ENSF 612 – Fall 2021 Final Report

Sentiment Analysis of IMDb Movie Reviews using NLP

GRAYDON WT HALL

University of Calgary, graydon.hall@ucalgary.ca, 30142310

ALEXANDER S LEAKOS

University of Calgary, asleakos@ucalgary.ca, 10083280

JARED P KRAUS

University of Calgary, jared.kraus1@ucalgary.ca, 10152270

The following table summarizes the specific contributions of each team member. It should be noted that all other tasks not mentioned in this table were shared collaboratively.

ABSTRACT

Context

User reviews are a key source of information for both customers and businesses. Due to the increasing volume of reviews, it is no longer practical to gain knowledge by manually reading these reviews, and thus it is essential to find ways to algorithmically perform tasks such as sentiment analysis.

Objective

The data analyzed in this report includes fifty thousand labelled IMDb movie reviews, plus one thousand manually scraped and labelled reviews procured by the authors of this report. The objective of this paper is to determine the optimal classifier for the sentiment of movie reviews, and to explore the root causes for misclassifications which occur.

Method

In this report, various natural language processing methods are explored to determine the sentiment of text. Two different datasets were used in this report. One data set with equal amounts of highly polarized positive and negative movie reviews and one dataset with randomly selected reviews. The textual contents of the reviews were preprocessed and tokenized. Various models were explored to vectorize the reviews including Bag of Words, Term Frequency Inverse Document Frequency (TF-IDF) and Word2Vec models. After, various classifiers were applied including Decision Tree, Random Forest and Naïve Bayes. The results were analyzed and the root causes for model misclassification were explored.

Results

For data labeling and collection, fifty thousand of the movie reviews were from an existing dataset of labelled movie reviews, while one thousand were manually labelled by the authors of this report. For the data pre-processing phase, the reviews were cleaned of noise, tokenized, and then turned into three types of feature vectors: Word2Vec, Bag of Words, and TF-IDF. Word2Vec produced the best results for the Random Forest and Decision Tree Classifier, while the Bag of Words vector produced the highest accuracy for the Naïve Bayes Classifier. After performing hyperparameter tuning and finding optimal parameters for all models, the Random Forest Classifier produced the highest accuracy on the one thousand new rows of data, while the Naïve Bayes Classifier produced the highest accuracy on both the original dataset and the combined dataset. The distribution of misclassifications was studied for two hundred reviews, and it was found that the two key causes were either weak sentiment or conflicting sentiment.

Conclusions

In this study, it was concluded that natural language processing methods can be used to effectively to determine the sentiment of textual reviews. In this report most optimal natural language process method was determined to be the use of the bag of words model followed by a Naïve Bayes model. The two main root causes for misclassification using this process were determined to be reviews with a conflicting sentiment and reviews with a vague sentiment.

1 INTRODUCTION

1.1 Problem Motivation

For any product or service, user reviews are a key business metric used to determine success and find out areas they must improve. User reviews are also key for consumers, allowing them to determine if a product or service is worth spending their money on. Examples of user reviews are everywhere in society. For any Amazon product, a shopper can read reviews by previous purchasers and quickly get an honest opinion of whether the product will meet their needs. For that same Amazon product, the company selling the item can analyze these reviews, and determine key issues to fix, or possibly areas which they have executed well. The company selling the product may also choose to analyze the reviews of a competitor, to asses where they stand relative to them. The sheer volume of reviews, however, makes classifying each and every review impossible. In the event of a product only having 5-10 reviews, reading them manually is no issue. However, in the digital world of today, there are far too many reviews to be read manually, and algorithmic measures of classifying these reviews is essential. As more people become computer literate, more reviews are placed online instead of simply by word of mouth, making the issue of algorithmic classification even more relevant. The following figure demonstrates the growth in online reviews, which shows the cumulative number of reviews submitted to Yelp from 2009 to 2020 (in millions)

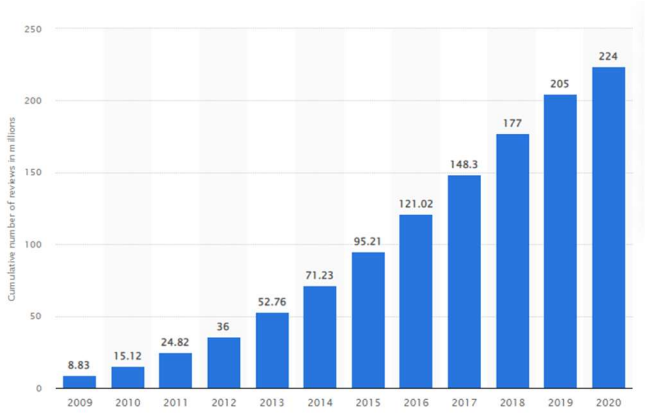


Figure 1 Cumulative number of reviews submitted to Yelp from 2009 to 2020 (in millions) [1]

1.2 Background

In the case of this report, movie reviews are used in place of product reviews, though the same issues discussed above apply. The dataset utilized in the report is the same as the dataset used by Hadi Pouransari and Saman Ghili in their paper "Deep learning for sentiment analysis of movie reviews" [2], published in 2015. The dataset consists of fifty thousand movie reviews, all labelled as either positive or negative. This dataset was also made famous by the Kaggle competition "IMDB Dataset of 50K Movie Reviews" [3]. For this report, one thousand rows of data were pulled from IMDb, classified by the authors of this report, and added to the dataset. Different classification models, as well as feature extraction techniques were utilized in order to try and achieve the optimal results.

1.3 Technical Approach

In this report three different models were trained and tested on movie reviews to determine the reviews sentiment. The first step taken in this report was data collection and data labeling. Fifty thousand labeled movie reviews were obtained from the Kaggle dataset "IMDB Dataset of 50k Movie Reviews" [3]. Further, an additional one thousand reviews were web-scraped from IMDb, labeled, and added to the dataset. The original data was filtered to have an equal number of highly polarized positive and negative reviews. The new data consisted of random reviews pulled from various movies in the last two years. The second step taken was to preprocess the combined dataset. Preprocessing consisted of cleaning the text with BeautifulSoup, making all words lowercase, striping punctuation and whitespace, stemming words with SnowballStemmer, tokenizing words, removing stop words, removing words less than three character and removing non alphabetical characters. The third step was to determine the best feature that complimented each model. The features were Bag of Words, TF-IDF, and Word2Vec. The fourth step was to hyperparameter tune

each model. The performance of the model's increased by an average of 5.5% after tuning them. The fifth step was to compare each models test results of the Kaggle dataset to the new web-scraped dataset. The model's performance on the Kaggle dataset was drastically better than the web-scraped data. The final step was to analyze the misclassifications of our best performing model. The model predicted a fair amount more false negatives than false positives. The main reasons for misclassification were determined to be reviews with conflicting sentiment and reviews with a weak sentiment.

