Open in Colab

Board Game Review Project

For as long as humans have been wandering the earth, they've been trying to find new ways to entertain themselves. Sure food, water, and shelter will get you through the day, but what is life without a little fun? Prehistoric gamers utilized tools like bones and shells to aid them in gameplay, and simple games like jacks and dice are some of the earliest known gaming types. Today games may have become a bit more sophisticated, but at their core the same principles still apply – challenge your mind, take on competitors, and most of all – have fun!

A board game is a game that involves counters or pieces moved or placed on a pre-marked surface or "board", according to a set of rules. Games can be based on pure strategy, chance or a mixture of the two, and usually have a goal that a player aims to achieve. Early board games represented a battle between two armies, and most current board games are still based on defeating opposing players in terms of counters, winning position or accrual of points. There are many different types and styles of board games. Their representation of real-life situations can range from having no inherent theme, as with checkers, to having a specific theme and narrative, as with Cluedo. Rules can range from the very simple, as in Tic-tac-toe, to those describing a game universe in great detail, as in Dungeons & Dragons. The amount of time required to learn to play or master a game varies greatly from game to game. Learning time does not necessarily correlate with the number or complexity of rules; some games, such as chess or Go, have simple rulesets while possessing profound strategies.

Board Games

Tried and tested

Though not quite as old as simple dice games, board games have been around for quite awhile, and are still going strong today!

One of the earliest known board games was game of Senet, found in Predynastic and First Dynasty burial sites in Egypt, and in hieroglyphs dating to around 3100 BCE. The game consisted of a board with 30 squares arranged in 3 rows of 10. It seems to have been a 2 player game, with at least 5 pawns per player (though sometimes more). Historians debate over the exact rules of the game, and like many things it's likely that the rules changed over time.

Stakes were somewhat higher when playing Senet than they are in modern board games, considering ancient Egyptians believed strongly in fate, and success in the game was thought to be strongly tied to protection by the gods!

Longlasting

Board games often morph over time, or fall in and out of favor. But some games have had long standing success in their ability to captivate and entertain!

Backgammon is one of the longest running games that is still played in modern times, but coming in with an even longer track record was the Royal Game of Ur, another game dating from the 3000s. Ur, a two-player strategy racing game, was popular in ancient Mesopotamia, and a version of the game called Aasha was still being played in Kochi, India, as late as the 1950s.

Life Lessons

Have you ever wanted to table flip the entire board after a night of Monopoly with the family? There's actually a reason for that!

Monopoly is actually an evolution of "The Landlord's Game," a board game created by one of America's first board game designers, Lizzie Magie, in 1904. Magie's game was based on the economic principles of Georgism, and was designed to show how landgrabbing enriched owners and impoverished renters. It was designed to provoke a sense of this system's unfairness in the children who played it, with the hope that they would take such lessons to heart as they grew older. So the table-flip rage you sometimes feel at family fun night isn't a fluke, it's just part of the game!

Magie sold Monopoly to the Parker Brothers in 1935 for $500, and it became their cash cow, launching them into the financial success their company would enjoy for over 100 years!

Video Games

A digital revolution

The first electronic digital computers, Colossus and ENIAC, were built during World War II to aid the Allied war effort. After the war computer program architectures stored at universities in the UK and US allowed computers to be reprogrammed for other tasks, which facilitated commercialization and the adoption of computers by universities, government organizations, and large corporations as the decade progressed.

Of course, computers could be more than purely functional! Not too long after the technological explosion brought on by these machines, the first computer games were born.

Early computer games were designed to demonstrate the power of these new machines, but perhaps the first game created solely for entertainment was William Higinbotham's "Tennis for Two," which allowed the player to bounce a virtual tennis ball (a point of light) over a virtual net (a vertical line) using a controller.

Rapid Evolution

As computers became more sophisticated, so too did the games people played on them!

Pong, a game in which featured 2 lines and a white square as its main visuals, was the first arcade video game to receive wide commercial success.

From there we had an explosion of arcade consoles in the late 70s and early 80s, as well as the rise of home consoles.

The popularity of arcade games eventually fizzled out, but home consoles kept going strong. 8-bit games that still maintain legendary franchise status such as Super Mario Bros. and The Legend of Zelda were released by Nintendo in the late 80s.

Home gaming continued to rise in popularity throughout the 90s, and new technologies led to games with increasingly impressive visuals, with games incorporating 3D graphics by the mid-90s.

Modern video games use new techniques to immerse the player into the experience. Games that incorporate AR

and VR allow the player to use their body to interact in a more natural way with the virtual world, rather than just by mashing buttons on a stationary controller.

As computer technology continues to advance, video games will certainly follow suit! We can only imagine what the next evolution of gaming will be!

Career Makers

For most people video games are just a hobby, but for a select few they can turn into a lucrative career! Esports have exploded in recent years, with top players cashing in on millions of dollars in prize money.

Championship matches can bring in arenas of tens of thousands of people, with online audiences in the millions! And they don't seem to be slowing down anytime soon.

If you have your sights set on a future of gaming as a profession, team video games may be the one for you!

About the Data

The dataset is commonly shared in the Kaggle and other data-science related communities but as far as this project goes the source of this data wasn't primarily given, and is only shared purposefully specific to this project. Hence, this dataset is stored and kept as is provided in the below link.

