#  LEGO Price Prediction

**Abstract**

Lego is the largest toy manufacturer on the planet. New releases and advertising increasingly skew towards adult collectors, the prevalence of individuals reselling Lego for profit is rising, and the Lego Company is increasing its media partnerships and licenses. Recent news coverage from the Wall Street Journal even considers Lego an “investment opportunity.”

In this project, the rise of Lego as a giant in the toy industry is analyzed. Additionally, the viability of Lego for collectible investing and the ability of predictive machine learning models to determine winners and losers is explored. Examining the pricing and features for Lego products over the past fifty years, this project considers the ability for trends in pricing, piece count, consumer ratings, and nostalgia to account for Lego’s rise. Furthermore, this project provides insights into the ability of modern machine learning methods such as regression, ensemble, and deep learning models to confidently predict a Lego product’s current pricing.

**Data Collection, Preprocessing, Exploration**

To begin, online APIs and web scraping pricing data from online Lego storefronts were explored. Some immediate challenges identified were sourcing more data and preprocessing that data for analysis and training. Some APIs such as that of Bricklink were locked behind verified seller status on the site, effectively pay-walling some data sources. Furthermore, data was inconsistently recorded and noisy, as well as affected by a particular sites user-base. Since sets could be in various conditions (New, Used, missing pieces, etc.) pricing was only considered for new, un-opened sets in original packaging. This may introduce some bias towards untouched products sought after by collectors, but also allows the inclusion of many recently discontinued and currently offered products by Lego itself.

The "LEGO Sets and Prices Over Time" Kaggle dataset was used as a starting point, and cleaned and updated using data from APIs to well-known second-hand Lego websites such as Bricklink and Brickset. In total, the original Kaggle dataset was 1.78 MB of information on 14,936 different Lego sets beginning in 1975. From the Kaggle dataset description, features included “include name, year, theme, theme group, subtheme, category, packaging type, number of instructions, retail availability, number of pieces, number of minifigures, user reviews (out of 5 stars), list price in USD, and latest resale price in USD.” Merging this with another dataset from Oczkoś, Wiktor et al. created a larger dataset of over 600,000 entries that included additional features such as age-rating and online demand. This dataset and accompanying research paper were helpful for determining additional sources of data, but since many of the pricing details were specific to Poland/Europe and recorded in Polish złoty, much of the data was discarded. While it would be possible to convert prices to USD, the additional impact of geographical price discrimination lead to this decision.

The high number of fields of data and the amount of noise in each attribute was difficult to sort through, so much work went into preprocessing and data exploration. Data examples with missing values for Current Price were thrown out due to the absence of ground truth labels to learn from. Entries with missing or invalid values for *Rating* and *USD_MSR*P were also removed as these would likely be the two best predictors for Current Pricing. This cut down the merged dataset to 3,225 useful entries as described by Figure 1 below.

| | | | | | |
|---|---|---|---|---|---|
| 1 | Set_ID | 3225 non-null | object | Set_ID | 3225 |
| 2 | Name | 3225 non-null | object | Name | 3019 |
| 3 | Year | 3225 non-null | float64 | Year | 25 |
| 4 | Theme | 3225 non-null | object | Theme | 92 |
| 5 | Theme_Group | 3225 non-null | object | Theme_Group | 13 |
| 6 | Subtheme | 2791 non-null | object | Subtheme | 380 |
| 7 | Category | 3225 non-null | object | Category | 4 |
| 8 | Packaging | 3225 non-null | object | Packaging | 12 |
| 9 | Num_Instructions | 3225 non-null | float64 | Num_Instructions | 24 |
| 10 | Availability | 3225 non-null | object | Availability | 6 |
| 11 | Pieces | 3224 non-null | float64 | Pieces | 1095 |
| 12 | Minifigures | 2315 non-null | float64 | Minifigures | 23 |
| 13 | Owned | 3225 non-null | float64 | Owned | 2709 |
| 14 | Rating | 3225 non-null | float64 | Rating | 21 |
| 15 | USD_MSRP | 3225 non-null | float64 | USD_MSRP | 81 |
| 16 | Total_Quantity | 3225 non-null | float64 | Total_Quantity | 96 |
| 17 | Current_Price | 3225 non-null | float64 | Current_Price | 1402 |
| 18 | lastUpdated | 3158 non-null | object | lastUpdated | 3153 |
| 19 | US_retailPrice | 3158 non-null | float64 | US_retailPrice | 78 |
| 20 | ownedBy | 3158 non-null | float64 | ownedBy | 2659 |
| 21 | wantedBy | 3158 non-null | float64 | wantedBy | 2026 |

