# Topic Modeling and Sentiment Analysis for Amazon Book Reviews

Andres Bermeo Marinelli

NLP Final Project, A.A. 2021-2022

## Contents

## 1 Problem Statement

The main aim of this project is to implement tools based on Natural Language Processing techniques to be used for the following tasks:

1. Help publishers and authors understand the topics of books being sold on Amazon to have a better idea of the current interests and overall market situation.

2. Classify user reviews in order to incorporate this knowledge in a collaborative based filtering technique where similar tastes between users are used to recommend new items.

To address these problems, we perform topic modeling and sentiment analysis on a corpus of book summaries and corresponding amazon reviews[1].

After studying the most frequent words in the corpus of book descriptions, we extract the K main topics that characterize them by means of Latent Dirichlet Allocation (LDA), along with the 10 most relevant words per topic. Consequently, we try to interpret the theme/meaning of each of these and analyze their association to the categories provided in the dataset. Lastly, by using the topic distribution of each document, we study how their popularity changes over time.

For sentiment analysis, we use RoBERTa to classify the reviews as positive or negative and compare them to grouped ground truth labels. Lastly, we fine-tune RoBERTa using HuggingFace's Trainer environment on our own data to improve the model. We compare it to a baseline classifier which always predicts the most frequent class.

## 2 Data Description

The dataset contains two separate files with data on books and their reviews. In particular, these two have a one-to-many relationship, and each review must correspond to an existing book in the other file.

The first file contains details about $\sim 212,000$ books. In particular, it contains information such as title, author, published date, description, and category. Around $68,000$ (30%) items have a missing description, so they are eliminated, since the main focus is to extract topics from the summaries of books. Furthermore, not all the books are in English. Since LDA implicitly assumes the same language for its probabilistic structure of document creation, we remove documents not in English after applying language recognition tools to the summaries. This leaves a total of $\sim 142,000$ items. In this subset, we study the frequency of the categories and find that the three most popular are fiction, history, and religion. However, the existing categories are 521, so we limit ourselves to showing the top 10 most frequent.

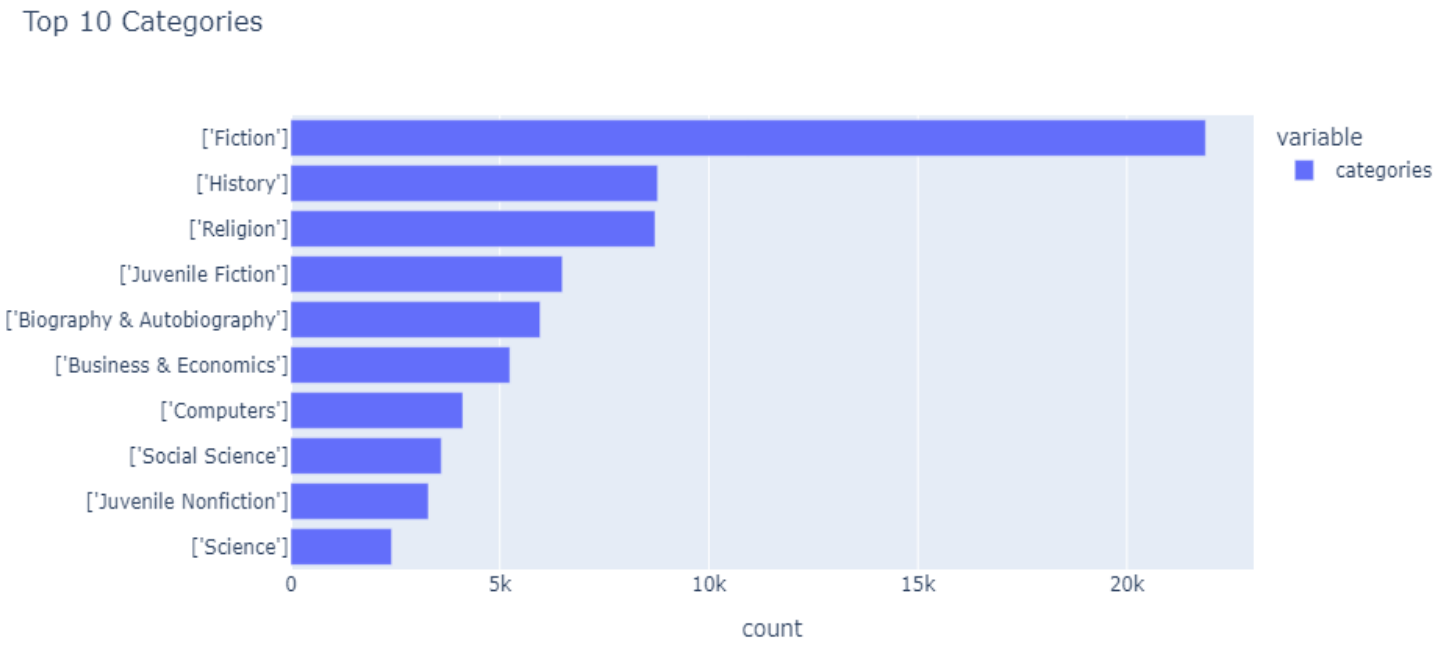


Figure 1: The top 10 most frequent categories of books. "Fiction", "history", and "religion" figure among the top 3.