2 RESULTS

The results section for this report will first explore the steps taken for labelling and collecting the new data that was obtained for this report. Next, the original and new data added will be introduced, explored, and contrasted. After this, the data preprocessing steps used to clean the data and then create feature vectors will be discussed. In the following section, the model's performance on the new, original, and new + original data will be summarized. To finish off this results section, the hyperparameter tuning done on each model will be explained, and the distribution of the misclassifications of the best performing model will be investigated.

2.1 Data Labelling and Collection

2.1.1 Approach

An additional one thousand unlabeled movie reviews were collected to be added to the existing fifty-thousand labeled reviews obtained from the Kaggle dataset "IMDB Dataset of 50k Movie Reviews" [3]. These additional reviews were collected by web-scraping the textual contents of movie reviews on IMDb. The python library IMDbPY [4] was used to web-scrape thirty reviews per movie from various movies released in the last two years. After collection, the sentiment of each new review was labeled by each author of this report. An author labeled each review either a "1" which represented a positive sentiment or "-1" which represented a negative sentiment. This resulted in three labels for each of the one thousand new reviews. Each review either had a unanimous decision or a split decision.

2.1.2Results

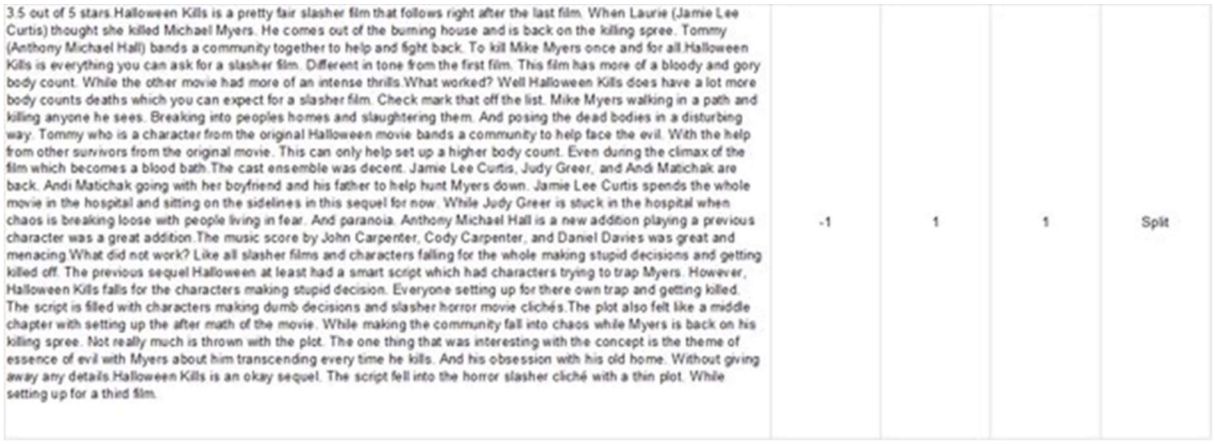
After labeling one thousand reviews, 55 had conflicting sentiment labels. There were three main categories for these conflicts.

1. *The author had a vague sentiment*: The review summarized the movie's plot and stakeholders and did not provide a sentiment. An example of this is provided in the following figure.

When it comes to films directed by Guy Ritchie (Lock, Stock and Two Smoking Barrels, Snatch), they can be hit and misses, with big misses including Swept Away, Revolver, Sherlock Holmes, and King Arthur, but the trailer for this film looked worth a try, even just for the star of Four Wedding and Love Actually playing completely against his usual type. Basically, American businessman Mickey Pearson (Matthew McConaughey) has formed a highly profitable marijuana empire in London and across England. The drugs kingpin is looking to cash out his business, and when the words gets out, several schemes, bribes and blackmails are triggered in an attempt to steal his domain. It is reported that Pearson has been shot dead, and private eye and aspiring screenwriter Fletcher (Hugh Grant) is employed by a newspaper to investigate. He visits Pearson's right-hand man Ray (Nicholas Nickleby's Charlie Hunnam) and threatens to expose the empire, he tells a full-length detailed backstory of the origins of the business, up until the present. This includes the many hidden marijuana plantations across the countryside, and Pearson's wife Rosalind (Downton Abbey's Michelle Dockery) being part of key decisions. Fletcher also details the many characters behind the scenes, including Coach (Colin Farrell), or trying to take it away, including Dry Eye (Henry Golding). The British crime underworld is close to exposure, but all is not as it seems, and Pearson may not actually be dead. Also starring Jeremy Strong as Cannabis King Matthew, Eddie Marsan as Mike, Jason Wong and Phuc and Jordan Long as Barman. McConaughey gives a good performance as the yank head honcho, there is good support from Hunnam, Dockery and Golding, but this film is pretty much taken by Grant, who with his cockney accent, goatee beard and sunglasses outshines all. It is nice of Ritchie to return his geezers gangsters and guns origins, but let's be honest, it is a bit of a rehash of what we've seen before, the dialogue is witty, there are amusing moments, and the editing is clever, but overall it just feels meh, an average action crime drama. Okay!	-1	-1	1	Split

Figure 2 Example of Vague Author Sentiment

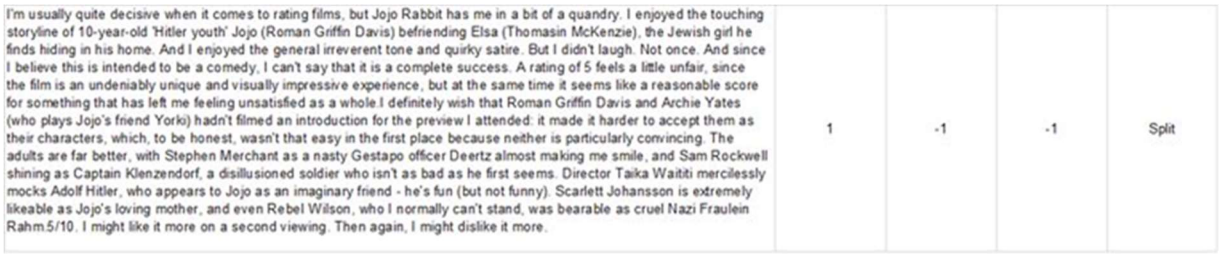
2. *The author had conflicting sentiment*: The review equally listed both the positives and negatives of a movie. An example of this is provided in the following figure.