*Figure 1: Attributes of cleaned dataset and number of unique values.*

To begin exploration, multiple plots were created exploring relationships between attributes and the target current price. There were several positive correlations between attributes such as *Pieces*, *USD_MSRP*, *Rating*, and *Quantity*, as depicted in Figure 2 below. As *Rating* and *Quantity* were values reported and aggregated from Bricklink users these measures were perhaps biased when trying to generalize to a wider population.

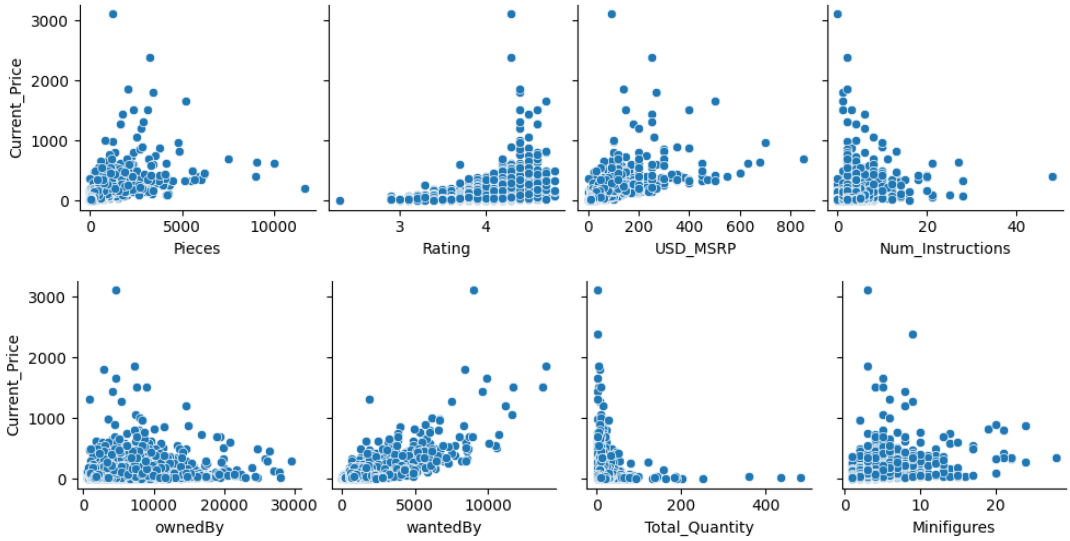
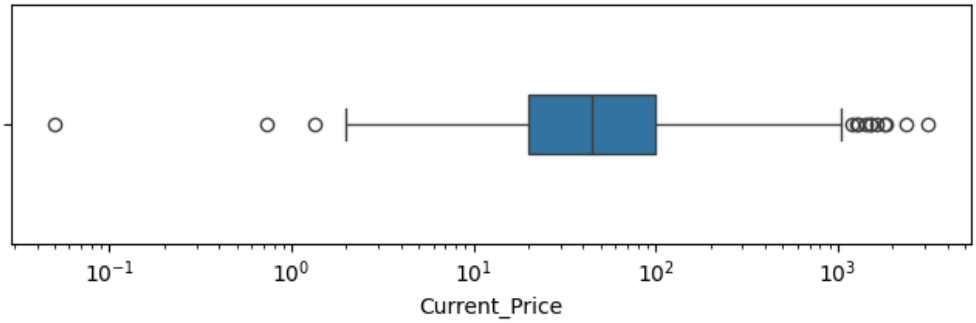



*Figure 2: Pair grid plots of key features vs. the target, Current_Price*




During this phase, the difficulty of the modeling task was also considered alongside finding pertinent features. As seen above, the target variable, *Current_Price*, while correlated to many of the recorded features, had incredibly high variance and was prone to outliers. This appeared to be in line with most collectibles investments which can vary wildly due to a wide range of factors.

