Block-By-Block Predictions: Data Mining with LEGO

Block-By-Block LEGO Predictions

Leveraging LEGO set data to predict the themes of unknown LEGO sets based on other feature attributes

Collin D. MacDonald

Student, George Mason University, cmacdon8@gmu.edu

This paper investigates the viability of predicting a LEGO set's theme based on other known feature attributes of the LEGO set such as the year it was released, the total number of parts in the LEGO set, and the Part-Color distribution of the LEGO set. Python, Pandas, and SKLearn were used to investigate the research question via K-Nearest Neighbors and Random Forest classification algorithms. Results were then analyzed, commented on, and ideas for future research were discussed.

CCS CONCEPTS • Artificial intelligence • Classification and regression trees • Experimentation

Additional Keywords and Phrases: LEGO, SKLearn, Pandas, Python, Data Mining, Prediction, KNN, Random Forests, Classification.

ACM Reference Format:
Collin D. MacDonald. 2021. Block-By Block Predictions: Data Mining with Lego: Leveraging LEGO set data to predict the themes of unknown LEGO sets based on other feature attributes. ACM, Fairfax, VA, USA, 10 pages.

1 INTRODUCTION

Since the 1960s, over 19,500 LEGO sets have been produced, across over 700 themes and sub themes. They ranged from generic themes such as City, to third-party licensed themes such as Star Wars, to in-house themes such as Ninjago. Given the diversity of LEGO sets and themes over the past 50 years, the following question was posed: "Could the theme of an unknown LEGO set be predicted using some known features of the same LEGO set?" A program was developed using Python, Pandas, and SKLearn that took in a variety of data regarding LEGO sets, LEGO parts, LEGO colors, LEGO themes, and their relationships as input and output the predicted theme and parent theme of a previously unseen LEGO set.

2 DATA SOURCES

All data for this project was found on ReBrickable, specifically https://rebrickable.com/downloads/ [1]. ReBrickable describes itself as a website that "will show you which LEGO sets you can build from the sets and parts you already own. You can choose from official LEGO sets or thousands of MOCs (My Own Creations) submitted by hundreds of designers. All MOCs include building instructions and full parts lists" [2]. ReBrickable provides an API to access the data they provide as well as CSV files that contain the same data. The CSV files are updated daily. Given that the goal was to classify as many LEGO sets as possible, the CSV file option was selected. This option provided easy access to bulk amounts of data. Twelve CSV files were provided by ReBrickable, five of which were used. Table 1 shows all twelve CSV files and which ones were used.

Table 1: CSV files available for download from Rebrickable

Data Sets Used In Project	Data Sets Not Used In Project
Themes.csv	Part_Categories.csv
Color.csv	Parts.csv
Sets.csv	Part_relationships.csv
Inventory_parts.csv	Elements.csv
Inventories.csv	Minifigs.csv
	Inventory_sets.csv
	Inventory_minifigs.csv

The data provided by Rebrickable was broken into three primary categories: Sets (Inventory_sets, Sets, and Themes), Minifigures (Inventory_minifigs and Minifigures), and Parts (Inventory_Parts, Parts, Colors, Part_Relationships, Part_Categories, and Elements). ReBrickable also provided a helpful diagram that showed the contents of each CSV file as well as its relationship to all the other CSV files. Using the diagram below (Figure 1), the CSV files that were most likely to help in the goal of predicting the "theme_id" of a given LEGO set were identified.

