ML and Board Games

Ana Morais

overview

- **Objective:** Build a model to predict average rating of board games on BoardGameGeek website
- **Features (numerical continuous):**
  - Year of publication
  - Game weight
  - Min players
  - Max players
  - Playtime
  - Age recommendation
  - # expansions
- **Features (categorical):**
  - Kickstarted (y/n)
  - Game category
  - Game subcategory
  - Game Mechanics

Implementation details

X-y split

y = “avgrating”
X.shape = (21925,183)

Test-train split

70% training data (15347 rows)
30% test data (6578 rows)

def compare_models()

Metrics:

- R-squared (test and training), MAE, MSE and RMSE

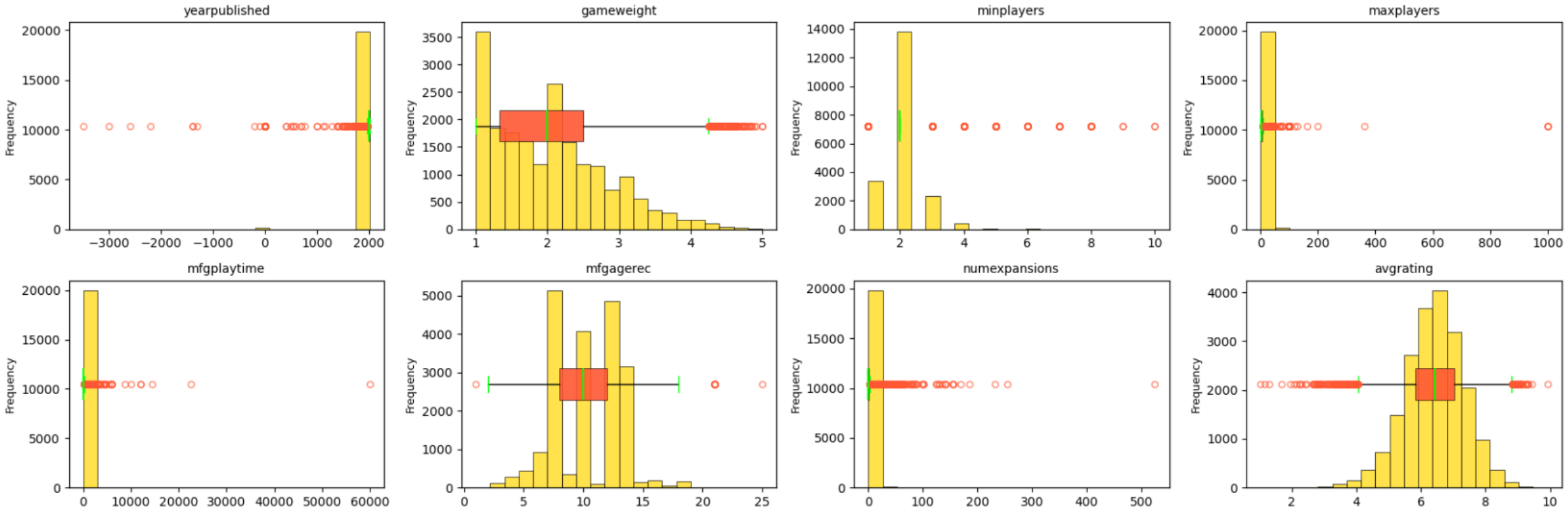
ML models (6):

