

Executive Summary

This report aims to investigate whether investing in Lego sets is a viable financial option. Exploratory data analysis, cluster analysis and sentiment analysis were all utilised throughout the report to aid in such. With the creation of a Tableau dashboard created to support a potential investor wishing to learn more about Lego.

To analyse the potential returns and risks associated with investing in lego sets, predictive models were developed, and Sharpe ratios used to assess risk-adjusted returns. This was then compared to the performance of more mainstream investment options, with the benefits and drawbacks of each discussed.

The report finds that Lego sets offer higher returns than several traditional investment options but have increased risk and time commitment relative to more traditional investment options. Additionally, it recommends that investors with sufficient time and expertise consider Lego investments as an option to add to their portfolio. For others, more accessible options such as ETFs may be more advisable.

Although the report identified challenges in the process such as incomplete data and difficulty in defining certain price measures. The report provides valuable insights for prospective investors seeking alternative investment options, highlighting the importance of further data collection and continual model development to ensure reliable predictions.

Extensive coding notebooks and dataset files were created and used throughout the report, to follow along and recreate results observed, files within this project's supplementary material can be found as follows. The additional files folder contains datasets used as well as the Tableau Dashboard, these are aptly named so easy to follow along with. The additional code folder contains the code used for each section of this report, and follows the same naming, with each file including the number and subheading name for ease of use.

File Name	Section(s) referred to
rebrickable.ipynb	2.1
brickowl_scrape.ipynb	2.2
price_prediction.ipynb	2.3 4.1 4.4.3
cluster_analysis.R	4.2
sentiment_scores.ipynb	4.3.3
amazon_scrape.ipynb	4.3
sentiment_analysis.ipynb	4.3
data_wrangling.ipynb	4.4.2
simple_approach_modelling.ipynb	4.4.2.2 4.4.2.3
sentiment_scores.ipynb	4.4.5

Table 0 - Showing a reference guide from additional files to the section they referred to.

Executive Summary.....	1
Introduction.....	6
1.1 Organisation.....	6
1.2 Goals.....	6
1.3 Constraints.....	6
Data.....	6
2.1 Rebrickable.....	6
2.1.0 Data Retrieval.....	6
2.1.1 Descriptive statistics:.....	7
2.1.2 Missing Occurrences.....	7
2.1.3 Outliers.....	8
2.1.4 Imbalance.....	9
2.2 Brick Owl.....	9
2.2.0 Data Retrieval.....	9
2.2.1 Descriptive statistics.....	10
2.2.2 Missing Occurrences.....	11
2.2.3 Outliers.....	11
2.2.5 Near-Zero-Variance.....	11
2.3 Brickset.....	12
2.1.0 Data Retrieval.....	12
2.3.1 Descriptive statistics.....	12
2.3.2 Missing Occurrences.....	12
2.3.3 Outliers.....	13
2.3.4 Imbalance.....	14
Methodology.....	15
3.1 Data Cleaning Strategies.....	15
3.1.1 Brickset.....	15
3.1.2 Brickowl.....	15
3.1.3 Joined.....	15
3.2 Data Ethics.....	16
Results.....	17
4.0 Data Visualization and Dashboard.....	17
4.0.1 Data and design Process.....	17
4.0.2 Tools and features.....	18
4.1 Exploratory Data Analysis.....	20
4.1.1 Data overview.....	20
4.1.2 Correlations and Unusual features.....	20
4.1.3 Variable creation.....	23
4.2 Cluster Analysis.....	23
4.2.1 Data Wrangling.....	23
4.2.2 Methods.....	23
4.2.3 Results.....	24
4.2.3.a Cluster 'Outliers':.....	27
4.2.3.b Cluster 'Retail':.....	27

4.2.3.c Cluster ‘Non-retail Star Wars’:	28
4.2.3.d Cluster ‘Retail Star Wars’:	28
4.2.3.e Cluster ‘Model Making’:	29
4.2.3.f Cluster ‘High Returns’:	29
4.2.3.g Cluster ‘Low Returns’:	29
4.2.3.h Cluster Comparison:	30
4.3 Sentiment Analysis	31
4.3.1 Data Wrangling	31
4.3.2 Frequent Items	32
4.3.2.a Methods	32
4.3.2.b Results	32
4.3.3 Frequent Itemsets	33
4.3.3.a Methods	33
4.3.3.b Results	34
4.3.4 Sentiment Scores	34
4.3.4.a Methods	34
4.3.4.b Results	34
4.4 Predictive Modelling	36
4.4.1 Measuring error	37
4.4.2 Simple approach	37
4.4.2.1 Data wrangling	37
4.4.2.2 Methods	38
4.4.2.2.a Regression models	38
4.4.2.2.b Tree Models	38
4.4.2.3 Results	38
4.4.2.3.a Regression Models	38
4.4.2.3.b Tree Models	40
4.4.3 Advanced approach	41
4.4.3.1 Data Wrangling	41
4.4.3.2 Methods	41
4.4.3.2.a Preprocessing	41
4.4.3.2.b Validation	42
4.4.3.2.c Feature importance	42
4.4.3.2.d Hyperparameter- Tuning	42
4.4.3.3 Results	43
4.4.3.3.a Preliminary model assessment	43
4.4.3.3.b Validation	45
4.4.3.3.c Feature importance	45
4.4.3.3.d Hyperparameter- Tuning	46
4.4.4 Choosing a model	46
4.4.5 Findings	47
4.4.6 Further considerations	49
Discussion	50
5.1 Fit-for-purpose	51

5.2 Future work.....	51
5.3 Limitations.....	52
5.4 Ethics.....	52
References.....	52
Glossary.....	55

Introduction

1.1 Organisation

This project works within the domain of consumer goods and collectibles, specifically on the Lego brand. Lego has been a strong competitor for many years in the toy manufacturing industry. The organisation's wide collection of products over the decades makes this a rich area for data analysis.

1.2 Goals

It has been said that "Lego is not just a toy but also a reasonable alternative investment with average returns comparable to stock returns"(Dobrynskaya & Kishilova, 2022) , and so the primary research question this project aims to answer is as follows: "Is investing in LEGO sets worthwhile? and if so, what sets or types of sets should be prioritised by a prospective investor in order to best maximise their returns?".

1.3 Constraints

To ensure a realistic outcome is achieved, it is important to consider the constraints of this project. The amount of data available to aid this project was very minimal. API calls were primarily used to accumulate a suitable amount of data. Through the use of API(s) the data is able to be extracted for usage, however these datasets contain a combination of different data types such as strings, factors and numeric values. The data must be cleansed properly in order for it to be used for analysis. Web scraping was another method used to accumulate some data, however scraping from different public sources introduced challenges around data variation and consistency.

Along with data collection, data merging could be a problem. Since the data is collected from several different sources, some datasets will need to be combined based on a matching key, typically a set's unique ID number. By doing this some rows of data could be lost or some rows will have missing values which would make it more difficult to build models with the data as they could be underfitted.

Data

2.1 Rebrickable

2.1.0 Data Retrieval

The database is retrieved via Rebrickable API and will mainly be used to create data visualisation in this project. This database is organised as a relational database. A relational database organises data across multiple tables which can be related to each other by a unique identifier called a primary key and a foreign key (IBM, 2023). The Rebrickable database contains eight tables with each representing an entity such as colours or parts. The below diagram, figure 2.1.0a illustrates the relationships between these tables.

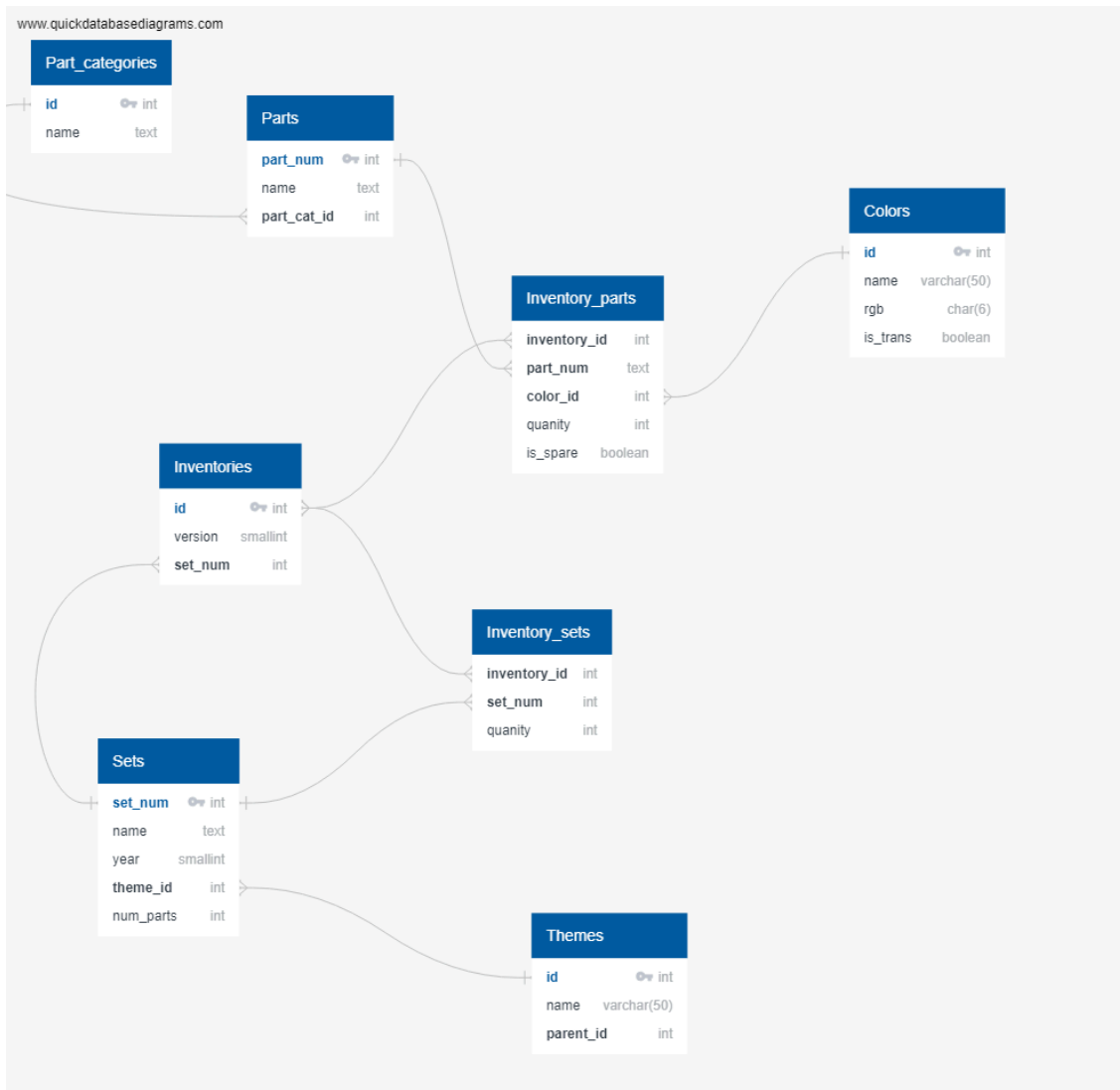


Figure 2.1.0.a - A relational schema diagram for the Rebrickable database. (Source: Image was recreated based on <https://rebrickable.com/downloads/>)

2.1.1 Descriptive statistics:

The database contains comprehensive information about Lego Sets, Themes, Colours, and Rebrickable Inventory's Parts from as early as 1949 to as recent as 2025.

There are 23085 unique sets, 463 unique themes and 267 unique colours. In a Lego set, on average there are 163 pieces, and 9 distinct colours. The highest number of pieces in a Lego set is 11,695 pieces (World Map 31203-1) and the smallest is 0. The sets with no parts are from a number of themes such as Key Chain, Gear, and Houseware and 240 other themes.

2.1.2 Missing Occurrences

For all tables in the database, there are only two columns containing missing values namely "img_url" from the *inventory_parts* table and "parent_id" from the *themes* table.

The first missing occurrence is not missing-at-random and could be explained by the difficulty of acquiring an image for some parts due to their rarity. The second occurrence is also missing-not-at-random, while most of the themes have a "parent_id", there are a number of themes which do not have a "parent_id" or rather, a parent theme. For example, the theme Star Wars could belong to a more general parent theme Technic or Advent or Mindstorms, or it could have no parent theme at all.

2.1.3 Outliers

Two main methods have been used to effectively identify outliers, these were IQR (interquartile range) and a histogram plot.

For IQR, the first and the third quartile which each represent one-quarter and three-quarters of the way through the list of all data are calculated. IQR is then calculated by subtracting the first quartile from the third quartile. The outliers would be identified if they present outside of the lower and upper bounds set by adding or subtracting IQR multiplied by 1.5 - which is a constant used to discern outliers, to the first quartile and third quartile respectively (Taylor, 2018).

For the second method, when the data is plotted as a histogram, outliers can be visually identified easily.

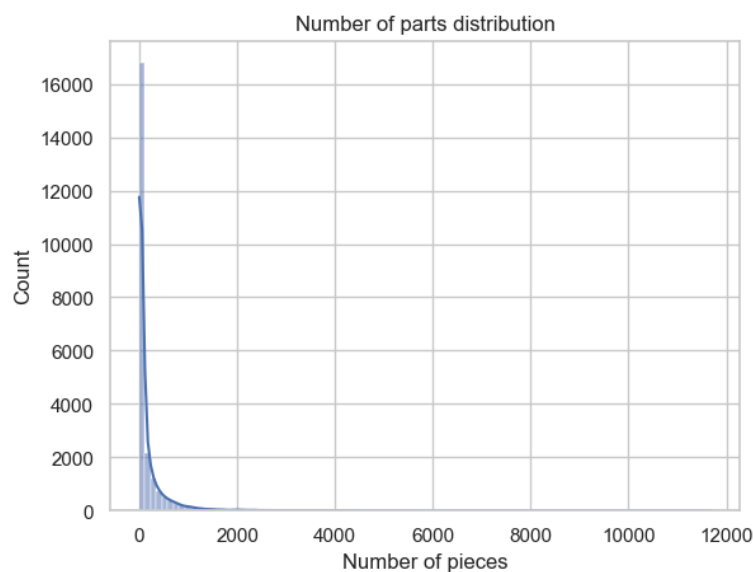


Figure 2.1.3.a - A histogram plot showing the distribution of number of pieces for Lego sets. Most of the observation lies around 0 to 1500 pieces. Outliers can then be filtered out to investigate.

	set_num	name	year	theme_id	num_parts
425	10307-1	Eiffel Tower	2022	721	10001
4470	31203-1	World Map	2021	709	11695

Figure 2.1.3.b - Dataframe after applied filtering by "num_parts" showing only rows with "num_parts" higher than 10,000

Sets such as the Eiffel Tower and the World Map both have very high piece counts compared to the rest of the sets within the dataset. However, it is important to understand that they are extreme values as opposed to outliers; they do not indicate data inconsistency or data entry mistakes.

2.1.4 Imbalance

Some categories have a disproportionate distribution in the Rebrickable database. There are a relatively large number of non-transient colours compared to a relatively small number of transient colours. Similarly, the majority of the Lego sets are variant number one (notated by a “-1” after the set number), indicating the original release, with some sets having up to 16 versions usually caused by regional releases of the same set with different packaging or instructions. (Rebrickable Help Guide: Set Variants and Inventories | Rebrickable - Build with LEGO, 2024).

There are also imbalances within the theme column. Theme id has high cardinality containing 390 different categories, with some popular themes such as Star Wars containing 960 sets compared to less popular themes such as Aquazone which only has 1 set.

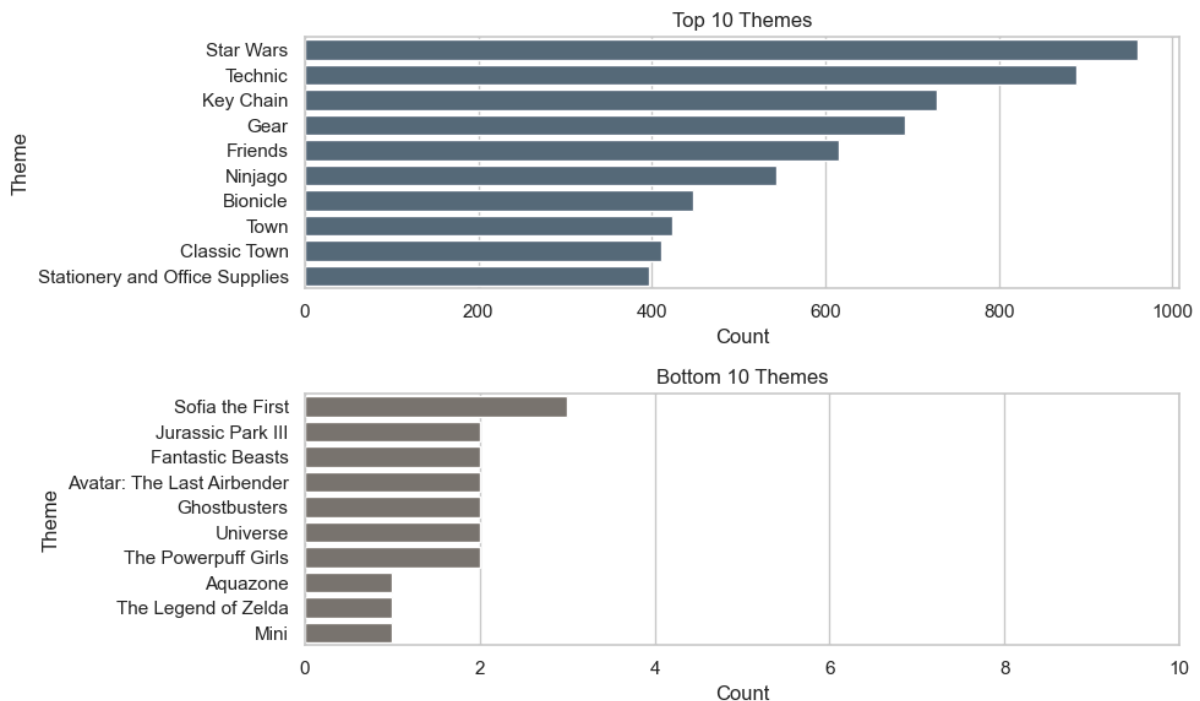


Figure 2.1.4.a - Showing the top 10 and bottom 10 themes by counting the number of sets. Note the different scales on the X-axis.

