ML Project Report :

BOOKS RATING PREDICTION

After initially working with the original dataset 'books' and encountering poor results, I decided to add the 'Genre' column to it. I used various datasets provided by the GoodReads website. I started with the 8 'by genre' datasets (*Children, Comics & Graphic, Fantasy & Paranormal, History & Biography, Mystery, Thriller & Crime, Poetry, Romance, Young Adult*). I transformed those JSON files to only keep the columns that I was interested in (Book_ID/genre), convert them to dataframes (and format them as such), and then merge them one by one with the initial dataset *(Note that I did not upload the whole transformation as this would require a very long jupyter notebook and lots of memory)*

Once all 9 datasets were merged, I realized there were more than 5,000 missing values. I then decided to use the 'Fuzzy book genres' dataset to fill in these missing values (*also provided by the GoodReads website*). Each row of this dataset had multiple 'genres' given by readers, so I chose to keep the first one listed (most of the time, the one most frequently given by readers). Once this dataset was merged with the other 9, I saved it in CSV format, and I used it as the reference dataset for the project.

1. CLEANING THE DATASET

~ INITIAL DATASET ~

- ✓ FIELD ERROR : Extra coma in "Author" Attribute

When uploading the dataset in Jupiter Notebook, I got an error message regarding extra fields : 12 were expected but 13 were found for 4 rows of the dataset (*Rows : 3350/ 4704/ 5879/ 8981*). The issues were caused by extra comas in the 'Author' attribute.
=> I decided to remove manually those 4 extra comas.

- ✓ 29 MISSING FIELD caused by double quotation marks.

After calling the 10 first lines of the dataset, I realised that the whole row of ID9 was located on the column 0 and other fields was filled with NaN. I then called the missing value of the dataset, and I obtained 29 rows with missing fields. After calling those rows, I realized that the conflict was caused by the double quotation marks.
=> I decided to replace them by 'single quotation marks' manually in the csv file through a Ctrl + H

~ FINAL DATASET ~

- ✓ MISSING VALUES : "Genre" Attribute

After merging the two datasets, the 'Genre' column had a significant number of missing values (1861). Initially, I decided to use conditional imputation: Books with missing 'Genre' that share the same author with other books (where the genre is specified) will be assigned the same 'Genre' as those books.
After this step, 1299 values were still missing. To avoid data deletion and to keep the data accurate, I decided to assign a new replacement value: 'Other'.

- ✓ EXTRA SPACE & SPECIAL CARACTERS

I removed the extra space in the title row "num_pages" & the special characters '.;;;'_ in the title + column "Publisher".

- ✓ ANOMALIES DETECTED

By printing the statistic description of the dataset, I realized that the 3 following attributes had – 0 – as minimum :
"Average_rating" / "num_pages"/ "ratings_count".
I can consider them as anomalies as 0 cannot be accepted as number of pages or average/ count ratings.
=> "Average_rating" &"ratings_count" :
I decided to remove the rows as when there was a '0' in "average_rating", there was also a '0' in "ratings_count". Those books have probably not been rated.
=> "num_pages" :
I decided to replace them by the mean as the value in the target column *("average_rating")* was well filled. Same for "ratings_count".
I also noted that "**Text_reviews_count**" had 0 as minimum, but I did not consider it as anomaly as readers can rate a book without leaving a text review.

ML Project Report :

BOOKS RATING PREDICTION

✓ OUTLIERS DETECTED

I considered the extravagant data following the same logic as for the anomalies :

>> average_rating : we did not modify as it is our target in this project.

>> ratings_count : we delete data above 4000000.

>> Num_pages : we replaced by the mean data above 4000.

>> Text_review_count : we removed data above 80000.

2. PRE-PROCESSING DATA

✓ NORMALIZING : Language_code attribute

I decided to group the English language and made 'eng-US' / 'eng-GB' and 'eng-CA' only one language together with 'eng'.

✓ ENCODING : language_code/ Title/ Author/ Publisher/ Genre

I decided to encode those attributes to have the possibility to try different combinations at the time of training the project. I choose the "Label Encoder" method as there were lots of values to encode for each attribute.

✓ EXTRACTION : Year from 'Publication_date"

I decided to extract the year instead of using the whole date. I didn't need to encode it as it is already a numerical value.

>> I decided to save the encoded data in a new dataframe. I will it use in the Classification part to avoid conflict with the regression part.