3.5 out of 5 stars.Halloween Kills is a pretty fair slasher film that follows right after the last film. When Laurie (Jamie Lee Curtis) thought she killed Michael Myers. He comes out of the burning house and is back on the killing spree. Tommy (Anthony Michael Hall) bands a community together to help and fight back. To kill Mike Myers once and for all.Halloween Kills is everything you can ask for a slasher film. Different in tone from the first film. This film has more of a bloody and gory body count. While the other movie had more of an intense thrills.What worked? Well Halloween Kills does have a lot more body counts deaths which you can expect for a slasher film. Check mark that off the list. Mike Myers walking in a path and killing anyone he sees. Breaking into peoples homes and slaughtering them. And posing the dead bodies in a disturbing way. Tommy who is a character from the original Halloween movie bands a community to help face the evil. With the help from other survivors from the original movie. This can only help set up a higher body count. Even during the climax of the film which becomes a blood bath.The cast ensemble was decent. Jamie Lee Curtis, Judy Greer, and Andi Matichak are back. Andi Matichak going with her boyfriend and his father to help hunt Myers down. Jamie Lee Curtis spends the whole movie in the hospital and sitting on the sidelines in this sequel for now. While Judy Greer is stuck in the hospital when chaos is breaking loose with people living in fear. And paranoia. Anthony Michael Hall is a new addition playing a previous character was a great addition.The music score by John Carpenter, Cody Carpenter, and Daniel Davies was great and menacing.What did not work? Like all slasher films and characters falling for the whole making stupid decisions and getting killed off. The previous sequel Halloween at least had a smart script which had characters trying to trap Myers. However, Halloween Kills falls for the characters making stupid decision. Everyone setting up for there own trap and getting killed. The script is filled with characters making dumb decisions and slasher horror movie clichés.The plot also felt like a middle chapter with setting up the after math of the movie. While making the community fall into chaos while Myers is back on his killing spree. Not really much is thrown with the plot. The one thing that was interesting with the concept is the theme of essence of evil with Myers about him transcending every time he kills. And his obsession with his old home. Without giving away any details.Halloween Kills is an okay sequel. The script fell into the horror slasher cliché with a thin plot. While setting up for a third film.	-1	1	1	Split

Figure 3 Example of Conflicting Author Sentiment

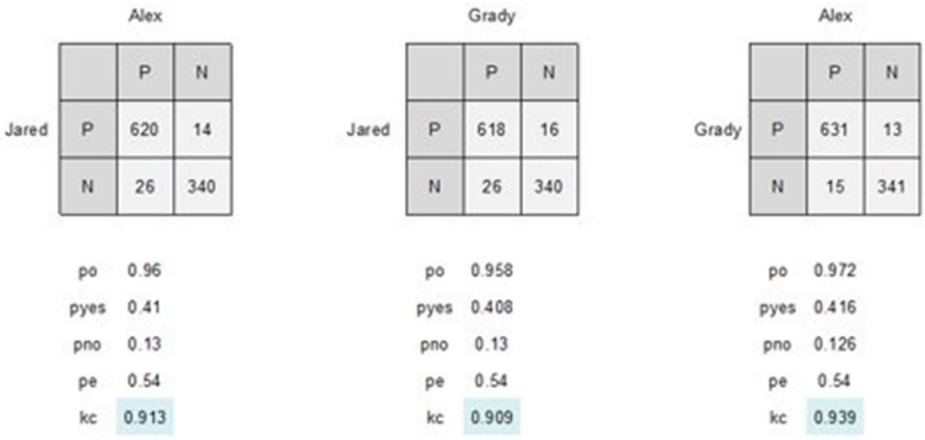
3. *The author was indecisive regarding his sentiment:* The review had a very neutral or indecisive sentiment. An example of this is provided in Figure the following figure.



I'm usually quite decisive when it comes to rating films, but Jojo Rabbit has me in a bit of a quandry. I enjoyed the touching storyline of 10-year-old 'Hitler youth' Jojo (Roman Griffin Davis) befriending Elsa (Thomasin McKenzie), the Jewish girl he finds hiding in his home. And I enjoyed the general irreverent tone and quirky satire. But I didn't laugh. Not once. And since I believe this is intended to be a comedy, I can't say that it is a complete success. A rating of 5 feels a little unfair, since the film is an undeniably unique and visually impressive experience, but at the same time it seems like a reasonable score for something that has left me feeling unsatisfied as a whole.I definitely wish that Roman Griffin Davis and Archie Yates (who plays Jojo's friend Yorki) hadn't filmed an introduction for the preview I attended: it made it harder to accept them as their characters, which, to be honest, wasn't that easy in the first place because neither is particularly convincing. The adults are far better, with Stephen Merchant as a nasty Gestapo officer Deertz almost making me smile, and Sam Rockwell shining as Captain Klenzendorf, a disillusioned soldier who isn't as bad as he first seems. Director Taika Waititi mercilessly mocks Adolf Hitler, who appears to Jojo as an imaginary friend - he's fun (but not funny). Scarlett Johansson is extremely likeable as Jojo's loving mother, and even Rebel Wilson, who I normally can't stand, was bearable as cruel Nazi Fraulein Rahm.5/10. I might like it more on a second viewing. Then again, I might dislike it more.	1	-1	-1	Split

Figure 4 Example of Indecisive Author Sentiment

A Cohen Kappa analysis was conducted on the split reviews. The results are summarized in the following figure:



Alex

Jared	P	N
P	620	14
N	26	340

po	0.96
pyes	0.41
pno	0.13
pe	0.54
kc	0.913

Grady

Jared	P	N
P	618	16
N	26	340

po	0.958
pyes	0.408
pno	0.13
pe	0.54
kc	0.909

Alex

Grady	P	N
P	631	13
N	15	341

po	0.972
pyes	0.416
pno	0.126
pe	0.54
kc	0.939

Figure 5 Cohen Kappa Analysis Results for Split Reviews

To resolve reviews with split labelling, the label with the majority of the three was selected as the final label for that review.

