Exploratory Data Analysis for Machine Learning

Rutwik Patel

About the data

- The data originally came from the Board Game Geek database, including 90,000+ board games, their description, and ratings.
- This data set was collected by R for Data Science (R4DS) - Online Learning Community and posted on their GitHub in March 2019. The .csv file can be found in Tidy Tuesday repository.
- R4DS selected games that have at least 50 ratings and were published between 1950 and 2016. The final data set has 10,532 rows and 22 columns.
- The data were split before this analysis: 80% train and 20% test

Data dictionary

Variable name	Type	Description
game_id	integer	Unique game identifier
description	string	A paragraph of text describing the game
image	string	URL image of the game
max_player	integer	Maximum recommended players
max_playtime	integer	Maximum recommended playtime (min)
min_age	integer	Minimum recommended age
min_players	integer	Minimum recommended players
min_playtime	integer	Minimum recommended playtime (min)
name	string	Name of the game

playing_time	integer	Average playtime
thumbnail	string	URL thumbnail of the game
year_published	integer	Year game was published
artist	string	Artist for game art
category	string	Categories for the game (separated by commas)
compilation	string	If part of a multi-compilation - name of compilation
designer	string	Game designer
expansion	string	If there is an expansion pack - name of expansion
family	string	Family of game - equivalent to a publisher
mechanic	string	Game mechanic - how game is played, separated by comma
publisher	string	Company/person who published the game, separated by commas
average_rating	float	Average rating on Board Games Geek (1-10)
users_rated	integer	Number of users that rated the game

Data exploration plan

This analysis is the initial step in an attempt to build a baseline model to predict game average ratings based on their characteristics.

1. Data Overview
2. Data Cleaning and Feature Engineering: Categorical Data
3. Data Cleaning and Feature Engineering: Numeric Data
4. Hypothesis Testing

Data overview

- The train set has 8,425 rows and 22 columns
- There are missing data only in most of the categorical variables

game_id	0
year_published	0
average_rating	0
playing_time	0
name	0
min_playtime	0
users_rated	0
min_age	0
max_playtime	0
max_players	0
description	0
min_players	0
image	1
thumbnail	1
publisher	2
category	79
designer	94
mechanic	751
artist	2238
family	2255
expansion	6236
compilation	8103

Categorical data

1. Data Cleaning:

- Remove features that are not useful to discriminate the target: *description*, *image*, *name*, *thumbnail*, *family*, *expansion*, and *compilation*
- Also remove *game_id*

	count	unique	top	freq
description	8425	8423	How could that have happened? Black Stories ar...	2
image	8424	8422	//cf.geekdo-images.com/images/pic2262580.png	2
name	8425	8314	Robin Hood	5
thumbnail	8424	8422	//cf.geekdo-images.com/images/pic2410035_t.png	2
artist	6187	3881	Franz Vohwinkel	141
category	8346	3310	Wargame,World War II	364
compilation	322	269	Traveller: The Classic Games, Games 1-6+	6
designer	8331	3978	(Uncredited)	442
expansion	2189	2106	Règlement de l'An XXX,Regulations of the Year ...	7
family	6170	3321	Crowdfunding: Kickstarter	312
mechanic	7674	2708	Hex-and-Counter	406
publisher	8423	4538	GMT Games	140

Categorical data

2. Feature engineering:

Counts derived from category aggregates

- Each columns have multiple values that are separated by commas
- Extract unique values and print out total number of these values for each column
- Derive new features that count number of artists, designers, and publishers of each game
- Remove columns: *artists*, *designer,* and *publisher*
- Remove rows that have missing values

```
Number of unique values of artist:        5416
Number of unique values of category:      83
Number of unique values of designer:      4476
Number of unique values of mechanic:      51
Number of unique values of publisher:     3045
```

