Supervised and Unsupervised Sentiment Analysis on Goodreads Review Text Final Report

Asia Paige | Malarvizhi Veerappan | Srilekha Sikhinam
Team 22

Motivation

Goodreads is an online catalog where users can search, review, and register books. This website also allows for quote and annotation searches. With so much text information offered on the website, our team was motivated to perform a sentiment analysis on Goodread reviews.

To do this, we used datasets from the goodreads website UCSD Book Graph website (See requested citations below). Our goal for this project was to see which machine learning method, supervised or unsupervised, was best for predicting sentiment from the review text.

Data Source

As stated above, our datasets came from the UCSD Book Graph website created by four researchers used for recommendation systems and fine grained sentiment analysis on book spoilers research. The data in these datasets were collected in 2017 and updated in 2019.

All data on this website is provided in a zipped json file and are of varying sizes. Datasets used by our team include: the authors dataset (17,457KB), books dataset (~2G), genre dataset (23,686KB), and reviews dataset (~5G). From each dataset, we pulled 250,000 rows. Important variables within the datasets included: **rating**, **review_text**, **genres**, and **author**.

Cleaning and Preprocessing

For both supervised and unsupervised models, the below cleaning and preprocessing steps were performed and utilized. For unsupervised models a few additional preprocessing steps were performed.

Cleaning steps taken include:

- dropping null values
- making all text lowercase
- removing any existing numerical digits
- removing additional spacing
- removing the newline character '/n'
- removing reviews that are not in English
- and, extending contractions within the text.

Numerical digits were dropped from text since they do not add any significance or weight to the words for this project. While working with the reviews in this dataset we identified many contractions, and in order to maintain meaning of each word during preprocessing, we decided to extend them.

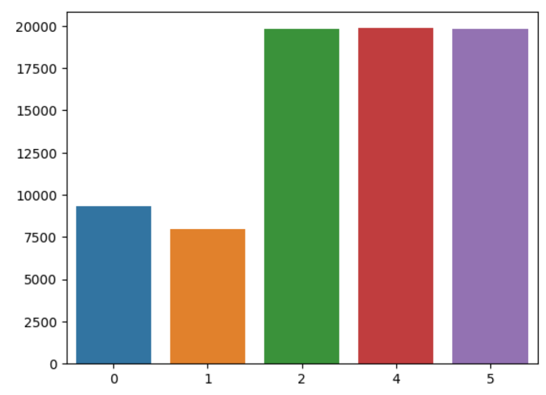
Preprocessing steps taken include:

- stratifying the data. Since stratification of the entire 15 million rows needed more computing capacity, we stratified 250,000 rows from the first 2,000,000 rows.
- adding a column of labels (positive, neutral, and negative) based on the rating
- removing stopwords
- and, lemmatizing/stemming words.

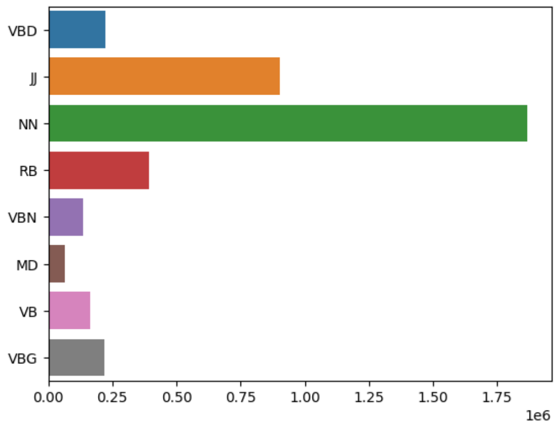
All books with a rating of 1 or 2 were given the label negative, ratings of 3 were given the label neutral, and ratings of 4 or 5 were given the label positive. For this project, we've decided to exclude reviews with rating 3 (neutral) to focus on binary classification.

Exploratory Data Analysis (EDA)

After filtering the data for reviews only in English and for ratings of 1, 2, 4, and 5, the total length of the dataframe is 76,860.