2.2 Data Comparison

2.2.1 Approach

In this paper, before completing analysis on the original and new dataset the contents of each of these datasets was examined. The original dataset was binary, consisting of only positive and negative sentiment review labels. The new dataset was labelled in a similar manner, with each new review given either a label of positive or negative. Once labelling was completed, to check for similarity of the original vs. new dataset, the split between positive and negative reviews was calculated, and the results of the original, new, and combined datasets was analyzed. A number of reviews were also randomly selected from both the original dataset and the new dataset and the contents of each were analyzed to determine if there was a noticeable difference in content.

2.2.2 Results

The original data set was comprised of fifty thousand IMDb movie reviews classified in a binary manner, with all reviews being labelled as either positive or negative. The reviews were also specifically selected to be used in sentiment analysis training, with reviews chosen being highly polarized with a positive label being given to movies with a score of >= 7 and a negative label given to movies with a score of <= 4. A maximum of thirty reviews per movie was included, and the dataset was constructed to be balanced, with twenty-five thousand being labelled as positive and twenty-five thousand being labelled as negative so randomly guessing would yield 50% accuracy [5] [2]/

Table 1 Summary of Original Dataset

Sentiment	Review Count	% of Total
Positive	25,000	50.0
Negative	25,000	50.0
Total	50,000	100.0

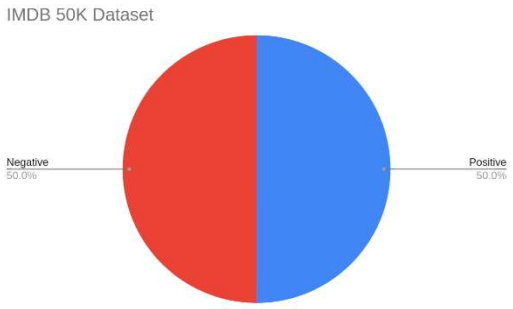


Figure 6 Summary of Sentiment Split for Original Dataset

The new dataset was constructed from one thousand randomly selected reviews across forty different movies released from 2019 – 2021 so as not to overlap with the original dataset. To remain consistent with the original dataset, no more than thirty reviews per movie were analyzed. Each review was labelled according to the methodology included in section 2.1.

Table 2 Summary of New Dataset

Sentiment	Review Count	% of Total
Positive	639	63.9
Negative	361	36.1
Total	1,000	100.0

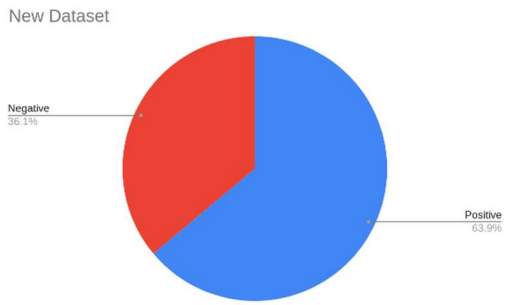


Figure 7 Summary of Sentiment Split for New Dataset

Finally, the new and original datasets were combined, resulting in a dataset which still remained quite balanced due to the volume of reviews in the original dataset.

Table 3 Summary of Combined Dataset

Sentiment	Review Count	% of Total
Positive	25,639	50.3
Negative	25,361	49.7
Total	51,000	100.0

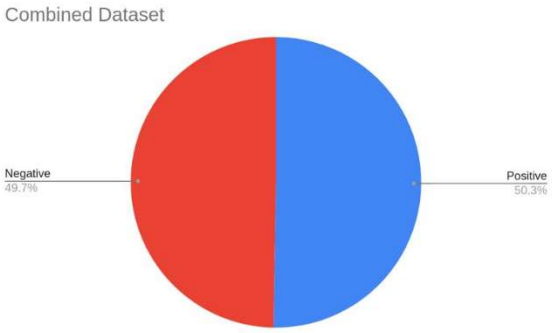


Figure 8 Summary of Sentiment Split for Combined Dataset

2.2.2.1 Differences Between Datasets

While the original dataset consisted of specifically chosen reviews to generate a balanced dataset, the new data was not hand selected, and will tend to reflect the "real-world" sentiment of how movie reviews are distributed. According to an analysis published by Nathaniel Johnston, an average rating on IMDb is 6.38 [6]. While this does not indicate that the average movie on IMDb is considered "good", it does indicate that reviews tend to skew closer to the "good" threshold of a rating of >=7 as opposed to a "bad" rating of <=5. As such, it is not surprising that in the new dataset, reviews tended to be weighted towards a more positive sentiment, with 63.9% classified as positive and 36.1% classified as negative.

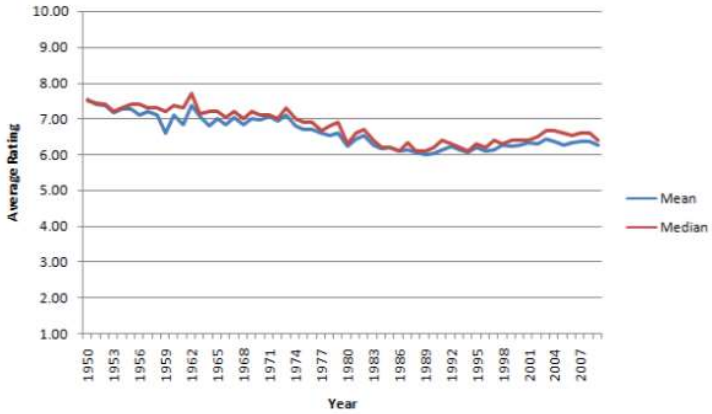


Figure 9 Average IMDb Rating by Year [6]

Example of a highly polarized negative review from the original dataset:

"This was the worst movie I saw at WorldFest and it also received the least amount of applause afterwards! I can only think it is receiving such recognition based on the amount of known actors in the film. It's great to see J.Beals but she's only in the movie for a few minutes. M.Parker is a much better actress than the part allowed for. The rest of the acting is hard to judge because the movie is so ridiculous and predictable. The main character is totally unsympathetic and therefore a bore to watch. There is no real emotional depth to the story. A movie revolving about an actor who can't get work doesn't feel very original to me. Nor does the development of the cop. It feels like one of many straight-to-video movies I saw back in the 90s ... And not even a good one in those standards."

Example of a negative review from the new dataset:

"Outside of the main plot, "The Addams Family 2" is pretty much what I expected. No masterpiece, but nothing terrible. Worth mentioning that there are some jokes that only adults will understand."