Categorical data

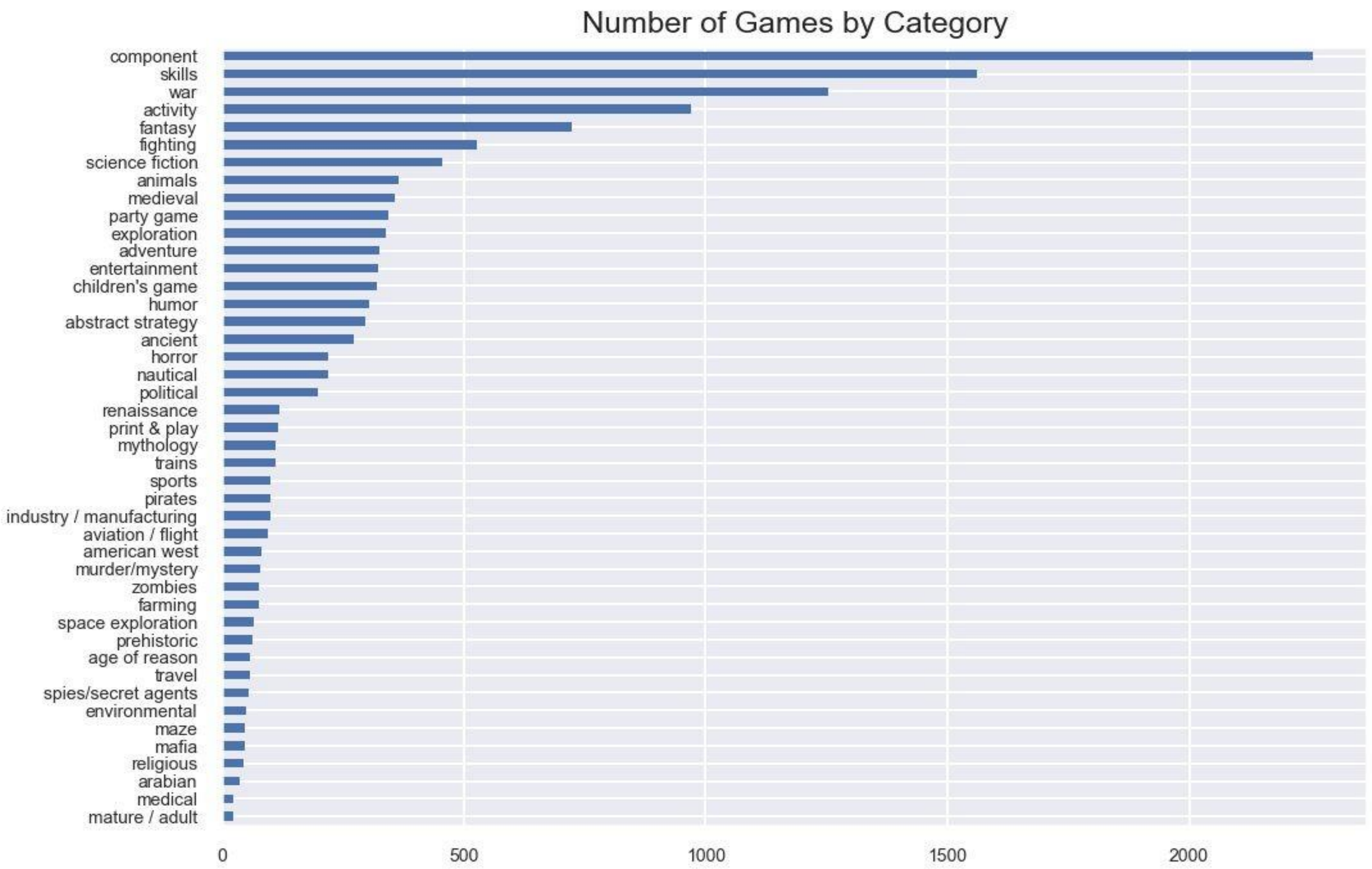
Categories derived from category aggregates