Following this, methods for scaling, normalizing, and encoding features were used. *Year* was encoded numerically starting with year 0 in 1975 to reduce the weight of values. Several of the attributes with smaller cardinality that were text-based were one-hot encoded. This included *Category*, *Packaging*, and *Availability*. Additionally, outliers were removed in the *Pieces* and *USD_MSRP* features, to account for extremely large collector LEGO sets. These sets, while highly sought after, would have an outsized negative impact on the ability of the model to generalize. Initially, the *Theme* attribute was also one-hot encoded, but due to the high cardinality of this set (92 different themes) this was changed .

**Model Selection and Hyperparameter Tuning**

Due to the nature of the data set, models with predictive capability despite noise and many features were favored. A baseline was established using Lasso regression on the data. Using k-fold cross validation, Lasso regression had a mean absolute error of around 33.826. Given that current prices for Lego sets in that data can range from under $1 to over $3000, this indicated price prediction was possible. Attempting to improve this baseline proved difficult due to its limited modeling power. Some options that were experimented with included tuning the hyperparameters, alpha and max iterations, as well as limiting the training features to more salient ones such as those observed in Figure 2.

Moving on from L1 regression, the possibility of using a random forest regressor was considered. With an ensemble learning method the complexity of the prediction task and noise of the dataset would likely be reduced through the addition of bias, hopefully reducing the variance seen across different datasets during k-fold cross validation. The random forest regressor was more computationally intensive, but also an improvement over the baseline, reducing error. The results of k-fold cross validation for the baseline, random forest regressor, and subsequent models is shown below.

| Model | Error |
|---|---|
| **Lasso Regression (baseline)** | 33.826 |
| **Random Forest** | 30.134 |
| **XGBoost** | 24.728 |
| **Dense Neural Network** | 56.085 |

To improve on the random forest, XGBoost, or Extreme Gradient Boosting, was used on the data. This led to significant performance improvements over the random forest, likely due to XGBoost's sequential generation of decision trees through gradient boosting allowing for subsequent weak learner trees to perform better. Additionally, opportunities for more hyperparameter tuning made this method more customizable with potentially more performance improvements.

Finally, deep learning with a densely connected neural network was attempted. A simple architecture of several linear layers was used to see if it could generalize better than gradient boosted methods on the data. However, multiple attempts to tune hyperparameters and modify the model architecture resulted in worse performance than both the XGBoost and random forest regressors. While removing features from the data allowed the deep learning approach to reduce loss, this method was eventually abandoned as the model appeared to be severely underfitting the data and struggling to learn any patterns compared to XGBoost.

**XGBoost**

As XGBoost proved to be the most successful model, attempts were made to tune hyperparameters to find the optimal setting for lowering error. To this end, a grid search method was used to find the best parameters across a wide array of selections. This ended up being one of the most computationally intensive and time-consuming parts of the project simply due to the size of the hyperparameter space being considered. In the end, this process was successful in finding a set of better hyperparameters. These hyperparameters included using 500 estimators with a max depth of 5, subsampling 80% of the data when constructing trees, gamma of 0.25 for regularization, and a learning rate of 0.01. This combination of hyperparameters successfully reduced model error below 20.

Another insight during this tuning process was further preprocessing the data to train on using generalizations based on knowledge of Lego. A Lego set's pricing can be heavily influenced by whether a set is discontinued by the Lego company. This is an easy observation, as, once a set is out of production, the forces of supply and demand work to quickly affect the price. However, how long a set remains in production can vary wildly based on popularity or the licensing of IP from other companies. Therefore, an attempt was made to exclude sets still in production or too far back in time, narrowing the time window so the factors for pricing would be largely the same across all the Lego sets in the train and test set. This method would also reduce the likelihood that XGBoost was reducing its error simply by copying the retail price of sets still in production (which would have not yet experienced a price bump from being discontinued). Through this

method, by constricting the time period being considered to a space of five years from the 2010s to the 2020s (e.g. 2016-2021), error was significantly reduced to 14.963.

