Hassna Boudalil
Yilun DU
Djihene Beladjine
Arnelle Nguefang Tchouamo
Kaoutar Zeroual

Instructor: Hanna Abi Akl

Book Rating Prediction Machine Learning Project

I- Introduction

The dataset is a curation of Goodreads books based on real user information which can be found at the following address : https://sites.google.com/eng.ucsd.edu/ucsdbookgraph/home. In this study, we will train a model that predicts a book's rating. Each book has an average score, and our goal is to analyze the dataset to predict a book's score by using machine learning tools. Our database contains 11 127 book records and 11 attributes of each book.

GitHub Repo:
Jupyter notebook and detailed analysis are available here:
https://github.com/hassnabdl/PythonLabsProjectS23.git

II- Data Cleaning

1. Data analysis

1.a. Incorrect format while loading dataset

Before starting the analysis, the data and variables must be examined. Thus the first step in Exploratory data analysis is data cleaning. After downloading the data contained in the file in the Jupyter notebook and converting the .csv file to pandas.DataFrame, the first thing we notice is that 4 lines have an extra comma on different features, so these 4 lines contained 13 columns instead of 12 : line 3350, line 4704, line 5879 and line 8981. We checked each of these lines and found that all errors are caused by author names containing a comma. Therefore, we used double quotes to surround these author names so that `pandas.read_csv()` doesn't identify the comma in author name as a delimiter. We have corrected these 4 lines in the original CSV as logically as possible manually. The corrected dataset file is renamed as `books_corrected.csv`

1.b. Missing values and duplicate

The ISBN/ISBN13 is the unique ID of a book in the world, so it can be used to examine if there are duplicated books in the dataset. We have found no missing values (NA) to deal with and no duplicate to remove in the dataset.

1.c. Format consistency and

- **average_rating:** For the feature "average_rating" we observe some books with an average rating equal to 0. For the case where the book has not

received a score evaluation, i.e. rating_count=0 and average_score=0, we have left the records as they are. Because they are also useful information which can teach our model how to predict the average rating of a book with no rating.

- **num_pages:** Same thing with the feature "num_pages" we observe 76 books with a number of pages equal to 0. As it is impossible to have a book with 0 pages in the real world, these should be treated as records with missing values. We decided to replace the missing value with the average number of pages of all the books with non-zero pages.
  Additionally, the column name `num_pages` contains 2 space characters at the beginning, which we removed in data cleaning. Because it will cause problems when doing indexing like `df_book.num_pages`
- **ratings_count:** There are misrecorded books that have `ratings_count`= 0 and `average_rating`≠0. It is impossible to score a book at 0 on the Goodreads website (the minimum score that can be set is 1), so logically `ratings_count`= 0 must lead to `average_rating`=0. Consequently, we dropped those lines because they are mis-leading for the machine learning model.
- **language_code:** There are 27 different language codes. We decided to combine the different English languages (`eng`, `en-US`, `en-GB`, `en-CA`) into a single column: `eng`. Because we assumed that readers don't care which type of modern English is used to write the book. On the other hand, we've left `enm` , which refers to Middle English (c. 1100 - c. 1500) and has a big difference with modern English.
- **publication_date:** when `pandas.DataFrame` was created, this column was processed as type `str`. The date is converted to pandas `datetime` object by convention at this stage. Some date errors such as “11/31/2000” and “6/31/1982” have been corrected additionally.
- **authors:** There are 5 books whose author names are marked as `”NOT A BOOK”`. After searching these 5 items’ names on Google, we found that all of them are CDs, not books. Therefore, these 5 records were removed from the dataset.

2. Feature selection and feature engineering

In order to observe correlation between different attributes and identify the most relevant attributes that have an impact on the average rating of a book and thus increase the efficiency of the models, we took a look at the correlation matrix, which shows a correlation of 86% between ratings_count and text_reviews_count, and a small correlation of 15% between num_pages and average_rating. The correction between isbn13 and ratings is negligible.

We then visualized the columns one by one, looking at the relationship of each feature with the average score using **scatterplots**, to see how the average score is affected by each variable for the numerical values. We observed a positive correlation for each numerical value (num_pages, ratings_count and text_reviews_count) with the average

score: the points in the scatter plot appear to be randomly dispersed for low values for the 3 features, but form an ascending line for medium to high values. So **there's a correlation between these numerical values and we've decided that these 3 numerical values have an impact on the average ratings of books** and are important for our model.