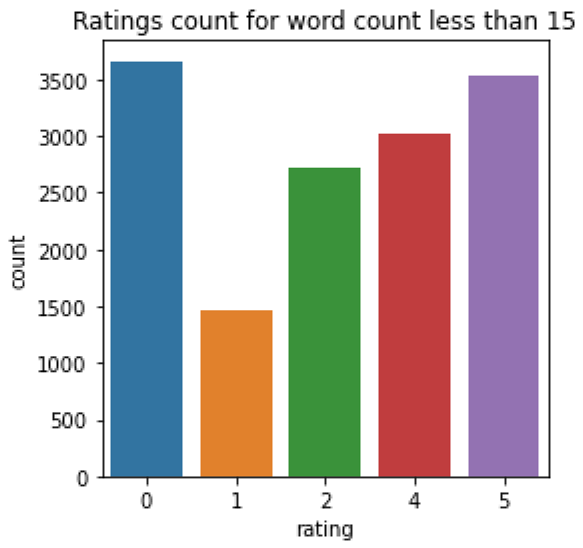


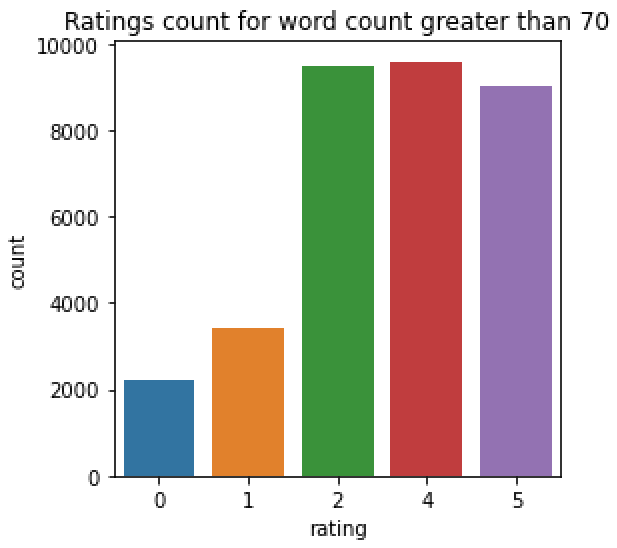
To check for imbalances in our data, the length of the strings within the review_text column were checked. We found that most reviews fall between 500 and 1000 words. There is a low number of reviews that fall between 2,000 and 6,000 words. This shows that review_length is consistent with only a few outliers. Checking the count of ratings was also explored to check for further imbalance. Ratings of 2, 4, and 5 are all even with a count of 2,000 ratings each. Ratings of 1 have the lowest rating count in the dataset with a count of 7,500 ratings. Since ratings of 1 are combined with ratings of 2, the ratings in the dataset are mostly balanced and we don't believe an imbalance will be a major issue.



Parts-of-Speech (POS) tagging was also done for further exploration. There are 8 main parts of speech: Noun ('NN'), Verb ('VB'), Adjective ('JJ'), Adverb ('RB'), Proposition ('IN'), Conjunction ('CC'), Pronoun ('PRP'), and Interjection ('INT'). Nouns are the most common part of speech in all of the reviews with adjectives coming in second. This is an interesting find since we believed that adjectives would be overall more common in reviews.

In addition, we checked to see the balance of good and bad reviews based on word count. We plotted two graphs to see the distribution of words with count less than 15 and words with count more than 70 (average word count across all ratings is 57). As observed from the graphs below, you will see that the low ratings seem to have text reviews with less descriptive words and higher ratings have more descriptive words.





Supervised Learning

Objective

Naive Bayes, Logistic Regression, and Support Vector Machine (SVM) models were chosen for sentiment analysis comparison and evaluation. SVM and Logistic Regression are commonly used in text classification and Naive Bayes is a good baseline for most classification models. The SVM model is often used for binary text classifications because it is able to predict well in high dimensional feature spaces and eliminates the need for feature selection. Logistic Regression is an easily trained efficient model that is quick and predicts well. We noticed that SVM took a long time to run with all samples and decided to reduce sample size to 50,000 for Logistic Regression and Naive Bayes models. Please note that df.sample without the random_state parameter was used to pull 50,000 samples.

Additional Preprocessing

Feature scaling using StandardScaler in sklearn was attempted for the SVM model. This was attempted due to scaling helping SVM make better overall predictions by reducing large values within the reviews. Running the model without scaling was also attempted for SVM.

Vector Representations

Vectors were represented in three different ways: Tfidvectorizer, Bag-of-Words (Countvectorizer), and Word2Vec vectors. In order to create the Word2Vec word vectors, sentence vectors were first created from all the word vectors and then averaged.

Accuracy, precision, recall, and F1 scores were used for evaluation with emphasis on accuracy.

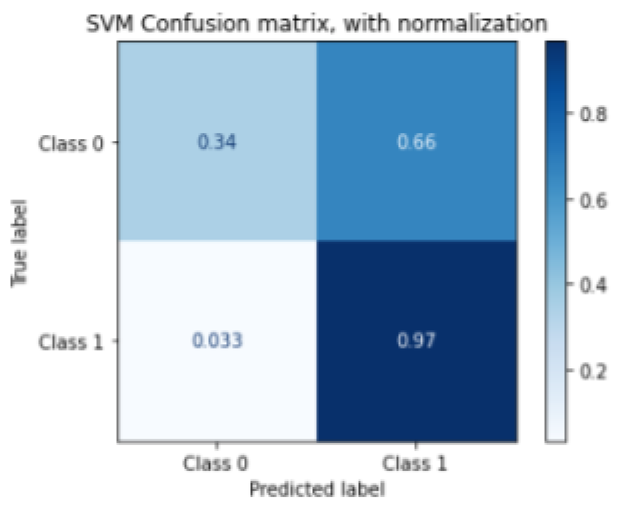
Methods and Evaluation

For supervised machine learning, predictions were made using two methods:

1. Sample reviews with stop word removal and lemmatization/stemming
2. Sample reviews without stop word removal or lemmatization/stemming.

The columns 'review_text' was used as a model feature and 'ratings' were converted into "good" or "bad" sentiment used for model prediction. A confusion matrix was then generated for each model to evaluate prediction success and failure across the different vectorizers used.

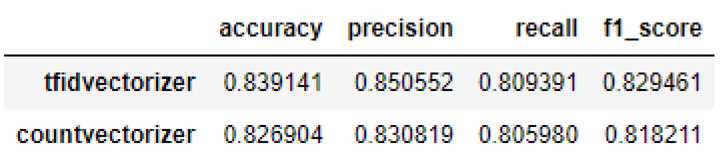
Support Vector Machine (SVM)



Tfidf Vectorizer Confusion Matrix

A ten-fold Cross Validation was done for this model initially using 10,000 samples from the cleaned and preprocessed 250,000 dataset on all vectors. Results from this validation showed that the Tfidvector performed the best while the Word2Vec model performed the worst with a difference of .25. Due to this and the fact that the Word2Vec vector would kill the kernel in great lakes cloud computing, it was decided to drop the use of the Word2Vec vector.

