Project 1 : Book average rating

Group n°8

Oluwatobi Olufunmilayo
Luis Lanzuela
Daniel Selim-Merlusca

Github link :

https://github.com/DanielSelimMerlusca/ML_Book_Rating_Project/tree/main

Machine Learning Project Report

Data preparation and cleaning process in Excel

Excel data cleaning before modeling in Python.

The main cleaning process before loading the file into python processing has been done on ‘Authors’ and ‘Publishers’ columns. The goal on these columns was to have a consistent set of data points that wouldn’t interfere into the creating categories groups in these columns, while having a defined protocol of transformation in the data that would be discretionary and could be executed mechanically.

Authors:

- Kept the first author listed when a collaboration would exist.

  (‘example author1’~~/’example author 2 ‘~~ => ‘example author 1’)

Publishers:

- Kept the first publisher listed when a collaboration would exist

  (‘example publisher1’/’example publisher2 ‘ => ‘example publisher1’)

- Erased legal form on publisher’s name (“Inc. “, “LTD.”)

  (‘examplepublisher1 Inc.’ => ‘example publisher1’)

- Erased terms in parenthesis ‘()’ at the end of publishers

  (‘example publisher1(NY) ‘=>’example publisher1’)

· Note: A final review was conducted to harmonize the names of publishers, creating a more uniform dataset that facilitated stronger categorical groupings for analysis. However, we found this task to be extremely discretionary, potentially not scalable to a larger data set and not reproducible among different people performing the same task. As a result, we have decided to not implement this data processing step into the modeling. i.e.:

  - (‘Alfred A. Knopf Books’ = ‘Alfred A. Knopf Books ~~for Young Readers~~’) ?

  - (‘Addison Wesley ~~Publishing Company~~ ‘= ‘Addison Wesley ~~Professional~~’) ?

Publish date:

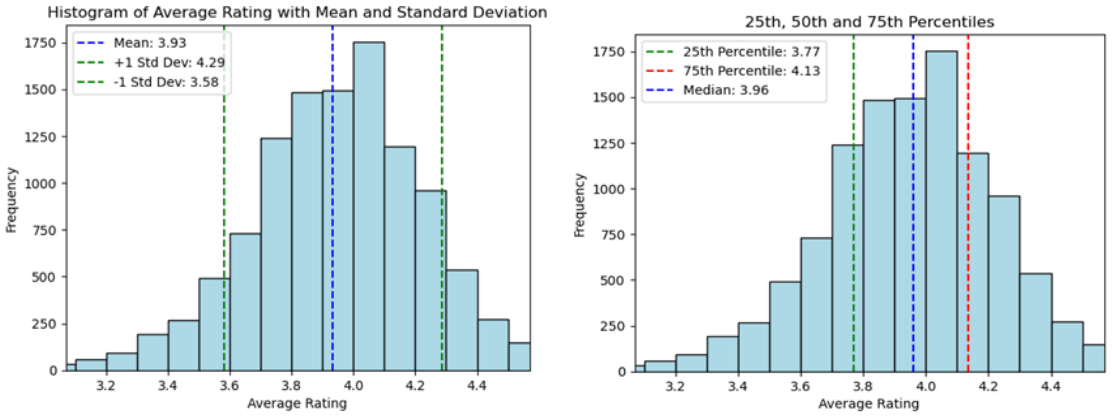
- Transform date format to numerical.