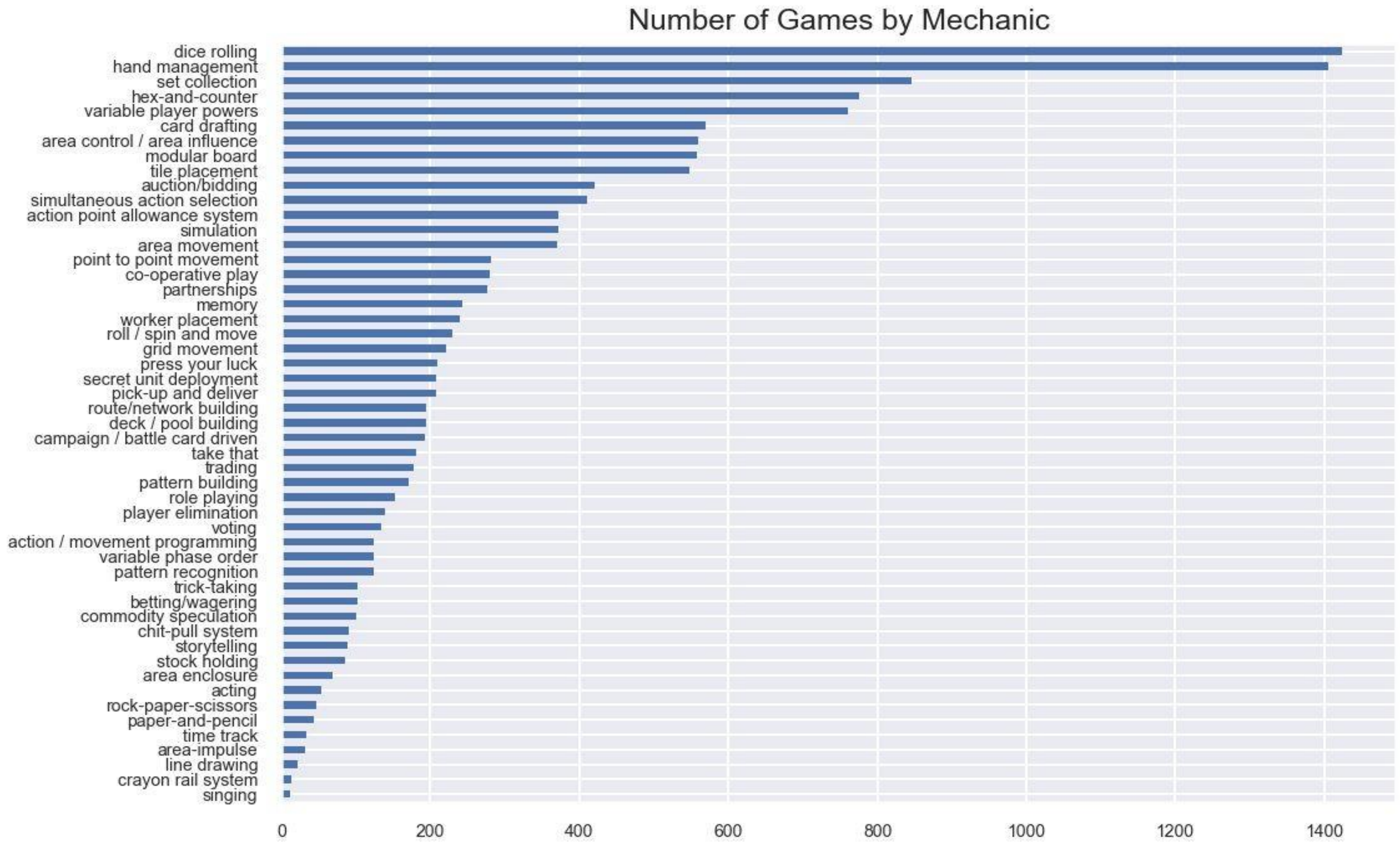
- Get a set of all unique values in each variable
- Create new columns based on these values
- Iterate through all rows and fill in dummy values for each new column
- Group these dummy variables if possible

Note: One game can be assigned to more than one category/ mechanic

The next two pages represent bar plots of 44 game categories (grouped from 81 categories) and 51 game mechanics.