## Results

Through analysis of the data, it was interesting that while there was a trend in rising prices and piece counts, this was not as pronounced as expected.

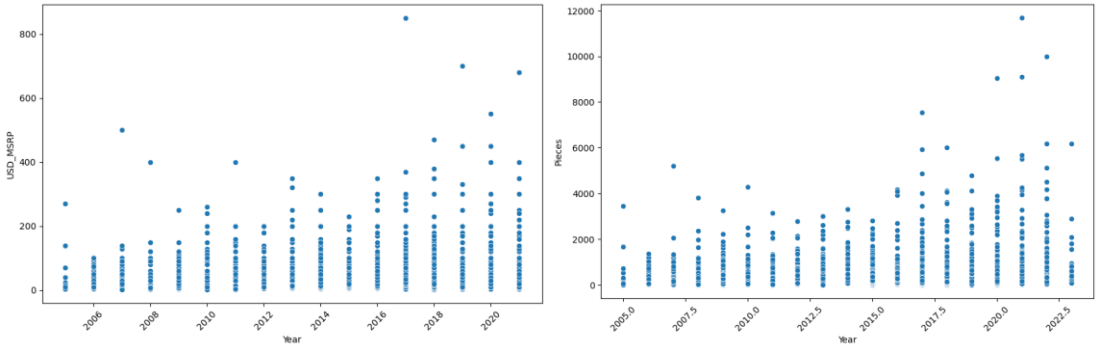
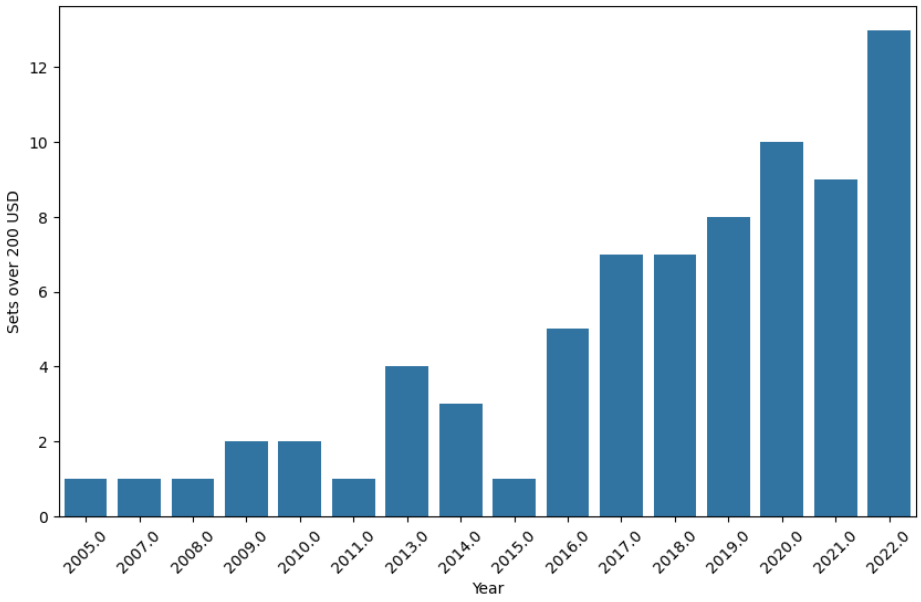



*Figure 3 and 4: MSRP and piece counts of Lego products from 2005-present.*

The distribution of the data suggested that while the prevalence of large and expensive sets had increased, a significant portion of Lego products have not deviated from past products. This is perhaps best shown by cutting out all sets over 200 USD MSRP. The trend for older products collecting value was also more obvious when only considering these larger, more expensive purchases. These findings were all consistent with other sources and expectations.




With this in mind, it is no surprise that predictive models struggled with this task. While most Lego sets gain value, the sets with most potential gain are thus often outliers in piece count and retail price the moment they are released. Since there are less than a dozen per year, it is also likely that significant perturbations in price can result from fluctuations in the popularity of the set theme.