Rows changes:

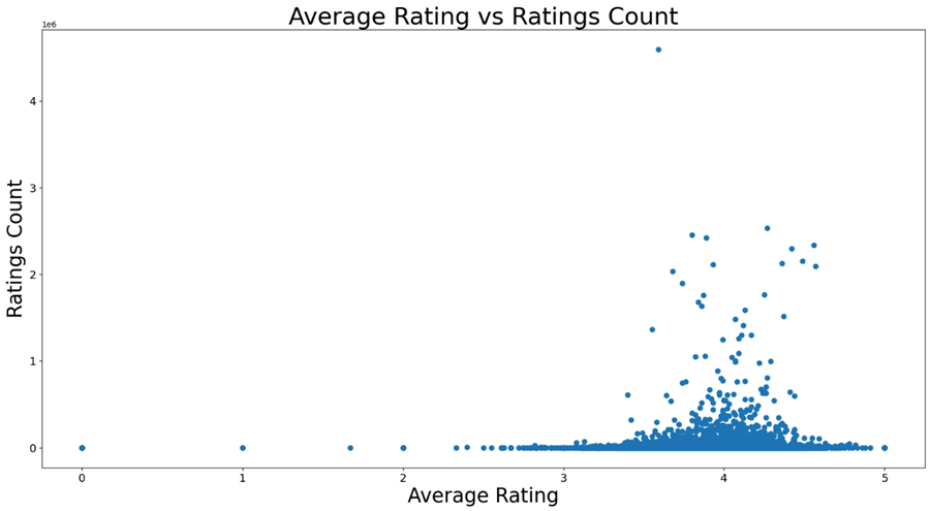
- Rows with ‘bookID’ [12224, 16914, 22128, 34889] had displaced columns to the right from the ‘average_rating’ column, so they were manually corrected.

DATA ANALYSIS

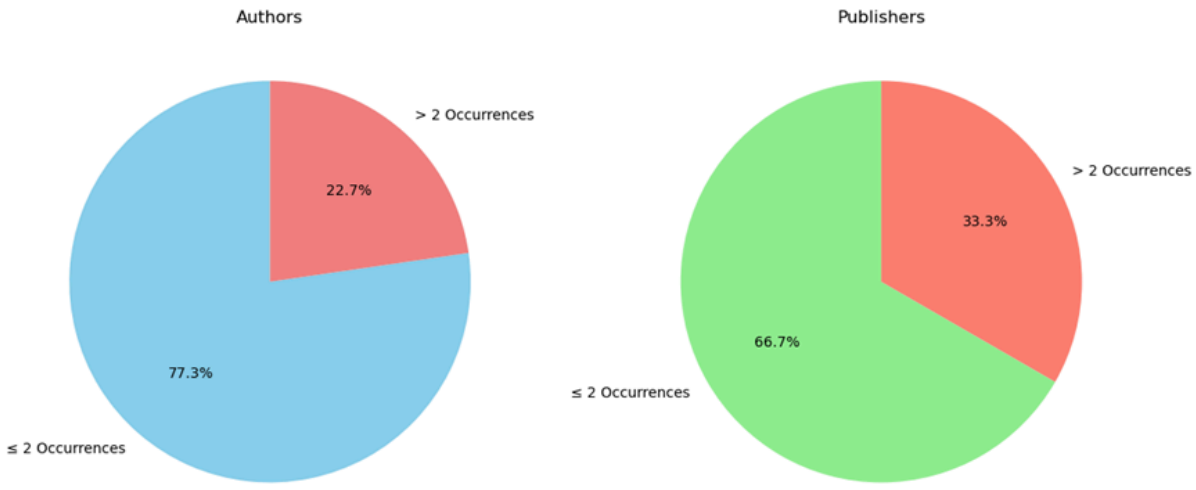
The distribution of the dataset for 'average_rating' exhibits a **left-skewed normal distribution**, with most ratings concentrated in the upper range (0-5). Specifically, the middle 50% of values fall within a 7.2% range around the average rating. This characteristic is advantageous for model creation, as it increases the likelihood of ratings clustering around the mean. However, it also introduces challenges in prediction due to the imbalances created by outliers.



We note that the books with higher **rating counts** tend to keep their rating values loyal to the bell curve. We can observe as well, that higher rating count books trend to be over the global mean of ratings (mean = 3.93).



The columns **‘Authors’** and **‘Publisher’** remain having a large number of categories (Unique authors count: 4219 and Unique publishers count: 1801), many of them with a small number of occurrences. This aspect of these columns represents a challenge when finding statistical significance of these features into the target.