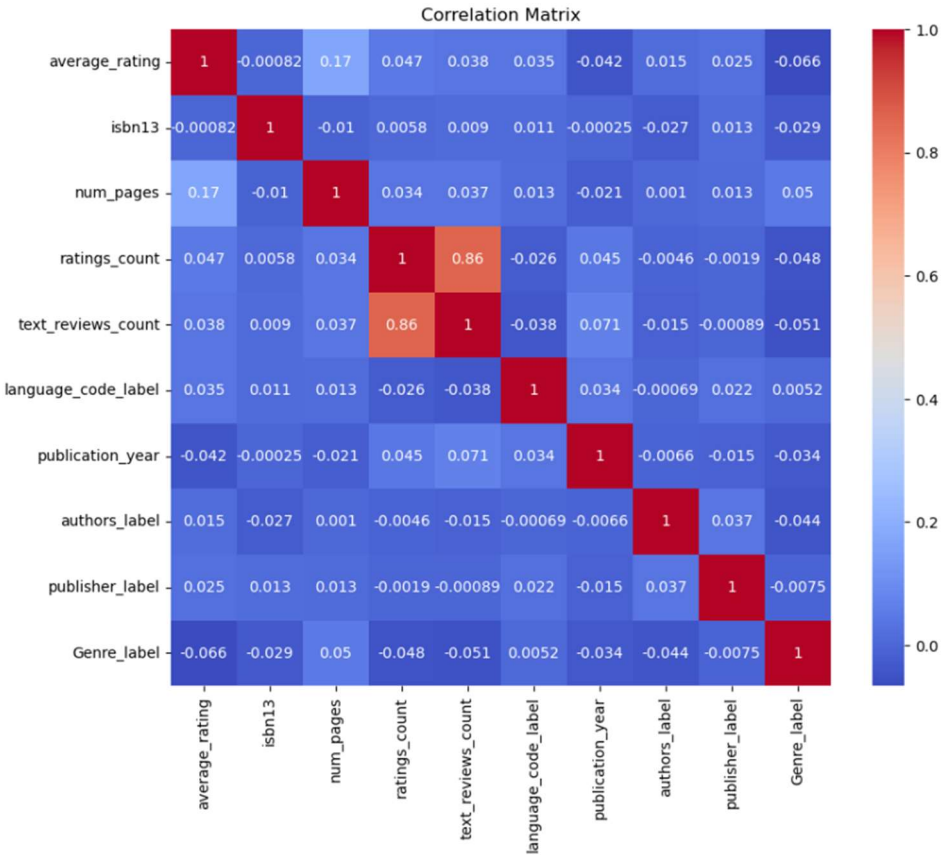
3. VISUALIZATION

✓ CORRELATION MATRIX

To have a representative correlation matrix, we need to have a maximum number of attributes represented.

According to the Correlation Matrix, we can observe that the **"ratings_count"** & the **"text_reviews_count"** are highly related (0.87). Among the other ones, the most related would be **"average_rating"** & **"num_page"** (0.17)

But more generally, the relation between the other attributes is very low. This suggests that the data are not really linearly related.



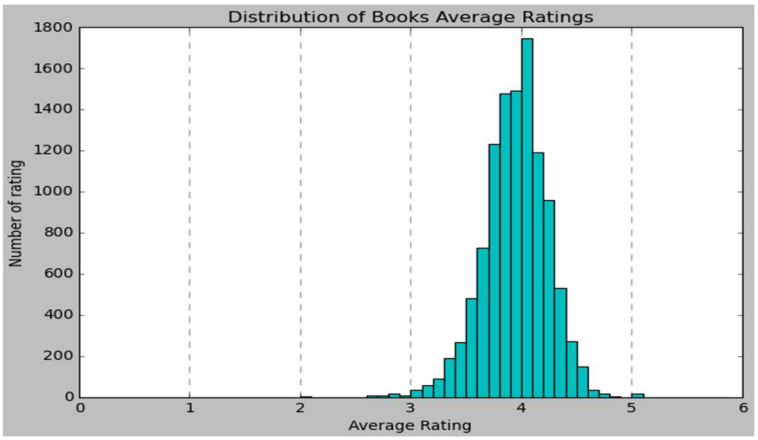
ML Project Report :

BOOKS RATING PREDICTION

✓ ANALYSING THE DATA

Average_rating :

According to the distribution of the average_rating, we can observe that the most given rates are between 3.8 and 4.2 *[4.0 being the most given (1805)]. The mean is 3.94. Ratings under 3.3 have been given less than 100 times : low ratings are in minority.*



Titles/ Authors :

We observe that half of the top 10 books is rated less than the mean (3.94). The 10 most represented authors of the dataset have not written the Top 10 rated books.