2.2 Brick Owl

2.2.0 Data Retrieval

The Brick Owl dataset was acquired from Brick Owl API. Brick Owl is the largest independent LEGO Marketplace for Lego Parts and Minifigures (Brick Owl | Brick Owl - LEGO Marketplace, 2024). It was primarily used for its market value data on Lego sets.

The API key was given by a representative from the site which allows requests for pricing information on sets. The process of retrieval was as follows.

Get information about all available "BOLD" which is each selling instance on the website. Then from the complete list of "BOLD", requests are made to get the pricing for "Set". If there is an instance, it is concatenated to the Dataframe.

```
params = {
    'key': my_key,
    'type': 'Set',
    'format': 'json'
}
```

Figure 2.2.0.a

```
params = {
    'key': my_key,
    'boid': '{}'.format(boid),
    'country': 'US',
    'format': 'json'
}
```

Figure 2.2.0.b

Figures 2.2.0.a and 2.2.0.b show two different parameters used to retrieve "BOLD" and "Set" respectively. "My_key" is the Brickowl API key.

The Dataframe contains information about the current price it is selling on the market, the available quantity, condition, set number, store logo, store URL, store id etc. The currency of the current listing price is in USD. Hence, we have utilised the Python Currency Converter API package (Prengère, 2024) to convert it into NZD.

Because there are multiple instances of the same sets. Some aggregations were then performed to prepare data for analysis. We first filter out by "con" (condition of the set new or used), and group by "set_number", aggregations used on price are min, max and mean and for "quantity" is sum. Now we have two dataframe, one for condition "new" and one for condition "old", then the two dataframes were outer joined together on "set_number".

	set_number	min_value_new	max_value_new	mean_value_new	quantity_new	min_value_used	max_value_used	mean_value_used	quantity_used
0	011-1	190.479899	190.479899	190.479899	1.0	36.514912	36.514912	36.514912	1.0
1	033-2	158.735104	158.735104	158.735104	1.0	NaN	NaN	NaN	NaN
2	087-1	69.000000	69.000000	69.000000	1.0	NaN	NaN	NaN	NaN
3	10014-1	100.141311	124.308747	116.252935	3.0	64.370000	92.379293	80.675979	5.0
4	10019-1	1669.021842	1669.021842	1669.021842	1.0	445.072491	445.072491	445.072491	1.0
...

Figure 2.2.0.c The dataframe after the transformation

2.2.1 Descriptive statistics

The Brick Owl data frame has 11,931 observations and 9 columns. Each observation is a Lego set with its average, minimum and maximum values on the Brick Owl marketplace. There are 9131 new-condition Lego sets and 7891 used. For the new sets, the average value is 146.34 NZD and for the used sets the average value is 85.74. The maximum values are 10,171 NZD (BatMan COMCON018-1) and 16,186 NZD respectively (Grand Carousel 10196-1).

2.2.2 Missing Occurrences

The missing occurrences for this data frame are not missing at random. As explained above in section 2.2. The outer join was performed for the two data frames with condition “new” and condition “old”. As a result, for sets which have market value for “New Condition” but not “Used Condition”, there will be missing values in columns for the used condition values and vice versa. For cases when a set has both used and new prices on the market, there will be no missing occurrences.

2.2.3 Outliers

Using the same strategies described in section 2.1, outliers can be identified visually using histogram plots.

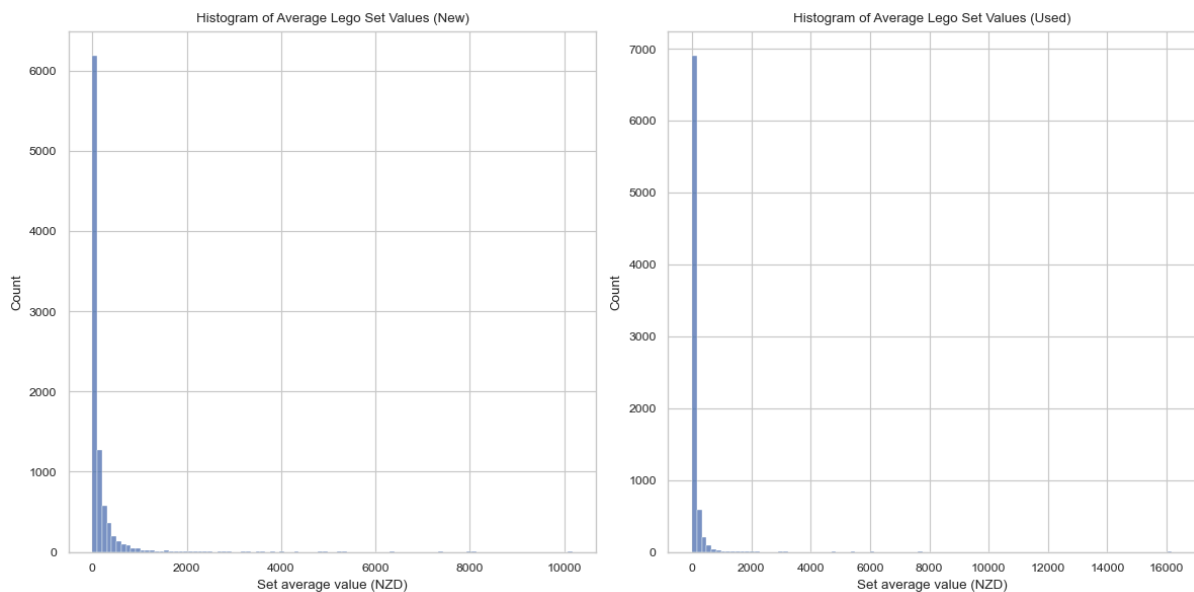


Figure 2.2.3.a

As seen most of the Lego sets have values, both new and used, between 0 and \$2000. However, there are a few extreme observations. When filtered, and with internet validation, it can be confirmed that some sets have extremely high values on the market. For example, Lego - Death Star II - 10143-1 has a current value for a new set of \$3692 USD on Brickset and \$8031 NZD on Brick Owl. This indicates that these outliers are meaningful and can inform important trends.

2.2.5 Near-Zero-Variance

There are small, but non-zero, variances in the “quantity” column for used sets. With a standard deviation of 4.75. Most of the data points are similar with 64% of observations having values between 1 and 3. This is the quantity of the sets on the market, and could be used as a feature for modelling which could potentially provide some information about the rareness of a set on the market. However, acknowledging the fact that the variable is almost constant, in addition to the presence of other potential features such as “availability” from Brickset which also provides the same information. The decision to exclude this feature was made.

2.3 Brickset

2.1.0 Data Retrieval

The Brickset dataset is retrieved via the brickset API using an R package called “Brickset”. This package provides functions to access data about Lego sets from the Brickset website. (Interface with the Brickset API for Getting Data about LEGO Sets, 2024).

The Brickset dataset contains comprehensive information about Lego sets such as theme, category, retail price, number of pieces, year released etc. This is useful for price prediction modelling later in the project.

2.3.1 Descriptive statistics

The dataset contains 19,409 entries and 36 columns.

For categorical columns, there are 158 unique themes, 16 unique theme groups, 956 unique subthemes, 7 unique categories and 9 unique ways of packaging a Lego set. The most frequently appearing theme is Gear, which includes 2999 sets. The most common packaging type is box and as expected, most sets were sold via retail.

For numerical columns, the earliest set was released in 1970 and the latest one was released in 2023. The median number of minifigures in a set is 2 and the maximum in a set is 80 (Figure Collection - Fabuland). The highest retail price for a set is \$849 USD (AT-AT - Star Wars) and the average is \$38.95.

2.3.2 Missing Occurrences

There are many columns containing missing values including subtheme, pieces, minifigs, retail price columns and date available columns.

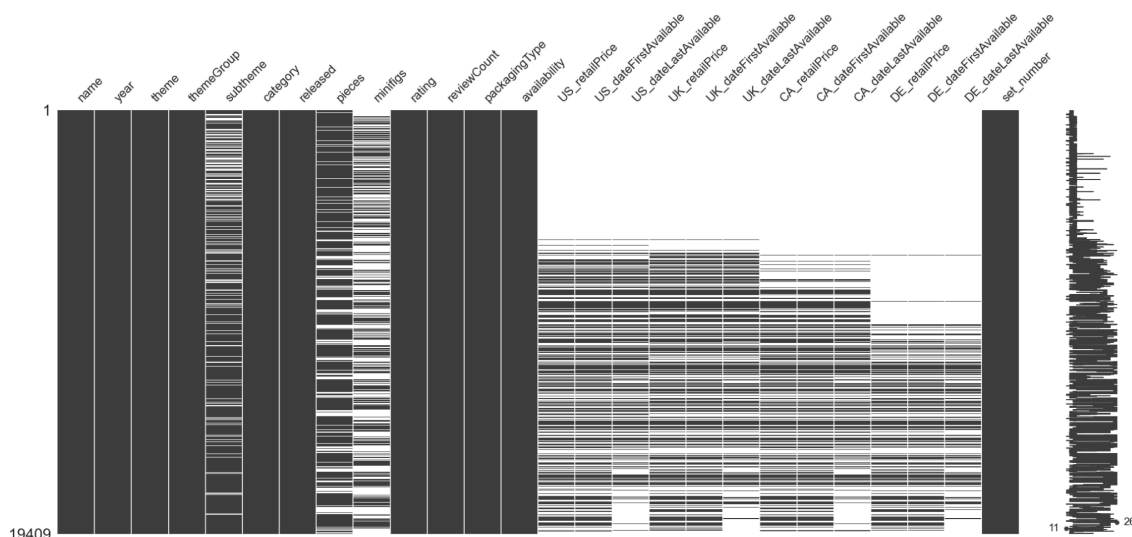


Figure 2.3.2.a - Visualizing missing values across columns in Brickset data.

As shown, in figure 2.3.2.a, the majority of missing values are in retail and date columns and mostly from the earlier sets. This is not missing at random considering Brickset was launched in 1997 and only started tracking information about Lego sets since then, hence

much information before that time was missing. Additionally, the missing values across different regions could be explained by the availability of certain sets across different regions. However, this cannot be confirmed.

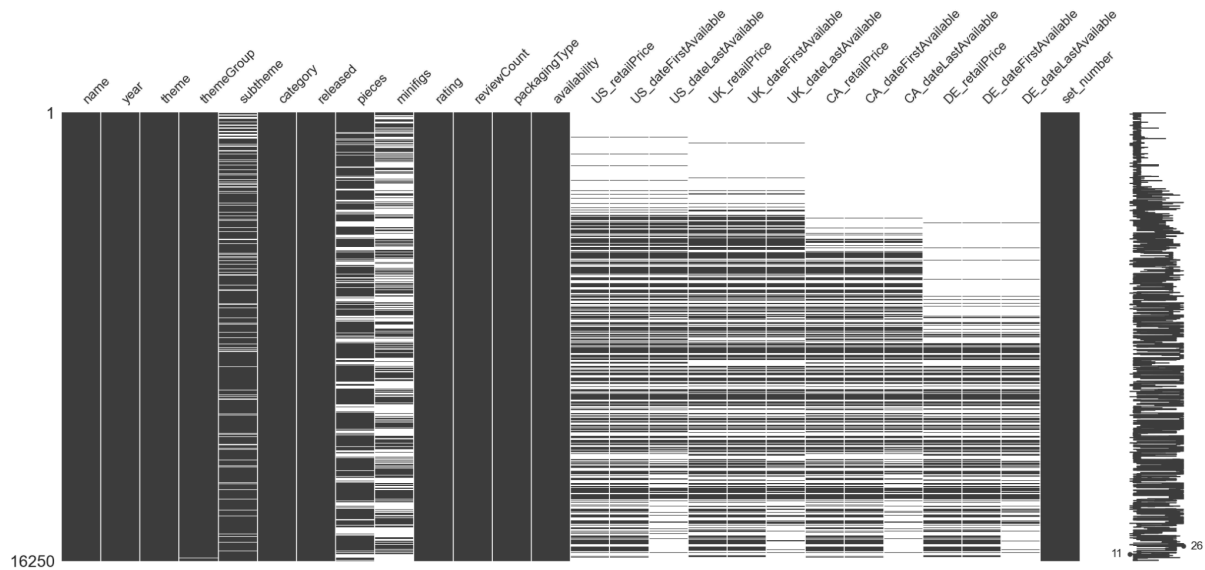


Figure 2.3.2.b - Visualizing missing values across columns in Brickset data after 1997, showing Brickset has picked up the tracking of values in retail and date available after its launch.

Missing values also appear in minifigs, pieces and subtheme columns. They are not missing at random. Not all Lego sets contain minifigs or pieces. For instance, sets from theme “Books” do not usually contain any pieces or minifigures. Similarly, not all sets have a subtheme.

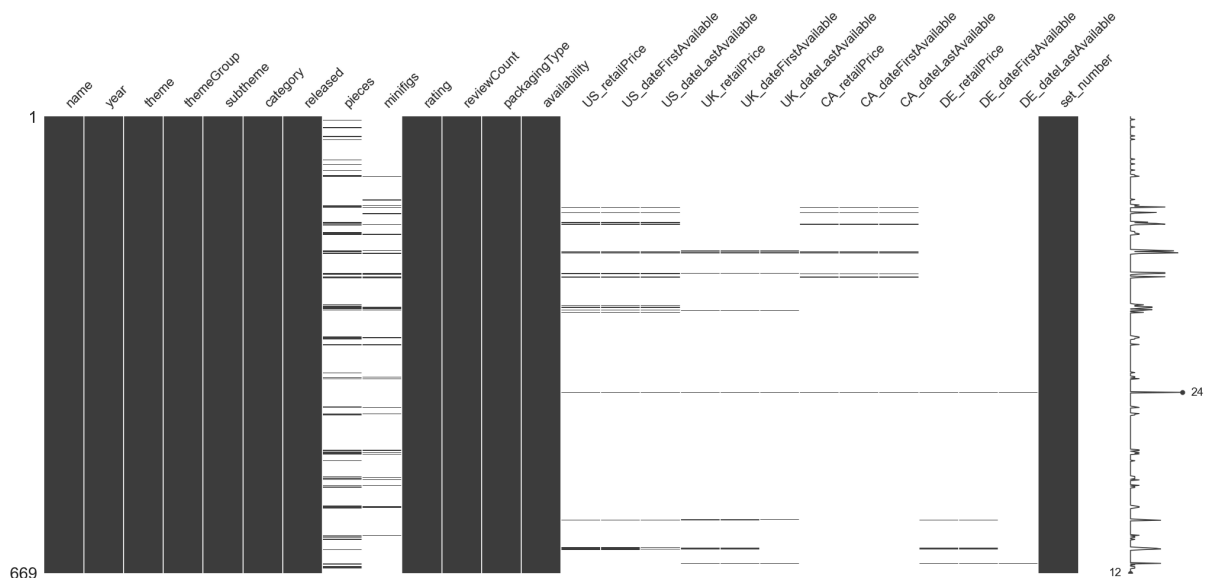


Figure 2.3.2.c - Visualizing missing values across columns in Brickset data for the “Books” theme

2.3.3 Outliers

Outliers were detected using previously discussed techniques for numerical columns, namely pieces and minifigures. The outliers in Brickset have similar meanings to the ones discussed in section 2.1.3.

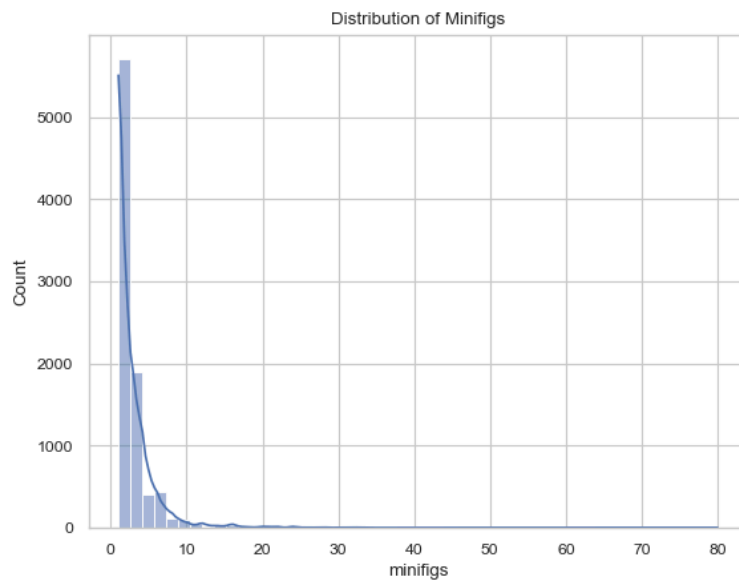


Figure 2.3.3.a - Histogram plot showing the distribution of minifigures

The set with the highest number of minifigures mentioned in section 2.3.1 is Figure Collection containing 80 minifigures. It provides useful information, and is not a representation of errors.

2.3.4 Imbalance

Disproportionate category distributions are appearing in theme, subtheme, packaging type, availability and category columns. Theme and subtheme are more extreme as discussed in section 2.1.4. The top 4 subthemes by frequency are Magazine Gift, Product Collection, Storage and Miscellaneous with observations between 290 to 540. However, there are many subthemes on the opposite end of the section with only 1, or very few, observations.

Similar situations appear in availability, packaging type and category columns. However, the gap between them is not as extreme as in the case above.

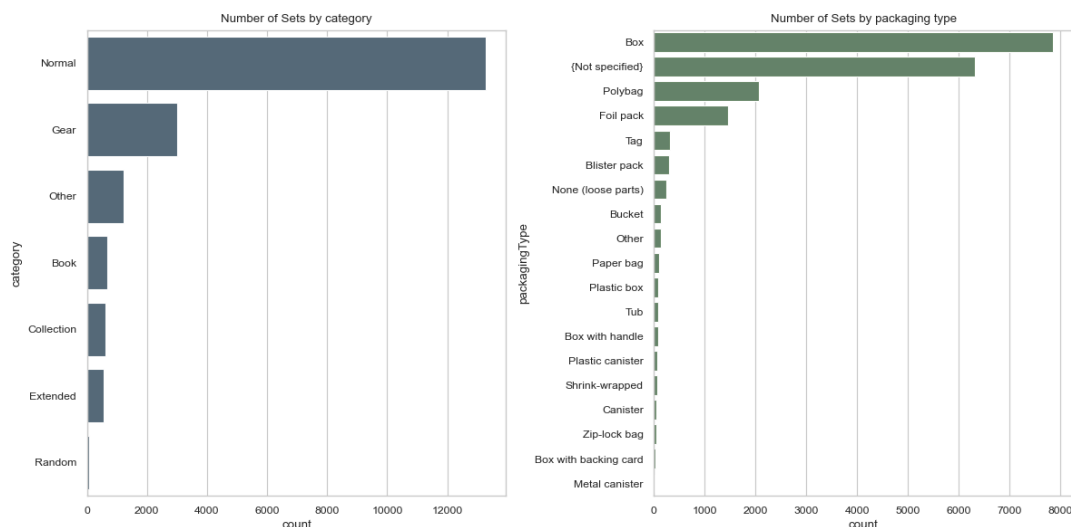


Figure 2.3.4.a - shows the imbalance in distribution within the category and packaging type columns.