> https://raw.githubusercontent.com/WolfDev8675/RepoSJX7/Assign5_1/data/games.csv

Primaries

In [62]:

```
#!usr/bin/python

''' Board Game Review Project '''
#

#imports
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plts
import warnings
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler,normalize
from sklearn.linear_model import LinearRegression as LR
from sklearn.linear_model import Ridge as R2
from sklearn.linear_model import BayesianRidge as BR2
from sklearn.svm import SVR
from sklearn.tree import DecisionTreeRegressor as DTR
from sklearn.ensemble import RandomForestRegressor as RFT
```

Presetting viewer

In [63]:

```
# Presetting viewer
pd.set_option('display.max_columns', None)
```

Loading dataset: data-specs, size and contents

In [64]:

```
# Loading dataset: data-specs, size and contents
d_set=pd.read_csv("https://raw.githubusercontent.com/WolfDev8675/RepoSJX7/Assign5_1/data/
```

```
games.csv")
#print(' Dataset example: \n')
d_set.head(20)
```

Out[64]:

	id	type	name	yearpublished	minplayers	maxplayers	playingtime	minplaytime	maxplaytime	minage
0	12333	boardgame	Twilight Struggle	2005.0	2.0	2.0	180.0	180.0	180.0	13.0
1	120677	boardgame	Terra Mystica	2012.0	2.0	5.0	150.0	60.0	150.0	12.0
2	102794	boardgame	Caverna: The Cave Farmers	2013.0	1.0	7.0	210.0	30.0	210.0	12.0
3	25613	boardgame	Through the Ages: A Story of Civilization	2006.0	2.0	4.0	240.0	240.0	240.0	12.0
4	3076	boardgame	Puerto Rico	2002.0	2.0	5.0	150.0	90.0	150.0	12.0
5	31260	boardgame	Agricola	2007.0	1.0	5.0	150.0	30.0	150.0	12.0
6	124742	boardgame	Android: Netrunner	2012.0	2.0	2.0	45.0	45.0	45.0	14.0
7	96848	boardgame	Mage Knight Board Game	2011.0	1.0	4.0	150.0	150.0	150.0	14.0
8	84876	boardgame	The Castles of Burgundy	2011.0	2.0	4.0	90.0	30.0	90.0	12.0
9	72125	boardgame	Eclipse	2011.0	2.0	6.0	200.0	60.0	200.0	14.0
10	2651	boardgame	Power Grid	2004.0	2.0	6.0	120.0	120.0	120.0	12.0
11	164153	boardgame	Star Wars: Imperial Assault	2014.0	2.0	5.0	90.0	90.0	90.0	0.0
12	115746	boardgame	War of the Ring (second edition)	2012.0	2.0	4.0	150.0	150.0	150.0	13.0
13	121921	boardgame	Robinson Crusoe: Adventures on the Cursed Island	2012.0	1.0	4.0	180.0	90.0	180.0	14.0
14	35677	boardgame	Le Havre	2008.0	1.0	5.0	200.0	100.0	200.0	12.0
15	28720	boardgame	Brass	2007.0	3.0	4.0	180.0	120.0	180.0	13.0
16	126163	boardgame	Tzolk'in: The Mayan Calendar	2012.0	2.0	4.0	90.0	90.0	90.0	13.0
17	150376	boardgame	Dead of Winter: A Crossroads Game	2014.0	2.0	5.0	210.0	45.0	210.0	14.0
18	68448	boardgame	7 Wonders	2010.0	2.0	7.0	30.0	30.0	30.0	10.0
19	18602	boardgame	Caylus	2005.0	2.0	5.0	150.0	60.0	150.0	12.0

In [65]:

```
print('_'.center(70,'_'))
```

```
print('\n\n Content information :: \n');d_set.info();print('_'.center(70,'_'));
print('\n');print('x'.center(70,'*'))
```

```
 Content information ::

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 81312 entries, 0 to 81311
Data columns (total 20 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   id                    81312 non-null  int64
 1   type                  81312 non-null  object
 2   name                  81271 non-null  object
 3   yearpublished         81309 non-null  float64
 4   minplayers            81309 non-null  float64
 5   maxplayers            81309 non-null  float64
 6   playingtime           81309 non-null  float64
 7   minplaytime           81309 non-null  float64
 8   maxplaytime           81309 non-null  float64
 9   minage                81309 non-null  float64
 10  users_rated           81312 non-null  int64
 11  average_rating        81312 non-null  float64
 12  bayes_average_rating  81312 non-null  float64
 13  total_owners          81312 non-null  int64
 14  total_traders         81312 non-null  int64
 15  total_wanters         81312 non-null  int64
 16  total_wishers         81312 non-null  int64
 17  total_comments        81312 non-null  int64
 18  total_weights         81312 non-null  int64
 19  average_weight        81312 non-null  float64
dtypes: float64(10), int64(8), object(2)
memory usage: 12.4+ MB
```

**********x**********

Noise and Vaccancy Study

In [66]:

```
# Noise and Vaccancy Study
print(" Noise and vacancy study ");print('_'.center(70,'_'));
print(' Null values per column \n\n',d_set.isnull().sum());print('_'.center(70,'_'))   #
8  fields has nulls
print(' Connection between nulls of various fields:-> [Name] \n',d_set[d_set['name'].notn
a()].isnull().sum());print('_'.center(70,'_'))
print(""" Connection between nulls of various fields:-> [yearpublished, minplayers,
        maxplayers, playingtime, minplaytime, maxplaytime, minage] \n""",
      d_set[d_set['yearpublished'].isnull()])
print('_'.center(70,'_'));
print('x'.center(70,'*'))
# ---- conclusion: nulls in name field is solo others are connected
```

```
 Noise and vacancy study
______________________________________________________________________
 Null values per column

 id                     0
type                    0
name                   41
yearpublished           3
minplayers              3
maxplayers              3
playingtime             3
minplaytime             3
maxplaytime             3
minage                  3
```

```
minage                3
users_rated           0
average_rating        0
bayes_average_rating  0
total_owners          0
total_traders         0
total_wanters         0
total_wishers         0
total_comments        0
total_weights         0
average_weight        0
dtype: int64
```

```
 Connection between nulls of various fields:-> [Name]
 id                    0
type                   0
name                   0
yearpublished          3
minplayers             3
maxplayers             3
playingtime            3
minplaytime            3
maxplaytime            3
minage                 3
users_rated            0
average_rating         0
bayes_average_rating   0
total_owners           0
total_traders          0
total_wanters          0
total_wishers          0
total_comments         0
total_weights          0
average_weight         0
dtype: int64
```

```
 Connection between nulls of various fields:-> [yearpublished, minplayers,
        maxplayers, playingtime, minplaytime, maxplaytime, minage]
           id       type                  name  yearpublished  minplayers  \
45972  53344  boardgame  Kirby the Board Game            NaN         NaN
46057  54165  boardgame  Clash of the Princes            NaN         NaN
46537  57161  boardgame              Showdown            NaN         NaN

       maxplayers  playingtime  minplaytime  maxplaytime  minage  users_rated  \
45972         NaN          NaN          NaN          NaN     NaN            0
46057         NaN          NaN          NaN          NaN     NaN           14
46537         NaN          NaN          NaN          NaN     NaN           18

       average_rating  bayes_average_rating  total_owners  total_traders  \
45972         0.00000                   0.0             2              0
46057         6.50000                   0.0            29              2
46537         6.97222                   0.0            35              0

       total_wanters  total_wishers  total_comments  total_weights  \
45972              0              3               0              0
46057              0              0               9              3
46537              1              4               8              4

       average_weight
45972             0.0
46057             2.0
46537             3.5
```

```
**********x**********
```