After analyzing the contents of reviews from the original dataset and the new dataset, within the original dataset many reviews have very strong sentiment identifying words such as "worst", "ridiculous", and "predictable" were easily classified as negative. Whereas in the new dataset reviews without any strong identifying words, and in some cases conflicting sentiment were also given a negative sentiment.

2.3 Data Preprocessing

2.3.1 Approach

The first step of pre-processing involved cleaning the data, so it was in a usable form. A generic data pre-processing function was created, which took the raw reviews and labels as an input, and produced a vector of cleaned, tokenized words as an output, along with numerically encoded labels. First, the BeatifulSoup library was utilized to remove any possible URLs, or preformatted text. Secondly, all words were made lowercase, stripped of trailing/leading whitespace, and any punctuation was removed via a regular expression. Following this, the words were stemmed using SnowballStemmer. Stemming is a procedure which reduces text dimensionality, allowing the words to be standardized. For example, the words program, programs, programmer, programming, and programmers could all be reduced to the standard stem of program. After stemming the words, each review was tokenized. Tokenization is the process of taking a string, and converting it into an array of individual words. After tokenization was completed, words of less than three characters were removed along with non-alphabetical words (e.g. numbers such as 1, 2, 3). The target

vector containing the labels of either 'positive' or 'negative' sentiment for each review was numerically encoded into the values 1 and 0.

The second step of pre-processing involved taking these cleaned, tokenized reviews, and converting them into numerical feature vectors which could be understood by the classification models. For this study, three types of feature vectors were produced: a simple bag of words vector, a TF-IDF vector, and a Word2Vec vector. The bag of words vector was created using PySpark's CountVectorizer function. For this bag of words model, a vector is produced where each index corresponds to a count for a given word. TF-IDF stands for Term Frequency – Inverse Document Frequency; a TF-IDF vector takes the bag of words vector one step further, and replaces the old index (which is simply a count of a given word in the corresponding vector) with a numerical value that reflects the word's importance in that review. For the Word2Vec vector, PySpark's Word2Vec library was used, with parameters of VectorSize equal to 100, and WindowSize equal to 5. Each word in the tokenized review was first assigned to a vector of size of 100. After this, the review vector was formed by averaging the vectors of every word in the review. [7]

2.3.2 Results

The following table summarizes how well each model performed using one thousand rows from each feature vector. It should be noted that the number of rows utilized in this step were limited to save computational resources. Also, Naïve Bayes was not able to be tested using the Word2Vec Feature vector, as the feature vector and model were incompatible in PySpark.

Table 4 Summary of Model Performance Utilizing Different Feature Vectors

Feature Vector	Naïve Bayes Accuracy (%)	Random Forest Accuracy (%)	Decision Tree Accuracy (%)
Count Vectorizer	78	60	68
TF-IDF	68	60	68
Word2Vec	n/a	76	71

Based on these results, the Decision Tree and Random Forest models had their hyperparameters tuned using the Word2Vec feature vector, while the Naïve Bayes model used the Count Vectorizer bag of words feature vector.

2.4 Model Performance

2.4.1 Approach

In order to assess the effectiveness of each model, Accuracy, Precision, Recall, and F1-Score will be calculated to get a comprehensive picture (bad word) of the selected model.

Accuracy is the simplest method for assessing model performance and is calculated by taking results of predicted / classified. While this method will give a depiction of how often a model will correctly classify inputted data, it fails to take into account imbalance in datasets, having a higher score for datasets which are imbalanced vs. balanced. The original dataset is balanced, so a high accuracy score on the original dataset should indicate strong model performance, however, for the new data, the data skews more positive, and relying on accuracy to assess model performance alone may fall short.

Precision is calculated by taking the True Positive / (True Positive + False Positive) and will give the proportion of movies classified as positive when this result was actually correct. Precision is an important metric especially when the outcome of misidentifying a positive is high. In the case of movie sentiment analysis, a higher weighting to Precision could be given to advertisers with a limited budget who wanted to limit their advertising and retargeting spend to only those users they can be confident enjoyed the movie.

Recall is calculated by taking the True Positive / (True Positive + False Negative) and identifies which proportion of positive movie sentiments were identified correctly. A high weighting to Recall would be given when trying to minimize the number of False Negatives. A higher weighting towards recall could be used by companies looking for negative reviews of their products, where the company wants to identify and respond to negative reviews online as efficiently as possible.

Finally, F1 Score is a combination of Precision and Recall and can be used when a balance between Precision and Recall is needed. F1 Score is calculated by taking the 2 * (Precision * Recall) / (Precision + Recall) - which can be an important measure

when both correctly identifying True Positives and minimizing False Negatives is important, and can be more useful than accuracy, especially on imbalanced datasets.

As the original dataset as well as the combined dataset are both balanced, there should not be many discrepancies between the scores for these datasets. However, as the new dataset is weighted towards positive reviews there will be some more noticeable differences in scores when evaluating the model performance on the new dataset alone. The metric best suited to predict the efficacy of these models is dependent on the intended use case of the model, and appropriate weighting should be given to each parameter when determining the best model.

2.4.2 Results

2.4.2.1 Naïve Bayes Model

The following table and summarizes the metrics for the Naïve Bayes model

Table 5 Summary of Results for Naive Bayes Model

Metric	Report Data (%)	New Data (%)	Combined (%)
Accuracy	85.3	51.9	85.0
Precision	85.3	74.0	85.0
Recall	85.3	65.2	85.0
F1-score	85.3	51.9	85.0

The model performed excellent on the Report data, however struggled significantly on the new data, leaning heavily to predicting false negatives.

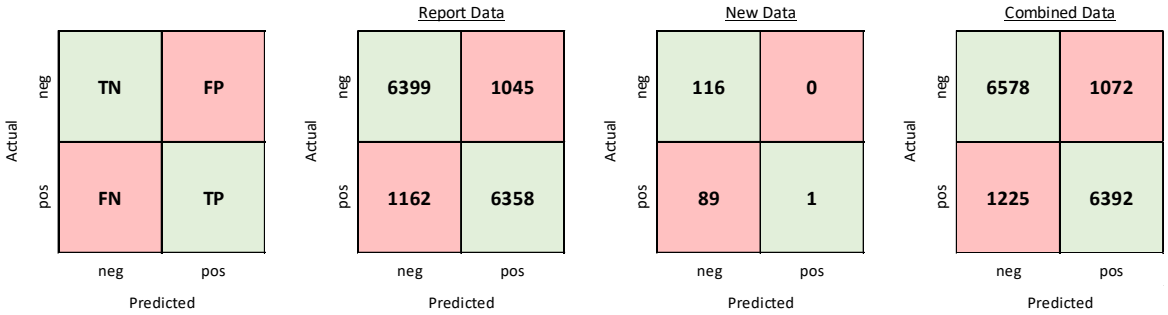


Figure 10 Confusion Matrices for Naive Bayes Model

As seen in the above confusion matrices, for the combined dataset, the model predicted more false negatives than positives.