The second file contains information about 3 million reviews of the books contained in the first file. In particular, we have info such as book id (being reviewed), title of the book, id of user reviewing, summary of the review, text of the review, and finally a score from 1 to 5. The scores are severely imbalanced with 5 star reviews being the most predominant class.

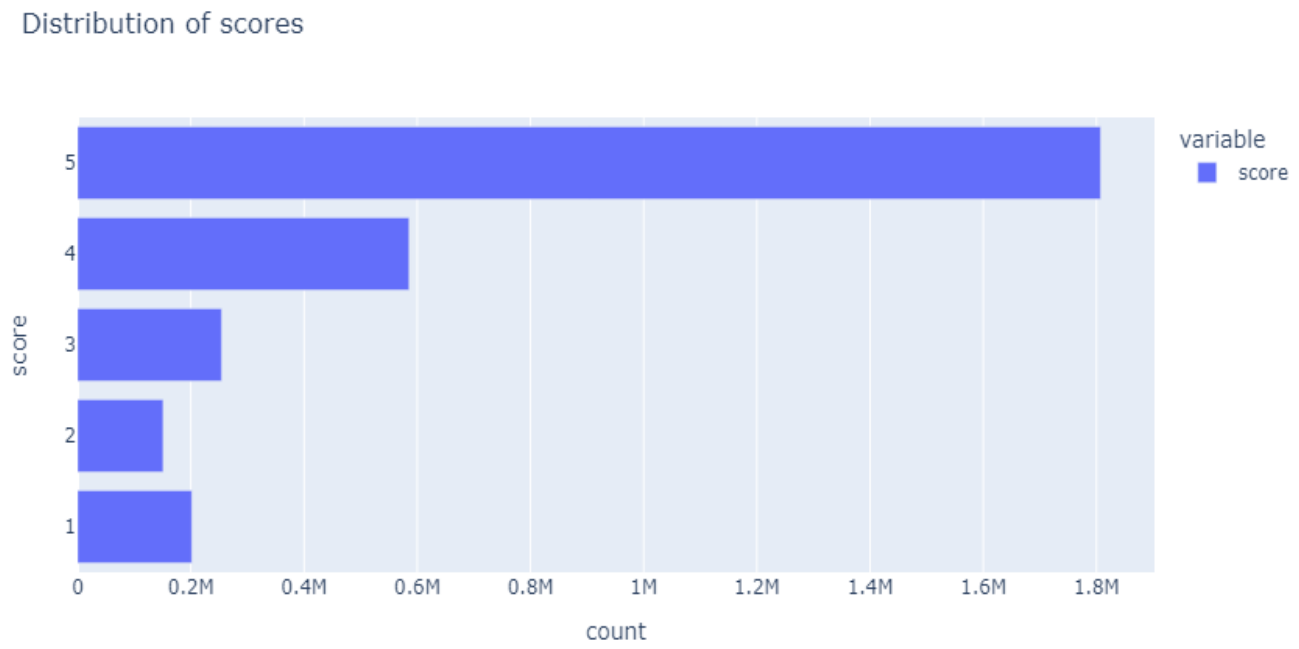


Figure 2: The distribution of scores shows that classes are imbalanced and the majority is composed of 5 star reviews.

## 3 Assessment and Performance Indexes

With regard to parameter tuning, for topic modeling, LDA requires as input the number of topics $K$. To choose this value, we vary $K \in [2, ..., 20]$ and use two metrics: UMass and CV score[2]. The optimal number of topics is reached when these two scores are locally maximized. However, we also keep in mind the principle of Occam's razor which prefers simpler, more explainable models, sometimes at the expense of a lower metric/score.

On the other hand, for sentiment analysis, we use accuracy, precision, recall, and f1-score. In particular, we use sklearn's classification report to quickly obtain a per-class score of the last three metrics as well as an overall accuracy of the model. In the end, when we compare the baseline with a fine-tuned instance of RoBERTa, we use balanced accuracy, micro, macro, and weighted f1-scores. This is done to properly study the effects of class imbalance.