The data set now has 5,608 rows and 109 columns


Number of Games by Category
component
skills
war
activity
fantasy
fighting
science fiction
animals
medieval
party game
exploration
adventure
entertainment
children's game
humor
abstract strategy
ancient
horror
nautical
political
renaissance
print & play
mythology
trains
sports
pirates
industry / manufacturing
aviation / flight
american west
murder/mystery
zombies
farming
space exploration
prehistoric
age of reason
travel
spies/secret agents
environmental
maze
mafia
religious
arabian
medical
mature / adult
0
500
1000
1500
2000


Number of Games by Mechanic
dice rolling
hand management
set collection
hex-and-counter
variable player powers
card drafting
area control / area influence
modular board
tile placement
auction/bidding
simultaneous action selection
action point allowance system
simulation
area movement
point to point movement
co-operative play
partnerships
memory
worker placement
roll / spin and move
grid movement
press your luck
secret unit deployment
pick-up and deliver
route/network building
deck / pool building
campaign / battle card driven
take that
trading
pattern building
role playing
player elimination
voting
action / movement programming
variable phase order
pattern recognition
trick-taking
betting/wagering
commodity speculation
chit-pull system
storytelling
stock holding
area enclosure
acting
rock-paper-scissors
paper-and-pencil
time track
area-impulse
line drawing
crayon rail system
singing
0
200
400
600
800
1000
1200
1400

Numeric data

Data description

	max_players	max_playtime	min_age	min_players	min_playtime	playing_time	year_published	average_rating	users_rated	num_artist	num_category	num_designer	num_mechanic	num_publisher
count	5608.000000	5608.000000	5608.000000	5608.000000	5608.000000	5608.000000	5608.000000	5608.000000	5608.000000	5608.000000	5608.000000	5608.000000	5608.000000	5608.000000
mean	5.010521	105.758559	9.955599	2.059379	91.313302	105.758559	2004.717725	6.546314	1166.660663	2.203994	2.651926	1.411733	2.600927	2.824893
std	7.543777	866.538797	3.301289	0.674542	848.267125	866.538797	11.284651	0.775103	3548.581155	7.690679	1.300462	0.802652	1.501255	3.683774
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1951.000000	2.339400	50.000000	1.000000	1.000000	1.000000	1.000000	1.000000
25%	4.000000	30.000000	8.000000	2.000000	30.000000	30.000000	2001.000000	6.051200	100.000000	1.000000	2.000000	1.000000	1.000000	1.000000
50%	4.000000	45.000000	10.000000	2.000000	45.000000	45.000000	2009.000000	6.548855	237.000000	1.000000	2.000000	1.000000	2.000000	2.000000
75%	6.000000	90.000000	12.000000	2.000000	90.000000	90.000000	2013.000000	7.065962	755.250000	2.000000	3.000000	2.000000	3.000000	3.000000
max	362.000000	60000.000000	25.000000	9.000000	60000.000000	60000.000000	2016.000000	9.003920	67655.000000	510.000000	14.000000	11.000000	12.000000	92.000000