conclusion: nulls in name field is solo others are connected/related to each others.
operation needed: separated handling needs to be done in cleaning these fields

Clearing dataset: noise and vaccancy cleaning

In [67]:

```
# Clearing dataset: noise and vaccancy cleaning
d_set_f0=d_set.dropna(axis='index'); # separately removes 41+3 index lines
print(' Pre-Cleaning size  : ',d_set.shape);print('_'.center(70,'_'));
print(' Post-Cleaning size : ',d_set_f0.shape);print('_'.center(70,'_'));
```

```
 Pre-Cleaning size  :  (81312, 20)
______________________________________________________________________
 Post-Cleaning size :  (81268, 20)
______________________________________________________________________
```

Exactly $44 = (81312$ lines of informations got removed

$-\ 81268)$

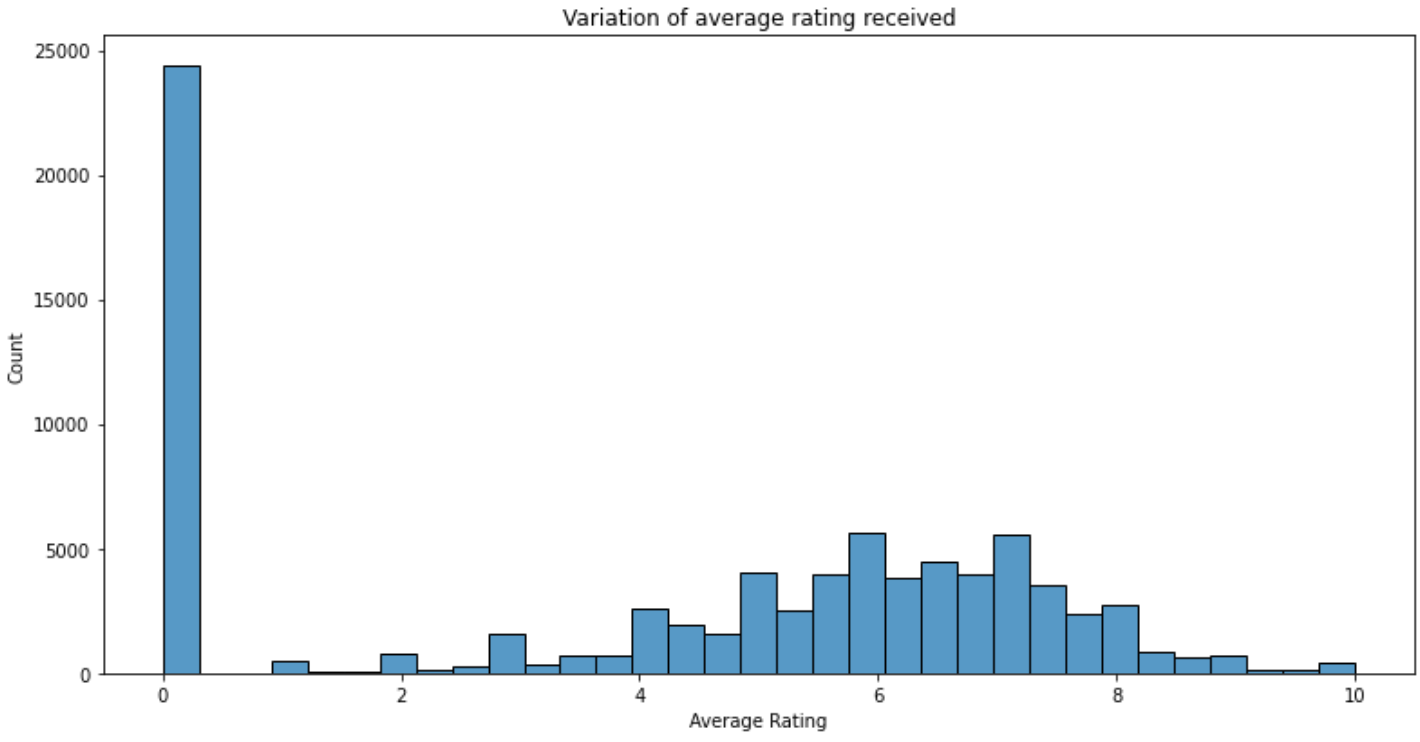
41 from "Name" field blank values

3 from the "NAN" values in "yearpublished", "minplayers", "maxplayers", "playingtime", "minplaytime", "maxplaytime", "minage" fields

Exploration of the data

In [68]:

```
# Exploration of the data
# EDA:#1
fig_1=plts.figure(figsize=(13,6.5));fig_1.canvas.set_window_title('EDA plot(1): Average r
ating histogram')
sns.histplot(d_set_f0['average_rating']);
plts.xlabel('Average Rating');plts.title('Variation of average rating received');
plts.show();
```



Quite a lot of games with Zero rating: this doesn't mean a bad game but specifically games that were never rated. Since their number is too large they might interfere with the proper interpretation of the machine learning algorithms.