Languages :

The 3 most popular languages are English, Spanish, and French. But we can consider English as the main one as almost 97% of the books represented are written in this language.

Pages :

The pie graph shows that almost half of the books contain between 200 and 400 pages (338 pages being the mean)
Only 1.9% have more than 1000 pages.

Scatter Plot :

Thanks to the scatter plot, we can see the relations between 2 attributes.
- average_rating vs ratings_count : The concentration of the plots between 3.8 and 4.2 is clearly confirmed. No linearity here.
- average_rating vs num_pages : big concentration of books with less than 500 pages and rated between 3 and 4.5.
- text_review_count vs. ratings_count : We observe that the plots are following a virtual line. A linearity is observed here. We could see that in the correlation matrix with a high score (0.87)
- average_rating vs. language code : This graph confirms the majority of English books. It also confirms that even for other language books, the majority is highly rated.

4. TRAINING & TESTING THE MODELS

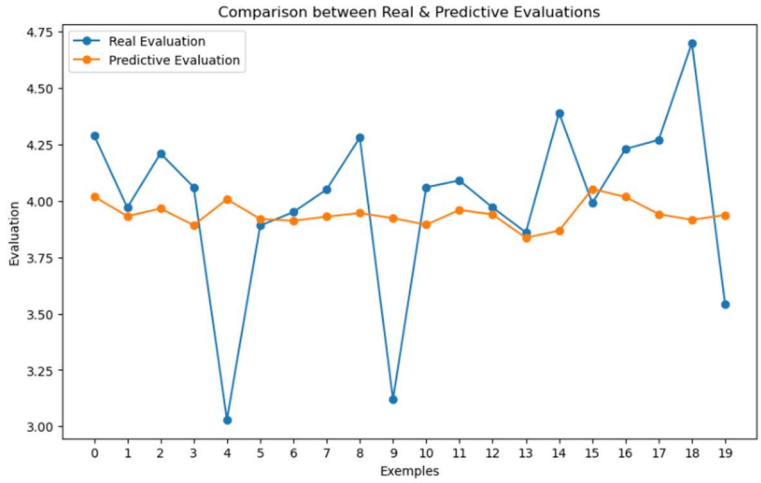
I decided to train the model with the following attributes :
>> ' average_rating' as a target
>> 'ratings_count'/ 'num_pages'/ 'language_code_label'/ 'Genre_label' as variables
We split the data in 2 parts : 80% for the training part & 20% for the testing one.

ML Project Report :

BOOKS RATING PREDICTION

LINEAR REGRESSION

This model is clearly not efficient. We can observe on the graph that the predictions are pretty linear, but the real value are not at all. We would have needed a R-squared close to 1 and a Mean Squared Error close to 0 to have an accurate model.



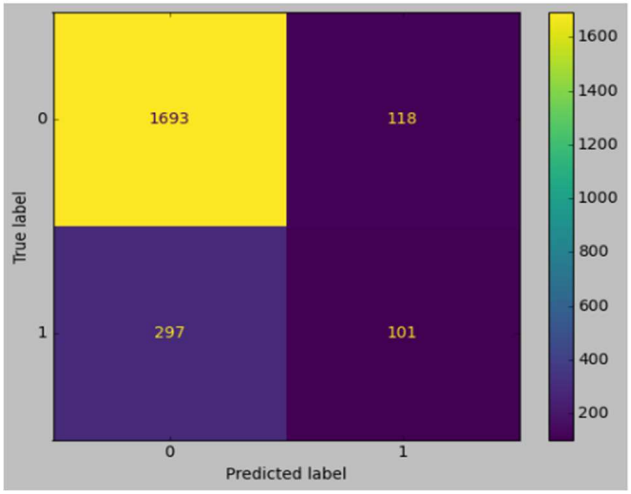
CLASSIFICATION MODELS

As I didn't get a good result with the Linear regression model, I wanted to tried classification models. For that, I had to make some adjustment in the data.
First, I had to transform the target into a binary classification. I choose the value of 4.2 as reference : Books rated 4.2 or above are considered Excellent books – Books rated under 4.2 are considered average or bad books.
I based my choice on the statistics : the mean is 3.94, the reference value had to be much higher. And according to the scatter plots, a reasonable reference would have been between 4.1 and 4.3. I split this range : 4.3 would have given better results, but it would have limited the chance for a book to be considered as an excellent book.