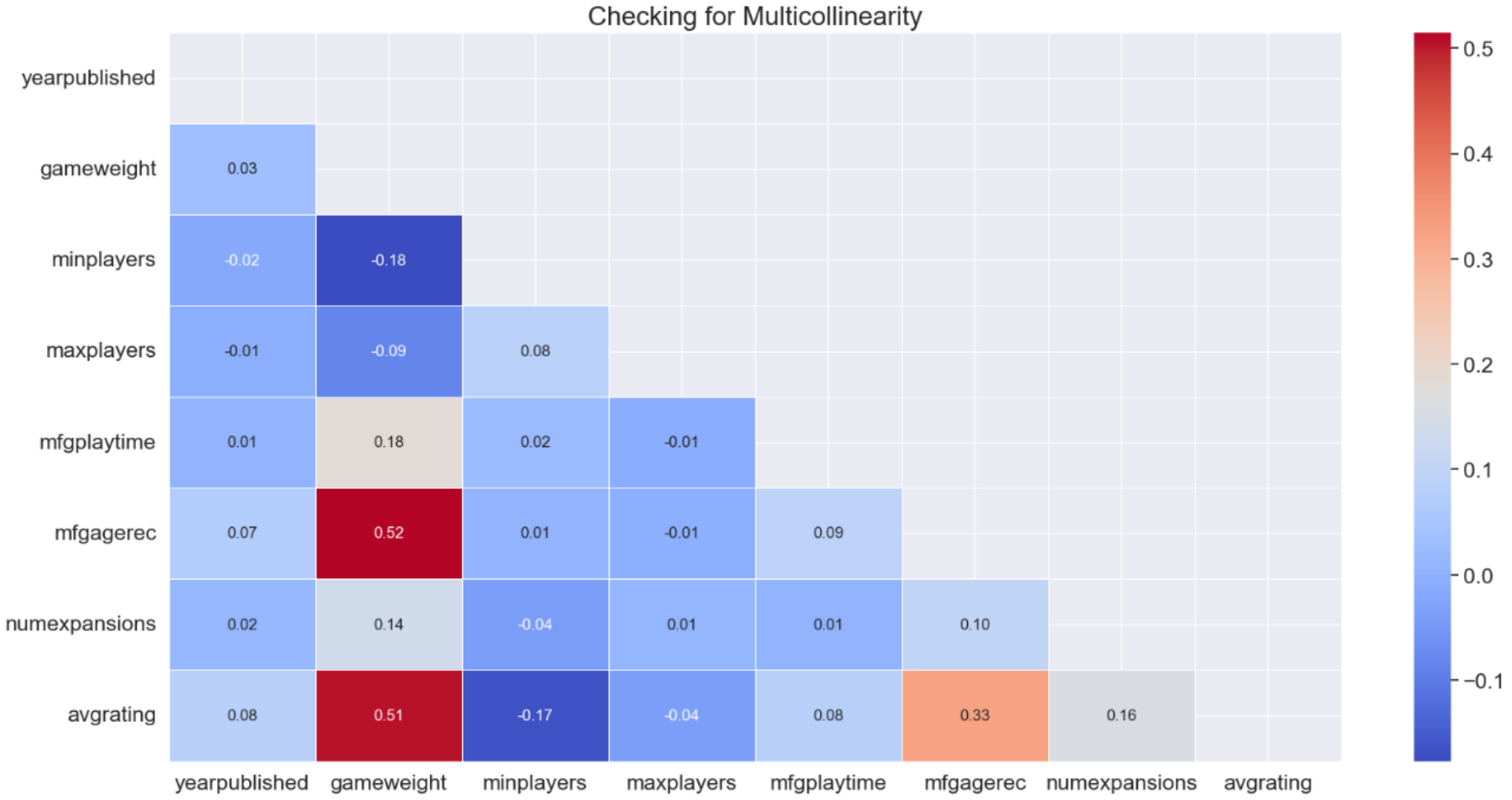
- Linear Regression, Ridge Regression, Lasso Regression, Decision Tree, KNN and XGBoost Regression

Model	R2	RMSE (Test)	RMSE (Train)	MSE (Test)	MAE (Test)
Linear Regression	0.406715	0.718534	0.700638	0.516291	0.543070
Ridge Regression	0.407593	0.718002	0.700666	0.515527	0.542599
Lasso Regression	0.021930	0.922573	0.906684	0.851140	0.722198
Decision Tree	0.111690	0.879220	0.008379	0.773029	0.658966
KNN Regression	0.365743	0.742931	0.585247	0.551946	0.560934
XGBoost Regression	0.492361	0.664650	0.631115	0.441759	0.496589

Frequency distributions

Numerical continuous features & target





Numerical continuous features & target

Categorical features

Checked for perfect multicollinearity (-1 | 1) – all good!

Top 5 positive correlations with target:

kickstarted	0.221434
solo_/_solitaire_game	0.214312
cat:strategy	0.209558
variable_player_powers	0.201060
miniatures	0.192414

Top 5 negative correlations with target:

roll_/_spin_and_move	-0.263116
cat:childrens	-0.199471
dexterity	-0.109082
electronic	-0.071736
card_game	-0.066718

StandardScaler & MinMaxScaler effects

Scaler	Model	R2	RMSE (Test)	RMSE (Train)	MSE (Test)	MAE (Test)
MinMaxScaler	Linear Regression	0.406715	0.718534	0.700638	0.516291	0.543070
StandardScaler	Linear Regression	0.406715	0.718534	0.700638	0.516291	0.543070
MinMaxScaler	Ridge Regression	0.407030	0.718343	0.700730	0.516017	0.542843
StandardScaler	Ridge Regression	0.406714	0.718535	0.700638	0.516292	0.543070
MinMaxScaler	Lasso Regression	-0.000205	0.932953	0.914426	0.870402	0.730522
StandardScaler	Lasso Regression	-0.000205	0.932953	0.914426	0.870402	0.730522
MinMaxScaler	Decision Tree	0.092045	0.888889	0.008379	0.790124	0.664204
StandardScaler	Decision Tree	0.093774	0.888043	0.008379	0.788620	0.662648
MinMaxScaler	KNN Regression	0.325762	0.765988	0.623200	0.586738	0.587247
StandardScaler	KNN Regression	0.294587	0.783497	0.631749	0.613867	0.599411
MinMaxScaler	XGBoost Regression	0.492361	0.664650	0.631115	0.441759	0.496589
StandardScaler	XGBoost Regression	0.492361	0.664650	0.631115	0.441759	0.496589