In [69]:

```
# EDA:#2
d_set_f0[d_set_f0['average_rating']==0].head()     #check:1
```

Out[69]:

	id	type	name	yearpublished	minplayers	maxplayers	playingtime	minplaytime	maxplaytime	minage	us

	id	type	name	yearpublished	minplayers	maxplayers	playingtime	minplaytime	maxplaytime	minage	us
13048	318	boardgame	Looney Leo	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
13068	579	boardgame	Field of Fire	2002.0	2.0	0.0	0.0	0.0	0.0	12.0	
13114	894	boardgame	LifeLinks	2000.0	1.0	8.0	10.0	10.0	10.0	8.0	
13116	897	boardgame	Dear Abby	1992.0	3.0	4.0	60.0	60.0	60.0	13.0	
13124	946	boardgame	Rolazone	1999.0	2.0	2.0	30.0	30.0	30.0	0.0	

In [70]:

```
# EDA:#2(contd.)
d_set_f0[d_set_f0['average_rating']>0].head()     #check:2
```

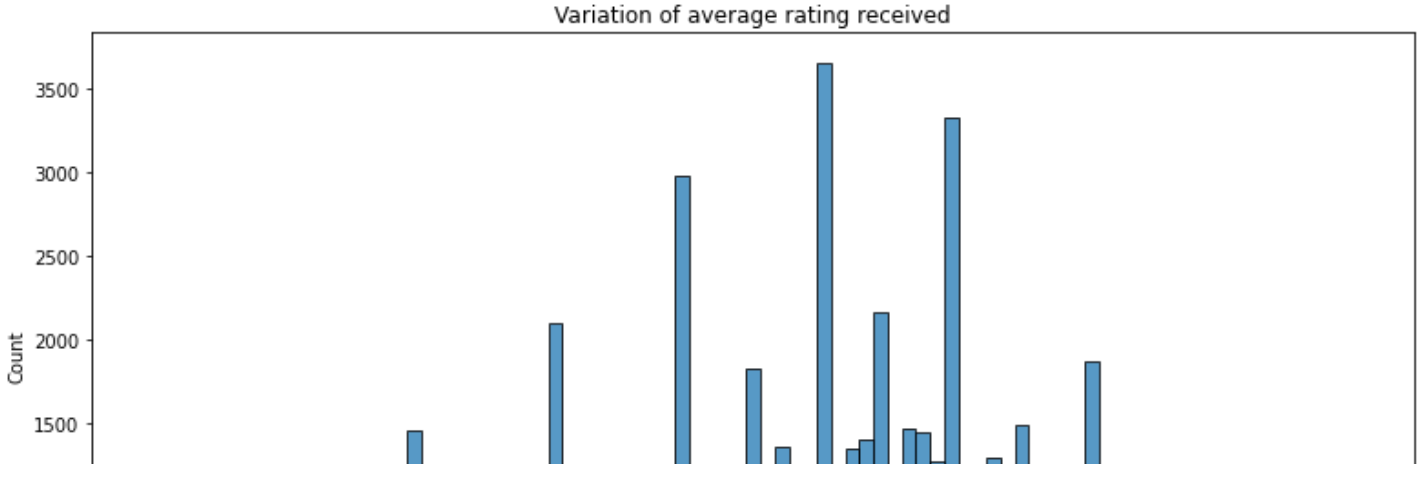
Out[70]:

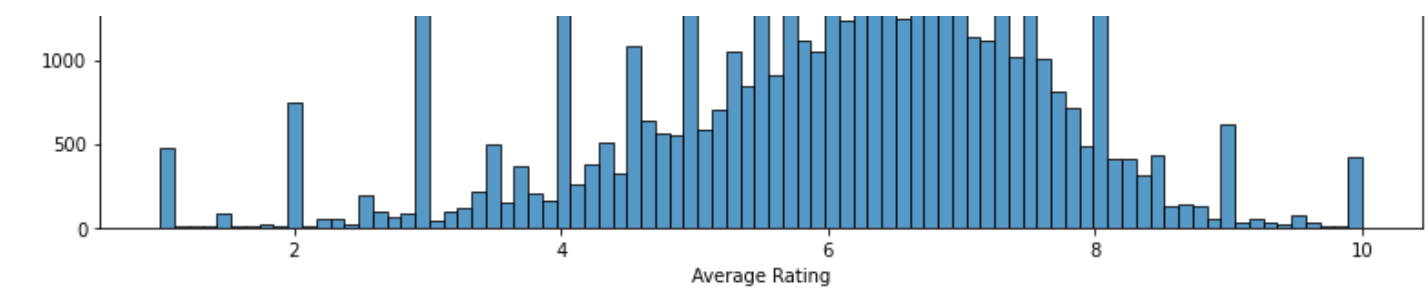
	id	type	name	yearpublished	minplayers	maxplayers	playingtime	minplaytime	maxplaytime	minage	u
0	12333	boardgame	Twilight Struggle	2005.0	2.0	2.0	180.0	180.0	180.0	13.0	
1	120677	boardgame	Terra Mystica	2012.0	2.0	5.0	150.0	60.0	150.0	12.0	
2	102794	boardgame	Caverna: The Cave Farmers	2013.0	1.0	7.0	210.0	30.0	210.0	12.0	
3	25613	boardgame	Through the Ages: A Story of Civilization	2006.0	2.0	4.0	240.0	240.0	240.0	12.0	
4	3076	boardgame	Puerto Rico	2002.0	2.0	5.0	150.0	90.0	150.0	12.0	

info determined: games with average_rating =0 also has users_rating =0. Hence, all those games that has no average rating can be contributed to also having no users rating for the game and therefore may not be a candidate to analysis.