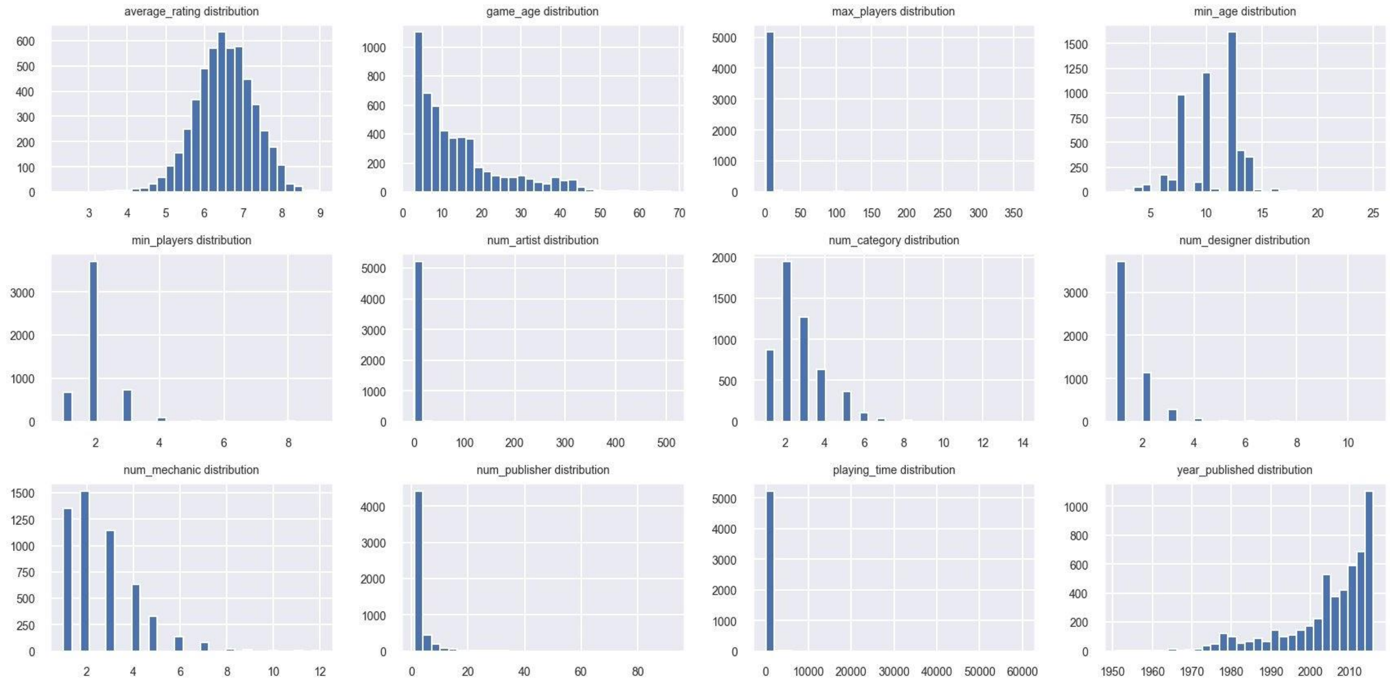
Numeric data

1. Data cleaning

- Derive *game_age* from *year_published*
- Remove *max_playtime*, *min_playtime*, and *users_rated*
- Select data that have non zero values

	max_players	min_age	min_players	playing_time	year_published	average_rating	num_artist	num_category	num_designer	num_mechanic	num_publisher	game_age
count	5240.000000	5240.000000	5240.000000	5240.000000	5240.000000	5240.000000	5240.000000	5240.000000	5240.000000	5240.000000	5240.000000	5240.000000
mean	5.102290	10.471756	2.070611	101.502481	2004.554008	6.525332	2.213550	2.663740	1.406870	2.624046	2.909542	14.445992
std	7.753493	2.441990	0.666585	858.286053	11.397775	0.765409	7.936809	1.316168	0.794444	1.512648	3.783639	11.397775
min	1.000000	2.000000	1.000000	1.000000	1951.000000	2.339400	1.000000	1.000000	1.000000	1.000000	1.000000	3.000000
25%	4.000000	8.000000	2.000000	30.000000	2000.000000	6.036300	1.000000	2.000000	1.000000	1.000000	1.000000	6.000000
50%	4.000000	10.000000	2.000000	45.000000	2009.000000	6.525335	1.000000	2.000000	1.000000	2.000000	2.000000	10.000000
75%	6.000000	12.000000	2.000000	90.000000	2013.000000	7.032408	2.000000	3.000000	2.000000	3.000000	3.000000	19.000000
max	362.000000	25.000000	9.000000	60000.000000	2016.000000	9.003920	510.000000	14.000000	11.000000	12.000000	92.000000	68.000000

Numeric data



Numeric data

- The target (*average_rating*) has a normal distribution
- Most features are right skewed
- Severe outliers