## 4 Topic Modeling

After having eliminated items without book descriptions and non-English text, we are left with $\sim 142,000$ elements. At this point, to obtain an even more descriptive summary of the book, we concatenate the summary of the book with the title by separating these two with a period.

Then we aggressively pre-process the text to reduce variance to a minimum and capture the most meaning with the word descriptors after running LDA. In particular, we normalize the text by lower casing, removing all symbols and punctuation (except for hyphens and apostrophes); we tokenize using spacy, we keep the lemma of each word, we remove stop words contained in the default list of spacy, removing also custom words such as "book", "author", "write", and "story". Finally, we find the joint collocations with $PMI \geq 1.0$ and represent them as bi-grams. This bound was chosen to limit the computational requirements and keep only the most meaningful collocations.

We construct the word cloud from the pre-processed descriptions to study the most frequent terms in order to have some prior ideas of some important topics/themes in the corpus.

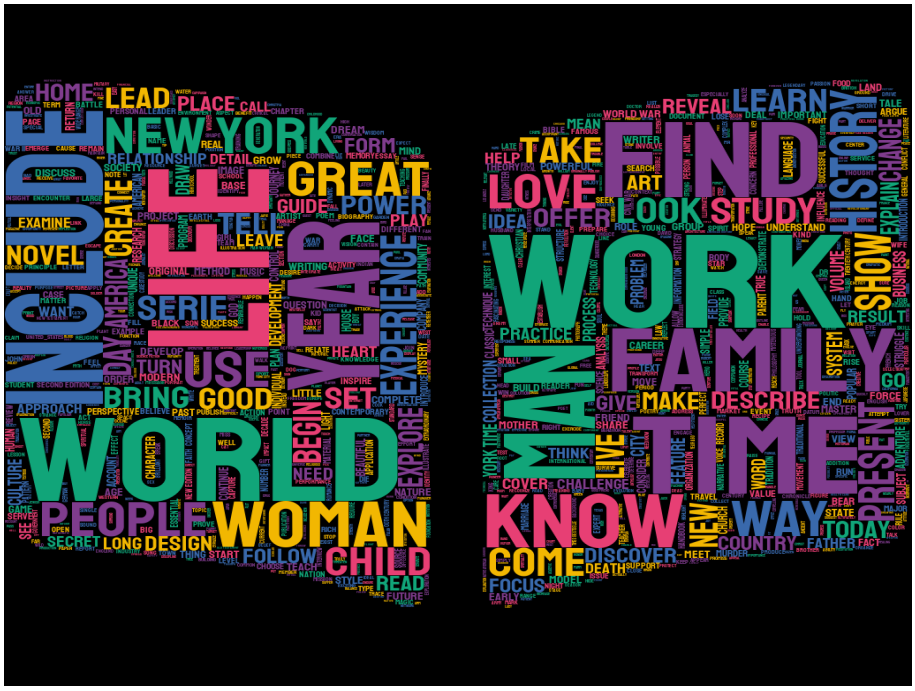


Figure 3: The word cloud of the most frequent terms in the corpus of book summaries. Terms such as "work", "world", "life", "history", "time", "love", "family" indicate the themes being discussed.

Some of the most frequent words such as "work", "world", "life", "history", "time", "love", "family", are very suggestive of the topics present in the corpus. These are powerful indicators of possible topics that we may find in the corpus. Also, we can see that they can be linked in some way to the most popular categories shown before.

After having prepared the corpus for LDA, we must determine the number of topics to extract. To achieve this, we loop over $K \in [2, ..., 20]$ and train LDA on the first $10,0000$ documents, and compute UMass and CV scores on the next $40,000$. We set $\alpha = 0.01$ and $\eta =$ "auto". The former tells us to expect around one topic per document while the latter is related to the specificity of words to topics and setting it to "auto" enables LDA to learn a prior from the corpus. We obtain the following curves for varying $K$:

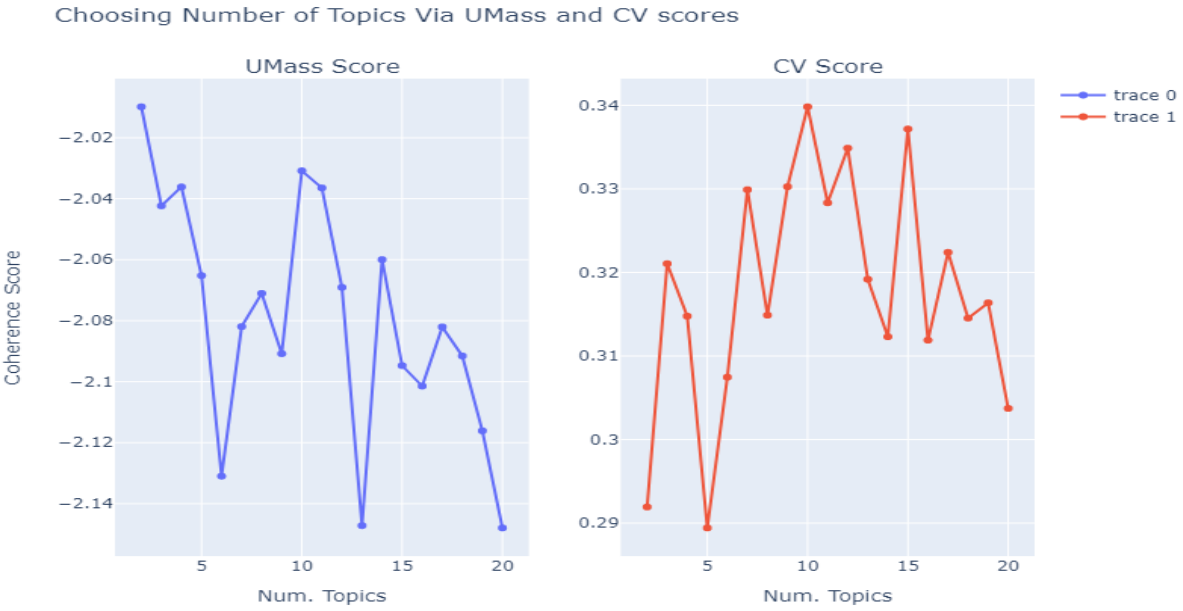


Figure 4: UMass and CV scores with varying number of topics. The maximum occurs at $K = 10$ which indicate the optimal number of topics.

As we can see, we obtain that the local maximum is at $K = 10$ topics for both UMass and CV score. Specifically, for the CV score, $K = 10$ represents a global maximum.

Finally, we can extract the top 10 descriptors for each topics and examine the results.

| | Word 1 | Word 2 | Word 3 | Word 4 | Word 5 | Word 6 | Word 7 | Word 8 | Word 9 | Word 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| Topic 1 | recipe | food | guide | cookbook | cook | garden | plant | dish | cooking | eat |
| Topic 2 | child | life | god | love | parent | help | dog | little | animal | baby |
| Topic 3 | social | theory | political | study | history | science | economic | culture | human | american |
| Topic 4 | music | art | film | work | artist | original | include | guide | song | history |
| Topic 5 | bible | god | church | christian | jesus | testament | biblical | spiritual | study | christ |
| Topic 6 | student | guide | edition | provide | business | design | language | include | information | system |
| Topic 7 | love | novel | man | life | find | woman | murder | family | young | new |
| Topic 8 | game | baseball | sport | guide | player | bird | quilt | include | color | history |
| Topic 9 | poem | poetry | poet | litarature | work | american | history | publish | english | essay |
| Topic 10 | war | history | american | world | life | man | america | battle | year | new |

As we can see, the topics are quite easy to interpret:

- topic 1: `cooking`.
- topic 2: `family`.
- topic 3: `socio-economic science`.
- topic 4: `fine arts`.
- topic 5: `religion`.
- topic 6: `student help`.
- topic 7: `fictional romance`.
- topic 8: `sport & outdoors`.
- topic 9: `poetry & literature`.
- topic 10: `american history`.

[1]We notice that `american history` and `religion` reconnect exactly to the categories found in the top 10 list. Looking at the top 40 categories, we realize that a lot of the topics found by LDA are either explicitly contained as a category or represent a mix of these. For example, `cooking` and `family` figure by the same name in the 11th and 13th position respectively (actually, it's *family & relationships*).

Other topics on the other hand, represent overlaps of categories. For example, `student help` is probably related to *juvenile non-fiction* and *self-help*; `socio-economic science` is an overlap of *social science* and *business & economics*; `fine arts` is related to *music*, *art*, and *performing arts*; `sport & outdoors` is a mix of *sport & recreation*, *health and fitness* and *nature*; `fictional romance` is probably linked to *fiction* and *juvenile fiction*; finally, `poetry & literature` is connected to *poetry*, *literary criticism*, and *language arts & disciplines*.

As we can see, the extracted topics are useful because they can help to summarize and group together the existing categories which are related. Furthermore, they can also condense information that in our case is given by analyzing 521 categories.

Finally, we analyze how the topics change throughout the years. In particular, we will analyze the last 12 years (for graphical and practical reasons). We associate to each document the most relevant topic and plot how many items belong to each topic per year. Below we can see the results:

[1]To differentiate between LDA topics and categories, we use bold font and italics, respectively.