In [71]:

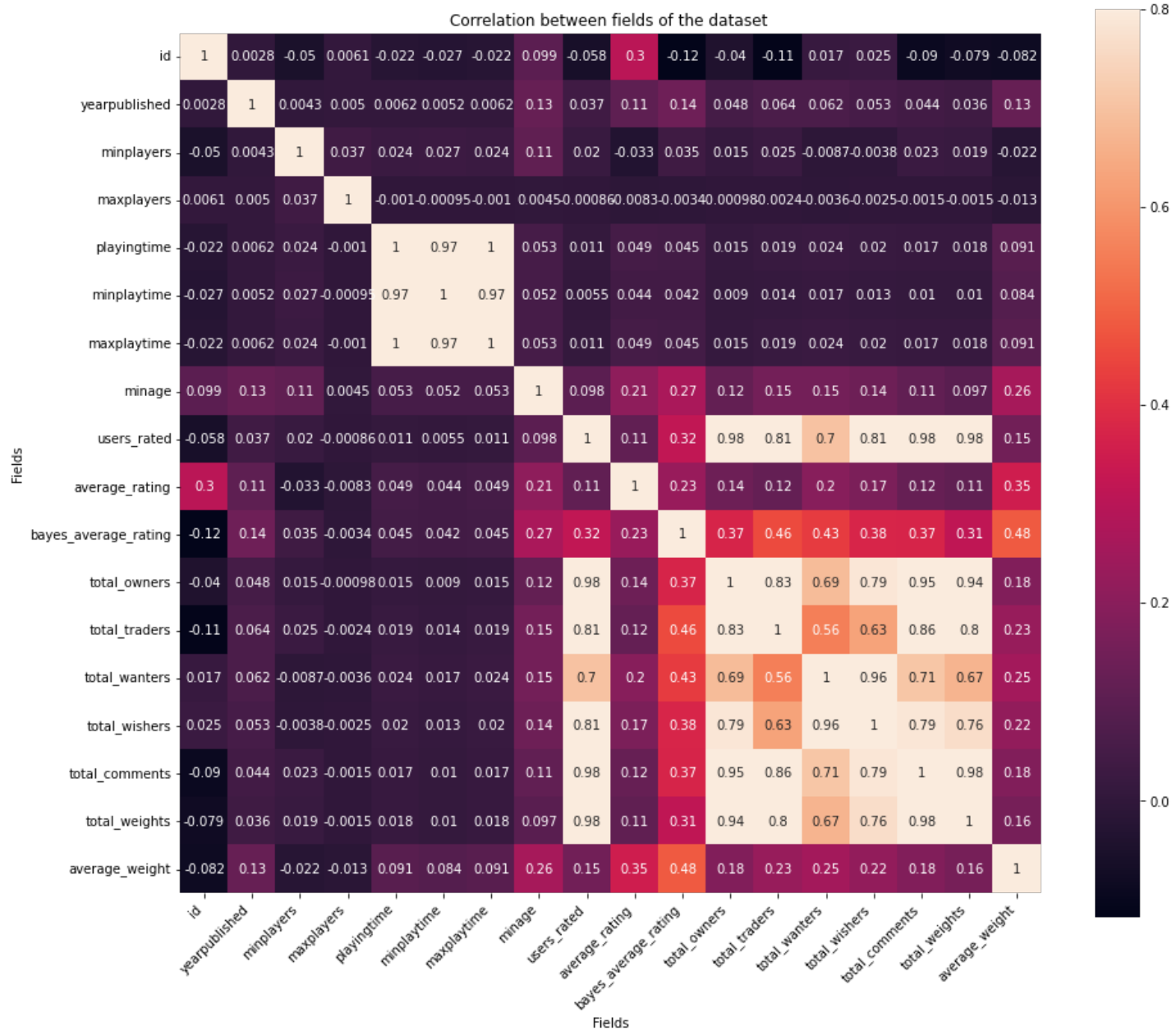
```
# EDA decision #1 (EDA#1 & EDA#2)
d_set_f1=d_set_f0[d_set_f0['average_rating']>0] # keeping only the games that have at lea
st an average rating
# EDA:#3
fig_2=plts.figure(figsize=(13,6.5));fig_2.canvas.set_window_title('EDA plot(2): Average r
ating histogram')
sns.histplot(d_set_f1['average_rating']);
plts.xlabel('Average Rating');plts.title('Variation of average rating received');
plts.show();
```





In [72]:

```
# EDA:#4
corr_info=d_set_f1.corr()
fig_3=plts.figure(figsize=(15,13));fig_3.canvas.set_window_title('EDA plot(3): Field Correlation information')
sns.heatmap(corr_info,vmax=0.8,square=True,annot=True);plts.xticks(rotation=45,ha='right');
plts.xlabel('Fields');plts.ylabel('Fields');plts.title('Correlation between fields of the dataset ');
plts.show();
```



info determined: average rating is the target variable for determination of a game quality.Secondarily, the name, type and id fields have no contribution. Also bayes rating and publishing year have very low correlation to the determining factors.