Numeric data

2. Feature engineering

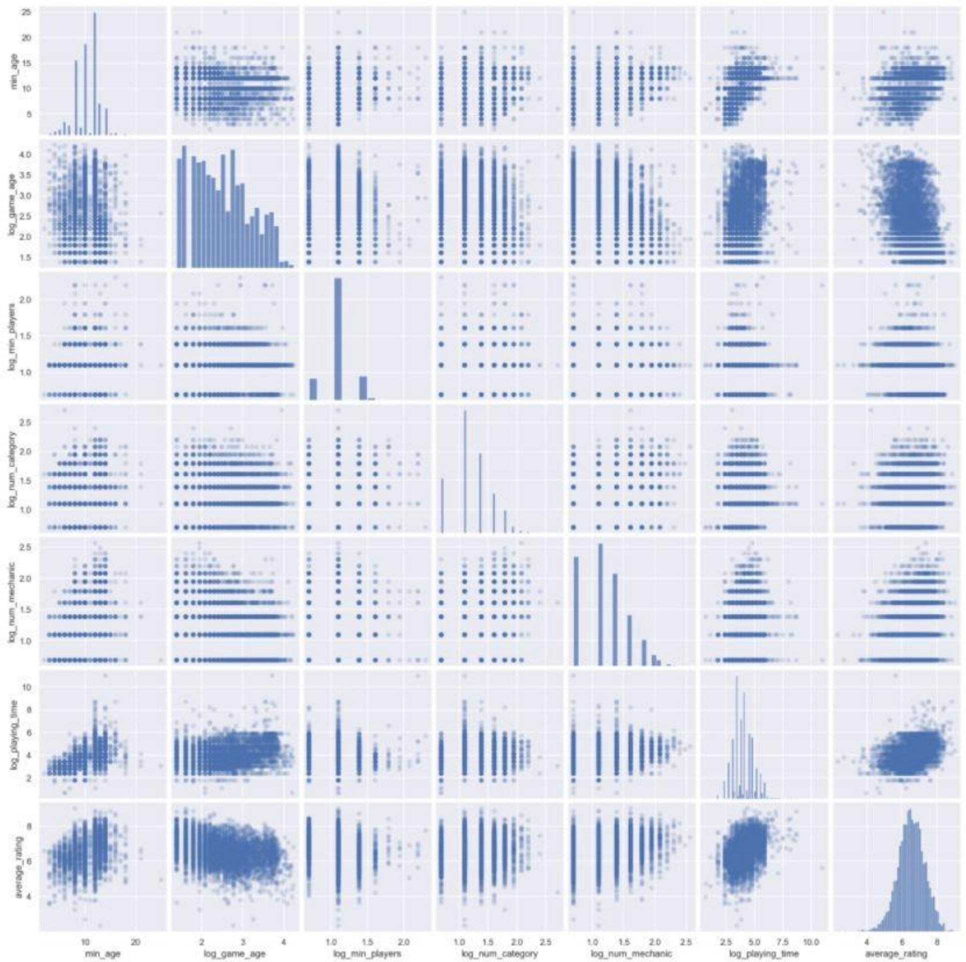
Log transformation for skewed variables

- Apply log transformation and check for skewness again.
- The result shows that log transformation does not work well for n*um_artist*, *num_designer*, *num_publisher*, and *year_published*

Next page present a pairplot of numeric features that have nearly normal distribution.

Numeric data

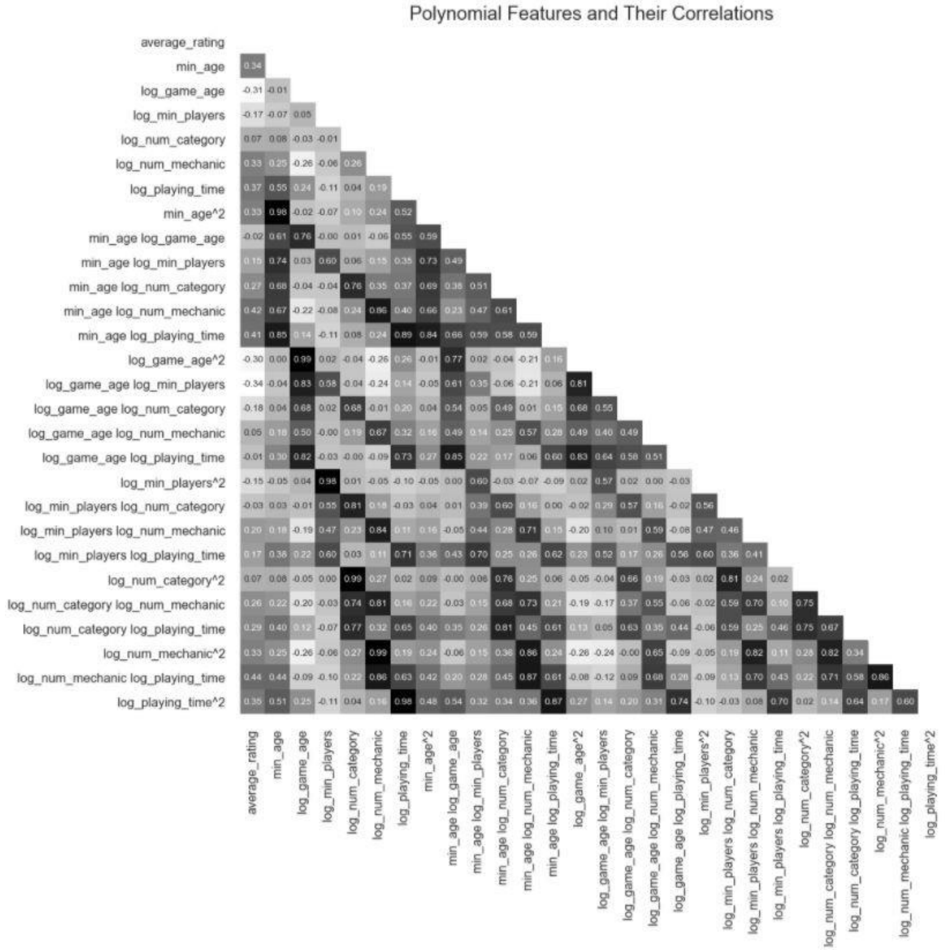
- No strong linear relationship between the features and the target. Linear regression might not be well-suited to this problem
- Might try adding polynomial and interaction terms and examine their correlation with the target



