of the board game.

It was interesting to use a model that I haven't used before, and not only that, but a model that was better than a Linear Regression model. I can

Acknowledgements

I would like to acknowledge Professor Hao Ji for providing great lectures slides to look back on and review in order to become successful in Machine Learning.