In [73]:

```
# EDA decision #2 (EDA#3 & EDA#4)
```

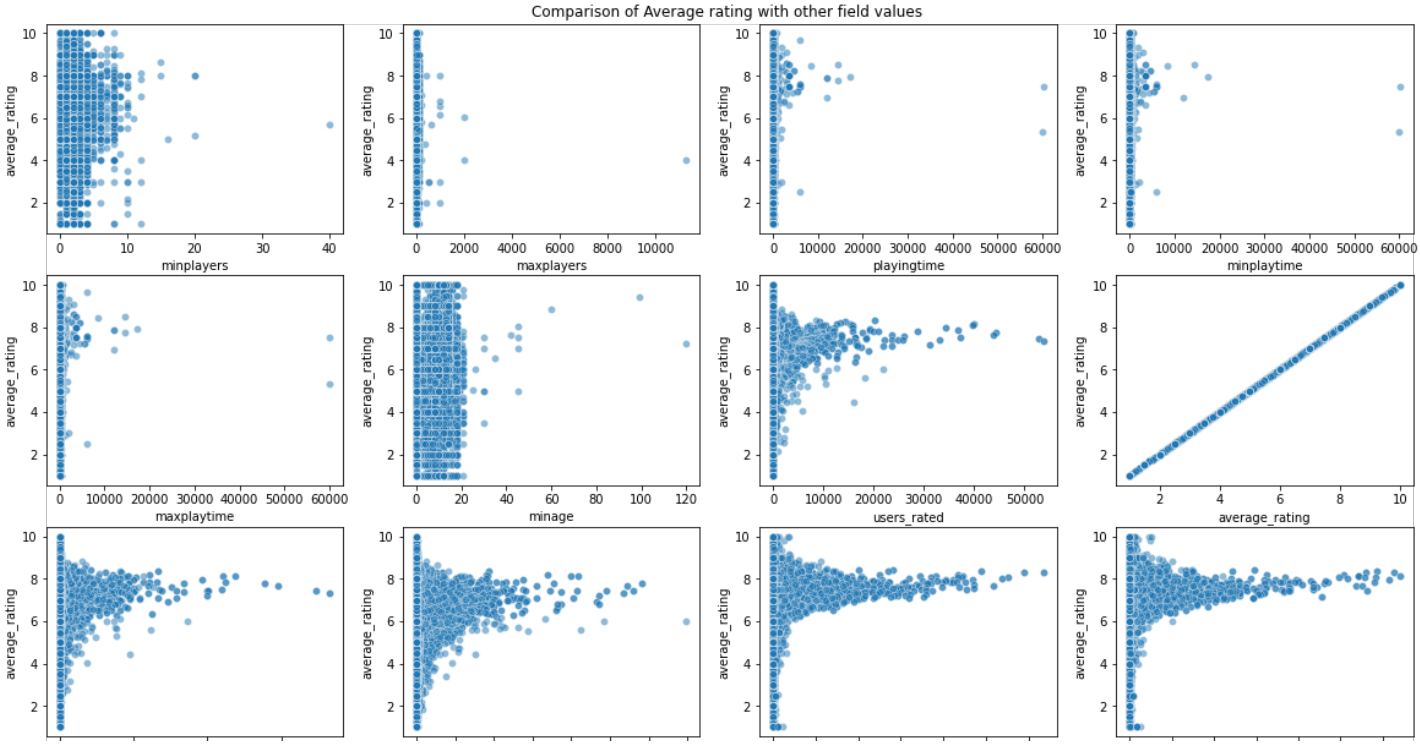
```
# EDA decision #2 (EDA#3 & EDA#4)
contributors=[a_column for a_column in d_set_f1.columns.tolist() if a_column not in ['id
','name','type','bayes_average_rating','yearpublished']]
affectors='average_rating'
full_X=d_set_f1[contributors].values
full_Y=d_set_f1[affectors].values
print(' Dataset components (After EDA decision: #2 ) : ');print('_'.center(70,'_'));
print(" Independent variables' value set \n",full_X);print('_'.center(70,'_'));
print(" Dependent variable's value set \n",full_Y);print('_'.center(70,'_'));
```

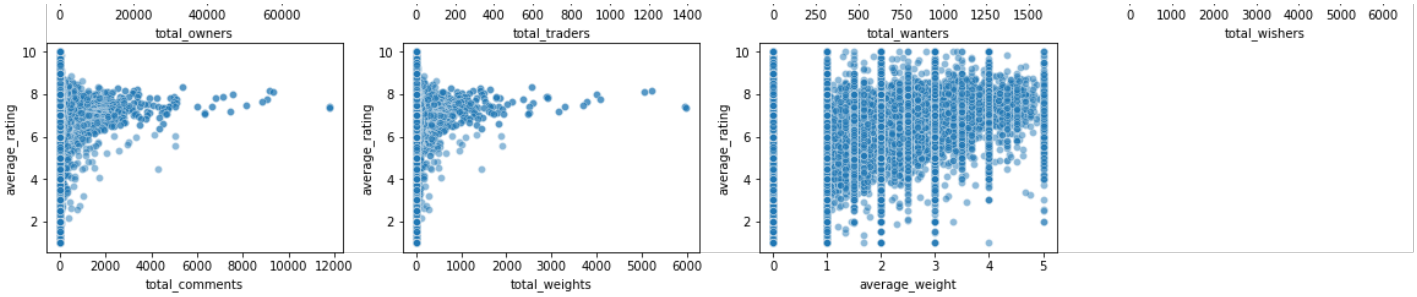
```
 Dataset components (After EDA decision: #2 ) :
______________________________________________________________________
 Independent variables' value set
 [[2.0000e+00 2.0000e+00 1.8000e+02 ... 5.3470e+03 2.5620e+03 3.4785e+00]
 [2.0000e+00 5.0000e+00 1.5000e+02 ... 2.5260e+03 1.4230e+03 3.8939e+00]
 [1.0000e+00 7.0000e+00 2.1000e+02 ... 1.7000e+03 7.7700e+02 3.7761e+00]
 ...
 [2.0000e+00 6.0000e+00 0.0000e+00 ... 2.0000e+00 1.0000e+00 1.0000e+00]
 [2.0000e+00 2.0000e+00 0.0000e+00 ... 1.0000e+00 0.0000e+00 0.0000e+00]
 [2.0000e+00 4.0000e+00 6.0000e+01 ... 0.0000e+00 2.0000e+00 1.5000e+00]]
______________________________________________________________________
 Dependent variable's value set
 [8.33774 8.28798 8.28994 ... 8.      7.      7.     ]
______________________________________________________________________
```

Visualization of the fields distribution

In [74]:

```
fig_4=plts.figure(figsize=(20,14.5));fig_4.canvas.set_window_title('Visuals plot(1): Visu
alization scatters of contributing fields ')
pos_counter=1 #preset value for 16 places 4x4 setup
warnings.filterwarnings('ignore')
plts.title('Comparison of Average rating with other field values ');
plts.tick_params(axis='x',which='both',bottom=False,top=False,labelbottom=False)
plts.tick_params(axis='y',which='both',left=False,right=False,labelleft=False)
plts.gca().spines['left'].set_visible(False)
plts.gca().spines['right'].set_visible(False)
plts.gca().spines['bottom'].set_visible(False)
plts.gca().spines['top'].set_visible(False)
for a_column in contributors:
    fig_4.add_subplot(4,4,pos_counter)
    sns.scatterplot(d_set_f1[a_column],d_set_f1[affectors],alpha=0.5)
    plts.xlabel(a_column);plts.ylabel(affectors);
    pos_counter+=1
plts.show()
```





from the plots it is apparent that there is very low correlation among the fields this is also viewed from the previously explored heatmap in EDA:4