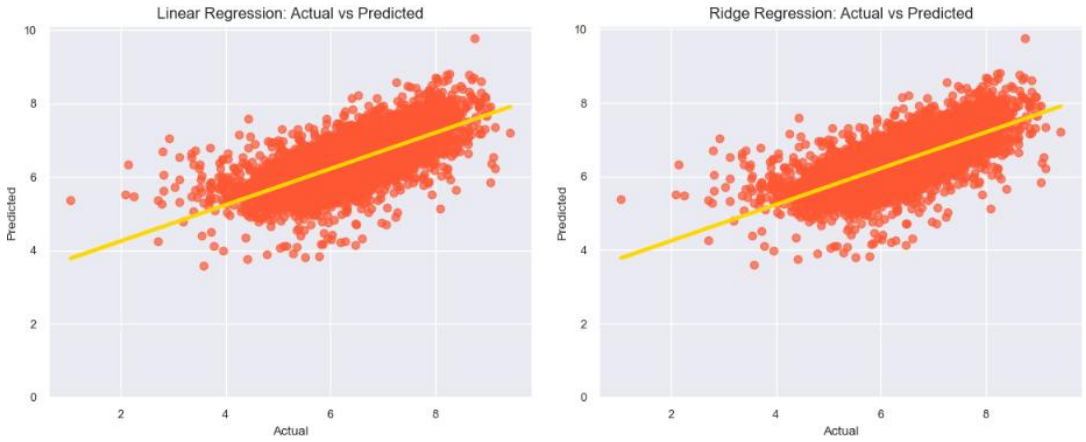
Model	R2	RMSE (Test)	RMSE (Train)	MSE (Test)	MAE (Test)
Linear Regression	0.406715	0.718534	0.700638	0.516291	0.543070
Ridge Regression	0.407593	0.718002	0.700666	0.515527	0.542599
Lasso Regression	0.021930	0.922573	0.906684	0.851140	0.722198
Decision Tree	0.111690	0.879220	0.008379	0.773029	0.658966
KNN Regression	0.365743	0.742931	0.585247	0.551946	0.560934
XGBoost Regression	0.492361	0.664650	0.631115	0.441759	0.496589

