Karefa Kargbo

Computer Science Capstone

Prof. Owrang

2019 April 29

BoardGameGeek Data Analysis

1. Introduction

Board Game Geek is the largest database on the internet about board games, cataloging tens of thousands of titles spanning decades of releases. In addition, the site has a vibrant community of users who spend time using the site to rate the games they've played, track their collections, and discuss their thoughts and strategies. The site's main user base consists of fans of more niche, complex board games rather than older, more traditional fare. As such, this data will be capturing only a subset of the board game market. This project is about seeing which variables, mechanics, themes, and categories will lead to more successful games. Success in this case was tracked by using 3 different variables, Ranking, Average Rating, and Owners. The variables in the dataset with the largest positive effect on these were a Roll and Move Mechanic, the game's Weight score (a measure of complexity, not mass), and the amount of Iterations a game has had respectively (Though that last one probably has a reversed relationship, with a game getting more iterations the more people own it).

2. Gathering Data

The dataset for this analysis was obtained by using a custom made Python script that utilizes a module created to gather data from BoardGameGeek. There was an application that I found called BGG1Tool, that would gather the data for me. However, when I attempted to use it, I found that at high game counts, it would fail at some point well below the amount of games I was trying to get data on. In fact, my script ran into the same problem, showing that it's likely a problem with their API. I had to alter my script to keep track of which games still needed to be taken, and rerun the function when it ran into a problem with the API. I still used BGG1Tool in order to get the list of games we're tracking. The script takes about 8 hours or so to gather data from the site, and store it in a .csv file, so getting a new dataset after doing a modification to the script can take a significant amount of time.

3. Dataset Information

BoardGameGeek has tens of thousands of games in its database, with over a thousand pages of games and expansions. However, we can't look at every game on this database. Many of them are obscure, random titles with very little information on them. More importantly, many of them are games that have yet to be released. In order to apply a reasonably limit to the dataset, I've decided to only look at games that have a ranking associated with them, as this means that the game has been rated enough times to gain a ranking. Even this massive restriction still leaves us with a dataset containing over 17000 games though, so there's no danger of not having enough data to work with.

4. Correlations

I decided that what I was going to look at which 5 variables in the dataset were the most positively and negatively correlated with each other and focus on regressing those variables. This test was performed using Excel, due to the simplicity in exporting, sorting, and using the data. In the end, the correlation between the datasets quite weak across the board, with most of them not reaching above a value of about 0.2. However, for completion's sake, I still performed a regression analysis on all of the top 5 (except for where there may have been problems of multicollinearity. Here are the 5 positive and negative correlations for each of the dependent variables we're looking at:

Positive Correlation

Ranking (Note that a lower rank is better, so a positive correlation is bad)

1. Roll and Move Mechanic: 0.24
2. Children's Game: 0.19
3. Trivia: 0.14
4. Movies/TV/Radio: 0.12
5. Party Game: 0.10

Average Rating

1. Weight: 0.48
2. Wargame: 0.23
3. Variable Player Powers: 0.19
4. Owners: 0.19
5. Miniatures: 0.19

Owners

1. Number of Iterations: 0.36
2. Number of Expansions: 0.29
3. Rating: 0.19
4. Variable Player Powers: 0.12

5. Hand Management Mechanic: 0.12

Negative Correlations

Ranking (Note once again that a lower rank is better, so a negative correlation is good)

Also, rating is very heavily correlated, but is due to the fact that it is a large, obvious, and public factor in determining rating, we aren't looking at it

1. Weight: -0.38
2. Owners: -0.34
3. Age: -0.21
4. Number of Expansions: -0.19
5. Variable Player Powers: -0.18

Average Rating

Once again, ranking is heavily correlated, but obvious

1. Roll and Move: -0.27
2. Children's Game: -0.22
3. Trivia: -0.15
4. Movies/TV/Radio: -0.14
5. Minimum Player Count: -0.14