Methodology

3.1 Data Cleaning Strategies

3.1.1 Brickset

The missing occurrences are mentioned in section 2.3.2. For missing values in date available columns, values are imputed across columns from different regions into one single column called dateFirst(/Last)Available. For instance, if a set has a first date available in the US and the UK but missing elsewhere, the new column would contain the value from the first region, US in this case. Similarly, this strategy of imputing across columns is performed to retailPrice columns.

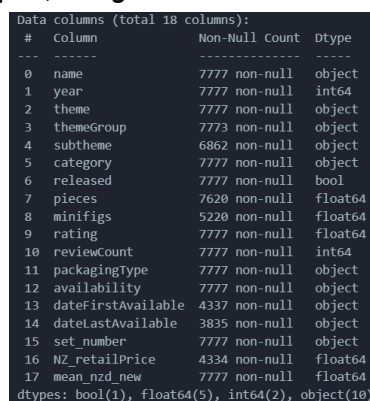
Within the Brickset dataset, sets had up to 4 retail prices available, United States, United Kingdom, Denmark and Canada, each in their own respective currencies. Each price was converted at the historical conversion rate using the Python Currency Converter package (Prengère, 2024) from their respective currencies to NZD, allowing for a “min retail price” to be calculated. The historical date at which conversions were completed was the last date which the set was available for retail purchase.

3.1.2 Brickowl

The missing occurrences in Brickowl is explained in section 2.2.2. Since the goal of the project is to build a prediction model to predict annual ROI given information about a Lego set, assuming that investors only purchase new sets (at retail price), without opening or playing those sets, this means that investors only resell sets with brand new condition. The columns of used condition in Brickowl dataset will not be included. In addition, sets with missing values in new condition will also be removed.

3.1.3 Joined

The data strategies are applied on the Brickset and Brickowl dataframe after performing an inner join on “set_number”. So that all the instances appearing in both tables are joined together in one dataframe. The columns are chosen as potential features excluding columns such as 'setID', 'number', 'numberVariant', 'bricksetURL', 'agerange_min', 'thumbnailURL', 'imageURL', 'height', 'width', 'depth', 'weight'. Which do not or have little predicting power.



```
Data columns (total 18 columns):
#  Column      Non-Null Count  Dtype
---  -
0  name         7777 non-null    object
1  year         7777 non-null    int64
2  theme        7777 non-null    object
3  themeGroup   7773 non-null    object
4  subtheme     6862 non-null    object
5  category     7777 non-null    object
6  released     7777 non-null    bool
7  pieces       7620 non-null    float64
8  minifigs     5220 non-null    float64
9  rating       7777 non-null    float64
10 reviewCount  7777 non-null    int64
11 packagingType 7777 non-null    object
12 availability  7777 non-null    object
13 dateFirstAvailable 4337 non-null    object
14 dateLastAvailable 3835 non-null    object
15 set_number    7777 non-null    object
16 NZ_retailPrice 4334 non-null    float64
17 mean_nzd_new  7777 non-null    float64
dtypes: bool(1), float64(5), int64(2), object(10)
```

Figure 3.1.a - Summary of the joined data frame contains 7777 observations and 18 columns.

For categorical columns, missing values will be filled with a new category called “Unknown”. For missing piece counts, Brickeconomy API was used.

	data
availability	promotional
currency	USD
current_value_new	6.47
current_value_used	4.13
current_value_used_high	4.96
current_value_used_low	4.05
forecast_value_new_2_years	8.29
forecast_value_new_5_years	10.36
minifigs	[rac061]
minifigs_count	1
name	Racer
pieces_count	30
price_events_new	[{'date': '2024-09-27', 'value': 6.47}, {'date': ...
price_events_used	[{'date': '2024-09-29', 'value': 4.13}, {'date': ...
retired	True
rolling_growth_12months	3.29
rolling_growth_lastyear	-3.4
set_number	30473-1
subtheme	City
theme	Juniors
upc	673419251075
year	2016

Figure 3.1.b - The response call on a set with missing value in pieces (30473-1) using Brickeconomy API including information needed (piece_count and set_number) to fill into the joined dataframe.

For dateFirstAvailable and dateLastAvailable, firstly a filter is applied only including observations which have values in dateLastAvailable, doing so will ensure the data only contains sets which have retired. Next step is to create a calculated column called “life_span_days” which is calculated by subtracting dateFirstAvailable by dateLastAvailable. Lastly, remove those dateLastAvailable and dateFirstAvailable columns as we now have those information in “life_span_days”.

For minifigures, not all Lego sets have minifigures as discussed in section 2.3.2, the missing values are filled with value 0.

In addition to data cleaning, the outliers identified are meaningful, hence will not be removed. However, it is important to acknowledge their impacts upon model building and that certain strategies such as log-transformation would be beneficial.

Another issue is imbalance distributed in some categorical columns such as theme or subtheme. Again, no action will be taken in this step. However, this is also important to acknowledge their impact during encoding as they are not ordinal variables but rather nominal variables which have no hierarchy in their order. Encoding techniques such as label encoding should be avoided (Jarapala, 2023).

3.2 Data Ethics

Several ethical considerations were addressed throughout this project, mainly concerning the data collection process. The majority of data used in this project was collected through APIs and Web Scraping.

Data collected via an API is an ethical practice, as the providers give open access to their data for public use assuming users will comply with the terms of service. Additionally, data we collected through APIs do not contain any personal data which could breach privacy.

Amazon was mainly used as a data source for web scraping but data collected via web scraping can sometimes be unethical because it can violate terms of service, invade user privacy, and can cause legal issues. Fortunately, Amazon allows web scraping as long as the scrapers are compliant, meaning that scraping data behind login walls, personal information or any other sensitive data is illegal and violates Amazon's terms of service. The methods used to web scrape the data from Amazon does not violate any of Amazon's policies, there is no names, contact information or any way of tracking each user. Each row of data is kept anonymously to each user since the only information of interest is the data of each LEGO set sold on Amazon.

Results

4.0 Data Visualization and Dashboard

To view and use this dashboard, refer to the file named "my_lego_dashboard.twbx" in the additional files folder of this reports supplementary material.

4.0.1 Data and design Process

All datasets were utilised to create the dashboard via relationships connection. The data sources are Brickset, Brickowl, Amazon Reviews and Rebrickable and they are connected using a common identifier "set_number". The Amazon Reviews data shows sentiment score of each review on a Lego set and contains variables such as sentiment score and reviewed country. The process of acquiring the data will be discussed in section 4.3 when we dive deeper into sentiment analysis.



Figure 4.0.1.a The relational schema shows the datasets used and their relationships.

The primary target audience for this dashboard is an individual who is new to LEGO and is interested in exploring investment opportunities in Lego sets. Therefore, the dashboard

consists of background information about Lego sets and its evolution overtime, the investability of sets within different themes, regional trends, and insights to start investing.

4.0.2 Tools and features

The tool used to create the interactive dashboard is Tableau and the dashboard contains 4 main pages with easy navigation between pages. The first page is a descriptive trend of lego sets, minifigures, and pieces evolution overtime. The second page then provides users with information about Lego colours and their ranking overtime, due to a big number of colours available, it is overwhelming for the users to observe and identify trends, hence the flexibility of adjusting how many colours they want to see is added as a filter.

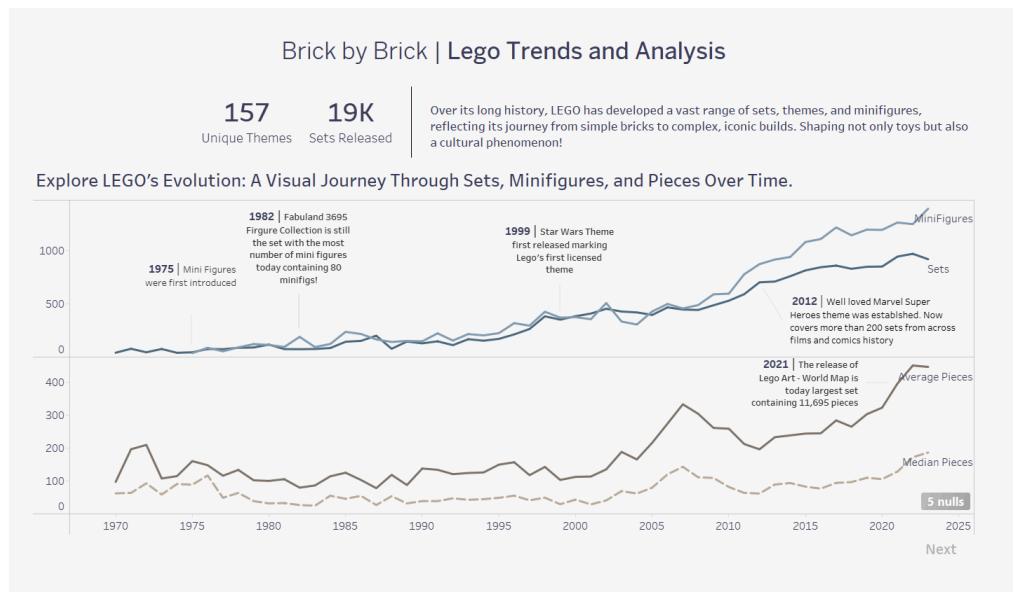


Figure 4.1.0.b The first page of the Lego dashboard provides descriptive trends about Lego sets over time.

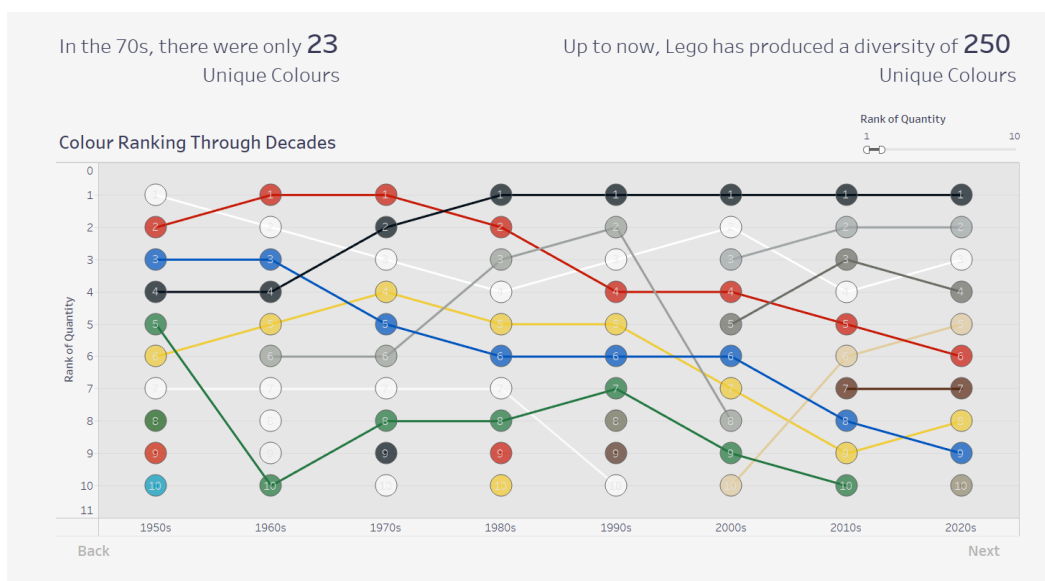


Figure 4.1.0.c Second page of the dashboard shows down trends in red and blue colours while black and grey are gaining popularity.

On the next page, users are introduced to the impact of theme and set availability on returns. A scatter plot illustrating the relationship between average retail price and average ROI by theme is provided, along with an interactive feature that allows users to click on a theme to drill down and explore the individual sets within it for deeper analysis.

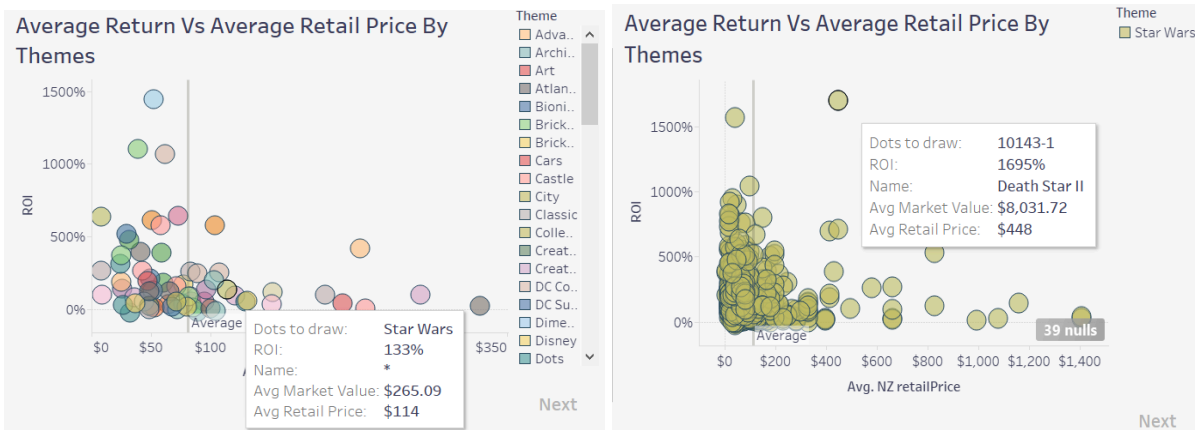


Figure 4.0.2.d - The scatter plot before drill down (left) showing themes hovering on Star Wars, and the scatter plot after drill down (left) showing sets within Star Wars theme - Death Star II has the highest ROI.

Finally, to offer insights into the regional trends in different parts of the world, the map displays the density of sellers and buyers in the world. The map itself is also a filter allowing users to click on a certain region to show the most loved and most hated themes in that country. This is particularly useful from an investor's point of view. For instance, if an investor lives in Canada and is looking into investing in Lego, he would benefit from information about the number of competitors and potential buyers, as well as the more favourable themes in Canada.

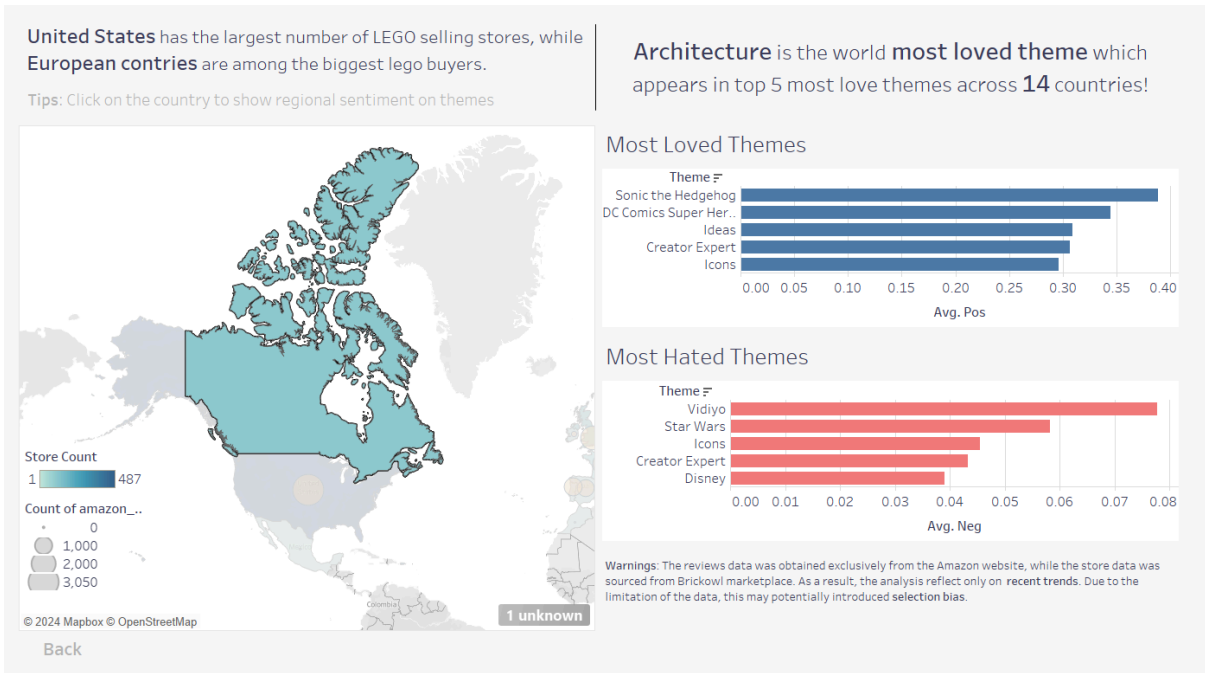


Figure 4.0.2.e - The last page of the dashboard when clicked on Canada to display the country's most loved and most hated themes.

4.1 Exploratory Data Analysis

4.1.1 Data overview

Exploratory data analysis is an important part of gaining understanding about our data with relation to potential target variables. The data aimed to be used for modelling would be the joined dataset from Brickset and Brickowl mentioned in section 3.1. The target variable chosen is annualised return on investment, the reason for this will be discussed further in section 4.4.

$$\text{Annualized ROI} = \left(\left(\frac{\text{Market Value}}{\text{Retail Price}} \right)^{\frac{1}{\text{Years Since Retirement}}} - 1 \right) \times 100$$

Figure 4.1.1.a - The formula to calculate annualised return on investment (target variable).

Moreover, the distribution of the target variable is highly right skewed, transformation methods are considered to be applied when building the model.