FEATURE SELECTION

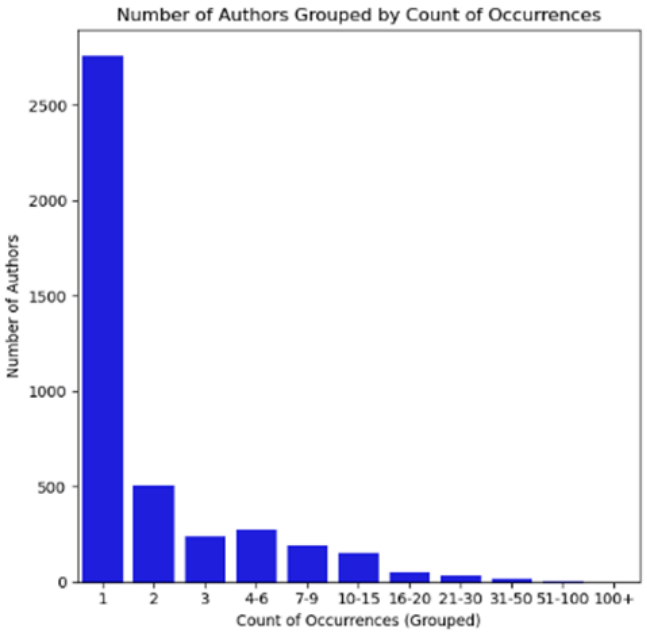
For feature selection, the following columns have been dropped, as we consider that it doesn’t provide any relevant information to the model and only increases dimensionality to data set:

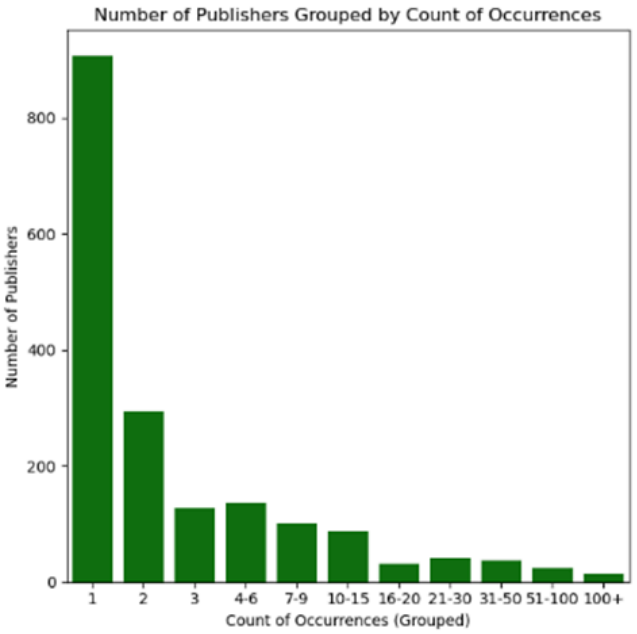
- **‘Title’, ‘isbn’, ‘isbn13’**. *In the case of ‘title’, we consider a complex task to extract any relevant information but could possibly have value by classifying the title in function of the word classifications that it contains.