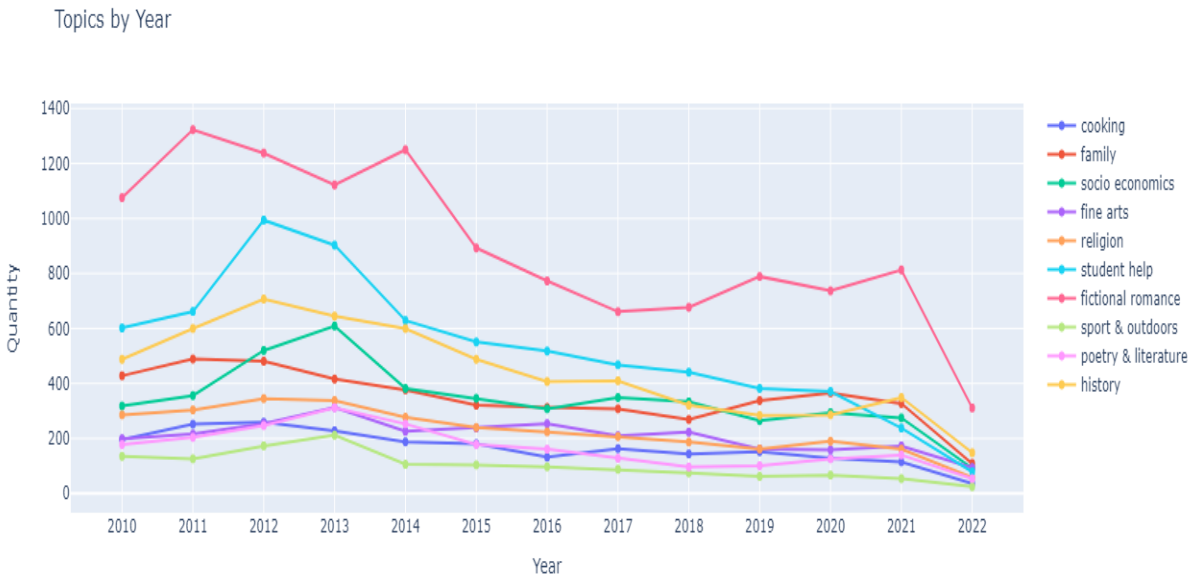


Figure 5: Frequency of topics per year since 2010. The overall trend is downwards, however, 2 years ago, `fictional romance` increased.

Some notable observations from the bar chart above:

1. In 2021, the number of books related to `fictional romance` and `family` has gone up. This could be due to the difficult times due to the pandemic which undoubtedly left lots of people with desires of affection and human contact[3] as well as an intriguing story to occupy their time during lockdown/quarantine.

2. There was a noticeable increase in quantity of books related to `student help` in 2012. This could be linked to the boom in popularity of massive open online courses - MOOCs in those years[4]. It could be that the boom in MOOCs cause an increase in the production of texts for helping students in various aspects.

3. Not many books related to `sport & outdoors` are produced. This could be due to the fact that this is a theme in which people prefer to learn by doing rather than reading.

4. All the trends are decreasing and the sheer quantity of books are also less. This could be due to lack of data for these years or perhaps it could reflect a general trend of diminishing interest in books due to the internet.

## 5 Sentiment Analysis

We have a dataset of 3 million reviews, with an associated score from 1 to 5 stars. We will focus on trying to predict the sentiment of a review based on the review summary instead of the entire text itself. This is done mainly for computational reasons.

We try to create word clouds for positive and negative reviews, mainly to understand the type of sentiments we expect to find. One of the first difficulties is how to group 3 star reviews. In similar applications, it is customary to group 3 star reviews together with 4 and 5 stars and classify them as "positive". However, a quick sampling of 3 star reviews shows that they are associated to both negative and positive sentiments. Therefore, they were marked as "neutral", while 1-2 and 4-5 star reviews were classified as "negative" and "positive" respectively.


(a) (b)

Figure 6: Negative reviews word cloud (a) and positive reviews word cloud (b) reveal the words associated to negative and positive emotions respectively.

For the negative sentiment word cloud (a) we have words such as "dissappointing", "bad", "boring", which indicate the emotions associated to these reviews. However, we also observe the presence of words such as "great" and "good", which seems counterintuitive at first glance. However, a quick search-and-find of the documents containing these terms shows that they are always preceded by "not" or "not so", in order to form a negative sentiment. This could have been avoided through the use of joint collocations.

For the positive sentiment word cloud (b), we see words such as "good", "great", "wonderful" and "excellent", which is a nice insight into the type of sentiments that are communicated with positive reviews.