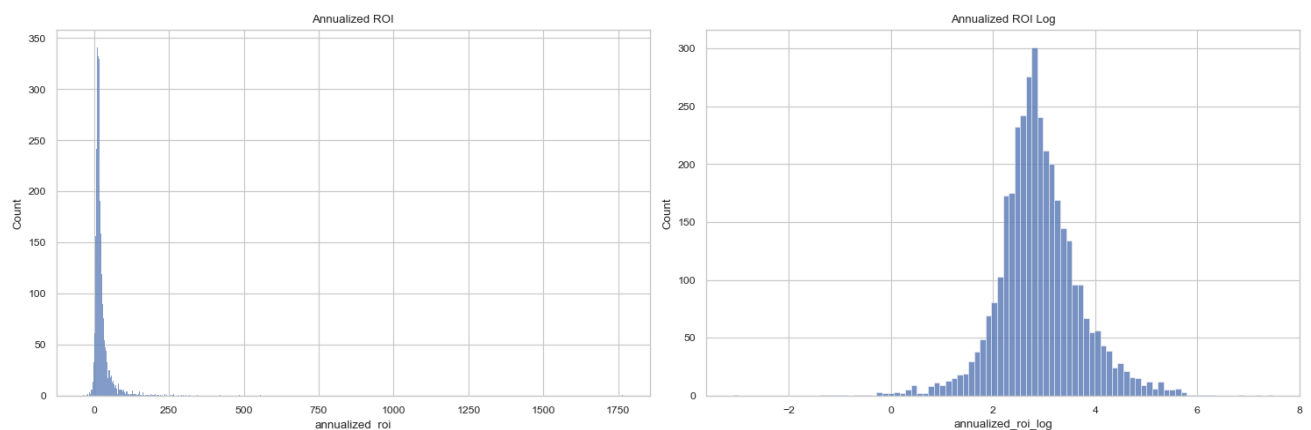


Figure 4.1.1.b - The actual distribution (left) vs the log-transformed distribution (right) of the target variable.

4.1.2 Correlations and Unusual features

During the process of exploring the features. There are three types of features available in this dataset, numerical, categorical and boolean. Each feature has a number of methods to explore their correlations with the target variable.

There is a weak positive correlation between the *year* and the target variable, and a weak negative correlation between *year_since_last_available* and the target variable. However, for the log-transformed target variable, the strength of those correlations are much stronger.

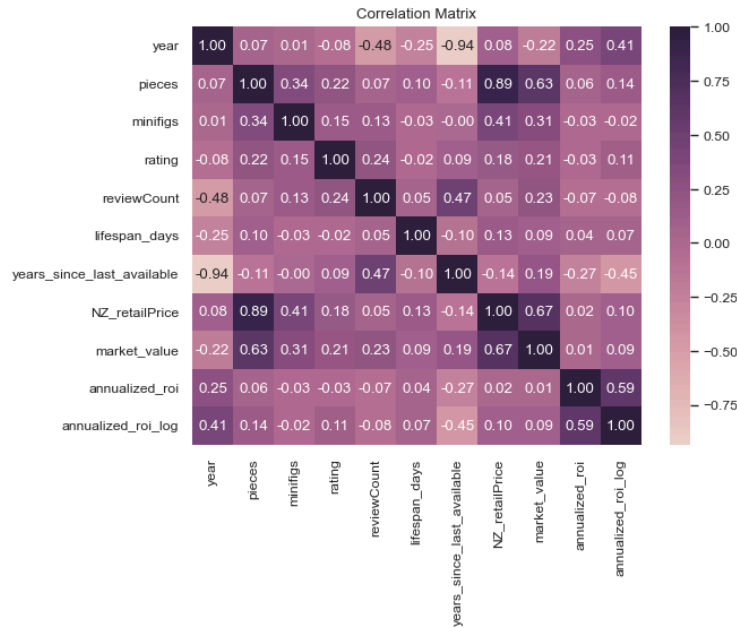


Figure 4.1.2.a - A heatmap showing correlation between numerical features against anualized_roi and anualized_roi_log.

Furthermore, scatter plots of each feature versus the target are utilised to look for any underlying patterns with the target variable. The strength of the correlations are affected by the presence of zero values in the features. A pattern shows that later-year sets typically have better annualised return on investment. There are some extremely high annualised ROI for sets with only 0 or 1 pieces. An example for this is the 'COMCON018 Batman' which is extremely rare and only given away for those who attended the comic con event in NY and held a special benchmark before the launch of *lego super heroes* in January 2012. (LEGO COMCON018 Batman, 2024). And lastly, sets containing between 0 to 10 mini figures have higher values.

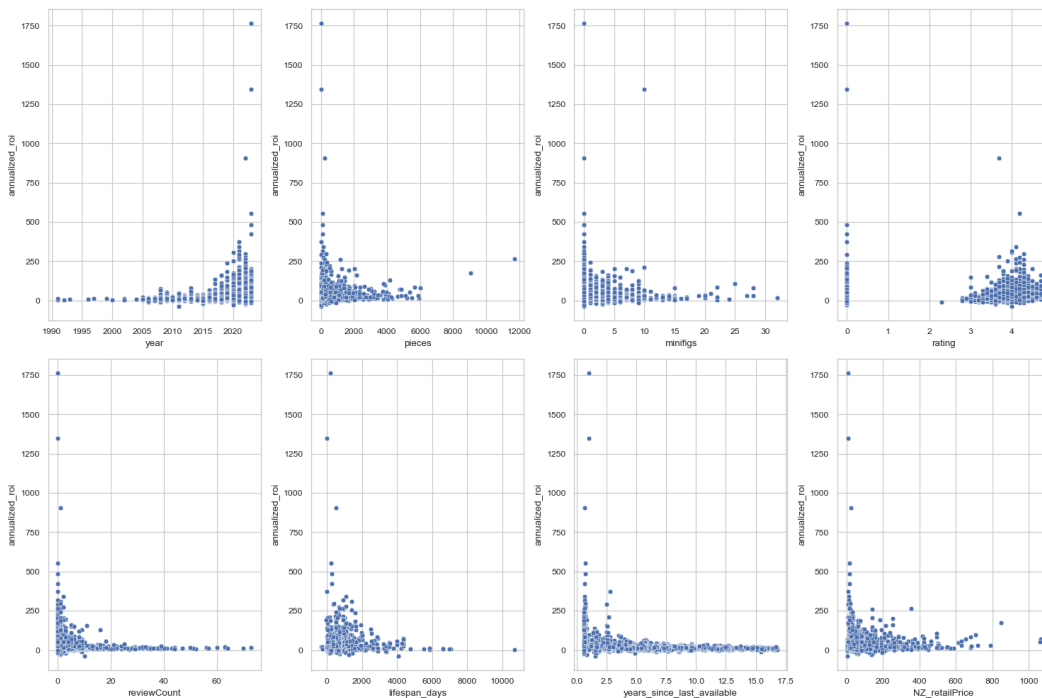


Figure 4.1.2.b - Pair plots of numerical features against the target variable.

While no interesting trend was found, an unusual feature was identified - rating. The majority of rating values are between 3 and 4. However, there are a large number of sets with rating value of zero. According to Brickset, the rating values are between 1 and 5. So the value zero actually means “no reviews” instead of a very low rating score (Huw, 2020). The solution for this is to create a new boolean variable called “rated” which has value 1 if the set's rating is between 1 and 5 and 0 otherwise.

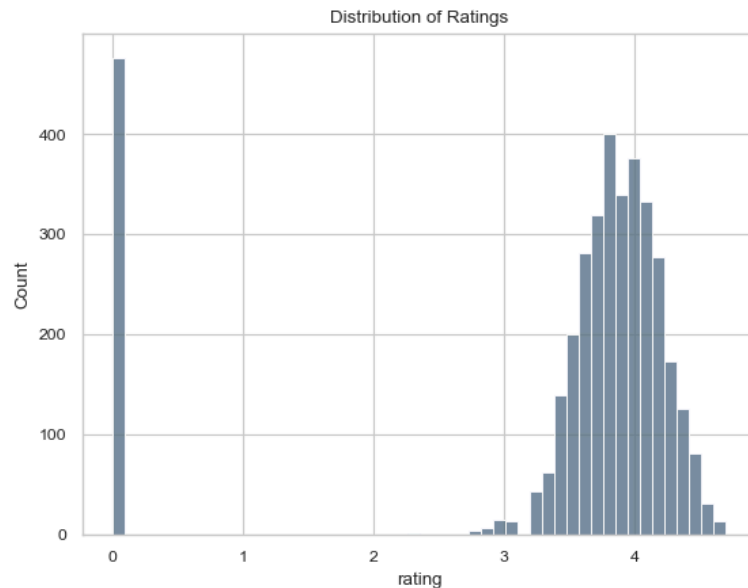


Figure 4.1.2.c - Shown the unusual distribution of rating at zero value.

For categorical features, the approach is to look at each category within a categorical feature and observe its impact on the target variable. While most of the categorical features do not have a significant variance with respect to the target variable, the theme feature has the highest variation between each theme with BrickHeadz stood out to have the majority of the sets with highest annualised ROI.

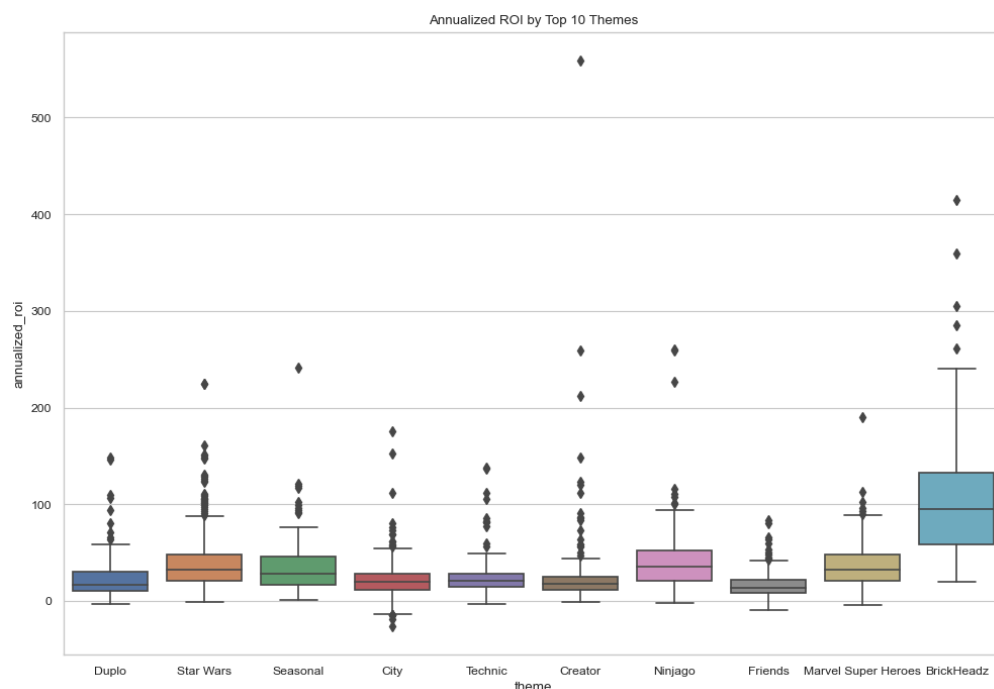


Figure 4.1.2.d - Boxplot showing the top 10 themes and their sets distribution against annualised ROI (removed outliers for better visual).

4.1.3 Variable creation

Initially, the attempt to create a new feature called “price_per_piece” was implemented to decrease multi-linearity as there is a very high correlation between retail price and pieces. However, a new problem occurred when dividing sets with zero pieces. As mentioned in section 2.3. The sets with zero pieces are meaningful sets such as collectable minifigures. It is a challenge to determine the “price_per_piece” for such sets. Therefore, the decision to move forward with this feature creation was postponed until a better solution is found.

4.2 Cluster Analysis

In this project, cluster analysis is used to reduce the complexity of large datasets. To effectively analyse such datasets, data wrangling is essential to prepare the data before the use of any modelling techniques. The analysis was done on several key variables namely pieces, minifigs, return, time_available, num_reviews, overall, min_retail_price, theme, year, themeGroup, and availability. By grouping together similar observations we were able to get meaningful insights on our dataset.

4.2.1 Data Wrangling

Some of the data used is categorical in nature and therefore couldn't be directly included while performing a cluster analysis and needed to be encoded before use. One-hot encoding was used for this purpose. This method prevents any ordinal relationship between categories which makes it suitable for use. The data was also scaled so that all data points can contribute equally to the analysis which will improve the interpretability of the model.

4.2.2 Methods

Different clustering algorithms were tested to identify unique groups in the dataset. Following the evaluation of their results, the best performing model was chosen, a choice that is critical for the analysis. The process began with K-means clustering. This method was initially chosen due to its ability to handle large datasets efficiently. The optimal value of ‘k’ was chosen using the elbow graph method (ZalaRushirajsinh, 2023). A cluster plot was made using this k value. The plot showed huge overlapping of the different clusters. Changing the k value does not address the issue, making it seem unsuitable for this dataset. Refer to Figure 4.2.a.

Next, Hierarchical clustering was applied. Given that most variables were continuous, Euclidean distance was utilised in this approach. A dendrogram was created to visually assess cluster relationships and determine the number of clusters. Refer Figure 4.2.b of section 4.2.3. This method faces computational inefficiencies with large datasets and so was not suitable.

The next algorithm used was the DBSCAN (Density-Based Spatial Clustering of Applications with Noise) algorithm. This ended up being the most suitable algorithm due to its ability in identifying clusters of different shapes and handling outliers. Silhouette scores were calculated for different combinations of epsilon distance and minimum point density

(Rousseeuw, 1987) . Initially, an epsilon distance of 1 and a minimum point density of 6 was used, resulting in three clusters, refer figure 4.2.c. After further trials, the parameters were adjusted to an epsilon distance of 0.7 and a minimum point density of 6, refer figure 4.2.d.

After encoding categorical variables and scaling numerical variables the dataset has become quite large and therefore it seemed better to use a PCA-transformed dataset. PCA reduces dimensionality and helps us focus on the most important features that explain the variance in the data. The algorithm was applied to the first and second principal components of the dataset as they capture the most variance in the data compared to the other principal components. The results of the clustering were visualised in plot 4.2.d.

4.2.3 Results

As seen in Figure 4.2.a the cluster overlaps despite changing the value of k. This is probably because k-means clustering assumes that the clusters are spherical and equally sized. The actual clusters are not spherical and are differently shaped making this to not be the most optimal algorithm for this dataset. We are also majorly dealing with continuous data. This type of data doesn't work well with k-means algorithm as it can be hard for the algorithm to identify boundaries and may not cluster points into distinct groups leading to overlapping regions.

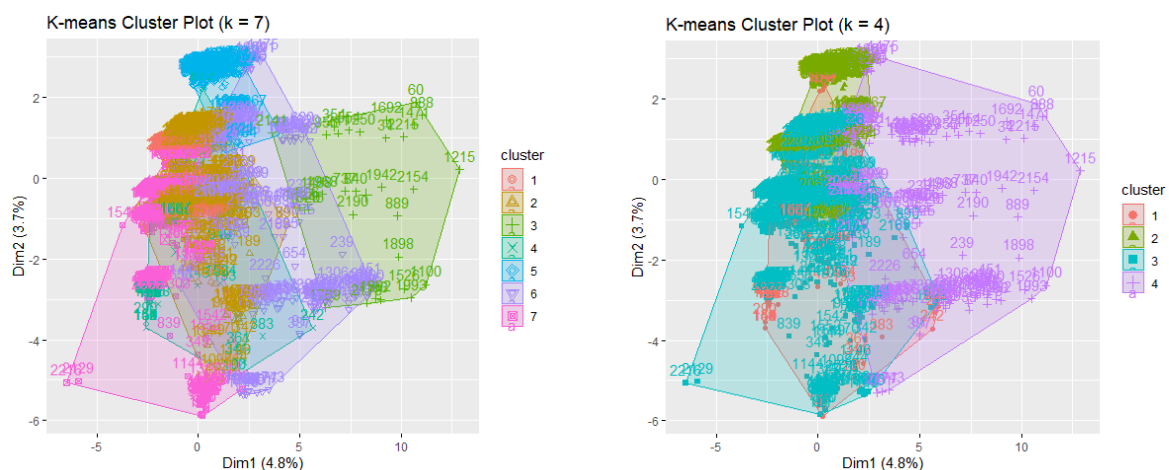


Figure 4.2.a
K means cluster plot showing overlapping of clusters for different k values

As seen in Figure 4.2.b we can see that there is an overly crowded dendrogram which is hard to read and get meaningful insights from. This could have been resolved using methods like pruning, however, insufficient guidance on best practices for pruning caused a hindrance in applying this technique effectively. Furthermore, constructing a dendrogram by itself required sorting and managing a large number of clusters and distances which added to the computational burden. Due to the size of the dataset, this algorithm required a significant amount of time to construct the plot which impacted the overall efficiency of the analysis.

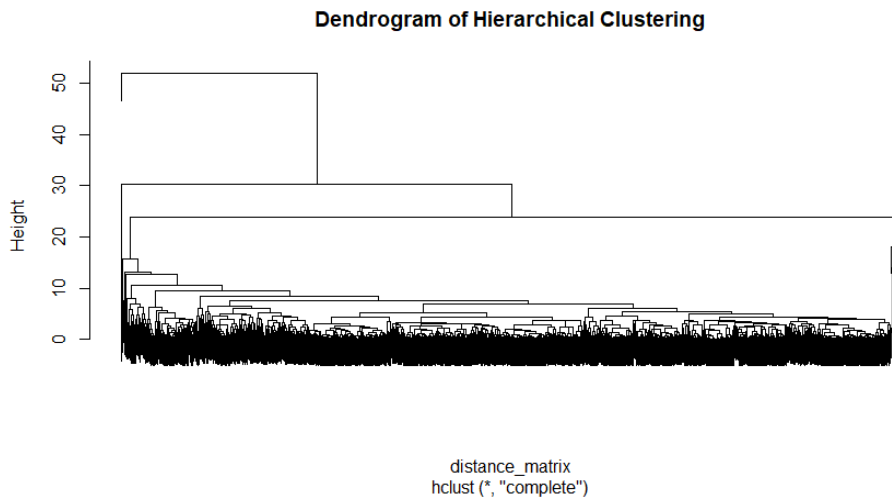


Figure 4.2.b
The Dendrogram of Hierarchical clustering

As seen in figure 4.2, Most data points were grouped in one large cluster, while the other cluster contained only a few points. This large cluster shows the existence of smaller groups within it. These findings show that using an epsilon distance of 1 and minimum point density of 6 are not the best parameters for this investigation.