Standardisation of Data

In [75]:

```
# Standardisation and Normalization of the data
scaler=StandardScaler()
scaled=scaler.fit_transform(full_X)
full_X=scaled
#full_X=normalize(full_X)
```

Training on Models

Part:1 Splitting Data

In [76]:

```
# Training on Models
#Part:1 Splitting
trainX,testX,trainY,testY=train_test_split(full_X,full_Y,test_size=0.21,random_state=52)
print(' Training set size for Independent variables  : ',trainX.shape);print('_'.center(
70,'_'));
print(' Training set size for Dependent variables    : ',trainY.shape);print('_'.center(
70,'_'));
print(' Testing set size for Independent variables   : ',testX.shape);print('_'.center(7
0,'_'));
print(' Testing set size for Dependent variables     : ',testY.shape);print('_'.center(7
0,'_'));
```

```
 Training set size for Independent variables  :  (44946, 15)
______________________________________________________________________
 Training set size for Dependent variables    :  (44946,)
______________________________________________________________________
 Testing set size for Independent variables   :  (11948, 15)
______________________________________________________________________
 Testing set size for Dependent variables     :  (11948,)
______________________________________________________________________
```

Part:2 Model Creation (Multiple models)

Regression is a statistical method used in finance, investing, and other disciplines that attempts to determine the strength and character of the relationship between one dependent variable (usually denoted by Y) and a series of other variables (known as independent variables).

Linear regression

The representation is a linear equation that combines a specific set of input values (x) the solution to which is the predicted output for that set of input values (y). As such, both the input values (x) and the output value are numeric.

The linear equation assigns one scale factor to each input value or column, called a coefficient and represented

by the capital Greek letter Beta (β). One additional coefficient is also added, giving the line an additional degree of freedom (e.g. moving up and down on a two-dimensional plot) and is often called the intercept or the bias coefficient.

For example, in a simple regression problem (a single x and a single y), the form of the model would be:

$$y = \beta_0 + \beta_1 * x$$

Ridge regression

Ridge regression is a way to create a parsimonious model when the number of predictor variables in a set exceeds the number of observations, or when a data set has multicollinearity (correlations between predictor variables).
Ridge regression belongs a class of regression tools that use L2 regularization. The other type of regularization, L1 regularization, limits the size of the coefficients by adding an L1 penalty equal to the absolute value of the magnitude of coefficients. This sometimes results in the elimination of some coefficients altogether, which can yield sparse models. L2 regularization adds an L2 penalty, which equals the square of the magnitude of coefficients. All coefficients are shrunk by the same factor (so none are eliminated). Unlike L1 regularization, L2 will not result in sparse models.
Ridge regression uses a type of shrinkage estimator called a ridge estimator. Shrinkage estimators theoretically produce new estimators that are shrunk closer to the "true" population parameters. The ridge estimator is especially good at improving the least-squares estimate when multicollinearity is present.

Bayesian Ridge

In the Bayesian viewpoint, we formulate linear regression using probability distributions rather than point estimates. The response, y, is not estimated as a single value, but is assumed to be drawn from a probability distribution. The model for Bayesian Linear Regression with the response sampled from a normal distribution is:

$$y \sim N(\beta^T X, \sigma^2 I)$$

The output, y is generated from a normal (Gaussian) Distribution characterized by a mean and variance. The mean for linear regression is the transpose of the weight matrix multiplied by the predictor matrix. The variance is the square of the standard deviation σ (multiplied by the Identity matrix because this is a multi-dimensional formulation of the model).

The aim of Bayesian Linear Regression is not to find the single "best" value of the model parameters, but rather to determine the posterior distribution for the model parameters. Not only is the response generated from a probability distribution, but the model parameters are assumed to come from a distribution as well.

Support Vector Machine

In machine learning, Support Vector Machines are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. In Support Vector Regression, the straight line that is required to fit the data is referred to as hyperplane.
The objective of a support vector machine algorithm is to find a hyperplane in an n-dimensional space that distinctly classifies the data points. The data points on either side of the hyperplane that are closest to the hyperplane are called Support Vectors. These influence the position and orientation of the hyperplane and thus help build the SVM.

Decision Tree Algorithm

Decision Tree is one of the most commonly used, practical approaches for supervised learning. It can be used to solve both Regression and Classification tasks with the latter being put more into practical application. It is a tree-structured classifier with three types of nodes. The Root Node is the initial node which represents the entire sample and may get split further into further nodes. The Interior Nodes represent the features of a data set and the branches represent the decision rules. Finally, the Leaf Nodes represent the outcome.

Random Forest

A random forest is a machine learning technique that's used to solve regression and classification problems. It utilizes ensemble learning, which is a technique that combines many classifiers to provide solutions to complex problems. A random forest algorithm consists of many decision trees. The 'forest' generated by the random forest algorithm is trained through bagging or bootstrap aggregating. Bagging is an ensemble meta-algorithm that improves the accuracy of machine learning algorithms.

The (random forest) algorithm establishes the outcome based on the predictions of the decision trees. It predicts by taking the average or mean of the output from various trees. Increasing the number of trees increases the precision of the outcome. A random forest eradicates the limitations of a decision tree algorithm. It reduces the overfitting of datasets and increases precision. It generates predictions without requiring many configurations in packages (like scikit-learn).