Owners

1. Rank: -0.34
2. Hex-and-Counter Mechanic: -0.07
3. Wargame Category: -0.06
4. Children's Game: -0.05
5. World War II: -0.04

5. Regression

The regression that I performed on the data was done using a Python module called Statsmodels, which allows you to perform many different statistical analysis techniques. Regression is a method that tries to find the how much one or more variables affect one another assuming all other variables are held constant. Specifically, I used the Ordinary Least Squares method, or OLS method, which attempts to minimize the sum of the squares of the differences observed between the predicted data and the actual data. While Statsmodels was able to output the regression data in a very nice format, exporting it to a format that I could use outside of my Python IDE was harder than expected. I ultimately simply took a capture of my screen. While being only loosely correlated with the data a lot of the time, nearly all of the relationships found were found to be statistically significant. However, this is not really a huge boon, as many relationships can be statistically significant while not actually showing anything (All statistically significant means is that this pattern is not likely to be a trick of the sample). Some variables had to be removed from some regressions due to potential problems of multicollinearity, which is when one predictor in a multiple regression model can be predicted from the others with a decent degree of accuracy. This can lower the predictive power of the individual predicted coefficients in the model (but not usually of the model itself). The different images for the regression tables are shown below, with minor code snippets:

```
In [76]: X = GameData[["Hex-and-Counter Mechanic", "Wargame Category", "Childrens Game Category", "World
War II Category"]]

In [77]: X = sm.add_constant(X)

In [78]: model = sm.OLS(y, X).fit()

In [79]: predictions = model.predict(X)

In [80]: model.summary()
Out[80]:
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                 Owners   R-squared:                       0.008
Model:                            OLS   Adj. R-squared:                  0.007
Method:                 Least Squares   F-statistic:                     32.69
Date:                Mon, 29 Apr 2019   Prob (F-statistic):           3.43e-27
Time:                        23:03:15   Log-Likelihood:            -1.6628e+05
No. Observations:               17010   AIC:                         3.326e+05
Df Residuals:                   17005   BIC:                         3.326e+05
Df Model:                           4
Covariance Type:            nonrobust
============================================================================================
                               coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------------------
const                     1465.1938     38.021     38.536      0.000    1390.669    1539.719
Hex-and-Counter Mechanic  -669.9230    148.550     -4.510      0.000    -961.097    -378.749
Wargame Category          -303.8578    123.182     -2.467      0.014    -545.307     -62.408
Childrens Game Category   -907.7959    124.455     -7.294      0.000   -1151.741    -663.850
World War II Category      -77.8471    160.349     -0.485      0.627    -392.148     236.454
==============================================================================
Omnibus:                    28917.930   Durbin-Watson:                   0.969
Prob(Omnibus):                  0.000   Jarque-Bera (JB):         31760292.564
Skew:                          11.710   Prob(JB):                         0.00
Kurtosis:                     213.388   Cond. No.                         5.53
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
"""
```

```
In [58]: X = GameData[["Roll and Move Mechanic", "Childrens Game Category", "Trivia Category", "Movies/
TV/Radio Category", "Min Players"]]

In [59]: y = targets["Rating"]

In [60]: X = sm.add_constant(X)

In [61]: model = sm.OLS(y, X).fit()

In [62]: predictions = model.predict(X)

In [63]: model.summary()
Out[63]:
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                 Rating   R-squared:                       0.142
Model:                            OLS   Adj. R-squared:                  0.142
Method:                 Least Squares   F-statistic:                     563.0
Date:                Mon, 29 Apr 2019   Prob (F-statistic):               0.00
Time:                        22:10:13   Log-Likelihood:                -21289.
No. Observations:               17010   AIC:                         4.259e+04
Df Residuals:                   17004   BIC:                         4.264e+04
Df Model:                           5
Covariance Type:            nonrobust
============================================================================================
                               coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------------------
const                        6.8613      0.021    328.960      0.000       6.820       6.902
Roll and Move Mechanic      -0.7379      0.027    -27.378      0.000      -0.791      -0.685
Childrens Game Category     -0.6540      0.025    -26.240      0.000      -0.703      -0.605
Trivia Category             -0.5500      0.041    -13.476      0.000      -0.630      -0.470
Movies/TV/Radio Category    -0.3451      0.031    -11.148      0.000      -0.406      -0.284
Min Players                 -0.1844      0.010    -19.150      0.000      -0.203      -0.166
==============================================================================
Omnibus:                      618.303   Durbin-Watson:                   0.870
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              942.106
Skew:                          -0.348   Prob(JB):                    2.66e-205
Kurtosis:                       3.918   Cond. No.                         15.1
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