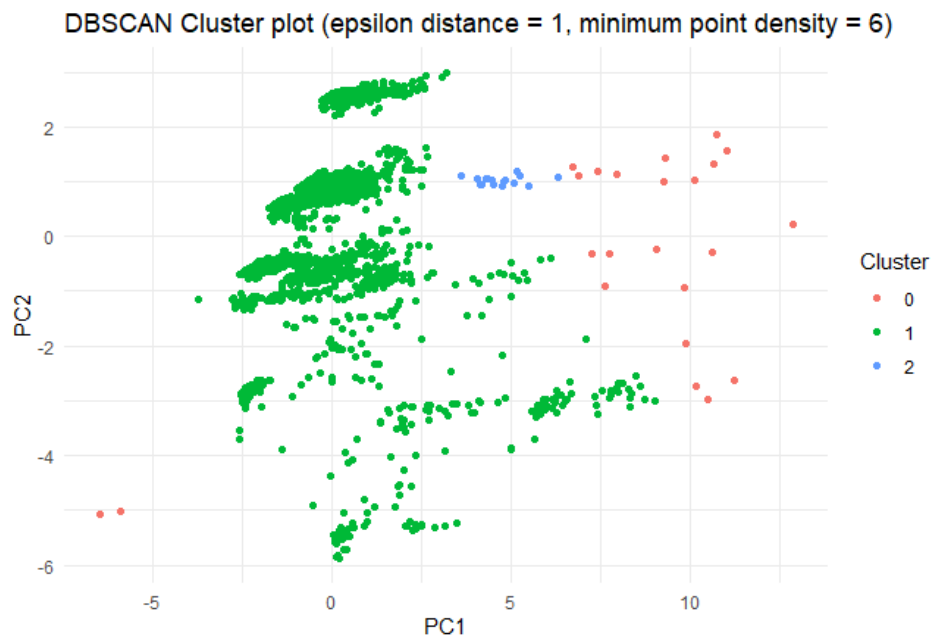


Figure 4.2.c
DBSCAN cluster plot (epsilon distance = 1, minimum point density = 6)

As seen in figure 4.2.d after careful adjustments of the parameters we are able to get 7 well formed clusters in the data. This configuration is well suited for our analysis as it clearly separates distinct groups which can help us gain meaningful insights of distinct groups.

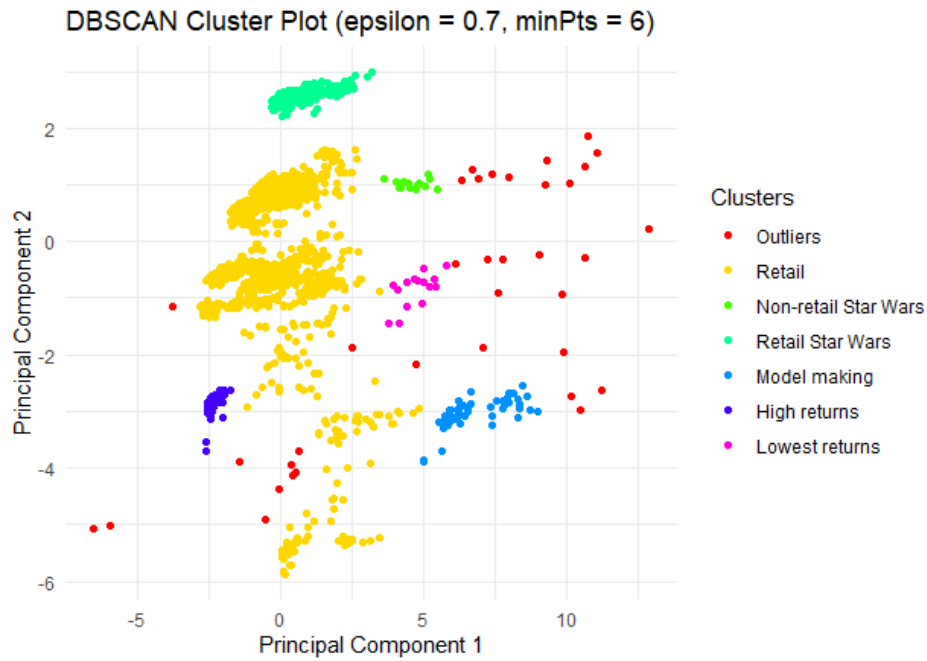


Figure 4.2.d
DBSCAN cluster plot (epsilon distance = 0.7, minimum point density = 6)

The DBSCAN algorithm resulted in 7 clusters that have been named based on the feature that best explains the cluster. The cluster names and the number of data points in each cluster are mentioned in Table 4.2.a. The Silhouette score was used to assess the quality of clustering.

Looking into each cluster the following observations were made:

CLUSTER NAME	NUMBER OF DATA POINTS
Outliers	36
Retail	1728
Non-retail Star Wars	17
Retail Star Wars	278
Model Making	46
High Returns	106
Low Returns	15

Table 4.2.a
Number of data points in each cluster

CLUSTER	PIECES COUNT	MARKET PRICE (\$)	RETAIL PRICE (\$)	RETURN (%)
Outliers	2716.555	959.094	394.862	144.995
Retail	344.507	137.902	56.42	168.896
Non-retail Star Wars	1871.411	927.739	288.942	234.678
Retail Star Wars	420.809	200.136	68.99	249.434
Model Making	1734.108	787.765	192.776	370.881
High Returns	37.934	49.788	11.984	652.440
Low Returns	2075.4	560.369	289.365	103.647

Table 4.2.b
Data collected on the different clusters

4.2.3.a Cluster 'Outliers':

Data points of this cluster have the highest piece count and market price. Visualising this cluster shows it contains primarily outliers or “noise points”. This cluster includes data points from themes like Star Wars, Creator and Harry Potter. Data points in this cluster are available as either retail or LEGO exclusive. With this information we can say that this cluster contains data points that don't conform to patterns found in other clusters.

THEME	THEME GROUP	COUNT
Star Wars	Licensed	11
Collectable Minifigures	Miscellaneous	4
Harry Potter	Licensed	3
Other	-	18

Table 4.2.c
Top 3 themes, theme groups and their count in cluster 'Outlier'

4.2.3.b Cluster 'Retail':

This is the largest cluster. This cluster has 10 themes in it. The most frequent themes are City, DC Comics Super Heroes, and Disney, with City being the largest, accounting for 391

of the 1728 data points. The availability of the data points in this cluster is majorly in retail with a few data points found in retail-limited and LEGO exclusive.

THEME	THEME GROUP	COUNT
City	Modern day	391
Disney	Licensed	77
DC Comics Super Heroes	Licensed	70
Other	-	1190

Table 4.2.d
Top 3 themes, theme groups and their count in cluster 'Retail'

4.2.3.c Cluster 'Non-retail Star Wars':

This cluster is made up of just two themes which both belong to the 'Licensed' theme group. This cluster was named after this point of commonality. Among the two themes Star Wars made up 16 of the 17 data points and are available as LEGO exclusive sets. The other theme that is a part of this cluster is Marvel Super Hero which is available in retail. This cluster also has the second highest average market price.

THEME	THEME GROUP	COUNT
Marvel Super Heroes	Licensed	1
Star Wars	Licensed	16

Table 4.2.e
Themes, theme groups and their count in cluster 'Licensed (excluding Star Wars)'

4.2.3.d Cluster 'Retail Star Wars':

This cluster only contains Star Wars retail sets. The average market price, retail price, and returns do not show any remarkable trends and seem to fall within the range of other clusters. This suggests that despite Star Wars being a popular theme the retail sets may not stand out significantly in terms of pricing or returns compared to other LEGO sets.

THEME	THEME GROUP	COUNT
Star Wars	Licensed	278

Table 4.2.f
Themes, theme groups and their count in cluster 'Star Wars'

4.2.3.e Cluster 'Model Making':

This cluster features a piece count that falls on the upper end of the spectrum. This also suggests that the sets are complex and detailed. It has 5 themes as a part of it out of which the top 3 themes which contain the most data points belong to the 'Model Making' theme group. These being model making sets explains the high average piece count of this cluster. The other theme groups are 'Historical' and 'Miscellaneous'. In this cluster, the market and retail prices are on the upper end, and the returns are also impressive. This suggests that the sets of this cluster appeal to consumers interested in higher-value, intricate LEGO sets. The mention of high returns also suggests that investing in these sets can be beneficial.

THEME	THEME GROUP	COUNT
Advanced models	Model making	15
Creator Expert	Model making	19
Icons	Model making	9
Other	-	3

Table 4.2.g
Top 3 themes, theme groups and their count in cluster 'Model making'

4.2.3.f Cluster 'High Returns':

This cluster has the highest returns of all clusters. This cluster also has the lowest market price and retail price. This cluster has only two themes which both belong to the 'Miscellaneous' theme group. One of these themes are collectable minifigures which have a low piece count. The sets of this cluster are available through retail channels.

THEME	THEME GROUP	COUNT
Collectable Minifigures	Miscellaneous	58
Dimensions	Miscellaneous	48

Table 4.2.h
Themes, theme groups and their count in cluster 'High Returns'

4.2.3.g Cluster 'Low Returns':

This cluster has the lowest returns, with an average market price. There are 9 themes that belong to this cluster. All other themes except icons are found as LEGO exclusive sets. Icons theme is available as retail. This cluster also has a piece count that is on the higher end.

THEME	THEME GROUP	COUNT
DC Comics Super Heroes	Licensed	5
Disney	Licensed	2
Icons	Model making	2
Other	-	6

Table 4.2.i
Top 3 themes, theme groups and their count in cluster 'Low Returns'

4.2.3.h Cluster Comparison:

Clusters 'Outliers', 'Non-Retail Star Wars' and 'Retail Star Wars' all have Star Wars as their largest themes. Star Wars of cluster 'Outliers' and cluster 'Licensed' are available as LEGO exclusive sets whereas in cluster 'Star Wars' it is found in retail. This suggests that Star Wars is a very popular theme that has a diverse form of availability. This indicates that different consumer segments are interested in Star Wars, ranging from collectors seeking exclusive items to casual buyers looking for accessible retail options.

Other themes, such as Disney and Collectible Minifigures, have most of their data points concentrated together but they also have a few points distributed across other clusters. This indicates that while these themes are primarily associated with a specific cluster, they still have some representation in other categories.

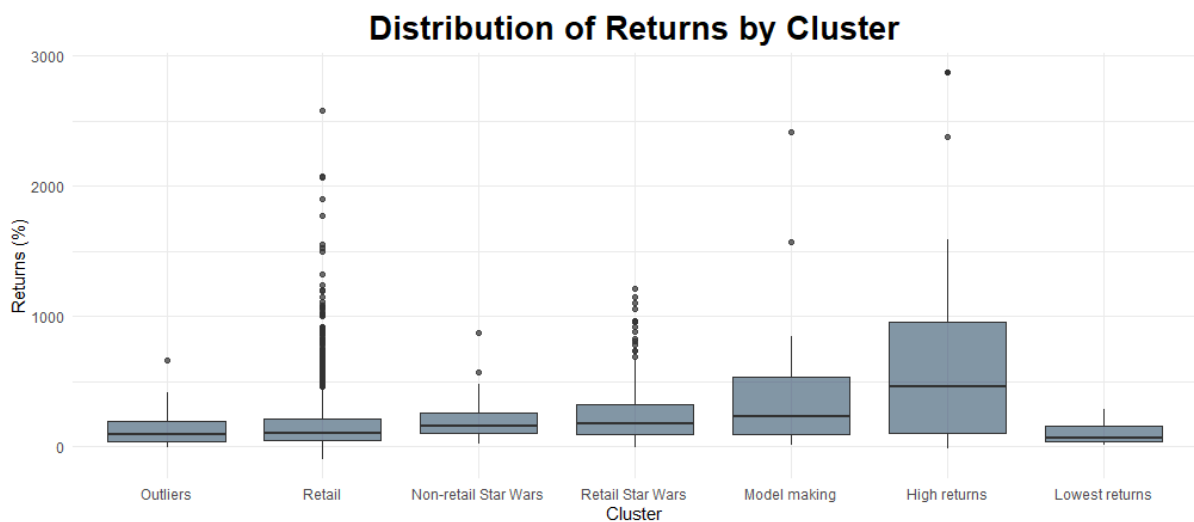


Figure 4.2.e
Box plot of Returns (%) of each cluster

As mentioned before, cluster 'High Returns' has the largest returns compared to the other clusters. Cluster 'Model Making' also has returns on the higher end. However, the market price and retail price of cluster 'Model Making' is greater than that of 'High Returns'. This indicates that while investing in more sets from the 'High Returns' cluster may yield greater

returns, those interested in owning a limited number of sets with similar returns might find it more advantageous to invest in the 'Model Making' cluster.

4.3 Sentiment Analysis

Analysing customer reviews is an important process for this project. The insights gained from examining reviews can be valuable. The reviews used for this sentiment analysis were collected from Amazon via web-scraping, each review is rated from 1 to 5 stars. With the provided information, reasons as to why the customer gave that review can be acquired. Insights from analysing product reviews could be useful for investors, as there could be some correlation between a set worth investing and its reviews.

4.3.1 Data Wrangling

The data must be acquired prior to performing the sentiment analysis. The data of interest is contained within each product's review page. The steps to collect the amazon reviews involved these steps: From the base page of all LEGO products, scrape each link and append it to a python list. After all the links to the products have been obtained, go into each link which will direct to the individual product page and go to the page that displays all of the reviews. Scrape all the reviews on each page and save it into a Pandas dataframe. Once all the review data is in the pandas dataframe, the data types for each column need to be changed from an 'object' to either a string or an integer type.

Across the review data set, there are several different languages to analyse, due to customers purchasing LEGO from Amazon globally. These different languages are not able to be analysed to get any meaningful results. The 'langdetect' and 'deep-translator' python modules were used to translate all of the reviews from their original languages to English. The initial plan was to use the 'translate' module and use its auto detect language function to detect what language the customer wrote in as a language code in the form of ISO 639, and then pass the language code into the 'Translator' function to translate from the customer's language to English. However, after extensive testing, this module failed to automatically detect languages correctly. In many cases it was misinterpreting the customer's language for a completely different language, so when the code attempted to translate it to English, a completely unexpected English sentence was returned.

The translate module was also not able to be used because it has a 5000 character translate limit, after using the 5000 character limit there is a 24 hour cooldown which is not a sufficient time to be able to complete the project by the deadline. The deep-translator module was used instead because there is no character-translate limit and it is able to support the usage of Google Translate which is one of the most reliable translating services, some reports show Google Translate reaching 94% accuracy (Harby, A, 2023).

The langdetect module is able to accurately detect what language a foreign sentence is written in and return the language as a language code. This language code could then be passed into the translator so the translator knows exactly what language to translate from. Dask was used in this process to optimise this process of language translation using parallelization.

4.3.2 Frequent Items

4.3.2.a Methods

This section is aimed towards identifying associations in Amazon customer reviews using a fairly basic algorithm. This algorithm looks at all the customer reviews in 1, 2, 3, 4 or 5 star ratings and finds the most commonly used n number of words across all the reviews. This could show insight as to why products get certain ratings. In this case the algorithm found the 20 most common words in each star rating. The 'stopwords' package from the NLTK module (Natural Language Toolkit) was used to remove meaningless words (e.g. 'a', 'the', 'I', 'that', etc) from the algorithm, meaning that there wouldn't be any meaningless words in the output.

4.3.2.b Results

One Star	Two Star	Three Star	Four Star	Five Star
lego: 86 box: 53 set: 41 product: 32 missing: 29 pieces: 20 bought: 18 damaged: 18 time: 18 packaging: 17 amazon: 16	lego: 85 box: 30 set: 28 missing: 25 pieces: 24 product: 19 good: 17 bricks: 15 figure: 15 beatbits: 15 children: 14 bought: 13 bags: 13 time: 13 price: 13	box: 62 pieces: 61 fun: 50 price: 47 build: 44 well: 44 instructions : 44 small: 39 nice: 39 great: 36	great: 210 good: 182 price: 178 fun: 169 pieces: 160 build: 138 little: 130 really: 128 instructions : 121 box: 117 mario: 114	great: 1276 fun: 800 build: 755 good: 666 gift: 555 price: 551 son: 550 pieces: 540 well: 522 building: 508 quality: 507 together: 455

Table 4.3.2.a - Frequent Items Results of Interest

The most common words from the one star reviews imply that these customers had problems with Amazon or they thought that it was just a bad product. The output(s) of interest are shown in table 4.32.a for this and the following ratings.

The most common words from the two star reviews suggest that these customers had similar issues from the customers who gave one star reviews. It is interesting to note that 'beat bits' is among the most common words. BeatBits are a LEGO 2x2 flat tile with a design on the surface to represent a unique album cover. There are 104 different BeatBits which are obtained via purchasing a Collectable MiniFigure (mystery box). Or a BeatBox(more expensive but contains more BeatBits). Since the word 'beatbits' was within the most common words among two star reviews, it could be evidence to show that BeatBits are not good LEGO products and potentially not worth Investing in.

The results from the three star reviews don't show many items of interest, however, it is worth noting that 'small' is among the results. It could mean that customers thought that the product they purchased was smaller than they expected.

The four star results show positive words like 'great', 'good', and 'fun' which would clearly show that the product itself was good. Interesting to note that 'mario' was among the results. This could mean that many customers liked the LEGO Mario set and could potentially be a good set for investors to invest in.

The results from the five star reviews are all positive words, it is clear that the customers are very satisfied with the products they bought.

4.3.3 Frequent Itemsets

4.3.3.a Methods

This section is aimed towards identifying frequent patterns/associations in Amazon customer reviews to discover any potential reasons why customers left 1, 2, 3, 4, or 5 star ratings. The Apriori algorithm was applied to the review data, this approach was used to identify pairs of words or features commonly mentioned by many customers in their reviews, which could aid in explaining the reasoning behind different ratings. This algorithm starts working by counting each individual word across all the reviews in each star rating and storing the counts for each word in a table. The algorithm will then only keep the words if their count is the same or higher than the minimum support level. To generate the 'frequent pairs' the algorithm combines each frequent item with another frequent item to form a pair, each pair will only be kept if it is the same or higher than the minimum support level. It is also possible to form 'frequent triplets' or more, but creating groups bigger than pairs can make it difficult for each group to remain above the support threshold so many of the groups could be lost.

One Star	Two Star	Three Star
(('box', 'damaged'), 11) (('box', 'product'), 10) (('box', 'missing'), 9) (('missing', 'pieces'), 9)	(('bought', 'lego'), 7) (('disappointed', 'lego'), 7) (('children', 'lego'), 7) (('lego', 'price'), 7) (('bought', 'disappointed'), 6) (('lego', 'pieces'), 6)	(('lego', 'price'), 27) (('lego', 'really'), 25) (('lego', 'pieces'), 25) (('fun', 'lego'), 24) (('great', 'lego'), 20)

Four Star	Five Star
(('lego', 'set'), 136) (('great', 'lego'), 90) (('lego', 'price'), 84) (('fun', 'lego'), 83) (('great', 'set'), 80) (('fun', 'set'), 74) (('instructions', 'lego'), 71) (('price', 'set'), 65)	(('great', 'set'), 367) (('fun', 'lego'), 363) (('build', 'lego'), 357) (('lego', 'quality'), 312) (('lego', 'pieces'), 291) (('lego', 'price'), 286) (('fun', 'set'), 282) (('good', 'lego'), 273)

<pre>(('good', 'set'), 63) (('build', 'lego'), 63)</pre>	<pre>(('build', 'set'), 258)</pre>
--	-------------------------------------

Table 4.3.3.a - Results of interest for the Apriori Algorithm.