Moving on to the performance of machine learning methods, XGBoost was by far the most effective. At its best, the model could predict product prices in USD with a mean absolute error of less than 15. Examining the visualizations of trees produced by XGBoost also provided key insights into the most pertinent features for price prediction.

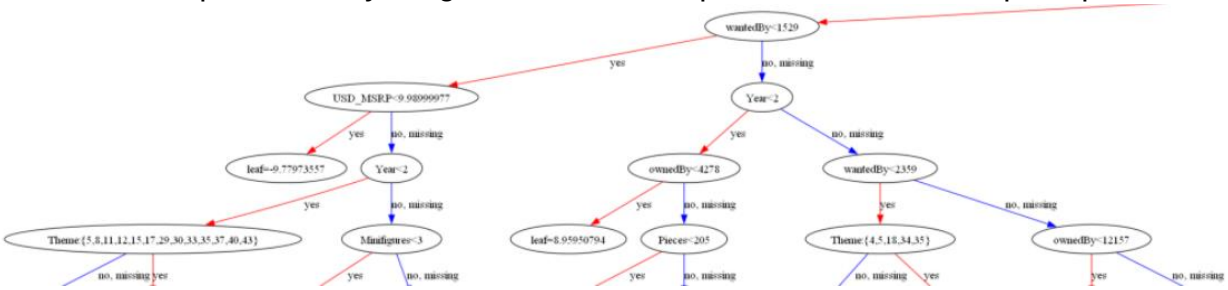



*Figure 4: Branch of a XGBoost decision tree.*

Above, in Figure 4, a single branch of an XGBoost tree depicts the features the model considered most important for pricing predictions. Nearly all trees generated by the model began by splitting on the *USD_MSRP*, or manufacturer's suggested retail price unsurprisingly. The tree also heavily uses the year of production, pieces, and minifigures, indicating that age and size of product are generally important. Beyond these, the model also began to pick up on the importance of a product's theme in determining its popularity, using this alongside the current supply, demand, and online rating to gauge consumer interest. Overall, the performance of this model alongside the trends observed in the data suggest Lego to be a viable investment. Investors with sufficient knowledge of what Lego products might be considered collectibles and which themes are popular may very well be able to generate reliable returns.

Meanwhile, the lasso regression and random forest models saw some success in prediction but using a densely connected neural network proved to be the worst option for this prediction task. Perhaps with a more complex model architecture and hyperparameter tuning, a model could be made to train on Lego prices, but the noise of the dataset and sparse relationships in the data made it unconducive to the model constructed in this project.

## Conclusion

This project demonstrated the possibility of predicting Lego pricing and the viability of Lego as an investment. Clear trends were observed in Lego pricing and size over time and models clearly picked up on learnable signals, both in our data and related studies, and future work should be able to continue developing better prediction systems.

Notably this project was limited by the available data, but with additional time, further details could be added. For instance, additional tags could be added to training data indicating the presence of rare minifigures or rare pieces in any set. Data on a product's performance during its manufacturing lifetime would also be helpful, beyond simply the supply and demand in online secondary marketplaces. Given more time and computational resources, it may even be feasible to train on detailed inventories of the pieces present in Lego sets. Of course, more time also allows for additional product

releases to further populate the data as the Lego company continues operating. On the other hand, more experimentation could be done to determine the effects of each feature on the model results. While this project attempted excluding and including features to simplify the problem and improve performance, the sheer number of features means that more possibilities can be explored in future work.

**References**


Chen, Tianqi and Carlos Guestrin. “XGBoost: A Scalable Tree Boosting System.” Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2016): n. pag.

Oczkoś, Wiktor et al. “Data on LEGO sets release dates and worldwide retail prices combined with aftermarket transaction prices in Poland between June 2018 and June 2023.” *Data in brief* vol. 52 110056. 11 Jan. 2024, doi:10.1016/j.dib.2024.110056

Wall Street Journal. “Lego Investing Is Booming. Here’s How It Works.” *The Wall Street Journal*, 13 Dec. 2022, www.wsj.com/video/series/in-depth-features/lego-investing-is-booming-heres-how-it-works/5F2B44FE-2789-46E2-B280-9CA089EAB458.