For categorical values we decided to rely on logic and general knowledge to select the appropriate features for our model and project and then implement feature engineering techniques to convert them to numeric values. The processing details are explained in what follows:

- Column **title** and **isbn** are dropped because we assume that these attributes don't have a significant impact on book choosing of readers. (We amit that some books' **title** can be more attractive than others, thus this attribute may have an impact on a book's average rating. But to process this, some NLP techniques such as *semantic analysis* are needed. Obviously it is out of the scope of the course and this project, so it is reasonable to drop attribute **title** at this phase.)
- **isbn13** is kept temporarily because it can be used as a unique identifier for each book in addition to the row index. This column will be very useful in some table operations such as "groupby" and "join" ("join" operation is implemented by function "merge" in `pandas.DataFrame`). It will be dropped after feature engineering because it has nothing to do with book rating.
- Frequency encoding is applied to the feature **language_code** because we assume that the books written in more popular language tend to attract more readers and thus have more convincing average ratings.
- **publication_date** is converted to book's age since its publication. Because usually the longer time that a book has been published, the more ratings it can accumulate, so that the more convincing the average rating should be. Therefore the **publication_date** is converted to the number of years in decimal values stored in a new column called **delta_date**. The original one is removed.
- Target encoding is applied to the attribute **publisher**. Because we believe the books from some publishers tend to have better quality and hence get higher average ratings. So it is logical to replace each publisher by the mean average rating of all its books, i.e. target encoding. The original one is replaced by column **publisher_encoded** after feature engineering.
- We applied a modified version of target encoding to column **authors**. Because each book can have more than one author, it is not possible to apply conventional target encoding to this attribute. Thus we converted this column from data type `str` to `list` so that each book has a list of authors. Then function pandas.DataFrame.explode() is implemented to impose that each single author name matches only one book in each row, where we can compute the mean value of all the books's average rating from a single author. This value is called **authors_avg_rating** in our notebook which signifies readers' rating for an author. At last, if a book has several authors, value of **authors_avg_rating** is computed by the weighted mean of authors' **authors_avg_rating,** with ratings_count as weights. The original attribute **authors** is replaced by **authors_avg_rating** after encoding**.**

III- Classification and Data imbalance

- **Classes combination:** we did 3 groups of classes that represent the ratings values . Group 1 which represents the “low group” combined the classes of rating values 0, 1, 2 and 3. Group 2 which represents the “medium group” has the class of rating value 4, and group 3 which represents the “high group” has the class of rating value 5. We therefore decided to modify the input and output of our prediction model.
  We decided to group in this way because the data was too unbalanced for the average score. But despite this, there was still far too much imbalance and we decided to use a method to remedy the problem
- **Data imbalance:** Our dataset shows a class imbalance, with books whose average rating is equal to 4 containing more data than others. To avoid poor performance due to data imbalance, we applied the SMOTE method and oversampling.

The oversampling involves increasing the representation of the minority class by duplicating instances from that class. The **Synthetic Minority Oversampling TEchnique** generates synthetic examples that are close in the feature space, drawing a line between the examples in the feature space and drawing a new sample at a point along that line.

However, when we used the SMOTE and oversampling in our models, we observed that accuracy has decreased, we deduced that it was because this technique puts more weight to the small class, making the model biased to it. We decided to use the models without applying oversampling.

IV- Model training

We compared the performance of different algorithms by comparing the results obtained in order to conclude on the best model. The explanation of all the methods and the package used is detailed in the notebook.

1. Model training & Model evaluation

We used one linear and 2 non-linear models and compared the effectiveness of each. To evaluate the models, we used the F1 score for each group (low, medium and high) and the overall accuracy that combines the precision and recall scores of a model.

1.a. Linear Model: Logistic Regression

We started by using a logistic regression model. This linear approach gave us results that were too low about 0% for the predictions of the low group (=3) and the high group (=5), but a very high prediction of 96% for the medium group (=4).
And the model's overall accuracy is 92%

1.b. Non-Linear Models: Decision Tree, Random Forest, and MLP

To capture more complex patterns in the data. We used the following non-linear models:

- **Random Forest Classifier**: We decided to use the Random Forest model because it combines multiple decision trees. It can handle complex and variable data sets, as in our case, and is capable of handling classification problems.

  This model gave us high results for all 3 groups: an F1 score of 60% for the predictions of the low group (=3) and 62% for the high group (=5), and a very high prediction of 97% for the medium group (=4).
  And the model's overall accuracy is 95% which is a very good score in our case.

- **MLP (Multi-Layer Perceptron) Classifier**: As with the random forest model, we decided to use this model because of its ability to model complex relationships between data.

  This model approach gave us results that were too low about 0% for the predictions of the low group (=3) and about 5% for the high group (=5), but a very high prediction of 96% for the medium group (=4). Results are similar to those of the linear model we used.
  And the model's overall accuracy is 92%.

V- Conclusion - Discussion :

In our case, we observed that the **Random Forest Classifier is the best and the most effective in terms of accuracy,** which is an expected case because this machine learning algorithm is a powerful ensemble learning algorithm that combines multiple decision trees to make predictions. It is highly used in classification due to its high accuracy, robustness, feature importance, versatility, and scalability. In our case, It demonstrated high precision, recall, and F1-scores across multiple classes, showcasing its ability to handle complex relationships within the data.

In conclusion, our work covered the different steps of making a book rating prediction model using 3 machine learning models which are : logistic regression (a linear model), Random Forest Classifier and MLP (Multi-layer Perceptron Classifier). And the model to be used to predict the average score of a book with the characteristics we have selected is the Random Forest model.