2.4.2.2 Random Forest Model

The following table summarizes the metrics for the Random Forest model

Table 6 Random Forest table

Metric	Report Data (%)	New Data (%)	Combined (%)
Accuracy	83.0	61.0	84.0
Precision	83.0	62.0	84.0
Recall	83.0	65.0	84.0
F1-score	83.0	61.0	84.0

The Random Forest Classifier model performed very well on the report data at 83% but its accuracy was drastically worse on the new data at 61%. Precision, Recall and F1-score were fairly consistent among datasets. The following figure gives the confusion matrices for the Random Forest model.

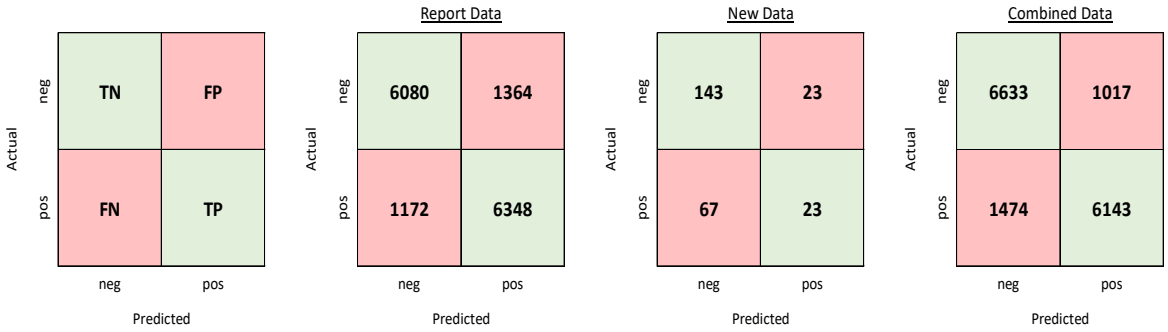


Figure 11 Random Forest confusion matrix

The combined dataset had significantly more false negatives than false positives. In other words, the model incorrectly predicted a review to have a negative sentiment many more times than a positive sentiment.

2.4.2.3 Decision Tree model

The following table summarizes the metrics for the Decision Tree model

Table 7 Decision Tree Table

Metric	Report Data (%)	New Data (%)	Combined (%)
Accuracy	73.9	54.4	74.4
Precision	73.9	55.9	74.8
Recall	73.9	53.5	74.4
F1-score	73.9	54.4	74.4

The Decision Tree Classifier performed relatively poorly compared to the other models analyzed, almost 10% worse accuracy than the ensemble method of Random Forest Classifier. Although its results were poor in comparison, it was quick to train and as a result could be tuned quickly with different parameters. However, even with extensive tuning the model could not compete with Naïve Bayes and Random Forest on any of the datasets.

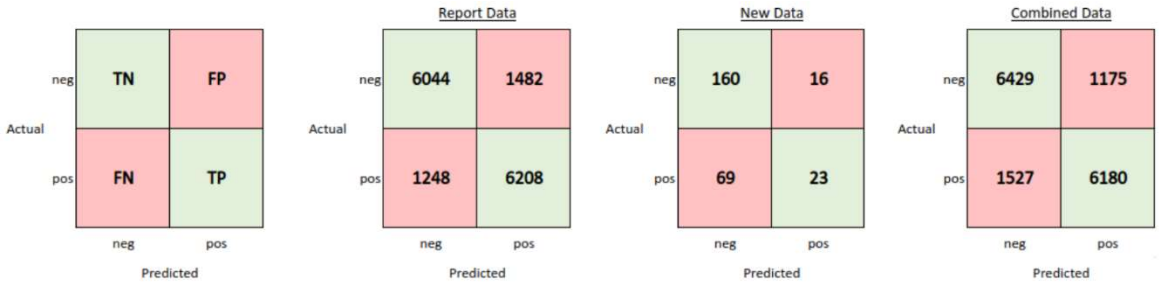


Figure 12 Decision Tree Confusion Matrix

2.5 Hyperparameter Tuning

2.5.1 Approach

2.5.1.1 Naïve Bayes

For the Naïve Bayes model, the hyperparameters which were tuned include the Laplace smoothing parameter, as well as the model type parameter. The Laplace smoothing parameter has a default value of 1, and was tested with values of 5, 7, 9, 12, 15, 17, and 19. The purpose of the Laplace smoothing parameter is to solve the "zero-probability problem", which arises for words which show up in the testing data that are not present in the training data. [8] For tuning the model type parameter, the tested values include 'Multinomial', 'Gaussian', and 'Complement'. The value of multinomial, corresponds to using the Naïve Bayes algorithm with a multinomial distribution, whereas using a model type value of 'Gaussian' refers to using a Gaussian distribution. A model

type parameter of "Complement" corresponds to an adaptation of a simple Multinomial Naïve Bayes model, which uses statistics from the complement of each class to compute the model's coefficients. [9]

2.5.1.2 Random Forest

After experimenting with various parameters in the Random Forest Classifier it was determined that "numTrees" and "maxDepth" had the largest impact on the model's performance. Therefore, these were the hyperparameters chosen to be tuned. The parameter "numTrees" represents the number of individual decision trees that will be created to operate as an ensemble. The parameter "maxDepth" is the maximum node depth of each decision tree. The library ParamGridBuilder was used with CrossValidator to determine the optimal parameters to be used to create the best possible model. For the "numTrees" parameter, a list of 200, 300, 400, and 500 was inputted to the grid. For the "maxDepth" parameter a list of 5, 10, 15 and 20 was inputted into the grid.

2.5.1.3 Decision Tree

The Decision Tree was able to be quickly iterated to test out which parameters increased model performance by the greatest amount. While training the model the parameters that affected the model the most effect on the results were "maxDepth" and impurity. "maxDepth" represents the maximum depth of each decision tree, where deeper trees are allowed to split more. It is important to find the right "maxDepth" parameter as too many decisions can cause a model to be overfit, whereas too few can result in poor accuracy. For the Decision Tree Classifier in this report a "maxDepth" of 5, 10, 20, and 30 were used.