Numeric data

Adding polynomial and interaction terms

- This plot shows that polynomial and interaction terms do not have significantly higher correlations with the target comparing to the original features



Numeric data

Binning numeric data that cannot be scaled by log transformation

- These are *num_artist*, *num_designer*, *num_publisher*, and *year_published*
- Apply dummy transformation to these bins
- New columns from these bins: *group_artist_three_or_more*, *group_designer_three_or_more*, *group_max_players_five_or_six*, *group_max_players_seven_or_more*, *group_publisher_four_or_more*, *group_year_published_between_2001_and_2009*, *group_year_published_between_2010_and_2013*, and *group_year_published_between_2014_and_2016*

Remove the original columns after transformation (log and binning). The data set now has 5,240 rows and 131 columns

Hypothesis testing

- Main purpose: check if there are differences in average ratings between one group and others
- Due to different variances between two groups, Welch's t-test is used
- Perform multiple tests across all categories, mechanics, and groups (derived from numeric data)
- Sample of hypotheses:
    - H_0: War games and other games have similar ratings on average
    - H_a: There is a difference in average ratings between war games and other games

Hypothesis testing

- Result tables are shown in the next three pages. These values are sorted by p-values with colored bars (green for positive values and red for negative ones)
- For those that have p-value < 0.05 and |t-value| > 1.96, we reject the null hypotheses
- The sign of t-value suggests the direction of the test. A positive sign means that the group of interest has higher average ratings than others. On the contrary, a negative sign means that the group of interest has lower average ratings than others.

category_name	t-value	p-value
children's game	-15.841916	0.000000
war	13.893726	0.000000
component	-10.584794	0.000000
humor	-9.138182	0.000000
party game	-7.005245	0.000000
animals	-6.487482	0.000000
trains	4.741813	0.000006
renaissance	4.690531	0.000007
activity	4.241476	0.000024
space exploration	4.478895	0.000032
fighting	3.980679	0.000077
industry / manufacturing	4.088935	0.000090
age of reason	3.980063	0.000221
ancient	3.669080	0.000289
abstract strategy	-3.629330	0.000328
medieval	3.532035	0.000461
fantasy	3.358155	0.000818
farming	3.341165	0.001299
science fiction	2.937326	0.003463
nautical	2.900014	0.004100
maze	-3.009117	0.004285
pirates	-2.329630	0.021775
political	2.278630	0.023735
mythology	2.144308	0.034218
spies/secret agents	2.004349	0.050047
entertainment	-1.937038	0.053610
religious	1.763866	0.084909
print & play	1.709978	0.090178
aviation / flight	1.708230	0.091375
skills	-1.677003	0.093654
exploration	1.409116	0.159621
environmental	1.298456	0.200263
adventure	0.972847	0.331317
mature / adult	-0.791506	0.437057
arabian	-0.638325	0.527225
murder/mystery	0.600392	0.550026
horror	-0.465960	0.641699
sports	0.423544	0.672805
medical	0.424722	0.675144
travel	0.345949	0.730699
prehistoric	-0.332909	0.740358
american west	0.231495	0.817530
mafia	-0.115298	0.908744
zombies	-0.002223	0.998233