Due to wifi connectivity issues, this model was only able to run once with all text no stop words removed for 'rbf' and 'linear' kernels. This model takes a while to train the more samples given to it, and with connection going in and out, great lakes were not able to go through this model multiple times. The confusion matrix and the count vectorizer vs. tfidf vectorizer dataframe are the findings that we did make with this model using all 76,000 reviews.



	accuracy	precision	recall	f1_score
tfidfvectorizer	0.839141	0.850552	0.809391	0.829461
countvectorizer	0.826904	0.830819	0.805980	0.818211

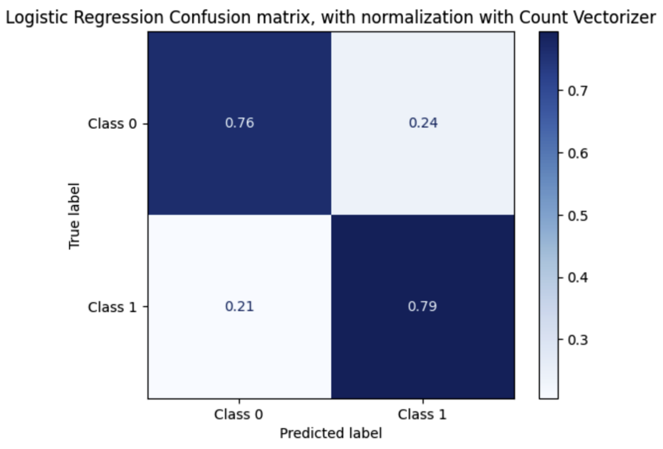
This model was trained using the 'rbf' and 'linear' kernel parameters (*clf_svm = SVC(random_state = 42, kernel = 'linear'/'rbf'*). The linear kernel had the better overall results with a difference of . 016 front the

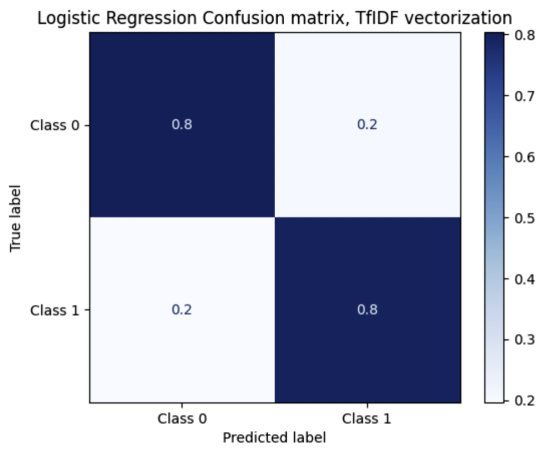
Tfidvector which is the best performing vector for both kernels. F1_score of this vector was .829 with recall and precision scores being .809 and .850 respectively.

The normalized confusion matrix of the Tfidvector showed a good ability to predict True Negatives but prediction failure in predicting True Positives.

Logistic Regression

Stop words were removed and stemming was specifically done for this model to make it more efficient and scalable. Snowball Stemming was decided over lemmatization here since lemmatization was not efficient when working with large data sets, however, we do notice that the output resulted in a few words that are not in "perfect English". This could widely be due to parts of speech not being implemented in the lemmatizer and so making the lemmetizer default parts of speech to noun.





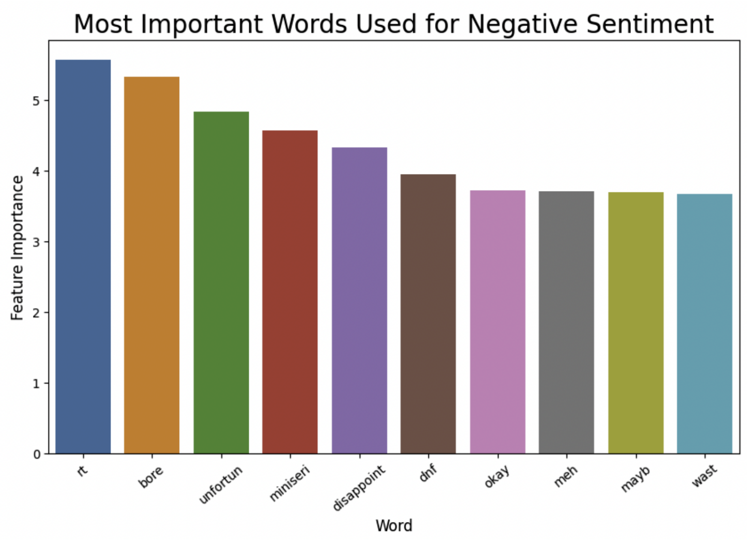
From the 50,000 sample, a ten-fold cross validation was performed between two vectors (TfIDF and Count Vectorizer). The above figures visualize the Logistic Regression confusion matrices between the two vectorizations.