Features of a Random Forest Algorithm

- **It's more accurate than the decision tree algorithm.**
- **It provides an effective way of handling missing data.**
- **It can produce a reasonable prediction without hyper-parameter tuning.**
- **It solves the issue of overfitting in decision trees.**
- **In every random forest tree, a subset of features is selected randomly at the node's splitting point.**

In [77]:

```
#Part:2 Model Creation (Multiple models)
model_list="""
        Linear Regression         : LR
        Ridge Regression          : R2
        Bayesian Ridge Regression : BR2
        Support Vector Regression : SVR
        Decision Tree Regression  : DTR
        Random Forest Regression  : RFT
        """
print('Creating models for : \n',model_list);print('_'.center(70,'_'));
models={'LR':LR(),'R2':R2(),'BR2':BR2(),'SVR':SVR(),'DTR':DTR(),'RFT':RFT()}
```

```
Creating models for :

        Linear Regression         : LR
        Ridge Regression          : R2
        Bayesian Ridge Regression : BR2
        Support Vector Regression : SVR
        Decision Tree Regression  : DTR
        Random Forest Regression  : RFT

______________________________________________________________________
```

Part:3 Training Data on Models (Multiple Models)

In [78]:

```
#Part:3 Training Data on Models (Multiple models)
print(' Model Training Status : ')
for key in models:
    models[key].fit(trainX,trainY)
    print('\tTrained Model : ',key)
print('_'.center(70,'_'));
```

```
 Model Training Status :
 Trained Model :  LR
 Trained Model :  R2
 Trained Model :  BR2
 Trained Model :  SVR
 Trained Model :  DTR
 Trained Model :  RFT
```

Part:4 Prediction tests from the Trained Models

In [79]:

```
#Part:4 Prediction tests from the Trained Models
predicts=dict.fromkeys(models.keys())
print(' Prediction Generation Status : ')
for key in models:
    predicts[key]=models[key].predict(testX)
    print('Prediction generated for : ',key)
print('_'.center(70,'_'));
#print('\n');input('Press any key to continue . . .');print('x'.center(70,'*'))
```

```
 Prediction Generation Status :
Prediction generated for :  LR
Prediction generated for :  R2
Prediction generated for :  BR2
Prediction generated for :  SVR
Prediction generated for :  DTR
Prediction generated for :  RFT
```

Part:5 Determining accuracy levels of models from the prediction

In [80]:

```
#Part:5 Determining accuracy levels of models from the predictions
for key in models:
    print("Score for %s :: "%key,models[key].score(testX,predicts[key]))
print('_'.center(70,'_'));
```

```
Score for LR ::  1.0
Score for R2 ::  1.0
Score for BR2 ::  1.0
Score for SVR ::  1.0
Score for DTR ::  1.0
Score for RFT ::  1.0
```

Error Study

In [102]:

```
from sklearn.metrics import mean_squared_error,mean_absolute_error,median_absolute_error,
r2_score
print("Model\t|\tMean Squared Error\t|\tMean Absolute Error\t|\tRoot Mean Squared Error\
t|\tMedian Absolute Error\t|\t\tR\xb2\t\t|\tAdjusted R\xb2");
print('-'.center(200,'-'))
for key in models:
    print(key,mean_squared_error(predicts[key],testY),
          mean_absolute_error(predicts[key],testY),np.sqrt(mean_squared_error(predicts[k
ey],testY)),
          median_absolute_error(predicts[key],testY),r2_score(predicts[key],testY),
          (1-(1-r2_score(predicts[key],testY))*(len(testY)-1)/(len(testY)-testX.shape[1]
-1)),sep='\t|\t')

print('_'.center(70,'_'));
print('x'.center(70,'*'))
```

```
Model | Mean Squared Error | Mean Absolute Error | Root Mean Squared Error | Median Absol
ute Error |  R²  | Adjusted R²
-------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------
-------------------------
LR | 1.797710233221451 | 0.9978717733042176 | 1.340787169248517 | 0.751112397854587 | -1.
416542990611756 | -1.419580884079672
R2 | 1.797613095678012 | 0.9984076087587764 | 1.340750944686600 | 0.752899586897160 |
```

```
-1.4544267287565686 | -1.4575122467695882
BR2 | 1.797688215712366 | 0.9979198658144208 | 1.340778958558183 | 0.7509622874887878 | -
1.4200687539359333 | -1.4231110797244884
SVR | 1.0267920612184704 | 0.7255848442136484 | 1.0133074860171865 | 0.5035623800403544 |
0.3359522258367066 | 0.3351174356412281
DTR | 0.9741211073429145 | 0.5478989966661086 | 0.986975737970754 | 0.2061700000000002 |
0.6117523655600701 | 0.6112642902569693
RFT | 0.43011816321762775 | 0.39076607208237724 | 0.6558339448500876 | 0.1970007750000011
3 | 0.7744326071472182 | 0.7741490410314964
```

**********x**********

from the detailed study of the errors generated in the prediction we can at least conclude that the Random Forest Ensemble Regressor is better than other classifies, but it is to be made sure none of the machine learning models are perfect and more development of their parameters could be made making them further intuitive of their predictions.

End Notes

Inferrencing from the above studies it must be mentioned that this project just skims the surface of what machine learning could do. But it also gives insight of the limitations of each models of machine learning. Also we derived our conclusions from average rating only, factors like invocation of new gamers to the board game field might also determine a board game's success. A Board Game is either like a purist's fantasy or a dinosaur's past time enjoyment in the world of video games of today. Still with the pandemic looming it created a way to discover the long lost origins of Games

Bibliography

- https://www.definitions.net/definition/board+game
- https://www.forbes.com/sites/erikkain/2012/04/19/are-board-games-better-than-video-games/?sh=45f23b96348f
- http://www.kotaku.com/5903243/board-games-are-better-than-video-games-in-so-many-ways
- https://blog.eyewire.org/video-games-vs-board-games-a-strategic-match-up/
- https://www.investopedia.com/terms/r/regression.asp
- https://machinelearningmastery.com/linear-regression-for-machine-learning/
- https://www.statisticshowto.com/ridge-regression/
- https://towardsdatascience.com/introduction-to-bayesian-linear-regression-e66e60791ea7
- https://towardsdatascience.com/unlocking-the-true-power-of-support-vector-regression-847fd123a4a0
- https://towardsdatascience.com/machine-learning-basics-decision-tree-regression-1d73ea003fda
- https://towardsdatascience.com/machine-learning-basics-random-forest-regression-be3e1e3bb91a

©Bishal Biswas(@WolfDev8675) _(b.biswas*94587@ieee.org)*