The Apriori algorithm was performed on five separate groups, where each group was its star rating. In the one star rating there were pairs that suggested something was wrong with Amazon's delivery services or problems that were out of LEGO's control, for this and the following section refer to table 4.3.3.a for this and the following results.

4.3.3.b Results

The two star rating results show that the customer was not satisfied with the product itself and there probably was not anything wrong with the delivery services. These results imply that the customer was disappointed, potentially due to the price or number of pieces.

The results in the three star ratings do not show any evidence of reasons to be rated 3 out of 5 stars. This may be because 3 stars is considered 'average' and there isn't much to be said positively or negatively about it.

The results in the four star ratings could suggest that the sets were rated four stars because customers got some enjoyment from the set itself. However, there is not any evidence to show why they got four instead of five stars. This could be because some of the customers are picky and do not give five star ratings often.

The results in the five star ratings are similar to the results in the four star ratings. The results show that the customers had lots of fun with their purchased set and it was likely a good price and quality.

4.3.4 Sentiment Scores

4.3.4.a Methods

The goal of this section is to investigate if there are any countries which tend to write more impolite reviews. The Natural Language Toolkit in Python comes with a sentiment analyzer. This was used to estimate how positive or negative each review was on a scale from 0 - 1. The sentiment analyzer calculates the positive, negative, neutral and compound levels for each translated review based on the words and phrases in each text. The two letter language codes used in this section are obtained from the langdetect module in Python which are in the form of ISO 639.

4.3.4.b Results

Plotting each language against each languages' average negative sentiment score showed that Polish(pl) and Dutch(nl) speakers wrote significantly more negative reviews than other languages as shown in figure 4.3.4.a.

Figure 4.3.3.b shows that Catalan and Norwegian speakers had a significantly higher average positive sentiment score, also worth noting that Catalan speakers had the lowest negative sentiment score as shown in figure 4.3.4.b.

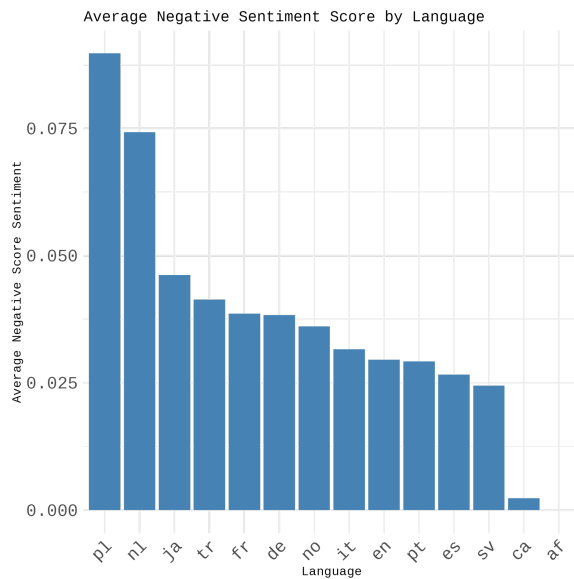


Figure 4.3.4.a

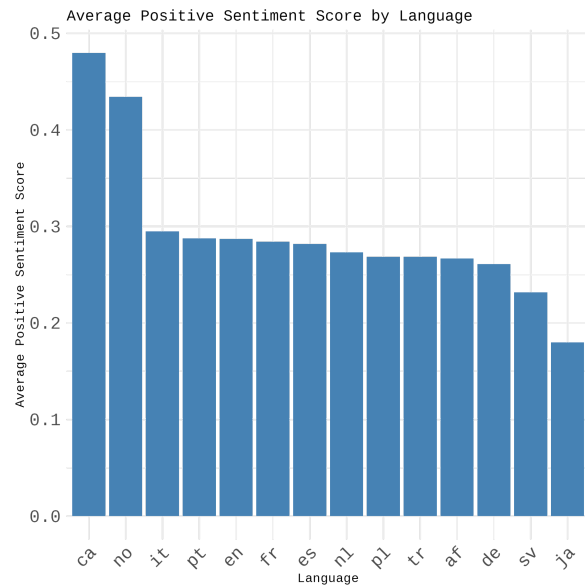


Figure 4.3.4.b

The purpose of Table 4.3.4.a is to compare each language's negative and positive sentiment score and determine if the people speaking that language are considered impolite or not. To calculate the 'is_rude' column, a custom metric was used in the form of $(average_neg * 10) - average_pos > 0.3$. If the result is greater than 0.3 then that language is considered rude. This metric was created to identify 'rude' languages by investigating if there was a significant difference between negative and positive sentiment scores. This is because a language with a high negative sentiment score are not considered rude if they also have a high positive sentiment score

The languages with a negative *rudeness_metric* value mean that they have a higher average positive sentiment score than negative sentiment score which means they are more polite than they are rude. Table 4.34.A shows that Polish and Dutch speakers are considered rude since their *rudeness_metric* is significantly higher than 0.3, also Japan's *rudeness_metric* is 0.282 which is close to 0.3, there is some evidence to suggest that Japanese speakers can be more rude in LEGO reviews. Spanish, Norwegian, Afrikaans and Catalan speakers have a negative rudeness metric which shows that they write more positive than negative LEGO reviews.

	language	average_neg	average_pos	count	rudeness_metric	is_rude
	<chr>	<dbl>	<dbl>	<int>	<dbl>	<lgl>
1	pl	0.0897	0.269	35	0.629	TRUE
2	nl	0.0742	0.273	32	0.469	TRUE
3	ja	0.0462	0.180	138	0.282	FALSE
4	tr	0.0414	0.268	93	0.145	FALSE
5	de	0.0383	0.261	698	0.122	FALSE
6	fr	0.0386	0.284	452	0.102	FALSE
7	it	0.0316	0.295	594	0.0216	FALSE
8	sv	0.0244	0.232	29	0.0128	FALSE
9	en	0.0295	0.287	2409	0.00808	FALSE
10	pt	0.0292	0.288	75	0.00476	FALSE
11	es	0.0266	0.282	507	-0.0158	FALSE
12	no	0.0361	0.434	12	-0.0733	FALSE
13	af	0	0.267	10	-0.267	FALSE
14	ca	0.00233	0.480	15	-0.456	FALSE

Table 4.3.4.a. - Identifying Rude languages based on the rudeness metric.

Table 4.3.4.b is the contrary of Table 4.3.4.a, it is used to determine polite languages as opposed to negative. The metric was used in the same way except ($average_neg * 10$) is subtracted off $average_pos$. If this value is greater than 0.3 then the speakers of the language are considered polite in Amazon LEGO reviews. The table shows that Catalan speakers write significantly more polite LEGO reviews than other countries. Afrikaans speakers could also be considered polite since their politeness metric is close to 0.3 at 0.267.

	language	average_neg	average_pos	count	politeness_metric	is_polite
	<chr>	<dbl>	<dbl>	<int>	<dbl>	<lgl>
1	ca	0.00233	0.480	15	0.456	TRUE
2	af	0	0.267	10	0.267	FALSE
3	no	0.0361	0.434	12	0.0733	FALSE
4	es	0.0266	0.282	507	0.0158	FALSE
5	pt	0.0292	0.288	75	-0.00476	FALSE
6	en	0.0295	0.287	2409	-0.00808	FALSE
7	sv	0.0244	0.232	29	-0.0128	FALSE
8	it	0.0316	0.295	594	-0.0216	FALSE
9	fr	0.0386	0.284	452	-0.102	FALSE
10	de	0.0383	0.261	698	-0.122	FALSE
11	tr	0.0414	0.268	93	-0.145	FALSE
12	ja	0.0462	0.180	138	-0.282	FALSE
13	nl	0.0742	0.273	32	-0.469	FALSE
14	pl	0.0897	0.269	35	-0.629	FALSE

Table 4.3.4.b - Identifying polite languages based on the politeness metric.

The results of this sentiment analysis show that Polish and Dutch speakers tend to write more impolite reviews on LEGO products sold on Amazon with some evidence to suggest Japanese speakers are also impolite, and Catalan speakers write more positive reviews with some evidence to suggest Afrikaans speakers are also polite too. It is also important to note the number of occurrences for each language, since the most polite and most impolite languages had less than 100 occurrences. It is possible that the more occurrences a language has, the more neutral their average review's sentiment score tends to get. This is because of the 'Law of Large Numbers', as the sample size increases, then the sample size will approach the population mean (Bhandari, P. 2023).

4.4 Predictive Modelling

Through prior analyses, it is apparent investing in Lego may be a viable option, however, this appears to depend on a number of factors. To determine which are most important when assessing a set's investability, as well as predicting the potential return of sets soon to retire, modelling methods such as linear regression, forests and gradient boosting can be used.

Market price (value in which a Lego set can be sold for) and an estimated return per year (measured as a percentage) models were created and tested during this project. Predicting a set's market price provides little inherent value to an investor as it does not indicate profit, nor does it show the timeliness of such. As such, focus was put on predicting the estimated return per year of a lego set as a percentage. This will be referred to as ROI PA, or return on investment per annum.

Additionally, this modelling was attempted using two approaches. With the first being a more simple rudimentary approach utilising available data. The second being a more in-depth method which also aimed to utilise incomplete data. A discussion of both and a comparison of their results will be conducted, with the best then being used in the recommendation of the investability of lego sets.

Additionally, to aid in the consideration of risk due to the nature of predicting financial information, a measure of “risk” or rather the variance of returns must also be taken into consideration to assess the investability of lego sets. For this analysis, standard deviation and the Sharpe Ratio will be considered to assess the risk adjusted expected return of investing in Lego sets (Sharpe, 1994). This will be discussed following the creation of soon to retire lego sets ROI PA predictions.

4.4.1 Measuring error

In order to determine the best model, it must first be determined the way in which the accuracy and effectiveness of a model will be measured. There are a few measurements that can be used, namely, the Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) can be used to assess the accuracy of predictive models. The first is useful as it shows how far away on average the estimates of our model are, with the second also being useful as it becomes weighted depending on the size of the error, with larger errors becoming proportionately worse in the calculation. MAE is therefore useful if there are concerns about outliers skewing the data, and RMSE is when assessing the all-things-considered accuracy of the model. Both will be of use throughout the model tuning and eventual model selection process.

Some of the categorical variables within the dataset, as previously discussed, have a significant number of levels. As a result of this, it is of high importance that models constructed do their best so as to not overfit on this data. In order to minimise the risk of such happening, cross validation of models will be conducted when assessing their errors so the errors being compared are based on not yet seen data, and thus a true test of a models ability to predict.

4.4.2 Simple approach

Although there may be over 20,000 lego sets in existence (Brickset, n.d.), the number of sets with complete data is much less. The first approach from which this project attempted to predict the investability of lego sets was a simple one, training a model using only data for which there is complete data

4.4.2.1 Data wrangling

The base dataset used for this approach was the previously outlined “Joined” dataset, with all rows containing missing values from the following variables deleted: pieces, minifigs, return, time_available, num_reviews, overall, min_retail_price, theme, year, themeGroup and availability.

The review data from brickset was processed, with the average ratings and sentiment scores of reviews for each theme calculated. Averaging was done across themes due to many sets and subthemes having little to no reviews. Each lego set was then assigned its themes ratings and sentiment scores as being its own. This sentiment analysis was completed using the natural language toolkit (NLTK, 2024), with sentiment scores taken from the title and review of all Brickset reviews.

The modified “Joined” dataset was then combined with the sentiment score review data, leaving a dataset containing 2227 legosets, or observations, which can be used for predictive modelling.

4.4.2.2 Methods

4.4.2.2.a Regression models

To limit the chances of overfitting occurring, the dataset was split into a training and testing dataset at a 70/30 split. Since there are many sub themes and themes with a limited number of observations. Any themes or subthemes only present in only one of the training or testing sets were removed, this was for two reasons. Firstly, it avoids errors which would come up in both testing and the creation of tree based models. Additionally, as this report will get to, a theme or subtheme with so few observations will not be considered for recommendation, so actionable insights are not being lost. This left a training set of 1062 and a testing set of 511.

Prior to the creation of a “model”, a baseline was created. This baseline model is simply the average of the response variable, which in this case is ROI PA. This acts as a comparison to models created to see how much explanatory power certain variables are actually able to provide over simply guessing. The first models to be tested were linear regression models. The first of these being a simple additive regression as a further baseline for future progression. Following this, log-transformed variables and interaction effects will be used.

Principal components analysis was implemented to attempt to model the data and had a significantly higher error, similar to that of our baseline, and thus was not pursued further.

4.4.2.2.b Tree Models

Both regression and boosting tree methods were attempted. The first of these was similar to that of the first linear approach, new interaction variables were created that were observed to correlate to a lower error rate in the regression approach.

Beyond this, generalised boosting will be used. Testing will be done using human intervention to lower the error by selecting hyper-parameters and variables in the model, with a grid-search approach then being used to tune these hyper-parameters. This grid search will be completed using 10-fold cross validation to ensure reliable and well-fit results.

4.4.2.3 Results

4.4.2.3.a Regression Models

Model	MSE	RMSE	MAE
PCA Model	186.38	13.65	9.98
Null Model (guessing the mean)	171.00	13.08	9.56
Linear Regression (First implementation)	131.96	11.49	8.37
Linear Regression (With interaction effects)	128.54	11.38	8.31
Linear Regression (After regularisation)	121.58	11.03	8.12
Regression Tree	119.63	10.94	7.47
Generalised Boosting Tree	104.98	10.25	7.30
Generalised Boosting (Tuned Hyperparameters)	103.70	10.16	7.24

Figure 4.42.3a

The mean ROI PA for the dataset was 20.31%, this indicates that on average, lego sets will appreciate 20.31% each year. As a predictive model, this gave a Mean Absolute Error (MAE) of 9.56 and Root Mean Squared Error (RMSE) of 13.08. These will serve as the baseline error measures to which we can compare other models.

Using linear regression an MAE of 8.37 was obtained. As expected, certain themes and subthemes could be seen to have varying effects on ROI PA. The negative coefficient on piece count of -0.0067 was not expected, as there had previously been observed a positive correlation between piece count and return, indicating that on average with all else held constant, each additional piece in a lego set nets on average 0.0067% less growth in returns per year. It had also previously been observed that there was a correlation between the value of a set and its number of minifigures. However, The coefficient of minifigures came out to be -0.66 for this model, meaning that on average with else held constant, each additional minifigure a set has will decrease its annualised percentage return by 0.66%. As explained by the Brick Economy “pop factor”, the negative coefficient on time since available of -2.21 means that the annual return of a set declines over time (BrickEconomy, n.d.), and the negative coefficient on time available of -0.50 indicates that the longer a set is available for purchase, the lower its return is.

By introducing an interaction term to account for the fact that the effect piece count, time since available and retail price have on ROI PA is dependent on one another, the testing set MAE was lowered to 8.31. This can be further improved upon by using regularisation to limit the variables and their coefficient sizes to reduce overfitting the model to our training data, which can often happen with high dimensionality data. Prior to regularisation, the training error was only 6.03. The new simplified and optimised linear model had a training error of only 7.38, this limited overfitting translated to a testing MAE of only 8.12, a significant improvement on this type of model.

4.4.2.3.b Tree Models

This first regression tree performed well, with a MAE of 7.47, higher than that first observed by our linear regression model. After tuning the Complexity Parameter (CP) to 0.02, this lowered the MAE down to 7.47.

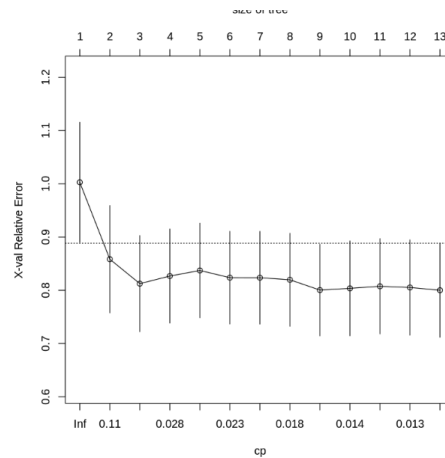


Figure 4.4.2.4.a showing the error relative to the CP value for the implemented regression tree model

A similar project predicting market price of lego sets found all tree methods to outperform all regression methods (Tang, 2024). Even though the dataset used in this project differs from that used in this project,

Using generalised boosting, an initial MAE of 7.45 was obtained. Some modifying of hyper-parameters and variables gave a lower MAE of 7.30. Namely, subtheme was removed due to the overfitting it caused, likely due to the limited number of observations several subthemes have. This is likely because Following the grid search, the MAE was further lowered to 7.24. The key difference the hyper-parameter tuning made was toning down the relative feature importance of the theme.

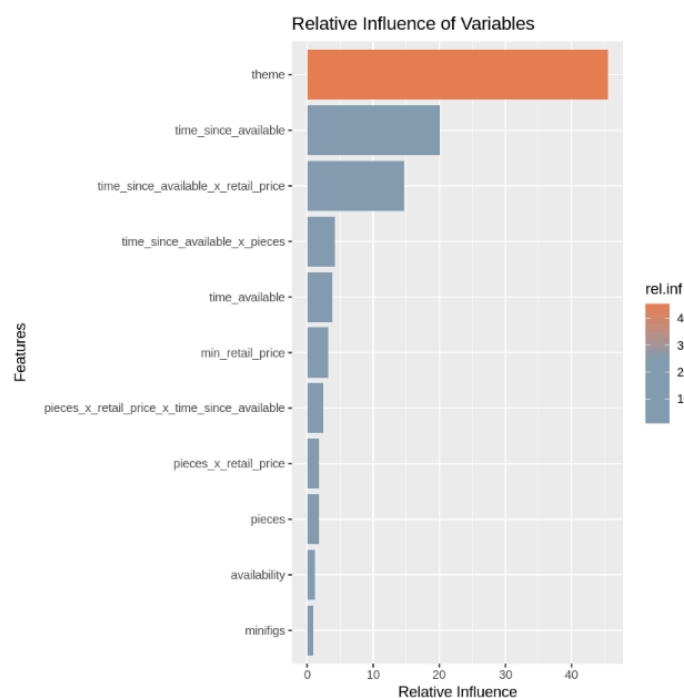


Figure 4.4.2.4.b showing the relative feature influence of the final generalised boosting model

The tuning of parameters decreased the theme's relative feature importance from 63 down to 46, with the key being significantly increasing the minimum observation of nodes up to 25 and having a high interaction depth and number of trees of 7 and 200 respectively to further generalise the model and not overfit it to the training set. Time since available and retail price are seen to also be important variables in the model.