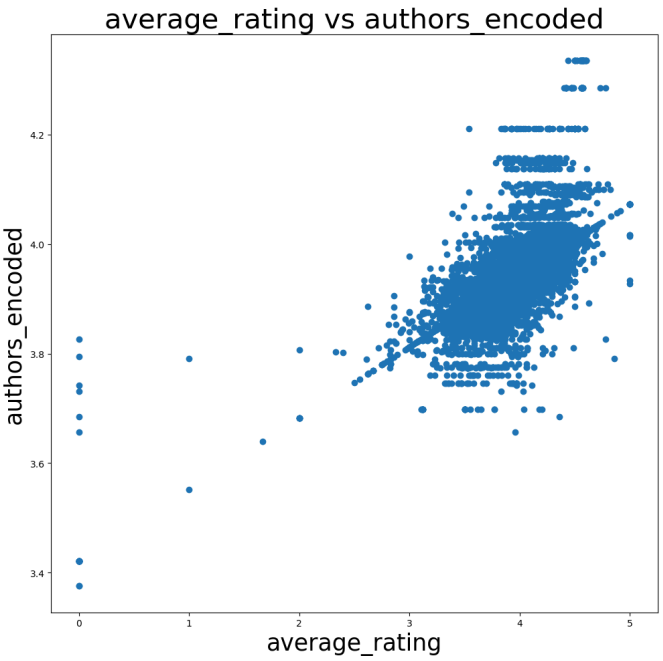
FEATURE ENGINEERING

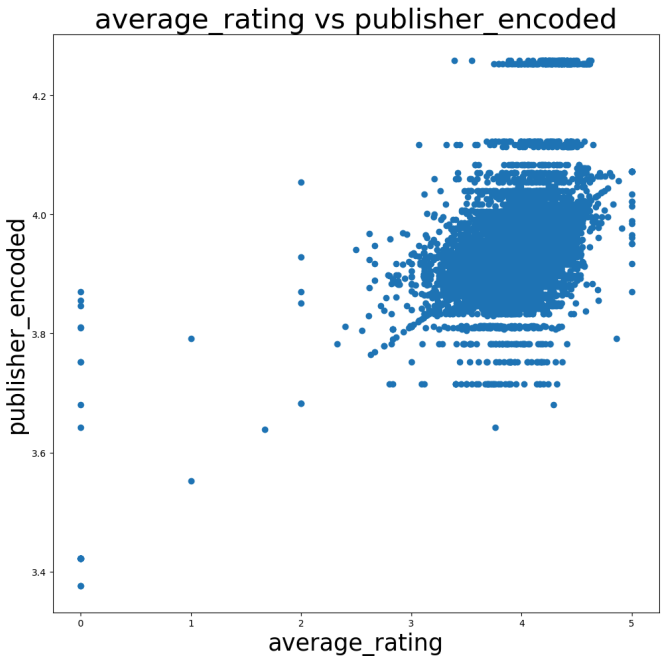
‘Authors’, ‘Publisher’ and ‘language_code’: These features have been encoded to allow the model to interpret its values and improve its statistical inference. We used a weighted average of each category inside these features, using “scikitlearn.targetencoder” .

This encoding allows to deal with high categorical features and helps to reduce overfitting, through its weighting component, each category is encoded based on a shrunk estimate of the average target values for observations belonging to the category. The encoding scheme mixes the global target mean with the target mean conditioned on the value of the category. This is especially relevant when dealing with ‘Authors’ and ‘Publisher’ features, where the number and proportion of categories with low occurrences is very high. Giving more relevance to the higher number of occurrences categories.









We can appreciate that those encodings gave some consistent approximation to the ratings of the books. Showing as well in the 45 degree line the ratings belonging to one occurrence in its category.

Model Training and Evaluation

The task was to predict the average ratings of books based on various features like authors, publishers, language_code, and publication_date. We started by treating this as a regression problem but later considered transforming it into a classification problem due to the nature of book ratings and buyer behavior. Through multiple iterations, we experimented with different models and data balancing techniques to optimize the results.

1. Initial Approach: Regression Problem

We began by treating the problem as a traditional regression task where the goal was to predict a continuous variable : average_rating. The correlation matrix gave us some insights on the importance of the authors and publishers in the dataset as well as other features like, the rating count or the text reviews count. After encoding the non numerical features, we started simply with linear regression.

- **Linear Regression** :

On the first trial, we dropped the book ID for the isbn13 and the authors as well. For the publishers we used target encoding and the language code with label encoding. The split between train and test was 70-30.

Initial results showed us R-squared (R^2) of 0.41 and a Mean Squared Error (MSE) of 0.073. The model could explain about 41% of the variance in the ratings, but it struggled to capture more complex patterns. With this result we knew that authors were definitely a key but we needed to encode it properly.