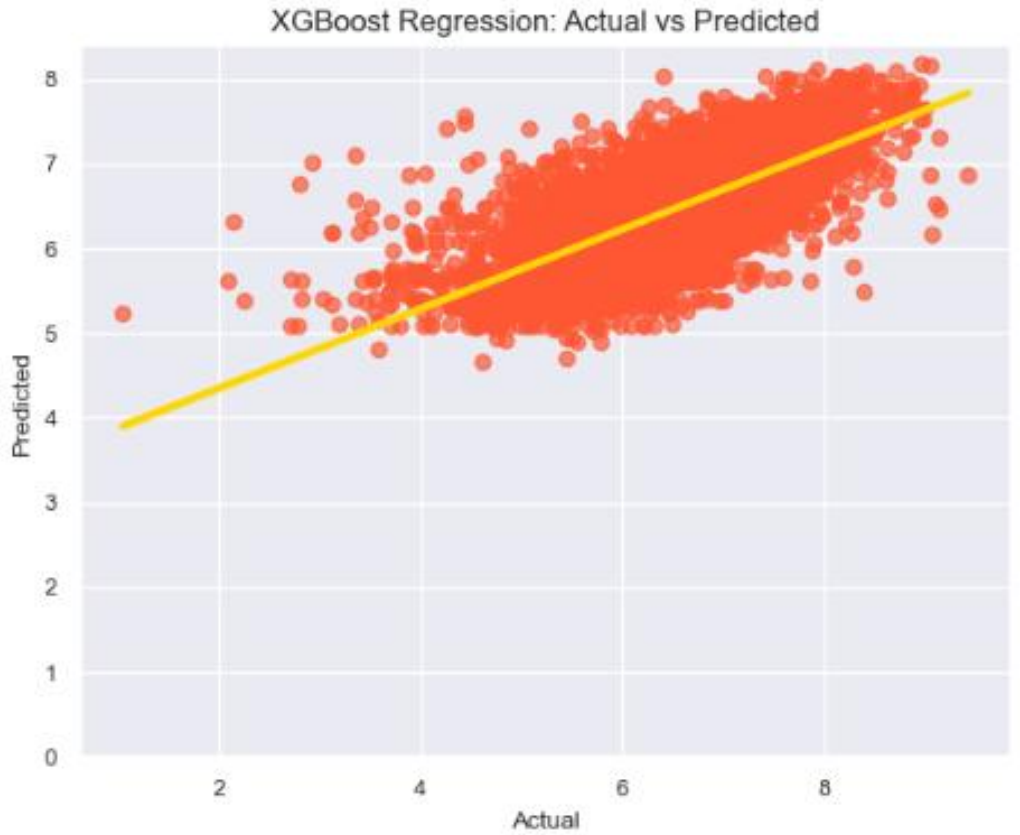
LogTransform

Model	R2	RMSE (Test)	RMSE (Train)	MSE (Test)	MAE (Test)
Linear Regression	0.490011	0.666186	0.647584	0.443804	0.491252
Ridge Regression	0.490617	0.665791	0.647606	0.443277	0.490862
Lasso Regression	-0.000205	0.932953	0.914426	0.870402	0.730522
Decision Tree	0.094388	0.887742	0.008379	0.788085	0.662931
KNN Regression	0.435768	0.700720	0.565257	0.491008	0.532330
XGBoost Regression	0.493974	0.663593	0.630758	0.440356	0.495603

Model	R2	RMSE (Test)	RMSE (Train)	MSE (Test)	MAE (Test)
Linear Regression	0.406715	0.718534	0.700638	0.516291	0.543070
Ridge Regression	0.407593	0.718002	0.700666	0.515527	0.542599
Lasso Regression	0.021930	0.922573	0.906684	0.851140	0.722198
Decision Tree	0.111690	0.879220	0.008379	0.773029	0.658966
KNN Regression	0.365743	0.742931	0.585247	0.551946	0.560934
XGBoost Regression	0.492361	0.664650	0.631115	0.441759	0.496589

Top3 ML models





XGBoost with log-transformed features

Time for some tuning

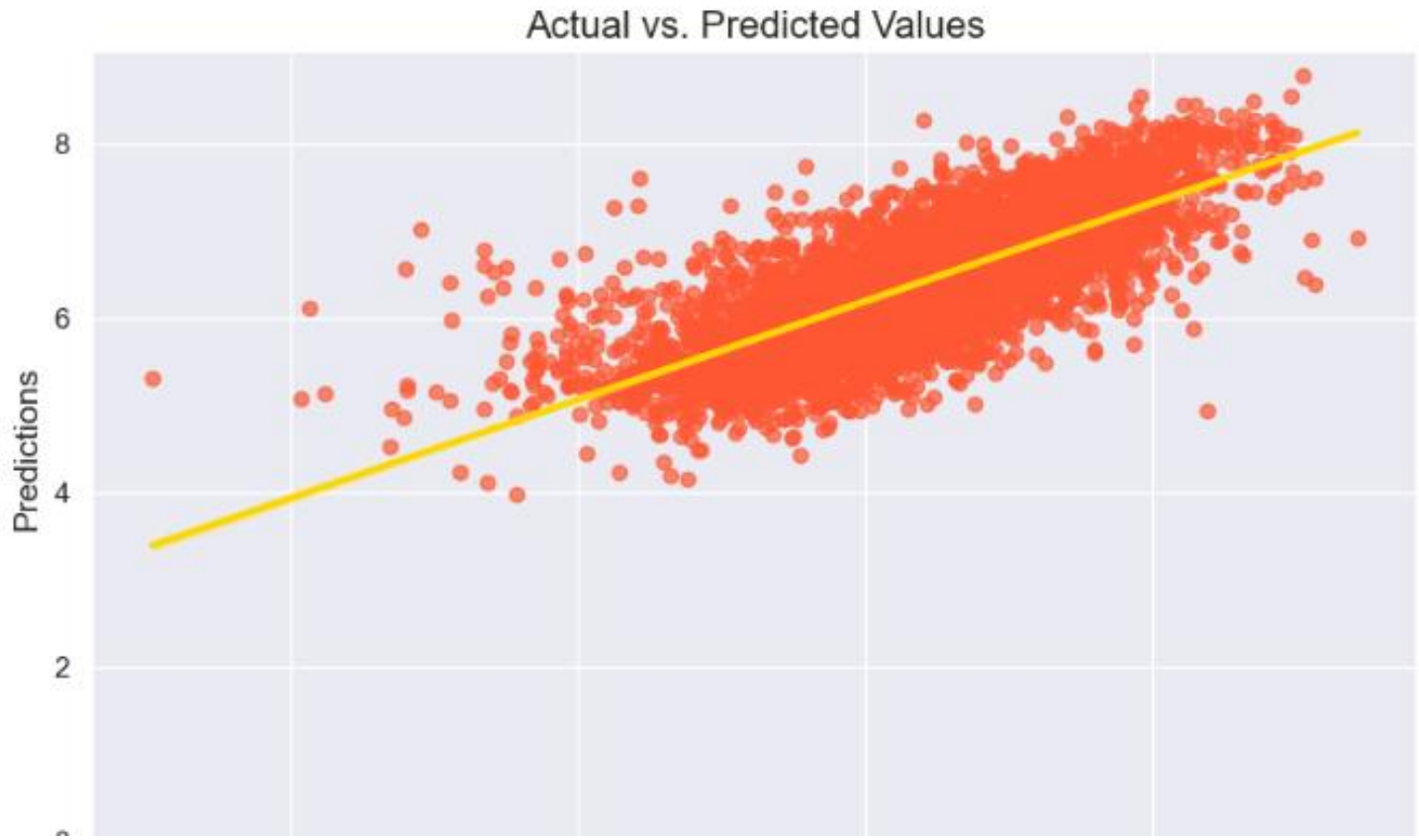
- **Method:** GridSearchCV
- Dictionary of parameters to test:
  - 'n_estimators': [100, 200, 300],
  - 'max_depth': [3, 5, 7],
  - 'learning_rate': [0.01, 0.05, 0.1],
  - 'subsample': [0.8, 1.0],
  - 'colsample_bytree': [0.8, 1.0],
- Goal:
  - Minimize MSE
- Result:

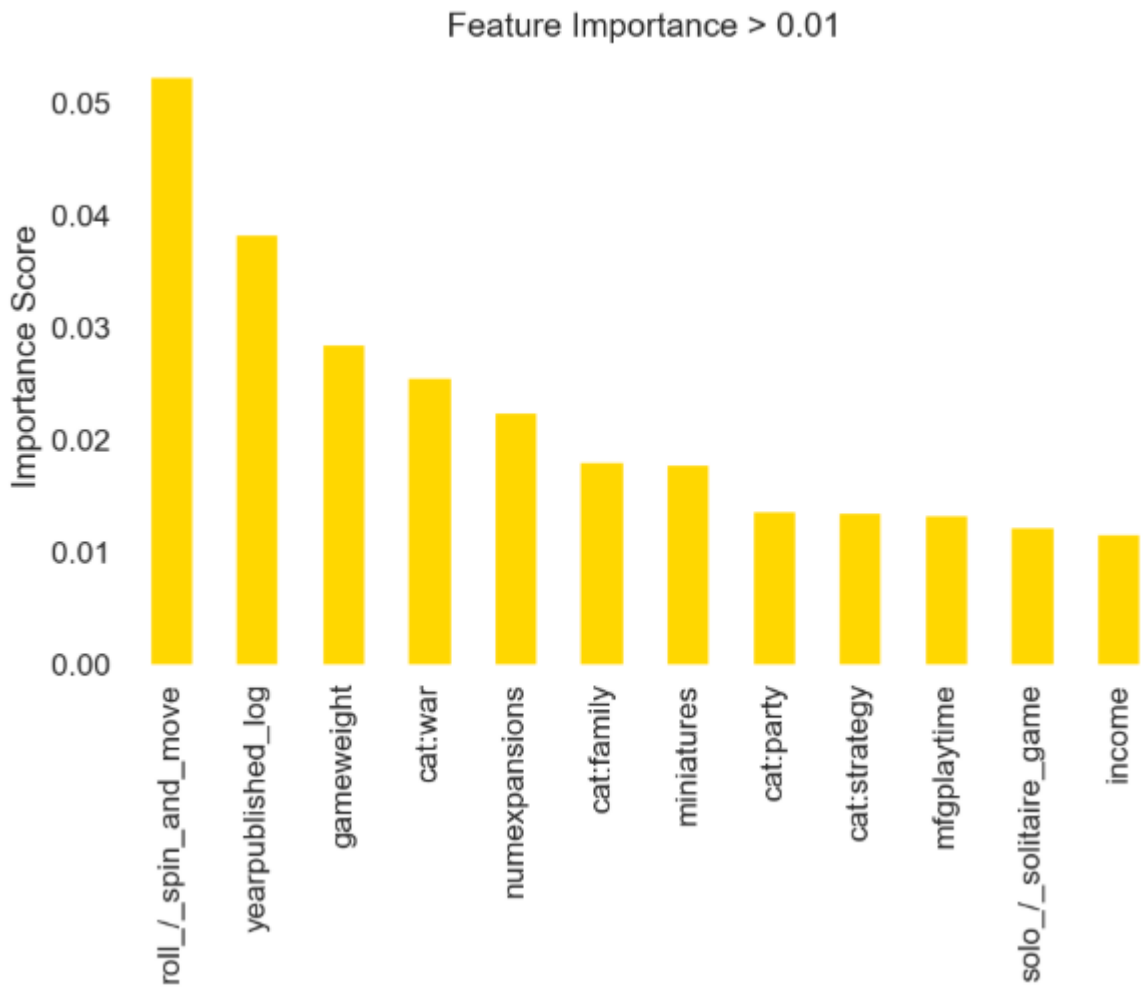
'n_estimantors': 300,
'max_depth': 7,
'learning_rate': 0.05,
'subsample': 0.8,
'colsample_bytree': 0.8