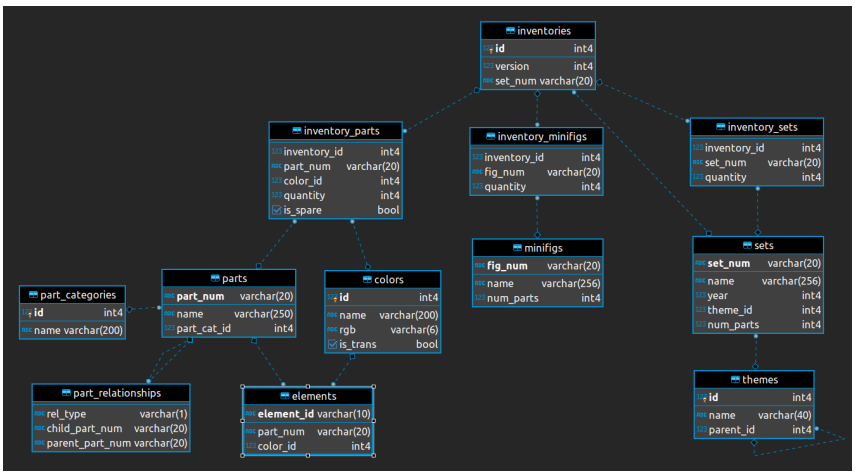

Figure 1: Schema diagram for all LEGO CSV files provided by Rebrickable.
(https://rebrickable.com/static/img/diagrams/downloads_schema_v3.png)

3 PROJECT SETUP & DATA PRE-PROCESSING

Python 3 was selected for this project given its suitability for rapid development and variety of third party data-focused libraries. Pandas (https://pandas.pydata.org) was selected as the library of choice for consuming the raw CSV files and getting them into a workable format. After the data was imported with Pandas, SKLearn (https://scikit-learn.org/stable/) was selected as the Machine Learning library of choice to analyze the data and create predictions from the data. Using Pandas' DataFrame components, the Sets, Inventories, and Themes CSV files were left-joined together on Sets. The Inventory Parts and Colors CSV files were also loaded into their own separate DataFrames for later use.

Because all the data being used was supplied by users of ReBrickable, there were some validation errors such as LEGO sets with "Infinite" amounts of parts due to numerical overflow. Pandas and Numpy were used to coerce as much

data as possible to Integer datatypes and then replace any invalid entries with zeros. Another challenge of using the ReBrickable datasets was that out of the approximately 19,500 LEGO sets provided in the Sets CSV, around 1,500 of those LEGO sets weren't true LEGO sets at all – they were everything from LEGO themed Books to LEGO DVDs to promotional giveaways and LEGO Pens. While these LEGO sets are still technically "sets", and some did have themes (such as LEGO Star Wars themed books), the decision was made to remove these LEGO sets and only train and test the model on standard LEGO sets. A regular expression was developed to filter out LEGO sets that did not follow the "traditional" numbering pattern of three to six digits, followed by a dash, followed by a single digit. After applying this filter to the joined DataFrame created above, about 18,000 LEGO sets remained. SKLearn was then used to split the pre-processed data into training and testing data [3]. The "theme_id" column of Sets was used as the target column (Y_Training and Y_Test) while the "year" and "num_parts" columns were selected as key features (X_Training and X_Test). Set names were not included as a feature because Set names often directly include or reference the Set's theme, especially for third party licensed LEGO sets from brands such as Disney.

4 MACHINE LEARNING IMPLEMENTATION & EXPERIMENTAL RESULTS

Once the data was imported, cleaned, and pre-processed, it was time to start implementing the actual classification algorithm, train the algorithm, and test the results. The first idea was to use K-Nearest Neighbors to try to determine the theme of an unknown set based on the specific parts in that LEGO set. The basis for this idea comes from the fact that many LEGO themes share a variety of parts, so perhaps the parts alone would be enough to predict a set's theme. However, there are over 44,000 different LEGO parts (and close to 1,000,000 unique elements if you look at possible colors for any given part). Even ignoring color differences, computing the One-Hot encoding of 44,000 parts for 18,000 sets would require over 792,000,000 calculations, which was not feasible. Because K-Nearest Neighbors does not work well in high dimensions, having 44,000+ features would have made such a matrix useless, even if the calculations could be feasibly done.