4.4.3 Advanced approach

In addition to the simple modelling approach taken, a more advanced one was also conducted which sought to mitigate some of the issues which the simple one seemed to have. The primary issue was missing and incomplete data, by using inference and other datasets, a larger dataset for model training can be created, improving accuracy. Although this approach was slightly different, the core goal and methodology remained the same, creating a means to determine the investability and expected return of a given lego set using available data.

4.4.3.1 Data Wrangling

It is worth noting that this approach didn't use sentiment analysis features due to their again being many missing data points, in combination with the fact that its effect on modelling was very insignificant.

The steps taken to handling missing occurrences in the datasets was discussed in depth in section 3.1. After cleaning, the dataset contains 3698 entries and 21 columns, with no missing values, available to be used to train a new set of models. This is a significantly larger dataset than that which was created for the previous approach, and it is hoped that this will deliver a more accurate model.

4.4.3.2 Methods

4.4.3.2.a Preprocessing

Before the model building process, several preprocessing steps are required, including encoding categorical variables and applying feature scaling.

The strategy used to encode categorical variables is target encoding. Target encoding is a useful encoding method when it comes to encoding categorical variables with high cardinality such as theme or subtheme observed in section 2.3.4. Each category is encoded using a shrunk estimate of the average target values for observations within that category. This encoding approach blends the global target mean with the target mean specific to each category, providing a more robust estimate that accounts for both the overall trend and the category-specific behaviour (TargetEncoder, 2024).

It is important to perform feature scaling because in some distance based algorithms such as Support Vector Machines or KNN regressor, having features in different scales will negatively affect model performance (YAĞCI, 2021). Tree-based and OLS algorithms are not impacted by feature scaling, as node splitting is performed on a single feature, the split of a feature is not influenced by other features. Considering various regression algorithms will be

tried, standard scaling was performed on all features. Standard scaler scales the features to mean zero and has a unit variance (Scikit-Learn, 2019).

Firstly, the attempt to fit the model without any scaling was made to later compare the impact of scaling to the models' performances. The dataset was first divided into a training set and a test set - test size is 20%, models will be trained on the training set and evaluate performances on the test set. Table 4.4.3.a shows various error metrics from the models.

After that, feature scaling was performed using Scikit-learn's StandardScaler. The results is summarised in table 4.4.3.b

4.4.3.2.b Validation

Moving forward, validation methods are used to ensure the models can generalise to unseen data and that the performance observed is not products of randomness. K- Fold cross validation is the method used for validation. K-Fold Cross-Validation divides the entire training data into K distinct subsets of roughly equal size before dividing the data into training and testing sets. Each subgroup is then separated into testing and training sets. The model is trained and tested on each subgroup. In actual use, this method produces K distinct models with K distinct outcomes. The average of each subset's individual metrics is the end result of the K-Fold Cross-Validation (Lemos, 2022).

4.4.3.2.c Feature importance

Based on the performance observed in figure 4.43.2.a, Gradient Boosting Regressor is selected as it has the lowest average MAE and relatively low variance. It is also important to understand how each feature influences the model's prediction. There are two ways of determining feature importance for trees algorithm. The first is impurity based feature importance and the second is permutation feature importance. The importance of a feature is computed as the normalised total reduction of the criterion brought by that feature, the higher, the more important the feature. This is also known as the Gini importance. impurity-based feature importances can be misleading for high cardinality features (Youssefi, 2023). The second important type is permutation which assesses the model performance by randomly suffering values of a single feature and then measuring the reduction in model performance (on test set). The greater the reduction the more important that feature is.

The next step is to test if the model is doing as good without those unimportant features.

4.4.3.2.d Hyperparameter- Tuning

The final step after selecting the best model would be hyper parameter tuning to optimise the performance of the model. Scikit Learn's RandomSearchCV and GridSearchCV were initially considered to be used to find the best parameter for the model. The list of the parameters is provided in table 4.3.3.a. However, Only RandomSearchCV would be performed since it will search the set of parameters randomly rather than exhaustive search, time and computing power will be saved with the trade-off that it might not find the true optimal parameters.

	Default	Search Range	Description
n_estimators	100	[100, 200, 300, 400, 500]	Number of boosting stages to be run
learning_rate	0.1	[0.01, 0.05, 0.1, 0.2]	Step size shrinkage
max_depth	3	[3, 4, 5, 6, 7]	Maximum depth of the individual regression estimators
min_samples_split	2	[2, 5, 10]	Minimum number of samples required to split an internal node
min_samples_leaf	1	[1, 2, 4]	Minimum number of samples required to be at a leaf node
subsample	1.0	[0.6, 0.8, 1.0]	Fraction of samples used for fitting the individual base learners
max_features	None	['auto', 'sqrt', 'log2', None]	Number of features to consider when looking for the best split

Table 4.3.3.a - Showing a set of hyper-parameters and each default setting and range to search for and to be tuned for the Gradient Boosting Regressor model.

(`Sklearn.ensemble.GradientBoostingRegressor` — Scikit-Learn 0.24.2 Documentation, n.d.)

4.4.4.3 Results

4.4.4.3.a Preliminary model assessment

	MSE	MAE	RMSE	R squared
Support Vector Regressor	5124.33	17.66	71.58	-0.02

Simple Linear Regression	4038.06	18.75	63.55	0.20
Single Decision tree	3223.41	14.61	56.78	0.36
Random Forest Regressor	3580.37	12.51	59.84	0.29
Gradient Boosting Regressor	3559.44	12.21	59.66	0.29
Dummy (Mean) Regressor	5032.40	20.06	70.94	-0.00

Table 4.4.3.a Summary of different models' performance on test set without feature scaling.

	MSE	MAE	RMSE	R squared
Support Vector Regressor	4884.38	14.79	69.88	0.02
Simple Linear Regression	4038.05	18.74	63.54	0.19
Single Decision tree	3292.46	14.55	57.38	0.34
Random Forest Regressor	3920.71	14.10	62.61	0.22
Gradient Boosting Regressor	3642.99	12.06	60.35	0.27
Dummy (Mean) Regressor	7145.41	29.39	84.53	-0.00

Table 4.4.3.a Summary of different models' performance on test set after feature scaling.

The improvement on the performance of the test sets for Support Vector Regressor is observed. Furthermore, Gradient Boosting Regressor has provided the best test MAE - which is the primary metric decided to evaluate the model performance.

4.4.4.3.b Validation

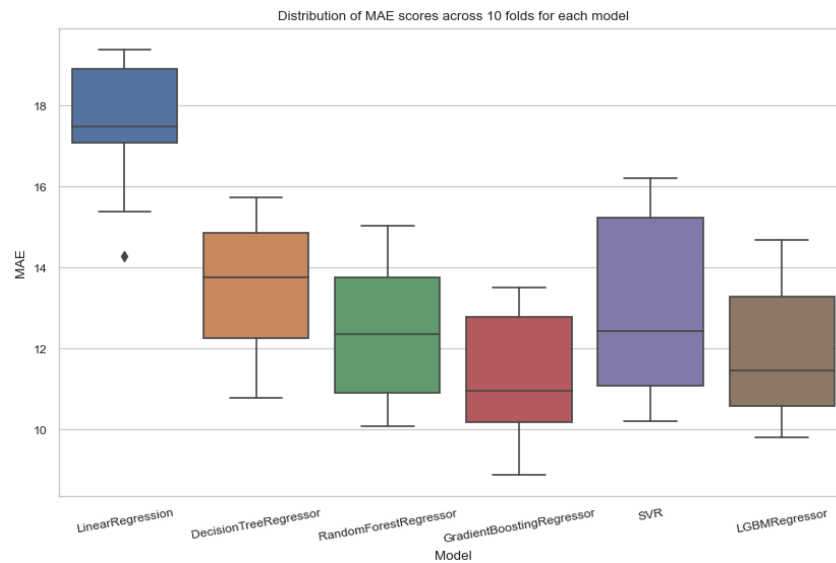


Figure 4.4.3.2 a - Showing the distribution of models MAE across 10 folds cross validation. Gradient Boosting Regressor has the lowest MAE, but considerably higher variance compared to SVR.

The results indicate that Gradient Boosting method has the lowest average MAE across 10 cross validation folds, and relatively low variance which could generalise well to unseen data.

4.4.4.3.c Feature importance

According to the impurity based feature importance, it shows category being the dominant feature with 40% of the normalised feature importance.

On the other hand, Permutation feature importance type suggests that Years Since Last Available, theme and pieces are the most important features.

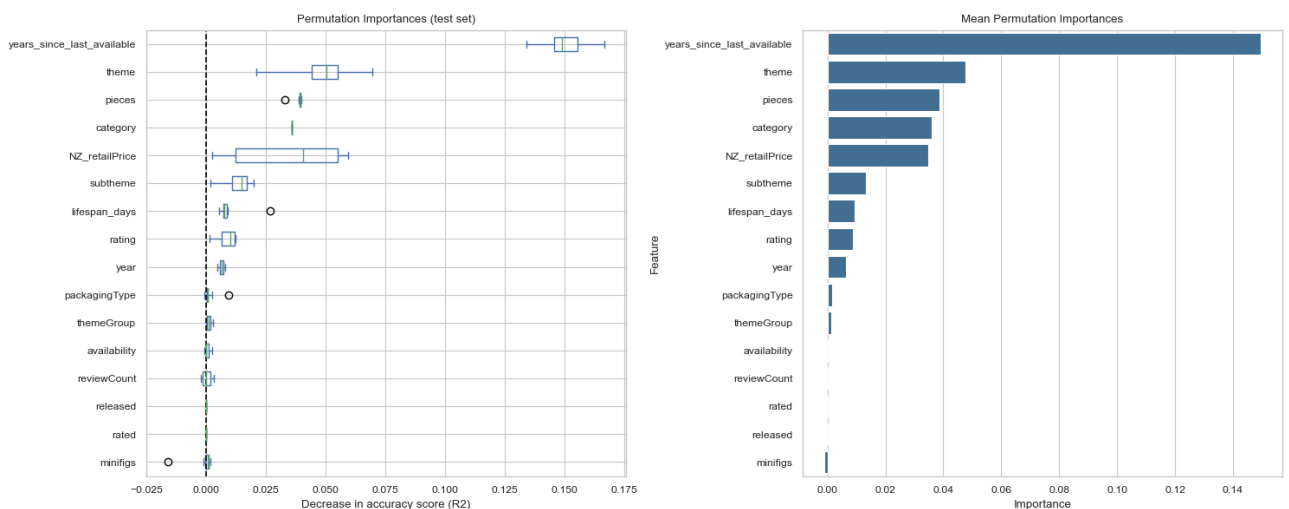


Figure 4.4.3.3.a - Showing the permutation importances using Gradient Boosting Model.

The average 10 fold cross validation MAE of the model excluding reviewCount, rated, released, availability, minifigs is 10.94. Compared to 11.24 from the model using all features. Therefore, those redundant features were removed from the model.

4.4.4.3.d Hyperparameter- Tuning

```

GradientBoostingRegressor
GradientBoostingRegressor(learning_rate=0.05, loss='absolute_error',
                           max_depth=6, min_samples_leaf=4, n_estimators=500,
                           random_state=421, subsample=0.8)

```

Figure 4.3.3.a - The result parameters returned by RandomSearchCV over specified parameters with cv=5, scoring=mae, n_iter=20.

After fitting the model with the new found parameters, the performance between the old and the new model are compared. Again, using 10-fold Cross Validation. It is observed from figure 4.3.3.b that the MAE and RMSE of the new model is slightly lower than those of the old model which means that the parameter tuning has optimised the performance of the model however not by a great extent.

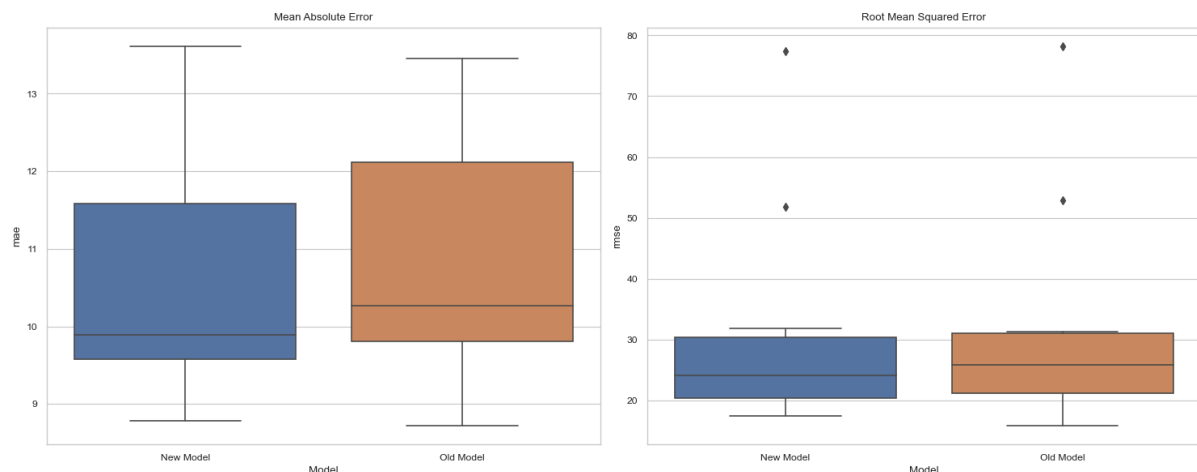


Figure 4.3.3.b - Comparing the MAE and RMSE of both models across 10-fold cross validation.

4.4.4 Choosing a model

After experimenting with both the simple and advanced approach. The simple approach was chosen to make predictions due to the following reasons. Firstly, it has a lower MAE of 10.14 compared to that of 10.62 of the advanced approach. Secondly, it is simpler and easier to deploy into production. For the advanced approach, there are many preprocessing steps needed, and a pipeline is required to streamline the preprocessing and training steps. Due to time constraints and scope of the project, the decision to go with the simple approach was made.

It is also worth considering that although the testing set MAE of the simple approach is lower than that of the advanced approach, this alone does not necessarily indicate that it is a better performing model. Since both approaches used different datasets that were processed in slightly different ways, the testing set for each was different, and so can't be directly compared with one another. It could simply be that the advanced approach testing set simply had more noise and was harder to predict. That being said, relative to the size of

other errors of models observed, the significance of the difference in MAE, in combination with the aforementioned data pipeline requirements, the simple approach will be utilised.

4.4.5 Findings

The Generalised Boosting Tree model from the simple modelling approach can now be used to predict the most investable soon to retire Lego sets.

To make predictions, data from the Brick Economy API on the 362 lego sets which are expected to retire at the end of this year (12/31/2024) was sourced. This Brick Economy data was then collated with existing Brickset review data to predict each soon to retire set's return. The below table shows the sets with the highest and lowest predicted annual returns 5 years after retirement.

	name	theme	set_number	min_retail_price	yr5_pred
	<chr>	<fct>	<chr>	<dbl>	<dbl>
42	Lotus Evija	Speed Champions	76907-1	32.92323	37.58280
47	Lamborghini Countach	Speed Champions	76908-1	32.92323	37.58280
315	1970 Ferrari 512 M	Speed Champions	76906-1	32.92323	37.40244
165	Pagani Utopia	Speed Champions	76915-1	35.96591	36.57252
190	Ferrari 812 Competizione	Speed Champions	76914-1	35.96591	36.57252
112	Porsche 963	Speed Champions	76916-1	35.96591	36.53255

Table 4.45a showing the sets with the highest predicted percentage annual return 5 years after retirement

	name	theme	set_number	min_retail_price	yr5_pred
	<chr>	<fct>	<chr>	<dbl>	<dbl>
163	Farm Animal Hospital	Friends	42632-1	47.95854	7.762043
49	Riding School	Friends	41746-1	47.95854	7.753714
356	Hot Dog Truck	Friends	42633-1	29.96959	7.215751
197	Space Research Rover	Friends	42602-1	77.94012	7.208421
222	Friends Advent Calendar 2024	Friends	42637-1	43.04144	4.998261

Table 4.4.5.b showing the sets with the lowest predicted percentage annual return 5 years after retirement

It can be seen from table 4.4.5.a, the sets with the highest predicted return are Speed Champions sets, with predicted returns all around 36-37% per year. These are highly popular among collectors as they are model replicas of iconic real life cars, so it makes sense for them to have a high predicted return. Counter to this, table 4.4.5.b shows sets from the theme Friends have very low predicted returns of only 4-7% per year. Like many other Intellectual Property based themes, these sets are not as popular among collectors and it therefore makes sense for these to be predicted poorly by the model. However, this

does not necessarily mean Speed Champions and Friends are the best and worst themes to invest in respectively, we must also consider a set's risk to judge its investability.

As with any investment, there is risk involved. For lego sets, there is a different risk associated with each set, with each of them having different themes and different associated variances in their returns. To account for this, the sharpe ratio can be used to determine which lego sets have the greatest predicted returns relative to their risk. The sharpe ratio for an investment is calculated as follows (Sharpe, 1994):

$$S = \left(\frac{R_p - R_f}{\sigma_p} \right)$$

Where R_p is a portfolio's return, R_f is a risk-free return and σ_p is the risk of a portfolio. To calculate the sharpe ratio for a lego set, R_p is the set's predicted annual return. The risk free rate R_f , typically refers to the return which could be obtained through government issued treasury bonds, which at the time of writing in New Zealand have a return of 5% (New Zealand Treasury, 2024). The risk of a portfolio, σ_p , is typically calculated as the variance of a portfolio returns over time, however, for lego sets we do not have data over time and we only have point in time cross-sectional data. Since theme is the key determinant for a set's predicted return, we can instead take the variance of a theme's returns to use as the risk of a portfolio. We can then calculate a sharpe ratio for each soon to retire lego set to determine which sets are predicted to have the highest risk to return ratio, and are therefore the most investable.

A sharpe ratio less than 1 typically indicates an investment is bad, 1-1.99 indicates a good investment, 2-2.99 indicates a very good investment and anything above 3 is typically referred to as an excellent investment (CMC Markets, 2024).