Best score:
MSE = 0.362881

R-squared =
0.567308 > 0.493974

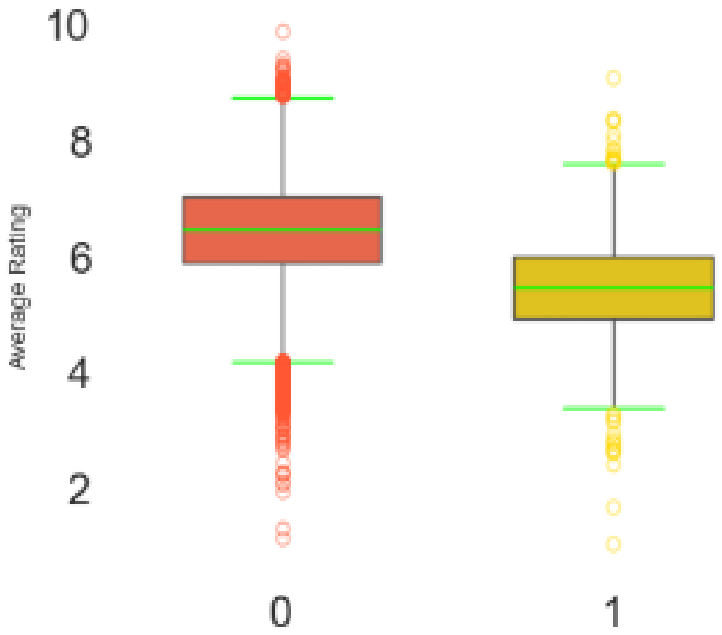
R-squared
0.567308



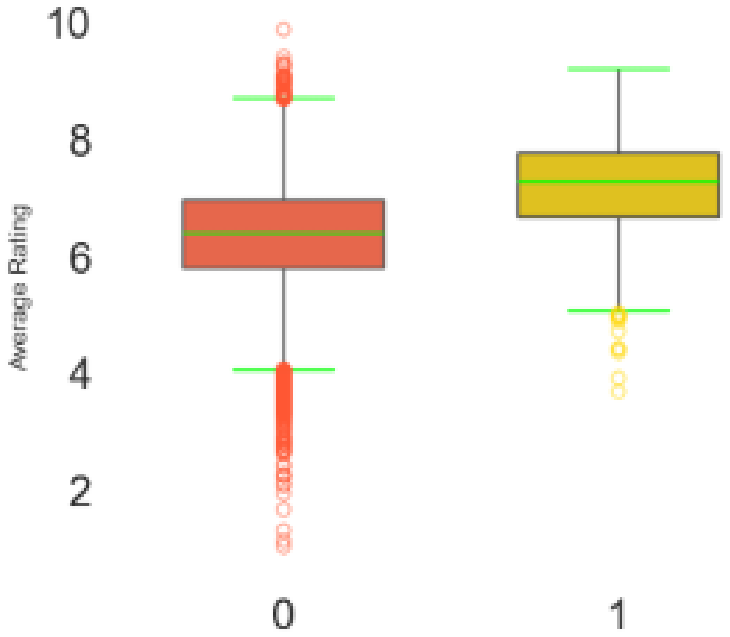


Top features

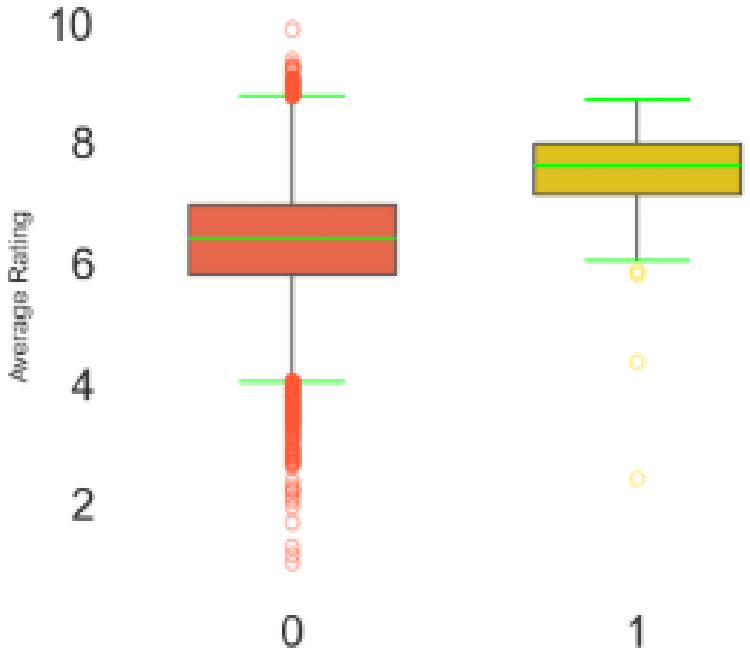
Roll Spin & Move



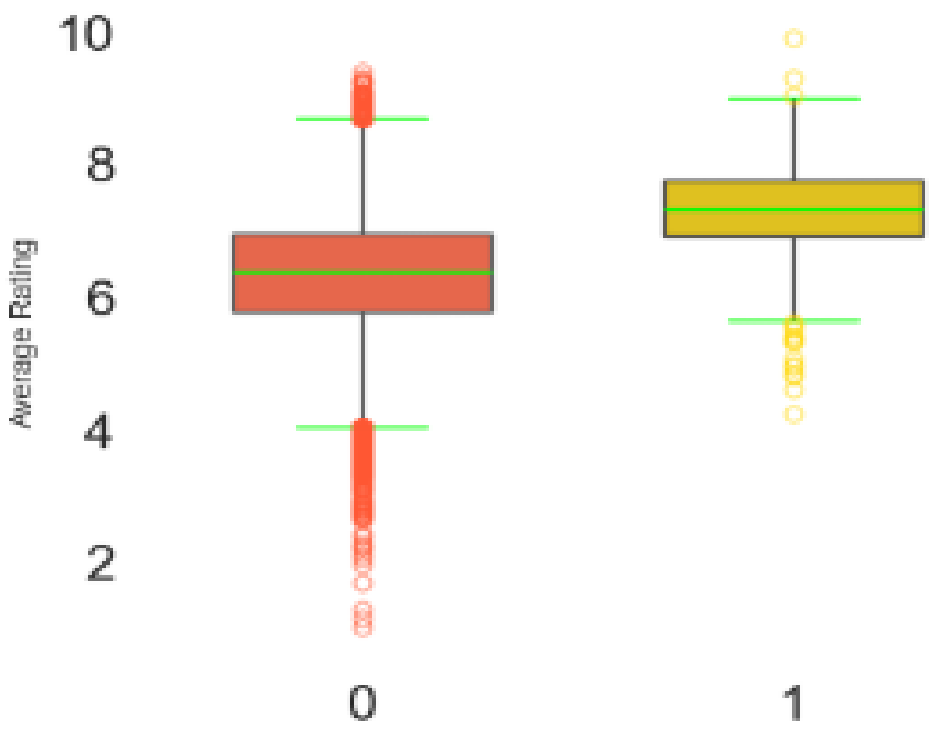
Miniatures



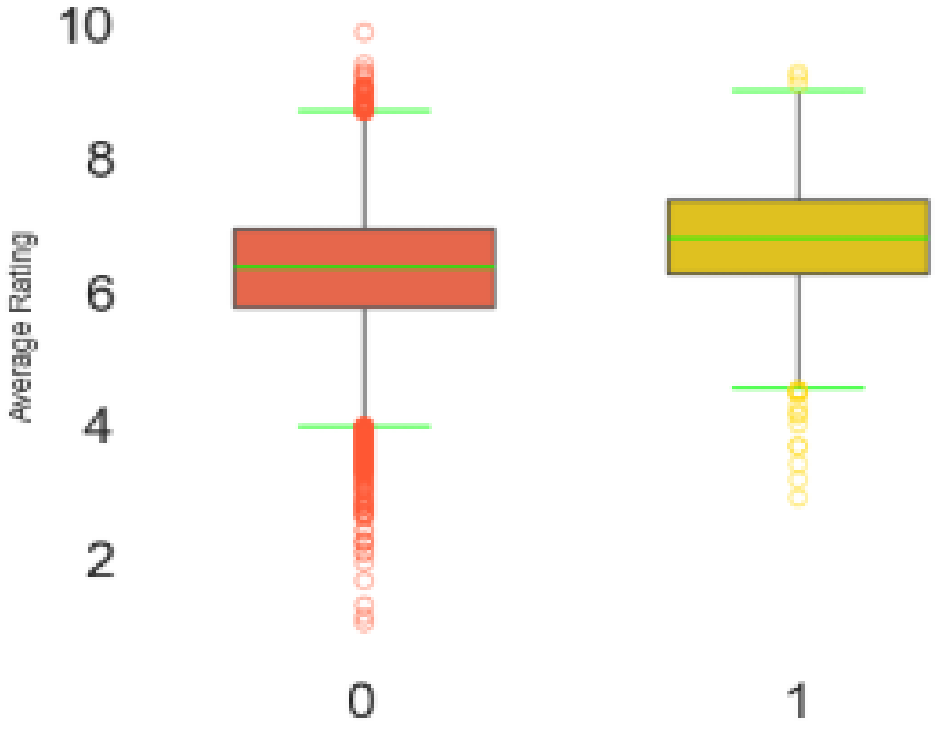
Income



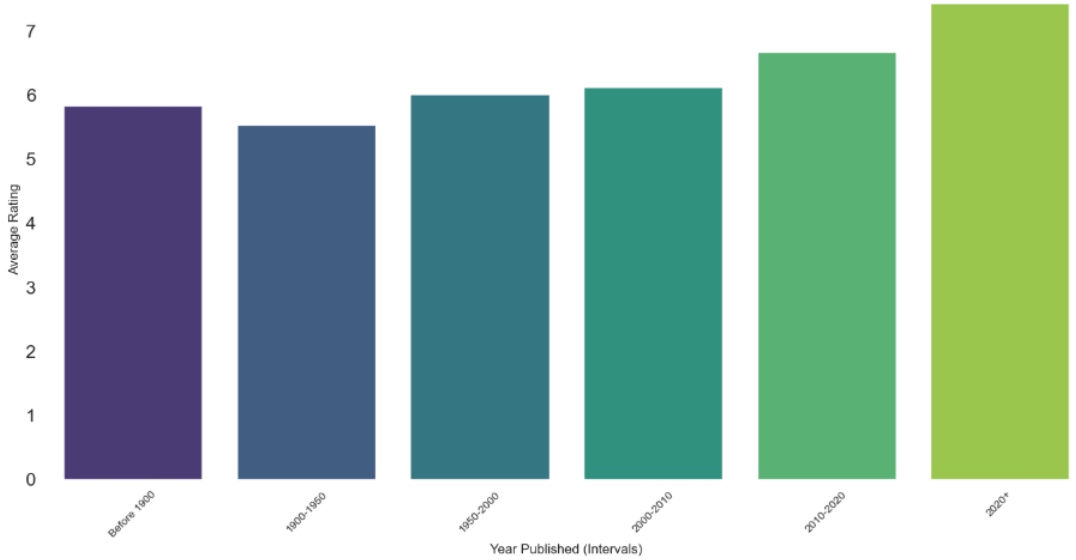
Solitaire games



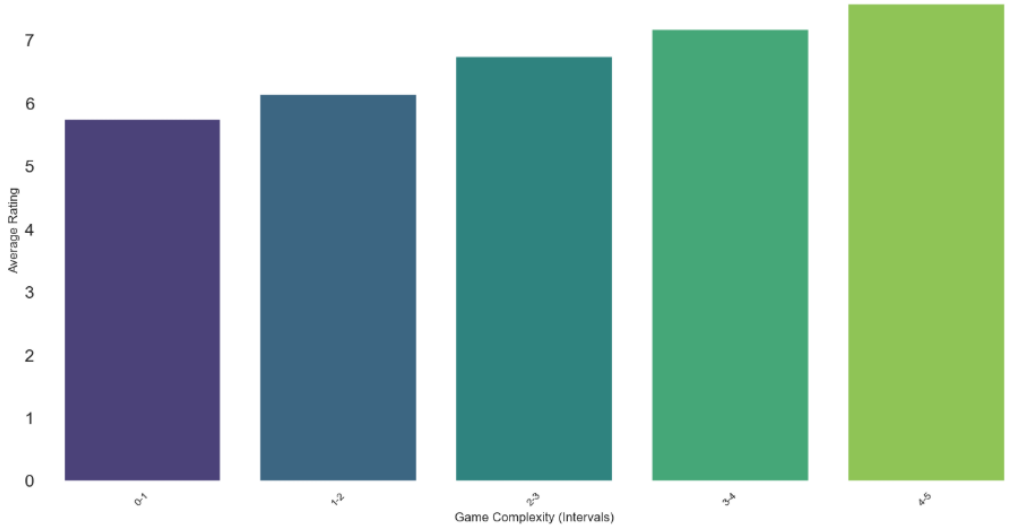
War games



Average Rating by Year of Publication



Average Rating by Game Complexity



Conclusions

Model Performance:	**Best Model**: XGBoost with log-transformed features. **Key Metrics**: •Mean Squared Error (MSE): 0.362881 •R-squared: 0.567308 (improved over baseline 0.493974). **Highlighted key features influencing ratings** (e.g., Game Complexity, Year of Publication and some categories and mechanics like War games, Miniatures, Income, Solo games and Roll, Spin & Move games).
2. Future Directions:	Explore tuning of other models like KNN Regression, Linear Regression and Decision trees to reduce overfitting. Statistical analysis for the effects of Top features on Average Ratings.
Main Takeaway:	Machine learning models can help predict average board game ratings, providing actionable insights for game developers and enthusiasts.

Thank you!