Since using One-Hot encoded LEGO part data as features was not viable, using LEGO color data as features was considered as an alternative option. While there are over 44,000 different LEGO parts, ReBrickable only recognizes 190 official LEGO colors as of December 2021. Using similar logic, it was hypothesized that every LEGO theme likely has a certain color distribution that is unique to that theme (For example, LEGO Ninjago makes heavy use of black, gold, and red colored LEGO pieces). For every LEGO set, the number of LEGO parts for each color was calculated and then normalized to create a "Part-Color Distribution". This matrix required around 3,400,000+ calculations, which is a 99.6% reduction when compared to the 792,000,000+ calculations that would have been required using the part data. Pairing Panda's Apply function with a custom function, all 3,400,000+ calculations were completed on the joined Sets-Inventories-Themes DataFrame in around 65 minutes [4]. K-Nearest Neighbors was then ran using the Part-Color distribution, release year, and total number of parts (192 features total) as input. A maximum accuracy of around 25% was achieved when tested from K = 1 to K = 99. The best accuracy was achieved at K = 1.

Given the partial success at predicting a Set's theme using K-Nearest Neighbors and Part-Color distribution, only the classifier was changed (features stayed the same). An SKLearn Random Forest classifier was implemented and then tuned via RandomizedSearch cross validation due to time constraints [5]. With only an hour of hyper-parameter tuning, the Random Forest classifier was able to achieve a 42% accuracy in predicting an unknown set's theme. Furthermore, the Random Forest classifier was able to achieve a 58% accuracy in predicting an unknown set's parent theme. A "parent" theme can be thought of as a generalized "umbrella" theme used to loosely organize themes into overarching categories. For example, a long running "parent theme" is LEGO City. The LEGO City parent theme has

included many subthemes over the years such as LEGO Trains, LEGO Safari, etc. The 58% accuracy for predicting a parent theme is encouraging, as it implies that most of the time the prediction was "close" even if the exact theme was wrong since it was still in the same parent theme.

5 LESSONS LEARNED & FUTURE RESEARCH

The most valuable lesson learned from this project was that it is important to take the time to fully understand the data and any pre-existing relationships in the data when working with a large unknown dataset. In the case of the ReBrickable datasets, this was understanding how all twelve CSV files were linked to each other through Parts, Sets, and Minifigures. This project also emphasized the importance of feature selection. While it might be theoretically possible to predict a LEGO Set's theme based solely on its parts, it is nor feasible to perform One-Hot encoding for 44,000+ parts. Also, knowing how to modify the data to fit the needs of the project cannot be understated (such as creating the Parts-Color distribution for this project). Lastly, it is important to remember that there is no one perfect classifier. Trying a variety of classifiers and knowing what classifiers work best for specific scenarios is a very important step when starting a Data Mining project.

One potential area of research in the future would be looking into how LEGO themes change over time and factoring that into the Machine Learning algorithm. Many LEGO themes are only out for a few years before being discontinued, so it would be interesting to test how well the current Random Forest algorithm takes this into account (if it does at all). Another area of research could be identifying re-named themes. Perhaps this could be done by using a K-Means algorithm with part-color distribution and theme data to identify groupings of themes over time that share similar characteristics. There may be some interesting results, as LEGO has historically re-named themes every so often (such as "LEGO Creator Expert" being renamed to "LEGO 18+" in 2021).

6 CONCLUSION

The initial question for this project was "Could LEGO set data be used to accurately predict the theme of an unknown LEGO set using other feature attributes?". Through a combination of data manipulation, preprocessing, and classification algorithms, the answer to this question is yes. Provided additional time and resources, the 42%-58% accuracy of the Random Forest classifier could likely be improved primarily through theme preprocessing (such as removing themes that only occur once).

ACKNOWLEDGMENTS

Thank you to Professor Sanmay Das and Teaching Assistant Bikram Adhikari for their support in CS484: Data Mining throughout the Fall 2021 semester.

PROJECT URL: https://drive.google.com/file/d/18jvBSL0rnCIyf8EOsfDRjQUiXNHWWNml/view?usp=sharing

REFERENCES

[1] About ReBrickable. Retrieved December 11, 2021 from https://rebrickable.com/about/

[2] Rebrickable LEGO Database Download. Retrieved December 9, 2021 from https://rebrickable.com/downloads/

[3] SKLearn. Retrieved December 9, 2021 from https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

[4] Pandas DataFrame Apply. Retrieved December 9, 2021 from https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.apply.html

[5] SKLearn. Retrieved December 9, 2021 from https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html