Sentiment Classification

Aditya Mundhara[1] Anubhav Tandon[1] Sidharth Shanu[1] Sushrut Barmate[1]
Tanmay Parashar[1]

[1]IIT Jodhpur
{Aditya, Anubhav, Sidharth, Sushrut, Tanmay}@iitj.ac.in

Abstract

The age of information is upon us. In the 20th century, We have seen a rapid rise of online users and online activity accompanied with a Boom in the user generated data. One such application is Twitter or now called X which is one of the biggest blogging site across the world. The generation of a large amount of User generated data demands a some sort of moderation on the type of data present on the platform. The use of machine learning algorithm to classify a text based on classes can be used to categorize said data , One such categorization can be done by Sentiment analysis. Sentiment analysis is a process of automatically identifying whether a user-generated text expresses positive, negative or neutral opinion about an entity (i.e. product, people, topic, event etc). The main focus of the project is to create classification models which can separate a text based on three sentiments mainly neutral, positive and negative where positive statement which have or show favorable opinion towards people and services and is generable favorable. Negative statement refers to the opposite where curse words or a bad or let down emotion is showed. The essence of the neutral emotion is hard to capture but can to summed up to the lack of positive or negative emotion in the text. We in this report have used Naive Bayes ,Multinomial Naive Bayes and Composite Naive Bayes from scratch and Naive Bayes ,Multinomial Naive Bayes, Composite Naive Bayes , Decision Tree, Random Forest and Support Vector Machines from Sklearn library.
Keywords: Sentiment, Negative, Positive, Neutral, Naive Bayes

Contents

1 Introduction

Sentiment analysis is a process of automatically identifying whether a user-generated text expresses positive, negative or neutral opinion about an entity (i.e. product, people, topic, event etc). The analysis can use a pre-grouped data to train or is constantly trained using new data by providing manual labels to the original label to train the algorithm.
Sentiment analysis can be used in a variety of service related application or user review applications. Some examples include:

- Customer feedback analysis to analyze feedback from surveys, online reviews, customer support tickets, and chat transcripts. Businesses can categorize feedback as positive, negative, or neutral and identify recurring themes or issues raised by customers. This information can be used to improve products, services, and customer experiences.

- E-commerce platforms and review websites often use sentiment analysis to analyze product and service reviews. By categorizing reviews as positive, negative, or neutral, companies can identify popular products, address product issues, and make data-driven decisions to improve customer satisfaction and loyalty.

- Sentiment analysis is used in political analysis and public opinion polling to gauge public sentiment towards political candidates, parties, policies, and current events. By analyzing social media discussions, news articles, and online forums, political analysts can track public opinion trends, identify key issues, and predict election outcomes.

Sentiment analysis provides excellent analytical and categorical opportunities for the data analysis and creating meaning out of a large amount of data. The statistics derived from these analysis affect very important decisions to be made by the organization and thus the need to improve said analysis will always be there. However, sentiment analysis is not a sure short method for such categorization as any Machine learning model is not 100% accurate. Thus the categorizations made should be rough and some interchange between classes should be accepted and allowed.

Models applied are

- Naive Bayes
- Multinomial Naive Bayes
- Decision Tree
- Random Forest
- Support Vector Machines (SVM)
- Logistic regression
- Complement Naive bayes

The organization of paper is as follows:

- Section 1 (Introduction) introduces us to the project and it's usecases in real life.
- Section 2 (Approaches tried) gives an overview on the approaches tried.
- Section 3 (Experiments and Results) gives an overview on the experiments that were done during the project and the results that were achieved.
- Section 4 (Summary) gives a summary on the overall project.