For the sentiment classification we use a pre-trained model from huggingface named "SiEBERT"[5] which is an english language sentiment classifier. This model was trained on 15 different english datasets, including reviews and tweets and outputs a binary classification - positive or negative. The model has its own tokenizer which can be readily used. In fact, this is the reason that joint collocations were not included, since the tokenizer may separate the bigram automatically.

Using the grouping described above, we remove all neutral reviews and fix a random subset of $100,000$ english reviews, to reduce computation time. Removing neutral reviews is an acceptable step since we are unable to classify them as positive or negative, and when studying the performance of the classifier, we need to be sure that the ground truth is correctly labeled, otherwise the performance metrics would not reflect the model's real capacity and fine-tuning may lead to undesired behavior.

We plot the distribution of the resulting sub-dataset to verify that it still follows the original distribution. As we can see it is still very imbalanced:

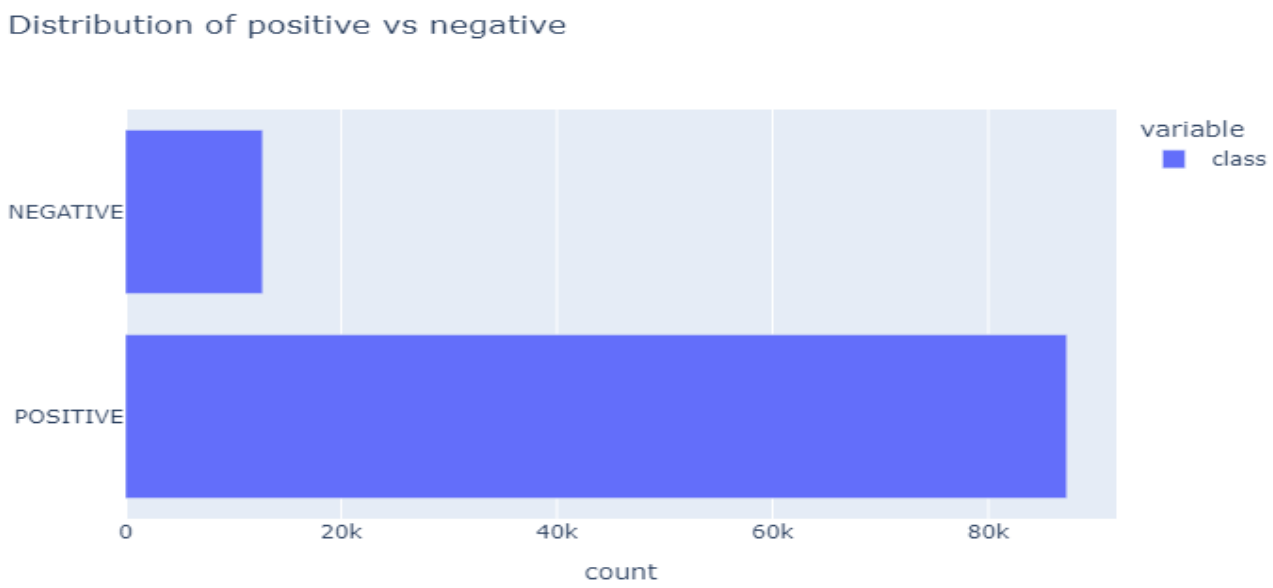


Figure 7: Distribution of positive and negative class.

Using the model's tokenizer to pre-process the review summary, we extract the predicted score for each review and run a classification report on the results to obtain metrics per-class.

| RoBERTa | precision | recall | f1-score | support |
|---|---|---|---|---|
| negative | 0.51 | 0.82 | 0.63 | 12720 |
| positive | 0.97 | 0.89 | 0.93 | 87280 |
| | | | | |
| accuracy | | | 0.88 | 100000 |
| macro avg | 0.74 | 0.85 | 0.78 | 100000 |
| weighted avg | 0.91 | 0.88 | 0.89 | 100000 |

This model has high values of recall for both classes and a high precision for the positive class. However, it has a precision of 50% for the negative class, which indicates that half of the time that the model classifies as negative, it is actually positive. This is largely due to the imbalance of the classes since the missing 10% of the recall for the positive class corresponds to around $8,500$ documents which are positive but classified as negative. However, the size of this misclassification is comparable to the size of the negative class, which will cause its precision to go down.