mechanic_name	t-value	p-value
area control / area influence	13.681888	0.000000
worker placement	12.531980	0.000000
simulation	11.917241	0.000000
variable player powers	11.257887	0.000000
deck / pool building	11.228836	0.000000
roll / spin and move	-10.953231	0.000000
action point allowance system	8.988847	0.000000
grid movement	9.043836	0.000000
dice rolling	8.376262	0.000000
hex-and-counter	7.678370	0.000000
card drafting	7.236475	0.000000
route/network building	7.371719	0.000000
campaign / battle card driven	7.395763	0.000000
variable phase order	7.407286	0.000000
pattern recognition	-7.046414	0.000000
area movement	6.606161	0.000000
co-operative play	6.594492	0.000000
trick-taking	-5.584741	0.000000
modular board	4.749033	0.000003
chit-pull system	5.020407	0.000003
crayon rail system	6.101341	0.000040
memory	-4.161645	0.000043
hand management	4.088635	0.000045
action / movement programming	4.203600	0.000050
pattern building	-4.014621	0.000088
stock holding	3.906744	0.000189
betting/wagering	-3.804758	0.000243
point to point movement	3.653005	0.000308
secret unit deployment	3.471693	0.000631
simultaneous action selection	3.098684	0.002062
tile placement	2.819089	0.004956
singing	-3.479737	0.005121
player elimination	2.795235	0.005905
set collection	-2.591833	0.009665
rock-paper-scissors	-2.586553	0.013116
acting	-2.472040	0.016953
partnerships	2.153588	0.032088
time track	2.053270	0.048476
role playing	1.959523	0.051902
paper-and-pencil	1.934789	0.060433
commodity speculation	1.693678	0.093399
storytelling	-1.496163	0.138264
auction/bidding	1.426467	0.154380
area-impulse	1.207543	0.239942
trading	-1.016744	0.310614
area enclosure	0.984194	0.328520
pick-up and deliver	0.741010	0.459464
voting	-0.721287	0.471988
take that	-0.422657	0.673047
press your luck	0.323775	0.746403
line drawing	0.253232	0.802941

Hypothesis testing

These tables show that on average:

- People like war games
- People do not like children’s games and component games
- People like games that use area control / area influence, worker placement, simulation, variable player powers, and deck / pool building
- People do not like games that use roll / spin and move mechanic
- People like games published between 2014 and 2016
- People like games that were designed by three or more artists

group_name	t-value	p-value
year_published_between_2014_and_2016	21.686049	0.000000
artist_three_or_more	10.349241	0.000000
year_published_between_2001_and_2009	-9.785744	0.000000
max_players_five_or_six	-9.068495	0.000000
publisher_four_or_more	6.625750	0.000000
year_published_between_2010_and_2013	5.153826	0.000000
designer_three_or_more	2.842000	0.004688
max_players_seven_or_more	-2.561497	0.010632

Hypothesis testing

- Since these features might have effects on each other, there need to be more analyses before jumping to a conclusion. For example, perhaps area control mechanic is mostly used in war games, or children's games are mostly played by rolling and spinning. War games might be more complex and need more artists to complete.

Further data engineering and analyzing

- Score game complexity by calculating weighted average of number of artists, number of designers, and number of publishers. Examine the relationship between this score and the target
- Reduce categorical data dimensionality and create interaction terms among them or with numeric data
- Apply mutual information regression for feature selection
- Apply Backward Stepwise Regression
- Build a pipeline to preprocess data and run the model on the test set

Conclusion

As shown in the analysis, linear regression might not be a good fit to this data set. However, it might be good enough as a baseline model. To collect a better dataset, one might request the Board Game Geek API and retrieve other features such as weight (complexity rating), number of reviews, or explore available data from Kaggle.

Jupyter Notebook for this analysis can be found here:
https://github.com/RutwikPatel13/Exploratory-Data-Analysis-for-Machine-Learning/blob/master/Project-1.ipynb