After testing the following models: Decision Tree, Random Forest, and Logistic Regression, I find that the results are better than with linear regression. However, although the accuracy rates are fairly positive (*Decision Tree: 74%, Random Forest: 81%, Logistic Regression: 82%*), the 'precision', 'recall', and 'F1 score' are rather poor. This makes our models ineffective and unreliable. *(See below ex. Random Forest)*

In order to try to improve my models, I've decided to implement oversampling using the **SMOTE method**. The same parameters have been kept *(attributes/reference rating*).

Random Forest Model



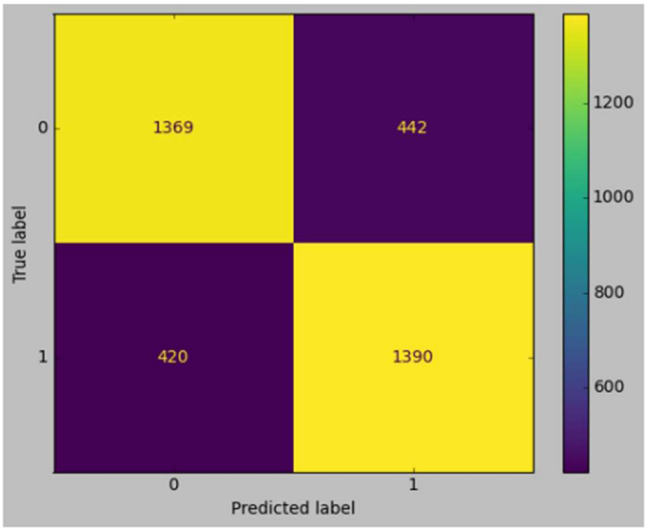
	precision	recall	f1-score	support
0	0.85	0.93	0.89	1811
1	0.46	0.25	0.33	398
accuracy			0.81	2209
macro avg	0.66	0.59	0.61	2209
weighted avg	0.78	0.81	0.79	2209

5. OPTIMIZATION : Oversampling with SMOTE Method

I notice that, thanks to oversampling, the results of my models have improved. The accuracy rates are slightly lower *(Decision Tree: 73% / Random Forest:76%),* but the 'precision,' 'recall,' and 'F1 score' have good results, making our models reliable. I also observe from the Confusion Matrix that the 'False Negatives' are lower than the 'False Positives,' further confirming the reliability of the models *(See below ex. Random Forest with Smote)*

As for the logistic regression model, it's a complete failure. With an accuracy of 50% and a confusion matrix with 0 for both true and false negatives, this result suggests that regression is definitely not a model to adopt for this type of problem.

Random Forest with SMOTE



	precision	recall	f1-score	support
0	0.77	0.76	0.76	1811
1	0.76	0.77	0.76	1810
accuracy			0.76	3621
macro avg	0.76	0.76	0.76	3621
weighted avg	0.76	0.76	0.76	3621

6. CONCLUSION : EVALUATION & IMPROVEMENT PROPOSAL

After attempting various combinations, the best results were obtained with the following columns: 'average_rating', 'language_code', 'num_pages', 'ratings_count', 'Genre'.
The linear regression model is clearly not suitable for this problem. This was initially evident through the correlation matrix, which showed very little linearity between the data.
On the other hand, classification models yield better results, even if they are not perfect.
Decision Tree and Random Forest models provide good accuracy; however, there is an imbalance in the data distribution, affecting the reliability of the models.
The use of oversampling with the SMOTE method corrects this imbalance and results in higher precision, recall, and F1-scores.
With or without SMOTE, logistic regression does not produce good results. Like 'linear regression', it's a model to be avoided for this type of problem.

To improve this model, I would propose the following solutions:
- Undersampling: Based on the statistical report, we observed that the target data is primarily skewed towards higher average ratings. Rebalancing the majority class could help stabilize the model. Given that oversampling has proven effective, it is reasonable to assume that undersampling might be effective as well.
- Hyperparameter Tuning: Throughout this project, we've seen that classification models yield the best results. Hyperparameter tuning, particularly effective for the Random Forest model, is worth exploring.
- Feature Engineering: Modifying the data combination, removing more outliers, and adding new columns are also worth exploring.
-Other Models: Trying out other models, ideally classification ones, could be beneficial given that this approach has so far been the most suitable for the problem at hand (*such as Gradient Boosting Classifier, XGBoost, etc.*).