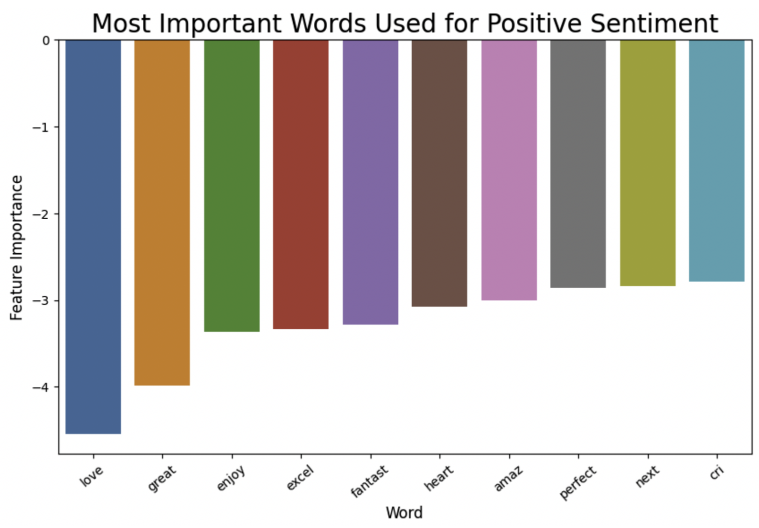
index	accuracy	precision	recall	f1_score
TfIDF vectorizer	0.8	0.8	0.78	0.79
Count Vectorizer	0.78	0.77	0.79	0.78

As observed from the confusion matrix above, it can be concluded that the TfIDF vector did perform slightly better over the bag of words vectorization, along with slightly more false positives and false negative percentages within the tfIDF vector.

Hyperparameter Tuning

Since the tfIDF vector had a better performance, the hyperparameter tuning was performed for two parameters specifically. One parameter being the C-value. Since the dataset is still fickle with the reviews, a list of low to high C-values were given (specifically, .01, .1, 1, 10). As expected, a lower C-value of 0.1 had resulted in higher accuracy scores, whereas as the C-values were getting bigger, the accuracy had dropped. As per the dataset, this does make sense as the dataset is not highly reliable even after all the cleaning as many "good" reviews contain either book titles or words in general that relate to "negative" sentiment.



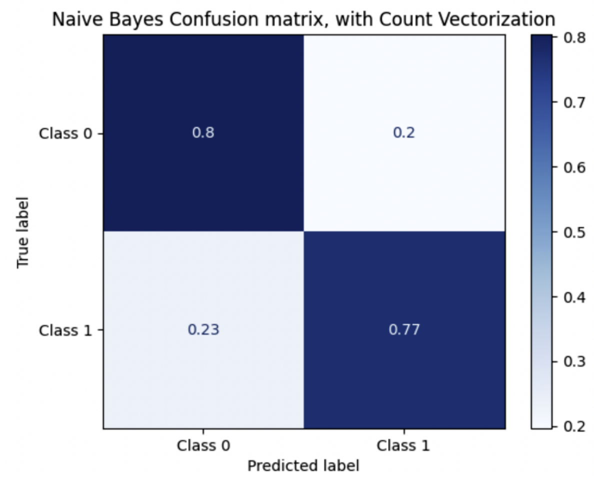


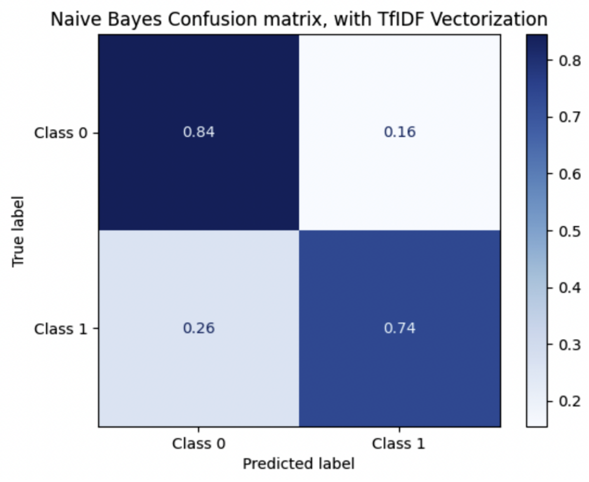
Another parameter that was planned to be tuned was the penalty (l1, l2). Since there was only one feature in the dataset (text reviews), there was not a much observed difference between the penalty hyperparameter. This does make sense since L1 combats overfitting by zeroing the weights of a few features, whereas L2 combats overfitting by sizing down the weights of a few features within the dataset. Since the text reviews was the only feature given into the model, it did not really make any difference by choosing a specific penalty score (L1 Accuracy: 80.2 %, L2 Accuracy: 80.5%).

With the hyperparameters now set for the data, a feature importance analysis was performed for the tfIDF vectorizer for both negative and positive sentiments.
It can be observed that words within the Negative and Positive Sentiment buckets makes sense, however since the words were stemmed, it can be observed that the words are not perfect english words. For example, words like "unfortun" (unfortunate), "wast" (waste), and "mayb" (maybe) contribute to top 10 words within the negative sentiment, however the words are incorrectly spelled due to the stemming. Similarly, the top ten words contributing to positive sentiment gives us more insight into the accuracy of the model, as we can easily relate words like "great", "love", and "enjoy" to positive emotions, but similarly words like "fantast" (fantastic) or "amaz" (amazing) are not perfect english words as these were derived from stemming.

Naive Bayes

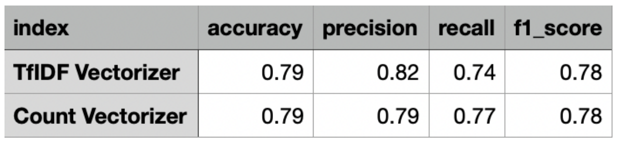
Similar to the Logistic Regression, the pre-processed data used for the Logistic Regression model was again re-purposed for the Naive Bayes, and a ten-fold cross validation was performed between two vectorizers (Count and TfIDF Vectorizer).