We then made a second trial, with the authors target encoded this time. We also changed the label encoding for the language code into binary encoding since there were only a few possible values. Results gave us R-squared (R^2) of 0.76 and a Mean Squared Error (MSE) of 0.030. This was definitely an improvement but with this setup it stabilized around these scores. So we thought of tackling this case as a classification problem to see if the results were more promising.

2. Transition to a Classification Problem

We tried to approach this classification problem from a business standpoint. Where books rated below 3 are rarely bought, those between 3 and 4 are bought conditionally, and those above 4 are often purchased. So we reframed the problem as a classification task with three categories:

- **Class 0**: Ratings below 3
- **Class 1**: Ratings between 3 and 4
- **Class 2**: Ratings above 4

We also tried to determine which errors we should avoid and why. Based on the classification we did, we concluded that for our use case False Positives were the most impactful. In a business point of view : Is selling a book that we predicted good and actually sells bad worse than predicting a book will not sell and we find out it actually sells really well ? Here

we had a case of real loss of money (False positive) vs potential loss of money (False negative). And a company would rather complain about a missed opportunity of a best seller than actually wasting my money on some low selling book. Both cases are valid but most of the time one would rather not waste money.

Classification Model :

We applied the Random Forest Classifier to predict the three rating classes. The initial accuracy was around 84.27%, with better performance on the majority class but weaker recall for the minority class (low-rated books).

The results were not that different from what was achieved by the linear regression. It looked a bit better but there was some obvious overfitting due to the class imbalance.

Data Balancing Techniques :

Since the dataset had class imbalance (far fewer low-rated books), we experimented with several data balancing techniques:

Oversampling :

We tried SMOTE, and it seemed to give the best results. The precision was slightly above 0.50 and the recall is stable 0.63. Adding hyperparameters to tune a bit the RFC improved slightly in accuracy, but the precision and recall remained quite stable.

Undersampling :

We used RandomUnderSampler and it gave by far the worst result regarding our case. It lowered the accuracy and drastically improved recall but at the cost of the precision. I regularly get 0.15 precision for 0.95 to 1.00 in recall.

Hybrid :

We used SMOTEEnn and SMOTETomek which gave similar results with lower precision and higher recall. But SMOTETomek performed a bit better overall.

Eventually, the oversampling strategy worked best. In a business standpoint we want to avoid False Positives (losing money) and the high recall for low precision seems to bring us many false positives, especially in undersampling and some hybrid techniques.

Gradient Boosting Classifier :

Gradient Boosting model was performed for class (0-3, 3-4, 4-5) ratings and (0-4,4-5) class. The model was trained in a 5-fold cross validation process, giving in both formats of the model classification (3 and 2 classes) an average score of 0.70 across Accuracy(0.691, 0.700) , Precision (0.686, 0.698) , Recall (0.691, 0.700) and F1 score (0.683, 0.6936).

The scores for the lower class in the 3 class classification however are very low due to under sampling of that class. Also the Recall scores in both models are low in many foldings on the upper class (4< average_rating), potentially giving many fake negatives, but making viable from a business standpoint from what was previously commented.

So we decided to get back to a regression problem since we didn't get significant improvements.

Using Regression Analysis for Predicting Book Ratings

This project aims to predict the **average rating** of books using a regression model based on several features from a dataset of book attributes. The goal is to find a robust model that can capture the relationship between the book attributes and their ratings, allowing for accurate prediction of ratings on unseen data. Key metrics such as **Mean Squared Error (MSE)** and **R-squared (R^2)** are used to evaluate the performance of the regression model.

Dataset and Features

The dataset includes several features about books, such as:

- **Text reviews count**: Number of text-based reviews for each book.
- **Publication date**: Year the book was published.
- **Number of pages**: Total number of pages.
- **Ratings count**: Total number of ratings a book received.
- **Publisher**: Categorical variable representing the book's publisher.