At this point, we only included soon to retire sets whose themes we had more than 10 observations of, as we otherwise could not gauge the variance in theme and calculate a risk metric. Additionally, only sets with a retail price of over \$100 were considered, this was two-fold. Firstly, there are limits placed on the quantity which a person can purchase of each lego set, typically 3 with a stand down period of 6 months, meaning a much harsher cap on investability if investing in cheap sets. Secondly, there likely wouldn't be enough liquidity in the market to dump a large number of cheap lego sets in the event the investor chooses to close their investment.

With that being said, using the 5 year predicted annual returns for the remaining soon to retire lego sets and calculating a corresponding sharpe ratio, the below table shows the most and least investable sets.

name	theme	set_number	yr5_pred	sharpe5yr	theme_sd
<chr>	<fct>	<chr>	<dbl>	<dbl>	<dbl>
Brachiosaurus Discovery	Jurassic World	76960-1	19.64031	2.611513	7.501517
Giganotosaurus & Therizinosaurus Attack	Jurassic World	76949-1	19.31378	2.567985	7.501517
Visitor Center: T. rex & Raptor Attack	Jurassic World	76961-1	19.06156	2.534362	7.501517
Sonic the Hedgehog - Green Hill Zone	Ideas	21331-1	26.09898	2.241447	11.621500
BTS Dynamite	Ideas	21339-1	25.68255	2.205614	11.621500

Table 4.4.4.c showing the sets with the highest predicted investability (sharpe ratio) 5 years after retirement

name	theme	set_number	yr5_pred	sharpe5yr	theme_sd
<chr>	<fct>	<chr>	<dbl>	<dbl>	<dbl>
Super Mario 64 Question Mark Block	Super Mario	71395-1	11.162736	0.3549668	31.30641
Princess Peach's Castle	Super Mario	71408-1	10.803227	0.3434832	31.30641
Larry's and Morton's Airships	Super Mario	71427-1	10.620702	0.3376530	31.30641
Diddy Kong's Mine Cart Ride Expansion Set	Super Mario	71425-1	9.953202	0.3163315	31.30641
Dry Bowser Castle Battle Expansion Set	Super Mario	71423-1	9.953202	0.3163315	31.30641

Table 4.4.4.d showing the sets with the lowest predicted investability (sharpe ratio) 5 years after retirement

Based on table 4.4.4.c, the most investable sets which are soon to retire are “Brachiosaurus Discovery”, “Giganotosaurus & Therizinosaurus Attack” and “Visitor Center: T. rex & Raptor Attack”. Although these Jurassic World sets have a significantly lower predicted percentage annual return than the Ideas sets, their sharpe ratio value is marginally higher. This indicates that they have a significantly lower risk and their risk-adjusted return is therefore about equal with that of the Ideas sets. The least investable sets shown in table 4.4.4.d which are soon to retire are “Dry Bowser Castle Battle Expansion Set”, “Diddy Kong’s Mine Cart Expansion Set” and “Princess Peach’s Castle”, all from the Super Mario theme. Not only do these sets have a very low predicted annual return of around 10%, the sharpe ratio values also indicate very high volatility and risk associated with them, and they should therefore be avoided by any prospective investors.

4.4.6 Further considerations

Regardless of the quality of models created and analysis completed, there are a number of additional considerations which affect the findings and subsequent applications of results. With the primary portion of analysis completed thus far on the sets which can allow for the greatest returns, this has failed to mention factors which can undermine the actual return which could be expected. The factors not yet considered include transport and storage costs of sets invested in.

As previously discussed, there is variance among where the cheapest prices can be found for a lego set, sometimes being in Europe or North America. Regardless, there is the

consideration that depending on where a set is purchased from there may be transportation costs. For the purposes of investment we are typically looking at placing large orders which will almost always reach the threshold for free delivery (Lego, n.d.), and so this cost is non-existent at best or negligible at worst.

Additionally, once these lego sets have been purchased there is the consideration that these must all be stored somewhere. For an investor, these are often stored in a spare room, garage or attic. There is the chance that a storage facility may need to be purchased to ensure adequate room. However, as with transport, an investor will be operating at a large enough scale that these costs are again negligible relative to the estimated profits.

Discussion

It has been shown throughout this report that there is sufficient potential for lego sets to be a viable investment option. Although the highest sharpe ratio values indicate that these are a “very good investment”, it is important to consider the opportunity cost of these investments. There may be other investment options readily available with comparable risk and returns, so these must be compared against investing in lego sets to then make an informed decision about the viability of investing in Lego. A table comparing some typical investment options can be seen below.

Investment Option	Predicted annual return (next 5 years)	Risk (σ)	Sharpe Ratio
Lego (Average of top 20 most investable sets)	24.87%	0.117	2.157
Standard and Poor’s 500 (S&P 500)	14.59%*	0.180	0.532
Dow Jones Industrial Average (DIA)	9.48%†	0.145	0.308
Gold Bullion	9.83%‡	0.164	0.295
Silver Bullion	7.93%§	0.280	0.105

Table 5.a comparing the expected annual return and risk of lego and other investment options

* (Morning Star, 2024)

† (Curvo, 2024a)

‡ (Curvo, 2024b)

§ (Curvo, 2024c)

For a true showing of the investability of lego sets, we must compare the estimated performance of such against some other common investment options, as shown in table 5.a. DIA and S&P 500 are both Exchange-Traded Funds (ETF) which effectively pools together multiple investments, lowering the risk compared to investing in individual stocks (Chen, 2003). It can be seen that these ETFs have slightly higher risk, however, they also have significantly lower annual returns, resulting in lower sharpe ratios which indicates that Lego

is more investable. The same is seen when comparing Lego to physical assets. It is worth highlighting that this is for the highest performing lego sets. Investing in bad lego sets, such as those in the Super Mario theme would be a much worse option than investing in ETFs.

It can clearly be seen that Lego is therefore more investable. That being said, it also requires significantly more time, effort and understanding. ETFs and physical assets are readily available to invest in online with minimal fees. However, there is significantly more logistics involved in the purchasing, storing and then selling of lego sets, not to mention the required understanding of which sets and types of sets are best to invest in.

The decision of whether or not to invest in Lego therefore becomes a personal choice dependent on the time in which a person has to make the most of such. For those with the adequate time to derive sufficient value from investing in Lego, it is an advisable practice. For those without this luxury, it may be best for them to stick to ETFs and more accessible forms of investment.

5.1 Fit-for-purpose

The initial question was “Is investing in LEGO sets worthwhile?”. The results of the project show that it can be worthwhile to invest in LEGO sets, however there are several things to consider before doing so. There are a few limitations to the model that can not be resolved, but the model is able to predict an annual return as a percentage while also showing how risky investing in a certain set can be. An investor can use the model to find sets with a high annual return and a low risk value to aid in their set purchasing decisions.

Therefore, this project has been fit for purpose and has been able to answer the question it initially set out to do. The extension of our project question was to answer “What sets or types of sets should be prioritised by a prospective investor to maximise their returns?”, which shown through the cluster analysis and predictive modelling section, this project has been able to deliver.

5.2 Future work

One significant challenge encountered was the bias resulting from limited data sources. The bias here refers to the systematic errors that can skew results resulting in inaccurate conclusions. To address this issue, the focus will be on obtaining additional datasets that are currently behind a paywall such as Bricklink. This can capture trends that a single source may miss and more data points can also increase the statistical power of the analysis and validate the findings. This will aid in reducing bias and enhance the models overall reliability. Moreover, it is important to continuously update the training dataset with the latest information as this will ensure that the predictions are robust and accurate in the face of evolving conditions. This will maintain the model's effectiveness over time.

To increase the predictive accuracy efforts can be directed towards examining more advanced modelling techniques like Artificial Neural Networks (ANNs) (Haykin, 2009). Through the application of ANNs, the aim is to strengthen the model's capability to identify complex patterns and dependencies within the data. While using such methods could

potentially enhance the performance, it may also present challenges in interpretability. However, a strong understanding of the dataset and the relationships between variables can effectively handle such complexities.

5.3 Limitations

Brick owl was the only source of market price data. Brick Owl had a few missing entries of market price data, however it was assumed that these were missing at random. Additionally, each Lego set had multiple active listings, in these cases the model would take the mean value of all the prices. This was a problem for some sets, as some have a wide range of prices and some of the prices were unrealistic which means that the unrealistic prices would influence the mean price. Sets could also be on sale or some websites offer cashback deals where the customer gets a small percentage of their money back after purchasing a set. Our database may therefore be overestimating retail price and underestimating returns.

Another issue was that some sets were included as a free 'bonus' from purchasing another set, the model was not able to account for this because there was no price linked to the 'bonus' item.

Another concern about investing in lego sets is that past performance is not indicative of future performance. In this case, the past performance of a lego set does not indicate that it will continue to do well in the future, like some other investments. Meaning that, just because lego sets have been a profitable venture in the past, there is no guarantee they will continue to be.

5.4 Ethics

The sentiment scores for each language could have potential to offend some people who speak certain languages as it was previously mentioned that some languages were considered 'ruder' than other languages. It is important to mention that these results are only comparing the languages across Amazon LEGO reviews, and not actually comparing the languages to other languages entirely. The sentiment analysis does not intend to offend anyone and is merely used for exploratory data analysis.

Additionally, the use of APIs to retrieve data was all done in accordance with each of their respective terms and conditions. The sites from which we utilised API have stringent policy against web scraping and an abuse of API calls to retrieve data for certain purposes. These were not violated in the course of the project to ensure our operation remained ethical, and our usage rights weren't rescinded.

References

Bhandari, P. (2023, June). Normal Distribution | Examples, Formulas, & Uses. Scribbr. <https://www.scribbr.com/statistics/normal-distribution/>

BrickEconomy. (n.d.). Sets retiring soon. <https://www.brickeconomy.com/sets/retiring-soon>

Brick Owl | Brick Owl - LEGO Marketplace. (2024). Brickowl.com; Brick Owl.
<https://www.brickowl.com/>

Brickset. (n.d.). Sets. Brickset.com.
<https://brickset.com/sets#:~:text=Brickset%20is%20the%20most%20established,over%20the%20last%2075%20years>

Chen, J. (2003, November 18). Exchange-traded fund (ETF) explanation with pros and cons. Investopedia. <https://www.investopedia.com/terms/e/etf.asp>

CMC Markets. (2024). Sharpe ratio. CFD Knowledge Hub.
<https://www.cmcmarkets.com/en-au/cfd/learn/trading-strategies/sharpe-ratio>

Curvo. (2024a, October). Backtesting for the European index investor. Superpower your savings through passive investing.
<https://curvo.eu/backtest/en/market-index/dow-jones-industrial-average?currency=gbp>

Curvo. (2024b, October). Backtesting for the European index investor. Superpower your savings through passive investing.
<https://curvo.eu/backtest/en/market-index/gold-bullion?currency=gbp>

Curvo. (2024c, October). Backtesting for the European index investor. Superpower your savings through passive investing.
<https://curvo.eu/backtest/en/market-index/silver-bullion?currency=gbp>

Dask Documentation (2024) Dask. <https://docs.dask.org/en/stable/>

Deep-translator 1.11.4 (2023) Python Package Index. <https://pypi.org/project/deep-translator/>

Dobrynskaya, V., & Kishilova, J. (2022). Lego: The toy of smart investors. HSE University.
<https://www.sciencedirect.com/science/article/pii/S0275531921001604#sec0060>

Harby, A. (2023) Slator Language Industry Intelligence.
<https://slator.com/resources/how-accurate-is-google-translate/#:~:text=Since%20its%20inception%20in%202006,Google%20Translate%20reaching%2094%25%20accuracy>

Haykin, S. (2009). Neural Networks and Learning Machines. 3rd Edition. Prentice Hall.
<https://dai.fmph.uniba.sk/courses/NN/haykin.neural-networks.3ed.2009.pdf>

Huw. (2020, November 12). Calculating set ratings. Brickset.com.
<https://brickset.com/article/54653/calculating-set-ratings>

IBM. (2023). Relational Databases Explained | IBM. IBM.
<https://www.ibm.com/topics/relational-databases>

Interface with the Brickset API for Getting Data About LEGO Sets. (2024). Github.io.
<https://jbryer.github.io/brickset/>

ISO639 Codes (October 2024). Wikipedia . https://en.wikipedia.org/wiki/ISO_639

Jarapala, K. N. (2023, March 27). Categorical Data Encoding Techniques. AI Skunks.
<https://medium.com/aiskunks/categorical-data-encoding-techniques-d6296697a40f>

Langdetect 1.0.9 (2021). Python Package Index. <https://pypi.org/project/langdetect/>

LEGO COMCON018 Batman. (2024). Brickset.com.
<https://brickset.com/sets/COMCON018-1/Batman>

Lego. (n.d.). Shipping & handling | Official LEGO® shop US. Official LEGO® Shop US.
<https://www.lego.com/en-us/page/shipping-handling>

Lemos, A. R. (2022, April 14). Cross-Validation. Medium.
<https://towardsdatascience.com/cross-validation-705644663568>

Machine Learning: When to perform a Feature Scaling? (2021, January 6). Atoti.
<https://www.atoti.io/articles/when-to-perform-a-feature-scaling/>

Metal Dragon BeatBox (2024) LEGO.com.
<https://www.lego.com/en-au/product/metal-dragon-beatbox-43109>

Morning Star. (2024, October). S&P 500 PR (SPX) risk | Morningstar. Morningstar, Inc.
<https://www.morningstar.com/indexes/spi/spx/risk>

New Zealand Treasury. (2024, September). Kiwi bond interest rates. New Zealand Debt Management - The Treasury.
<https://debtmanagement.treasury.govt.nz/individual-investors/kiwi-bonds/kiwi-bond-interest-rates>

NLTK. (2024, August 19). NLTK :: Natural Language Toolkit. <https://www.nltk.org/>

Peter J. Rousseeuw (1987, November) Silhouettes: A graphical aid to the interpretation and validation of cluster analysis.
<https://www.sciencedirect.com/science/article/pii/0377042787901257>

Prengère, A. (2024, October). CurrencyConverter. PyPI.
<https://pypi.org/project/CurrencyConverter/>

Rebrickable Help Guide: Set Variants and Inventories | Rebrickable - Build with LEGO. (2024). Rebrickable.com. <https://rebrickable.com/help/set-variants-and-inventories/>

Removing stop words with NLTK in Python (January, 2024) Geeks for Geeks.
<https://www.geeksforgeeks.org/removing-stop-words-nltk-python/>

Sample usage for sentiment (2024). Natural Language Toolkit.
<https://www.nltk.org/howto/sentiment.html>

Scikit-Learn. (2019). sklearn.preprocessing.StandardScaler — scikit-learn 0.21.2 documentation. Scikit-Learn.org.
<https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html>

Sharpe, W. F. (1994). The Sharpe ratio. The Journal of Portfolio Management.

<https://web.stanford.edu/~wfsarpe/art/sr/sr.htm>

sklearn.ensemble.GradientBoostingRegressor — scikit-learn 0.24.2 documentation. (n.d.). Scikit-Learn.org.

<https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingRegressor.html#sklearn.ensemble.GradientBoostingRegressor>

Tang, E. (2024). Lego Price Prediction. Brown University.

https://cs.brown.edu/media/filer_public/ba/7b/ba7b4638-d162-44b3-a302-81a46ef9a78d/tang_eric_capstone_abstract.pdf

Taylor, C. (2018). Detect the Presence of Outliers With the Interquartile Range Rule.

ThoughtCo. <https://www.thoughtco.com/what-is-the-interquartile-range-rule-3126244>

TargetEncoder. (2024). Scikit-Learn.

<https://scikit-learn.org/1.5/modules/generated/sklearn.preprocessing.TargetEncoder.html>

Translate 3.6.1 (2021). Python Package Index. <https://pypi.org/project/translate/>

YAĞCI, H. E. (2021, February 22). Feature Scaling with Scikit-Learn for Data Science. Medium.

<https://hersanyagci.medium.com/feature-scaling-with-scikit-learn-for-data-science-8c4cbcf2daff>

Youssefi, S. (2023, September 18). Permutation importance vs impurity-based feature importance. Medium.

<https://medium.com/@syoussefi600/permutation-importance-vs-impurity-based-feature-importance-1c1a8d027479>

ZalaRushirajsinh (2023, November 4). The Elbow Method: Finding Optimal Number of Clusters.

<https://medium.com/@zalarushirajsinh07/the-elbow-method-finding-the-optimal-number-of-clusters-d297f5aeb189>

Glossary

API (Application Programming Interface): A set of rules that allows programmers to interact with websites.

Cardinality: Refers to the number of unique values within a categorical column.

Clustering Analysis: The process of grouping similar data points based on shared features, without predefined labels.

Complexity Parameter (CP): A threshold in decision tree algorithms that controls tree pruning by setting the minimum improvement in model fit required to split a node.

Cross Validation (CV): A technique used to evaluate if a model will generalise well to unseen data.

Data Wrangling: The process of cleaning and transforming data before analysis.

Database: An organised collection of structured data, usually in the form of relational tables.

Encoding: The process of converting categorical variables into numerical values.

Euclidean Distance: The length of the line segment between two points.

Exploratory Data Analysis (EDA): The process of analysing and visualising datasets to uncover relationships or patterns before modelling.

Foreign Key: A field in a table that links to the primary key of another table.

Interquartile Range (IQR): The range between the first quartile (25th percentile) and the third quartile (75th percentile).

ISO 639: Two-letter universal language codes used to classify languages.

Log-transformation: Applying a logarithmic function to a variable.

Mean Absolute Error (MAE): A regression metric to evaluate model errors (not sensitive to outliers).

Mean Squared Error (MSE): A regression metric to evaluate model errors (sensitive to outliers).

Outlier: A data point that significantly stands out from the rest.

Overfitting: A model that is too complex and adapts to noise in the data, generalising poorly to unseen data.

Primary Key: A unique identifier of a table.

Principal Components Analysis (PCA): A technique that transforms correlated variables into a smaller set of uncorrelated components, capturing the most variance in the data.

Return: Also known as 'Return on Investment,' a ratio or value that defines an investment's gains relative to its cost.

R-Squared: A metric indicating the proportion of variance explained by the model.

Root Mean Squared Error (RMSE): A regression metric to evaluate model errors (moderately sensitive to outliers).

Sentiment Analysis: The use of natural language processing to determine the emotional or opinion-based content in text data.

Sharpe Ratio: A measure used to evaluate the risk-adjusted return of an investment.

Underfitting: A model that is too simple and fails to capture the true relationship in the data.

Web Scraping: Automating the process of extracting data from websites through programming.