The above two confusion matrices visualize the accuracy between both the vectors and it could be observed that the TFiDF vector performed slightly better predicting the true positives, whereas the count vectorization had performed slightly better with the predictions of true negatives. Although we observed similar results between both the logistic regression and the naive bayes, we see slightly higher precision scores for the TfIDF vectorizer for the Naive Bayes Model. Alongside, we can observe that the Naive Bayes model with TfIDF vectorization does a slightly better job at classifying True Positive values.

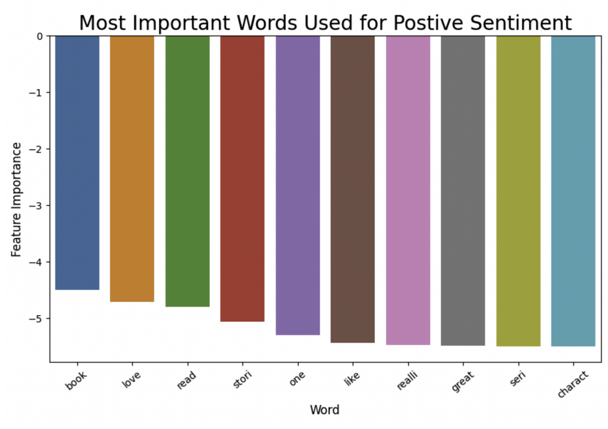
Hyperparameter tuning

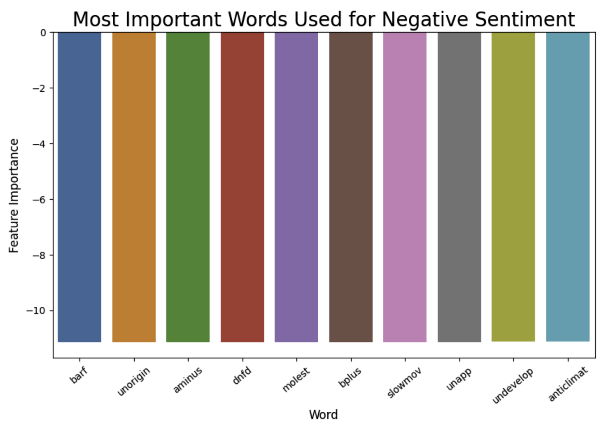


index	accuracy	precision	recall	f1_score
TfIDF Vectorizer	0.79	0.82	0.74	0.78
Count Vectorizer	0.79	0.79	0.77	0.78

The evaluation metrics were performed after the hyperparameter tuning of the alpha for the naive bayes models. However, even after the hyperparameter was done, there was not a significant difference between the various hyperparameters that were created, hence the default parameters were used. A feature importance analysis was then performed on the Naive Bayes model using Tfidf vectorization to check for important words within the negative and positive sentiments and see if they were classified correctly, similar to the logistic regression model.

As observed, some words categorized into the positive and negative sentiments do not really make sense such as “animus”, “unorigin”, “anticlimat” classified with the negative sentime list. This could be due to the fact that words were stemmed instead of being lemmatized, making incorrect predictions for a few test reviews, however the overall sentiment for the words classified in the positive and negative sentiment makes sense.

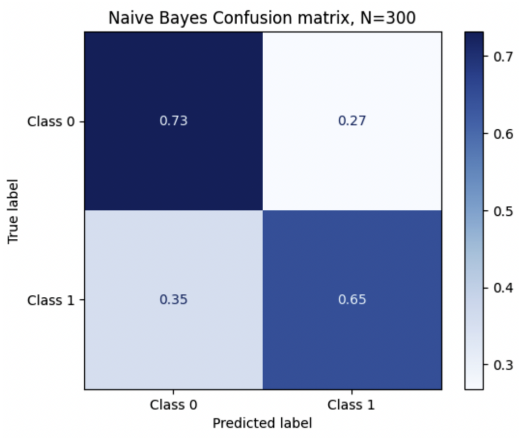


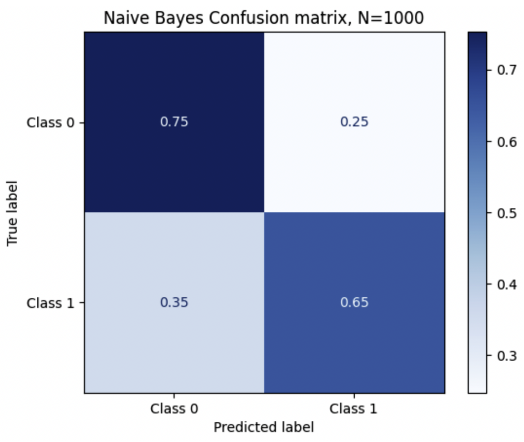


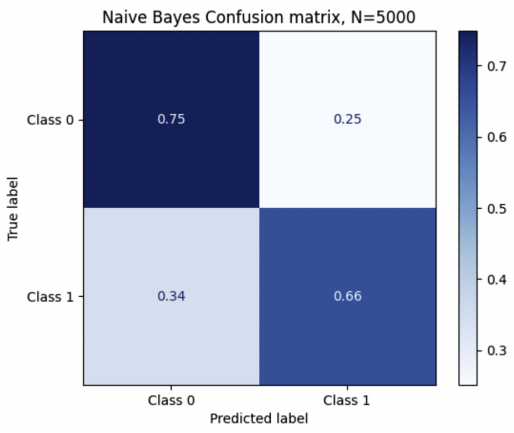
Failure Analysis

For the failure analysis, a randomized sample of 300, 1000, and 5000 reviews was taken from the 250,000 data reviews, and put through the Naive Bayes after TFIDF vectorization without stemming or stop words. We had then manually reviewed the reviews where the model had failed to predict correctly.