We fine-tune this model to our dataset by using huggingface's trainer[6] environment. We construct a train, validation, and test set by performing a random $60-20-20$ split. We set the learning rate $lr = 2e-5$, and train for 5 epochs, evaluating and saving our model at each of these. We obtain the following results:

| Epoch | Training Loss | Validation Loss | Accuracy | F1 | Precision | Recall |
|---|---|---|---|---|---|---|
| 1 | 0.206900 | 0.178979 | 0.937300 | 0.964278 | 0.959358 | 0.969248 |
| 2 | 0.169200 | 0.196870 | 0.940950 | 0.966765 | 0.950423 | 0.983679 |
| 3 | 0.136100 | 0.192735 | 0.941700 | 0.966914 | 0.958268 | 0.975719 |
| 4 | 0.112300 | 0.241649 | 0.942200 | 0.967200 | 0.958497 | 0.976062 |
| 5 | 0.094600 | 0.248630 | 0.939800 | 0.965667 | 0.961718 | 0.969648 |

Figure 8: Metrics for RoBERTa after fine-tuning on $60,000$ reviews. The metrics are computed on the validation set of $20,000$ items. After the first epoch, the model is clearly over-fitting as can be seen by the decrease in training loss and increase in validation loss.

After the first epoch, the model is already over-fitting, as evidenced by the decrease in training loss and increase in validation loss. We use the model weights in the first epoch to recompute the predictions.

Running the classification again on the test set, we obtain the following classification report:

| fine-tuned RoBERTa | precision | recall | f1-score | support |
|---|---|---|---|---|
| negative | 0.77 (+0.26) | 0.70 (−0.12) | 0.73 (+0.10) | 2561 |
| positive | 0.96 (−0.01) | 0.97 (+0.08) | 0.96 (+0.03) | 17439 |
| | | | | |
| accuracy | | | 0.93 (+0.05) | 20000 |
| macro avg | 0.86 (+0.12) | 0.83 (−0.02) | 0.85 (+0.07) | 20000 |
| weighted avg | 0.93 (+0.02) | 0.93 (+0.05) | 0.93 (+0.04) | 20000 |

As we can see, the model has noticeably improved compared to before. We have higher values for precision (51% → 71%) and f1-score (63% → 73%) for the negative class, and higher recall (89% → 97%) and f1-score (93% → 96%) for the positive class. Furthermore, the accuracy also increased (88% → 93% ).

The precision for the positive class decreased slightly (97% → 96%) and the recall dropped considerably (82% → 70%). However, the performance metrics indicate an overall improvement and are more desirable for an all purpose application.

We compare this model to a baseline classifier which predicts the most frequent class for all items. Instead of per-class metrics, we use global scores such as the f1-score. In particular, we use different flavors of f1-score: macro, micro, and weighted. All of these account for class imbalance in different ways. For the same reason, we also use balanced accuracy instead of normal accuracy. We show the final results below:

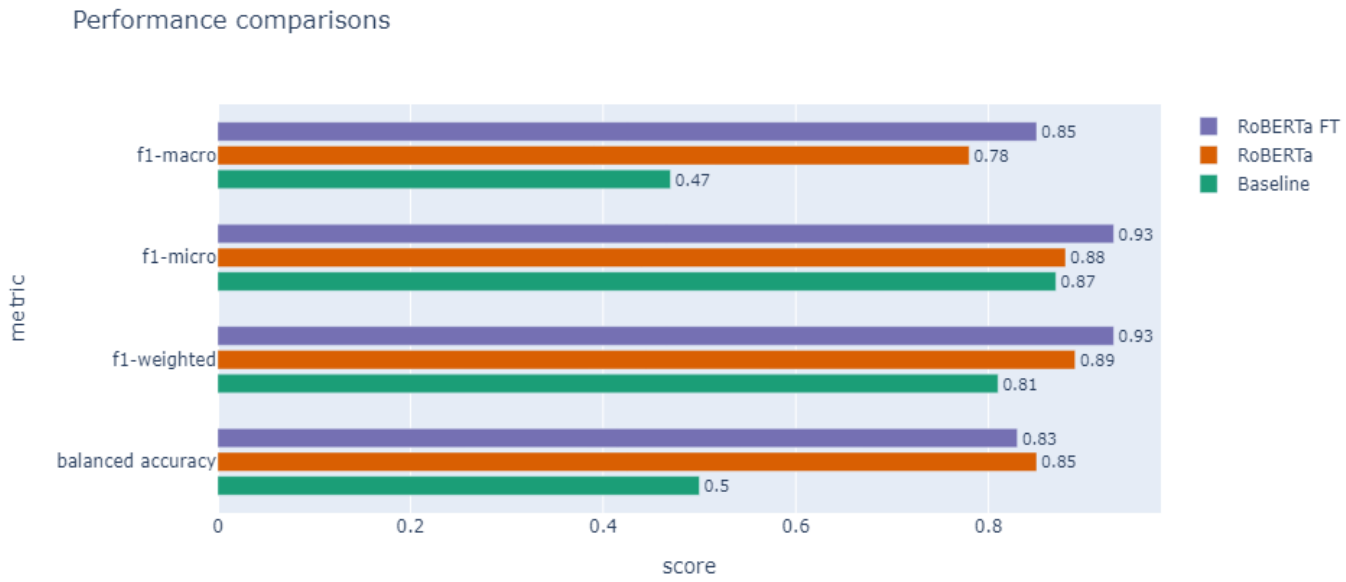


Figure 9: Comparison between the baseline most frequent classifier, RoBERTa, and RoBERTa fine-tuned. The latter model has the best overall performance.