```
In [51]: X = GameData[["Weight", "Age", "Number of Expansions", "Variable Player Power Mechanic"]]

In [52]: X = sm.add_constant(X)

In [53]: model = sm.OLS(y, X).fit()

In [54]: predictions = model.predict(X)

In [55]: model.summary()
Out[55]:
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                   Rank   R-squared:                       0.186
Model:                            OLS   Adj. R-squared:                  0.186
Method:                 Least Squares   F-statistic:                     971.0
Date:                Mon, 29 Apr 2019   Prob (F-statistic):               0.00
Time:                        21:59:24   Log-Likelihood:            -1.6700e+05
No. Observations:               17010   AIC:                         3.340e+05
Df Residuals:                   17005   BIC:                         3.340e+05
Df Model:                           4
Covariance Type:            nonrobust
==================================================================================================
                                     coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------------------------
const                           1.379e+04    111.902    123.231      0.000    1.36e+04     1.4e+04
Weight                         -1909.8402     42.718    -44.709      0.000   -1993.571   -1826.109
Age                             -121.9346      9.781    -12.467      0.000    -141.106    -102.763
Number of Expansions             -90.6682      4.968    -18.249      0.000    -100.407     -80.930
Variable Player Power Mechanic -1433.7355    109.136    -13.137      0.000   -1647.652   -1219.819
==============================================================================
Omnibus:                      593.881   Durbin-Watson:                   0.363
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              268.509
Skew:                          -0.009   Prob(JB):                     4.94e-59
Kurtosis:                       2.385   Cond. No.                         36.2
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

```
In [39]: X = GameData[["Number of Iterations", "Number of Expansions", "Rating", "Variable Player Power
Mechanic", "Hand Management Mechanic"]]

In [40]: y = targets["Owners"]

In [41]: X = sm.add_constant(X)

In [42]: model = sm.OLS(y, X).fit()

In [43]: predictions = model.predict(X)

In [44]: model.summary()
Out[44]:
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                 Owners   R-squared:                       0.205
Model:                            OLS   Adj. R-squared:                  0.204
Method:                 Least Squares   F-statistic:                     874.7
Date:                Mon, 29 Apr 2019   Prob (F-statistic):               0.00
Time:                        21:33:11   Log-Likelihood:            -1.6440e+05
No. Observations:               17010   AIC:                         3.288e+05
Df Residuals:                   17004   BIC:                         3.289e+05
Df Model:                           5
Covariance Type:            nonrobust
================================================================================================
                                   coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------------------------
const                        -3017.7203    208.786    -14.454      0.000   -3426.963   -2608.477
Number of Iterations          1560.9686     36.394     42.891      0.000    1489.633    1632.304
Number of Expansions           119.9280      4.374     27.419      0.000     111.355     128.501
Rating                         547.1752     33.017     16.572      0.000     482.458     611.893
Variable Player Power Mechanic 524.4883     94.142      5.571      0.000     339.959     709.017
Hand Management Mechanic       861.2538     73.802     11.670      0.000     716.594    1005.914
==============================================================================
Omnibus:                    26094.097   Durbin-Watson:                   1.118
Prob(Omnibus):                  0.000   Jarque-Bera (JB):         18804545.139
Skew:                           9.543   Prob(JB):                         0.00
Kurtosis:                     164.764   Cond. No.                         54.4
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
"""
```

```
In [34]: X = GameData[["Weight", "Wargame Category", "Variable Player Power Mechanic", "Miniatures
Category"]]

In [35]: X = sm.add_constant(X)

In [36]: model = sm.OLS(y, X).fit()

In [37]: predictions = model.predict(X)

In [38]: model.summary()
Out[38]:
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                 Rating   R-squared:                       0.251
Model:                            OLS   Adj. R-squared:                  0.251
Method:                 Least Squares   F-statistic:                     1423.
Date:                Mon, 29 Apr 2019   Prob (F-statistic):               0.00
Time:                        21:14:03   Log-Likelihood:                -20135.
No. Observations:               17010   AIC:                         4.028e+04
Df Residuals:                   17005   BIC:                         4.032e+04
Df Model:                           4
Covariance Type:            nonrobust
==================================================================================================
                                     coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------------------------
const                              5.3402      0.016    328.546      0.000       5.308       5.372
Weight                             0.4761      0.008     57.218      0.000       0.460       0.492
Wargame Category                   0.0464      0.018      2.603      0.009       0.011       0.081
Variable Player Power Mechanic     0.2865      0.020     14.436      0.000       0.248       0.325
Miniatures Category                0.3669      0.030     12.190      0.000       0.308       0.426
==============================================================================
Omnibus:                     1119.922   Durbin-Watson:                   0.896
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             2355.773
Skew:                          -0.446   Prob(JB):                         0.00
Kurtosis:                       4.590   Cond. No.                         12.1
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
"""
```

```
In [21]: X = GameData[["Roll and Move Mechanic", "Childrens Game Category", "Trivia Category",
TV/Radio Category", "Party Game Category"]]

In [22]: y = target["Rank"]
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-22-7c2cd9549d9e> in <module>
----> 1 y = target["Rank"]

NameError: name 'target' is not defined

In [23]: y = targets["Rank"]

In [24]: X = sm.add_constant(X)

In [25]: model = sm.OLS(y, X).fit()

In [26]: predictions = model.predict(X)

In [27]: model.summary()
Out[27]:
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                   Rank   R-squared:                       0.098
Model:                            OLS   Adj. R-squared:                  0.097
Method:                 Least Squares   F-statistic:                     367.8
Date:                Mon, 29 Apr 2019   Prob (F-statistic):               0.00
Time:                        19:45:20   Log-Likelihood:            -1.6787e+05
No. Observations:               17010   AIC:                         3.358e+05
Df Residuals:                   17004   BIC:                         3.358e+05
Df Model:                           5
Covariance Type:            nonrobust
============================================================================================
                              coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------------------
const                    7819.1829     40.112    194.933      0.000    7740.559    7897.807
Roll and Move Mechanic   3578.9839    149.004     24.019      0.000    3286.922    3871.046
Childrens Game Category  2859.5685    137.719     20.764      0.000    2589.626    3129.511
Trivia Category          2428.3206    234.910     10.337      0.000    1967.874    2888.768
Movies/TV/Radio Category 1527.3248    171.181      8.922      0.000    1191.791    1862.858
Party Game Category      1185.0895    133.891      8.851      0.000     922.649    1447.530
==============================================================================
Omnibus:                     4365.044   Durbin-Watson:                   0.188
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              739.205
Skew:                          -0.015   Prob(JB):                    3.05e-161
Kurtosis:                       1.979   Cond. No.                         6.83
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
"""
```

6: Conclusions

In the end, I’d say that while the rank and ratings might be able to be affected by various things, the amount of owners of a game has such a low correlation to almost everything that I’d say it’s hard to predict using these methods. Which honestly shows that the best way for people to get people to play your game is to make a good game regardless of its themes and other aspects (as can be seen by the fact that a low (better) ranking is decently correlated with more owners).