As observed, the Naive Bayes model does a better job at predicting true positive values over true negatives. Overall, the accuracy of the models is similar to the train/test run with a slightly lower accuracy within the randomized reviews, most likely due to the fact that stop words were not removed and stemming was not performed.







Here are 3 reviews that were predicted incorrectly:

Review	Actual Sentiment	Predicted Sentiment
Gotta say, this book was highly recommended to me by someone who loves Raymond Carver. Who I don't care for. So if you're a Carver fan, these finely tuned, beautifully written stories of quiet desperation and depression may move you. But it just wasn't my favorite.	Bad	Good
Perfect Chemistry didn't live up to my expectations. I read great things about the book, and how swoon-worthy was Alex that I couldn't help but expect a really good book. I was also inspired to read it since it was high in many of the lists GoodReads has about YA/NA and such... I don't know if any of you read Federico Moccia's book Three Meters Above Sky, but this sound A LOT like it. I couldn't help but feel like I was reading an American version of the same story. Granted, there are differences, especially in the ending, but there were many coincidences. However, I won't name them in case you do feel like reading "Tre metri sopra il cielo". Anyway, all in all, it wasn't a terrible read. It's just that I was expecting a bit more.	Bad	Good
This was slow going for me but it was my fault, not the book's. The tone was dreamy and weird and perfect for the plot. I really enjoyed the characters and I felt real emotions reading about them.	Good	Bad

We can notice in the first review, words such as “beautifully written”, “favorite”, “loves”, “recommended”, and “move you” might be related to predicted sentiment, however reading the review, we can see how the review consists of contractions that could have not been used in the training properly.

Similarly in the second review, we can words like “great”, “swoon-worthy”, or “good” can mislead the predictor to believe these are good reviews, however since there are not a lot of negative sentiment words within the review, the model had a hard time classifying it into the negative sentiment bucket.

The third review is a case of false negatives. However, we do see the reviewer mentioning how it is slow for him which was one of the most important words in the negative sentiment feature importance analysis.

Unsupervised Learning

Objective

With unsupervised learning, our objective was to use and compare different unsupervised learning models to understand what information we can find purely from the reviews without any ratings data. Our analysis focused mainly on generating topic models to see if the models are able to extract topics especially the ones we might know or those topics that more commonly appear in the text reviews. This might be useful to understand why certain types of books are liked or disliked by users (beyond a rating). In addition, we wanted to explore and see if the generated topic models can be used in any way to further improve performance of the supervised learning models. Since learning a new representation of the data can sometimes improve the accuracy of supervised algorithms, or lead to reduced memory and time consumption. For this analysis, we used Latent Dirichlet allocation (LDA), and K means clustering and the topic model outputs are manually reviewed to check if the generated topics make sense and whether words in a topic cluster correlate with each other.

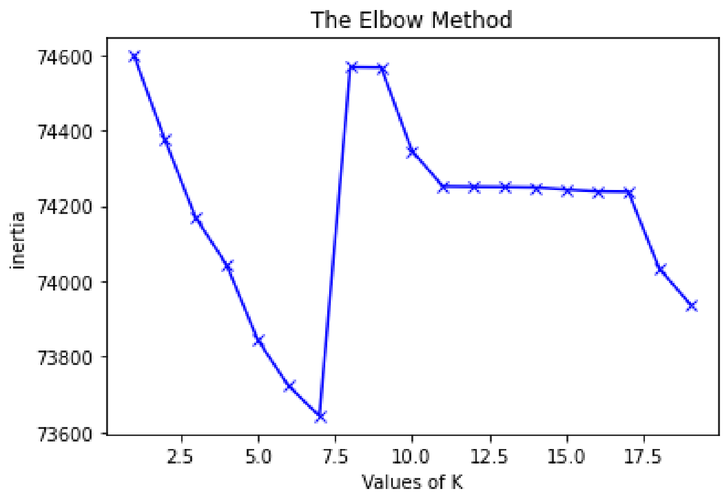
Data Source

The same good reads datasets were used after the preprocessing steps (discussed in section xx) to remove ratings “3” and restricting the reviews for the english books/reviews only. Additional preprocessing steps were implemented to remove noise from the dataset such as urls and extending the stop word list to include words that occurred more commonly than others. For the purpose of the unsupervised learning models, only the review_text column was used and rating and sentiment columns were used for visualization purposes.

Methods and Evaluation

K means clustering

First, we used the TF-IDF vector representation as input to the K-means clustering algorithm. Large text data are difficult to handle and analyzing such data for patterns requires converting text to numeric representation in the form of a vector using words as features. Here we choose TF-IDF for vector representation since TF-IDF weights the words based on its importance on both how often it is present in that document and also on how its overall presence is in the corpus.The first in this process was to decide on the optimum k that produces the best results. We used the elbow method for this purpose where we vary the number of clusters (k) from 1 – 20 to find the value of k with the lowest inertia value which is the sum of squared distances of samples to their closest cluster center. From the graph below, we see that as the number of clusters increases, the inertia value reduces to a point when it is almost parallel to the X-axis. The K value corresponding to this point is the optimal K value or an optimal number of clusters. Here in our case, we observed two optimal values, one at 10 and then another at 18.