We notice that both the model and its fine-tuned version performs better than the baseline in all the metrics.

In particular, we see that the balanced accuracy for the baseline is reduced to $1/n_{classes}$ which is the accuracy of a random classifier.

Both models also outperform the weighted and micro f1-scores, which is important due to the high imbalance of the classes. To briefly explain further, the baseline will correctly predict all the occurrences of the positive class, which account for approximately 85% of the dataset. Thus, if we are weighting the metrics by their class size, this will "hide" the null performance on the negative class. However, both models are able to outperform this and correctly predict the majority of the positive class and also 70% of the negative class (in the case of fine-tuned RoBERTa).

We are not surprised that the models outperform the baseline in the macro f1-score since this gives equal weights to the classes and the baseline misclassifies all the negative instances.

Lastly, the only area where the pre-trained model surpasses the fine-tuned model is in balanced accuracy. However, the difference is so small that it could be due to fluctuations in the testing set. In fact, we must remember that for the pre-trained model we compute the metrics on a test set which contained 100,000 items, while for the fine-tuned model the

test set only has $20,000$ reviews. A better approach would have been to fine-tune the model on $100,000$ items and then test it on another $100,000$ items. However, this was not done for computational reasons.

## 6 Results and Discussion

In the first section we extracted 10 topics from the corpus of book descriptions ( $\sim 140,000$ items) by maximizing UMass and CV scores. By analyzing the top 10 descriptors for each topic, we were able to draw connections to the categories of the dataset. In particular, we saw that all of the topics either appeared as a category, or encapsulated 2 or more categories which were very related. Furthermore, the topic descriptors also served to have a better idea of the type of words associated to each topic, which gives a deeper understanding of what a category means, rather than leaving it to interpretation.

However, in the absence of supervision (i.e the categories), this tool could enable authors and publishing companies alike to obtain information about the topics of a corpus and study how their frequency changes over time. This in turn could give an insight into which categories are more popular and worth publishing or writing about. In our case, we saw that the topic of `fictional romance` was produced more frequently during the years of covid, which indicates perhaps, that this would've been a good moment to focus on this thematic. Another interesting observation was the increase in 2012 of books related to `student help` which coincided with the boom of MOOC's. Finally, the data showed an overall decline in volume, which could be due to lack of data collection or could indicate a general trend towards reading less.

In the second section, we use a pre-trained RoBERTa classifier to predict reviews as positive or negative based on their summary. The model was originally trained on 15 different datasets which made it much more versatile in terms ability to classify text accurately. After classifying a random sample of $100,000$ reviews, we observed that the model had a precision of 50% for the negative class, which is the precision of a random classifier.

Subsequently, the model was fine-tuned for 5 epochs on a subset of $60,000$ reviews, and validated on $20,000$ items. After one epoch the model was at the optimal state compared to epochs later where it was overfitting. Using the model weights obtained in the first epoch, we classified reviews on a held out test set of $20,000$ documents and found it outperformed the model without fine-tuning.

Finally, we compared both models (with and without fine-tuning) to a baseline classifier which predicted the most frequent class. The fine-tuned model outperformed the other two in all of the aggregate metrics, such as f1-score weighted, micro, and macro. On the other hand, the balanced accuracy dropped slightly compared to the model without fine-tuning.

This tool gives us a simple method to evaluate user reviews and integrate this knowledge into a recommender system which is based on user similarity. Specifically, if two users with similar features, rate the same items as positive and negative, then we could recommend items that one user liked to the other.

## References


[1] https://www.kaggle.com/datasets/mohamedbakhet/amazon-books-reviews

[2] https://www.os3.nl/_media/2017-2018/courses/rp2/p76_report.pdf

[3] https://www.frontiersin.org/articles/10.3389/fpsyg.2021.798260/full

[4] https://onlinelearninginsights.wordpress.com/2012/12/21/what-the-heck-happened-in-2012-review-of-the-top three-events-in-education/

[5] https://huggingface.co/siebert/sentiment-roberta-large-english?text=I+like+you.+I+love+you

[6] https://huggingface.co/docs/transformers/training#additional-resources