The impurity measure was also a focus of hyperparameter tuning with both gini impurity and entropy tested. Impurity measure determines how nodes "spit" optimally at each level. Decision Tree Classifier offers two different types of impurity measures, gini, and entropy. Gini impurity measures how frequently any element of the dataset will be mislabeled when it is randomly labeled and is defined below, where *p(j)* is the probability of picking a datapoint with class j.

Equation 1 Gini Impurity Equation

$$Gini: Gini(E) = 1 - \sum_{j=1}^{c} p_j^2$$

Similar to entropy in thermodynamics, entropy in classification indicates the disorder of the feature with the target. Entropy is more complicated than gini impurity when training models due to logarithmic calculations, and may not be deemed worthwhile to train on more complex ensemble models, when tested alone the accruacy of the model when using entropy improved ~0.3%.

Equation 2 Entropy Equation

$$Entropy: H(E) = -\sum_{j=1}^{c} p_j \log p_j$$

2.5.2 Results

2.5.2.1 Naïve Bayes

Based on the hyperparameter tuning process discussed in the approach section, the optimal parameters chosen for the Naïve Bayes model were model type equal to "Complement" and smoothing equal to 12. The results of the model before and after tuning are summarized below. Only one thousand rows were used in each case due to computational limitations.

Table 8 Summary of Model Results Before and After Tuning for Naïve Bayes Model

Metric	Un-tuned Naïve Bayes Model	Tuned Naïve Bayes Model;
Accuracy (%)	78.0	85.0
Smoothing Parameter	1	12
Model Type	Multinomial	Complement

2.5.2.2 Random Forest

Based on the hyperparameter tuning process discussed in the approach section, the optimal parameters chosen for the Random Forest Classifier model were "numTrees" equal to 400 and "maxDepth" equal to 10. Applying these parameters improved the model's accuracy compared to a base model by 8% as show in the table below:

Table 9 Summary of Model Results Before and After Tuning for Random Forest Model

Metric	Un-tuned Random Forest Model	Tuned Random Forest Model;
Accuracy (%)	76.0	84.0
Number of Trees	10	400
Max Depth	5	10

2.5.2.3 Decision Tree

Hyperparameter tuning by choosing the optimal parameters of 20 for "maxDepth" and entropy for impurity improved the performance of the model by 4% over the base case.

Table 10 Summary of Model Results Before and After Tuning for Decision Tree Model

Metric	Un-tuned Decision Tree Model	Tuned Decision Tree Model;
Accuracy (%)	71.0	74.4
Impurity	Gini	Entropy
Max Depth	5	20

2.6 Distribution of Misclassification for Best Performing Model

2.6.1 Approach

In order to assess the root causes for misclassification within the model, two hundred misclassified records selected from the best performing model (the tuned Naïve Bayes model using Count Vectorizer features). The two main reasons identified for misclassification were "weak sentiment" and "conflicting sentiment". All two hundred rows were given one of these two labels. An example of a review with weak sentiment is given in the following figure:

Figure 13 Example of cleaned and tokenized review with weak sentiment

In this prior figure, the review is considered to have weak sentiment since it lacks any words that give a hint about the user's sentiment. These reviews generally stem from reviews where the reviewer simply provides a synopsis of the film, as opposed to

offering an opinion. An example of a review with conflicting sentiment is given in the following figure, where words associated with a positive sentiment are highlighted in green, with words corresponding to a negative sentiment are highlighted in red.

[giant,monster,fan,see,yeti,absolut,must,especi,hear,much,thank,good,bootleg,
market,abl,find,copi,pretti,easili,happili,surpris,upon,watch,flick,actual,dare,say,de
centdec,actual,name,cheesi,giantmonst,flick,kick,pretti,quick,yeti,found,pretti,
much,immedi,get,introduc,various,charact,consist,sleazi,one,good,one,girl,pretti,
much,one,downright,strike,beauti,girl,cheesi,scifi,film,faryeti,look,like,longhair,guy
,straight,origin,woodstock,concert,realli,hes,bad,dude,especi,introduc,world,kind,
funki,cagelik,thing,godzilla,despit,rude,awaken,doesnt,even,rampag,actual,rare,
destroy,anyth,whole,pictur,kinda,look,puzzl,tri,figur,thing,yeti,seem,understand,
english,pretti,nice,copi,dub,english,know,good,guy,bad,guy,arehowev,want,see,
giant,yeti,thing,hes,pretti,much,whole,movi,typic,lowbudget,fashion,seem,chang,
size,lot,depend,scene,there,even,bunch,fake,leg,shot,stand,therey,special,effect,
arent,greatest,definit,good,one,scene,yeti,smash,warehous,done,well,anoth,use,
window,build,ladder,step,climb,top,shatter,window,foot,often,shock,occup,insid,
one,sequenc,realli,look,much,much,better,bad,movieyeti,never,stoop,low,say,ape,
actual,time,even,come,close,genuin,silli,beauti,girl,caus,yeti,nippl,becom,erect,lift,
eyebrow,yeah,babi,manner,even,isnt,bad,kinda,even,get,laugh,viewerth,movi,
pretti,long,kind,thing,surpris,enough,doesnt,get,bore,stori,actual,good,watch,
utter,gorgeous,actress,screen,make,male,viewer,happyyeti,may,upper,echelon,
giant,monster,flick,definit,better,king,kong,ripoff,like,ape,queen,kong,far]

Figure 14 Example of Review with Conflicting Sentiment

In this case, several words with positive and negative sentiment are found, which may have confused the algorithm and caused the misclassification.

2.6.2 Results

The following figure gives the distribution of misclassification for the best performing model Naïve Bayes model.

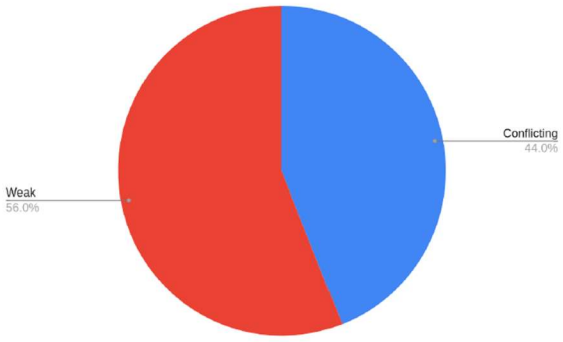


Figure 15 Distribution of Misclassifications for Best Performing Model

The misclassifications are close to evenly distributed, with 56% being caused by weak sentiment, and 44% being caused by conflicting sentiment.