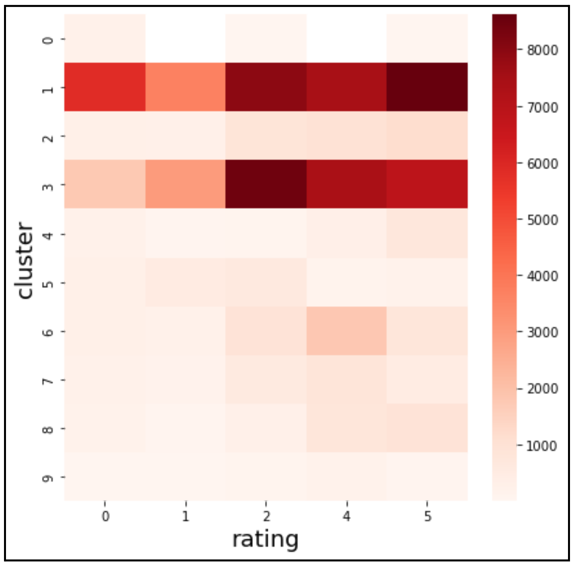


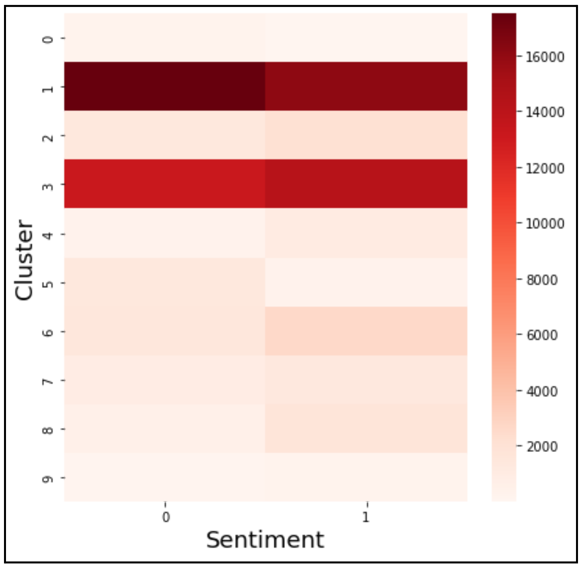
Using the value of k as 10 and other standard and default hyperparameters, we ran the K-means clustering model. The results were mixed in terms (as noted in the next section) where some topics showed more correlated words while others didn’t indicate much correlation. There are several reasons for this behavior. First, while the good reads dataset itself is well curated, the review text has a lot of noise in the form of several reviews in other languages even for English books. So in some cases there are large amounts of reviews that are in other languages that are not tagged with a language code and hence it was difficult to filter them out. We did try using “detect” to find the language to further clean the dataset but the algorithms that we tried took a long time and couldn’t perform this step within the time we had. The next interesting observation we made was that several ratings that had positive reviews were tagged as rating 0, 1 or 2. You see this in the next section when we correlate the topics with the ratings. Though the results did produce some well correlated results, cleaning the datasets further can yield much better results.

Latent Dirichlet allocation (LDA)
We trained LDA to use TF-IDF and bag of words inputs to observe differences in output. In the case of LDA, there is no need to identify a preferred optimal k. The output topic clusters seemed reasonable in terms of topics with similar words in each topic.

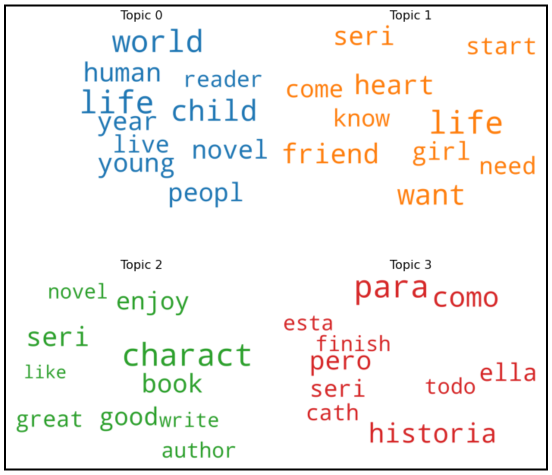
Unsupervised Learning Evaluation and Visualization

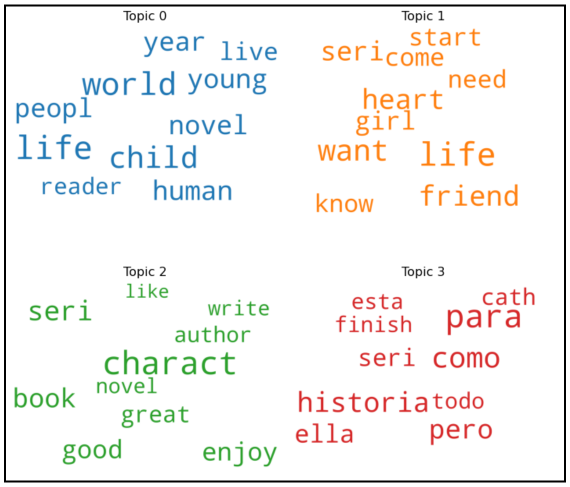
The topics that were generated by k-means were in general correlated well. We plotted two visualizations to see the correlation between the topics generated with the ratings and the sentiment (good or bad reviews). The visualizations show interesting correlations between ratings and topic 0 and 1. We observed strong correlations for topic 1 and all ratings. This is strange considering topic 0 and 1 has a number of positive words - 0 : {freebi, amazon, ebook, free, kindl, live, current, sourc, short, dnf}, 1 : {interest, favorit, amaz, fun, bore, dnf, beauti, write, better, best}. This is primarily due to improved tagging of good reviews with lower ratings.





The output from the LDA model was analyzed as a word cloud to show similar words in a topic. While LDA with both TF-IDF and bag of words generated relevant topics there weren't any stark differences in the topics and words within each topic to make any concrete conclusions on their performance. Further tuning of the model using hyperparameters and data cleaning might help understand the performance of these models better.