The target variable is the **average rating** of the books, which is a continuous variable.

Model Selection and Training

The model used for this task is a Multiple Linear Regression model. This model aims to model the relationship between the book features (independent variables) and the average rating (dependent variable) through a linear equation.

Evaluation

The model's performance was evaluated using **Mean Squared Error (MSE)** and **R-squared (R^2)**.

- **Mean Squared Error (MSE)**: Measures the average of the squares of the errors between predicted and actual values. A lower MSE indicates better accuracy.
    - **MSE**: 0.0637
- **R-squared (R^2)**: Represents the proportion of variance in the dependent variable that is predictable from the independent variables. An R^2 score closer to 1 indicates a better fit.
    - **R^2**: 0.524

Interpretation:

- The **MSE** of 0.0637 indicates that the model has a moderate prediction error, which suggests room for improvement in predictive performance.
- The **R^2 score** of 0.524 indicates that the model explains about 52.4% of the variance in book ratings based on the input features. While this is a decent result, it implies that other factors not captured by the current model could still influence the ratings.

2. Using Random Forest Regressor for a better performance.

After experimenting with different models, we utilized **Random Forest Regression**, which was optimized through hyperparameter tuning. The model's performance was evaluated using **Mean Squared Error (MSE)** and **R-squared (R^2)** to assess its accuracy and predictive power.

Hyperparameter Tuning

To optimize the Random Forest model, a **grid search** with 5-fold cross-validation was performed over the following hyperparameters:

- **n_estimators**: The number of trees in the forest.
- **max_depth**: The maximum depth of each tree.
- **min_samples_split**: The minimum number of samples required to split an internal node.
- **min_samples_leaf**: The minimum number of samples required to be at a leaf node.

Grid Search Results:

- **Best parameters**:
    - n_estimators: 300
    - max_depth: 20
    - min_samples_split: 5
    - min_samples_leaf: 2

Random Forest Regressor Model Performance

After tuning, the best Random Forest model was evaluated on the test dataset using **Mean Squared Error (MSE)** and **R-squared (R^2)** scores. The performance metrics were as follows:

- **Mean Squared Error (MSE)**: 0.0296
- **R-squared (R^2)**: 0.7500

Interpretation:

- **MSE**: An MSE of **0.0296** indicates that the model has relatively low error in predicting the average ratings, meaning the predictions are close to the actual ratings.
- **R^2**: The **R^2 score of 0.75** indicates that the model explains 75% of the variance in the average ratings, demonstrating strong predictive performance.

Model Evaluation Using Cross-Validation

Cross-validation is a critical step in model evaluation, especially for understanding how well the model generalizes to unseen data. In this project, we applied 5-fold cross-validation to assess the performance of our Random Forest Regression model more comprehensively.

Cross-Validation Results

The **R^2 scores** for each fold were as follows:

- **Fold 1**: 0.5705
- **Fold 2**: 0.5309
- **Fold 3**: 0.5384
- **Fold 4**: 0.5187
- **Fold 5**: 0.3734

Mean R^2 Score

The **mean R^2 score** across all 5 folds was **0.5064**, indicating that the model explains about 50.64% of the variance in the data on average. This suggests that while the model performs reasonably well, there is still room for improvement in capturing the remaining variability in the ratings.

Insights:

- **Consistency**: The R^2 scores for most folds were consistent, with a slight dip in Fold 5. This indicates that the model's performance is relatively stable across different subsets of the data.
- **Generalization**: The mean R^2 score of 0.5064 provides a reliable estimate of how the model will perform on new, unseen data. The cross-validation results suggest that the model generalizes reasonably well but may benefit from further optimization or additional features.

Conclusion

The Random Forest Regression model performed well in predicting the average ratings of books, with an R^2 score of 0.75 and a low MSE of 0.0296. This means that the model captures a substantial portion of the variance in the ratings, making it a robust model for this task.