3 DISCUSSIONS

3.1 Implications of Developed Model

As thought of by team member Alex Leakos, one use case for this model in the real word could be for Movie production companies to algorithmically determine the success of their movies on platforms where a rating system is not available, without having to physically read the thousands of reviews their film may receive online. There may be certain platforms or communities where a genre of movie performs better, and if a movie company produced several genres of movies, they could assess the overall sentiment of these movies and then proceed to allocate advertising spend as efficiently as possible.

Another scenario where this model could be useful (as thought of by team member Graydon Hall), would be to apply the sentiment analysis outside the movie industry, and apply the model to sentiment pertaining to publicly traded companies. By

assessing the sentiment associated with tweets and discussions pertaining to a publicly traded company, one may be able to correlate the sentiment to the stock price, and find a sophisticated trading algorithm to buy and sell the stock based on the daily sentiment surrounding it.

A third and final scenario suggested by team member Jared Kraus, would be for a company like amazon to use sentiment analysis to assess the overall sentiment with certain products. By doing this, they could identify specific products with positive sentiment, and then proceed to promote those items to more customers. Doing this will be a win-win scenario, where hopefully more products are sold, and shoppers on the site are recommended the best possible items.

3.2 Going Forward

In this section, possible areas which the report can be extended on will be discussed. Area number one which has been identified would be doing hyperparameter tuning on a larger subset of the data. In this report, only one thousand rows were used, due to computational limitations faced by using Databricks Community Edition. An area that would be excellent for explanation would be incorporating N-grams into the feature vectors. In the field of sentiment analysis, n-grams could be critical. For example, using N-grams of N=2 would allow for significant improvement for word combinations such as “don’t like” or “not great”. Other classifiers such as Gradient Boosted Tree Classifier or XGBoost could be also tested. Possibly the greatest area which could lead to improvement would be using multi-classification instead of a binary classifier. As discussed in the results section, the two main causes of mis-classification were identified as being “weak sentiment” and “conflicting sentiment”. Adding a third label of “neutral” sentiment could solve this problem, and thus produce much more accurate results.

4 CONCLUSIONS

The objective of this report was to find the optimal classifier for performing sentiment analysis on movie reviews, and explore reasons for misclassification of reviews. The dataset used consisted of fifty thousand movie reviews labelled as positive or negative, plus one thousand reviews manually scraped and labelled by the members of this group. For the data pre-processing phase, the reviews were thoroughly cleaned and tokenized, and then transformed into three types of feature vectors: a simple bag of words vector, a TF-IDF vector, and a Word2Vec vector. The three classifiers studied in this report were Naïve Bayes, Random Forest, and Decision Tree Classifiers. Regarding the feature vectors, it was concluded that Word2Vec produced the best results for Random Forest Classifiers and Decision Tree Classifiers, while the bag of words vector produced the best results for the Naïve Bayes model.

Each classifier was tuned to find the optimal parameters. For the Naïve Bayes Classifier, it was concluded that the hyperparameters producing the highest accuracy on the dataset were smoothing equal to 12, and model type equal to “complement”. For the Random Forest Classifier, it was concluded that the hyperparameters producing the highest accuracy on the dataset were number of trees equal to 400, and max depth equal to 10. For the Decision Tree Classifier, it was concluded that the hyperparameters producing the highest accuracy on the dataset were impurity equal to “entropy” and max depth equal to 20. After testing out all these tuned models on the datasets, the highest performing model on the one thousand new rows of data was the Random Forest model, achieving an accuracy of 61%. The highest performing model on the combined dataset and old dataset alone was the Naïve Bayes model, with accuracies of 85.0% and 85.3% respectively. This comes extremely close to the report this was based off, who achieved a maximum accuracy of 86.6% on the fifty thousand movie reviews, using a logistic regression classifier. [2]

In the last section of the lab, reasons for misclassification of the one thousand new data points was investigated. It was concluded that the primary reasons for misclassification were reviews either having weak sentiment or having very conflicting sentiment. Since the dataset consisting of fifty thousand labelled IMDb reviews consisted of polar reviews only (reviews clearly positive or negative), it is concluded that the primary reasons for the models in this study performing worse can be boiled down to the choice to not filter the one thousand reviews chosen, and strictly take one thousand reviews randomly. Lastly, it is concluded that based on the results of all the classifiers in this paper, Natural Language Processing is an excellent technique for determining the sentiment of reviews, especially in cases where the number of reviews makes them impractical to read and classify manually.

5 BIBLIOGRAPHY

[1] Statista, "Cumulative number of reviews submitted to Yelp from 2009 to 2020," [Online]. Available: https://www.statista.com/statistics/278032/cumulative-number-of-reviews-submitted-to-yelp/. [Accessed 09 12 2021].

[2] S. G. Hadi Pouransari, "Deep learning for sentiment analysis of movie reviews," Stanford, Palo Alto, 2015.

[3] Kaggle, "IMDB Dataset of 50K Movie Reviews," [Online]. Available: https://www.kaggle.com/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews. [Accessed 9 12 2021].

[4] IMDbPY, "IMDbPY," [Online]. Available: https://imdbpy.readthedocs.io/en/latest/. [Accessed 9 12 2021].

[5] R. E. D. P. T. P. D. H. Andrew L. Maas, "Learning Word Vectors for Sentiment Analysis," in *HLT Human Language Technology*, 2011.

[6] N. Johnston, "IMDb Movie Ratings Over the Years," 9 10 2009. [Online]. Available: http://www.njohnston.ca/2009/10/imdb-movie-ratings-over-the-years/. [Accessed 09 12 2021].

[7] A. Parrish, "Understanding word vectors," [Online]. Available: https://gist.github.com/aparrish/2f562e3737544cf29aaf1af30362f469. [Accessed 9 12 2021].

[8] V. Jayaswal, "Laplace smoothing in Naïve Bayes algorithm," 22 11 2020. [Online]. Available: https://towardsdatascience.com/laplace-smoothing-in-na%C3%AFve-bayes-algorithm-9c237a8bdece. [Accessed 9 12 2021].

[9] A. Spark, "API Reference," [Online]. Available: https://spark.apache.org/docs/latest/api/python/reference/index.html. [Accessed 09 12 2021].