Discussion (Next Steps)

The following are some insights and learnings that we gathered from the analysis of the different supervised and unsupervised models from the perspective of improving on the existing analysis:

1. **Data cleaning** has a big impact on the results of the models. As we added more steps to clean the data, the performance did improve. As next steps, we would have further removed shorter reviews, reviews in non English languages, creating a corpus of "book" specific stop words.
2. **Supervised Learning:**
   a. It was surprising to see the Logistic Regression and Naive Bayes models performing at similar accuracy. We expected the Naive Bayes model to outperform the Logistic Regression model as the data has a high bias.
   b. Not being able to experiment further with the SVM model was surprising. This model did have a higher accuracy and is known to work great with semi-structured data.
   c. With more time, we would like to explore making SVM more time efficient and scalable by trying different methods such as the Bagging Classifier provided by sklearn.
   d. Additionally, we would like to take neutral reviews into consideration for better model accuracy and sentiment classification.

3. **Unsupervised Learning:** For the unsupervised learning, we used the default hyperparameters for running models. As next steps, it would be useful to explore additional hyperparameters to see if they have an impact on the topic models and clusters generated.
4. **Additional feature inputs:** Originally we had plans to use genre and author name as additional features but wrangling this huge dataset made it difficult to do it within the allotted time. We did clean and merge the data in the dataset used for the analysis but the next steps would be to use these features and test model performance.
5. **Computing environment and handling of large datasets:**

   a. It was particularly difficult to handle large datasets of size 5 GB. While Greatlakes environment helped to some extent, use of parallel computing might have been better in this case.
   b. We would like to experiment further with the SVM model, with a more scalable Cloud Resource like AWS or Azure, and ideally create a user interface that will work with real-time Goodreads Reviews to perform sentiment analysis.

6. **Ethical considerations:**
   a. We generally observed that models seemed to predict positive reviews better than negative reviews which could signal some type of imbalance in the stratified dataset. So having a clear methodology for balancing the review data can help eliminate any bias in the model output.
   b. Though this dataset is anonymized, it's still unclear whether the users and authors provided consent for this data to be used for this type of analysis. Especially if models were built based on these reviews that were to be productionized , it can have an impact potentially on book sales or author reputation.
   c. Related to the above point, is the issue of model interpretability. In some of the models (especially unsupervised) it is hard to explain how the outputs are generated since they seem to behave like a balck box. This can further exacerbate the issue described in point b.

Statement of Work

Cleaning - **All**
Preprocessing - **All**
EDA - **All**
Data Manipulation & Approach Supervised - **Asia and Srilekha**
Data Manipulation & Approach Unsupervised - **Malar**
Data Visualization - **All**
Final Project Report - **All**

References

Abid, A. (2021, December 31). Fixing Your Machine Learning Model's Failure Points. Medium. https://towardsdatascience.com/fixing-your-machine-learning-models-failure-points-e3ec0a047895

Bakharia, A. (2018, June 10). *Visualising Top Features in Linear SVM with Scikit Learn and Matplotlib*. Medium. https://aneesha.medium.com/visualising-top-features-in-linear-svm-with-scikit-learn-and-matplotlib-3454ab18a14d

Li, K. (2018, May 18). *Model Tuning and what is it ?(using Python)* - Kelvin Li. Medium. https://medium.com/@kelfun5354/model-tuning-and-what-is-it-using-python-630e388e224a

Li, S. (2018, June 20). *Topic Modeling and Latent Dirichlet Allocation (LDA) in Python*. Medium. https://towardsdatascience.com/topic-modeling-and-latent-dirichlet-allocation-in-python-9bf156893c24

Mashalkar, A. (2021, December 16). Sentiment Analysis using Logistic Regression and Naive Bayes. Medium. https://towardsdatascience.com/sentiment-analysis-using-logistic-regression-and-naive-bayes-16b806eb4c4b

Mengting Wan, Julian McAuley, "Item Recommendation on Monotonic Behavior Chains", in *RecSys'18*.

Mengting Wan, Rishabh Misra, Ndapa Nakashole, Julian McAuley, "Fine-Grained Spoiler Detection from Large-Scale Review Corpora", in *ACL'19*.

Müller, A. C. (n.d.). *Introduction to Machine Learning with Python*. O'Reilly Online Learning. Retrieved October 24, 2022, from https://www.oreilly.com/library/view/introduction-to-machine/9781449369880/ch03.html

Penn Treebank P.O.S. Tags. (n.d.). Retrieved October 24, 2022, from https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html

Prabhakaran, S. (2022, March 8). T*opic modeling visualization – How to present the results of LDA models?* Machine Learning Plus. https://www.machinelearningplus.com/nlp/topic-modeling-visualization-how-to-present-results-lda-models/

Sharma, A. (2022, July 19). *A Beginner's Guide to Exploratory Data Analysis (EDA) on Text Data (Amazon Case Study)*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2020/04/beginners-guide-exploratory-data-analysis-text-data/

SIADS 542, Supervised Learning. (n.d.). Coursera. Retrieved October 24, 2022, from https://www.coursera.org/learn/siads542/home/

SIADS 543, Unsupervised Learning.. (n.d.-b). Coursera. Retrieved October 24, 2022, from https://www.coursera.org/learn/siads543/home/week/1

UCSD Book Graph. (n.d.). Retrieved October 24, 2022, from https://sites.google.com/eng.ucsd.edu/ucsdbookgraph/home