2 Approaches Tried

- **Naive Bayes**
  The Naive Bayes classifier [1] is the simplest (as the name suggests) and most commonly used classifier. Naive Bayes classifier works very well for text classification as it computes the posterior probability of a class, based on the distribution of the words (features) in the document. The model uses the Bag of words feature extraction . It assumes that the features are independent of each other. It uses Bayes Theorem to predict the probability that a given feature:
  P(label/features) = P(label)P(f1/label)...P(fn/label)/P(features)
  P(label) is the prior probability of a label or the likelihood that a label is observed . Given a feature, P(features—label) is the prior probability that feature set is being classified as a label. P(features) is the prior probability that a given feature set is occurred. Given the Naive assumption which states that all features are independent of each other , the equation could be rewritten as follows:
  P(label/features) = P(label)P(f1/label)...P(fn/label)/P(features)

- **Naive Bayes from Scratch and Sklearn Naive Bayes With Changes**
  Due to Poor result from the Naive Bayes Classifier, a change was implemented in the Naive Bayes Classifier while classifying the test data. Due to the vast and ambiguous nature of neutral classification, a small training set and uneven amount of neutral classification in the common words,

the change suggested was to ignore the neutral probabilities and classify the test data as neutral if the probabilities of the negative and positive are separable by a value times the number of words. This created better test results comparable to the Multinomial Naive Bayes and had a much better variance plot due to less number of neutral classifiers.

- **Multinomial Naive Bayes from Scratch and Sklearn**
  Multinomial Naive Bayes classifier is a specific instance of a Naive Bayes classifier which uses a multinomial distribution for each of the features. This is a much better classifier as it does not require any changes to produce good and consistent results.

- **Complement Naive Bayes from Scratch and Sklearn**
  In Complement Naive Bayes, instead of calculating the probability of an item belonging to a certain class, we calculate the probability of the item belonging to all the classes. This is the literal meaning of the word, complement and hence is called Complement Naive Bayes.

- **Decision Tree**[2]
  The training data space is represented in a hierarchical form in which a condition on the attribute value is used to partition the data. The condition on attribute values is the presence or absence of one or more words. The partition of the data space is done recursively until the leaf nodes contain certain minimum numbers of records which are used for the purpose of classification.

- **Random Forest**
  Random Forest is a commonly-used machine learning algorithm that combines the output of multiple decision trees to reach a single result.

- **Support Vector Machine**[2]
  The main principle of SVMs is to find out linear separators or hyperplane in the search space which can best separate the different classes. There can be several hyperplanes that separate the classes, but the one that is chosen is the hyperplane in which the normal distance of any of the data points is the largest, so that it depicts the maximum margin of separation.

- **Logistic Regression**
  It's a supervised machine learning algorithm used for binary classification tasks. It predicts the probability that an instance belongs to a given class based on the logistic function.

3 Experiments and Results

The Training Data has 10 columns out of which we use 3 columns mainly for classification Text , Selected Text and Sentiment. The Text column contains unfiltered text of the tweets containing links, punctuations and stop words and a lot of useless information but also strongly resembles the test Data. The Selected Text contains a filtered version of the Text data which removes most of punctuations and stop words. Sentiment contains the classification of the data into Three Classification "Neutral", "Positive", "Negative" . As to convert categorical data into Integer classification for better look up a map is created "Neutral" to 0 , "Negative" to 1, "Positive" to 2. The Sentiment Distribution for the three classes show an almost equal Negative and Positive Sentiment with Neutral sentiment being a bit greater than both. The word count for Text of the three classes shows a direct correlation with the number of classification. The word count for Selected Text of the three classes does not show a direct correlation with the number of classification. As, the number of neutral words are almost equal to the one obtained in Word count for Text . We can conclude the Selected text data can not highlight the Neutral sentiment from the Text hence uses all of the Text for Neutral Classifiers.

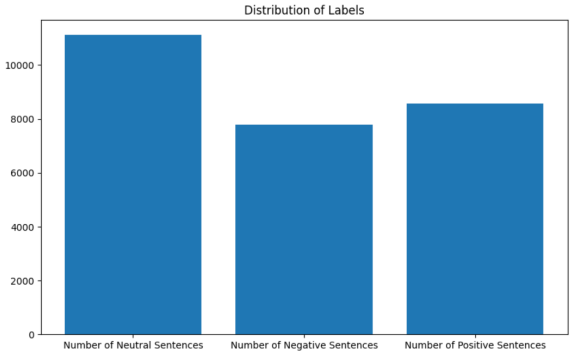


Figure 1: Distribution of labels in train data

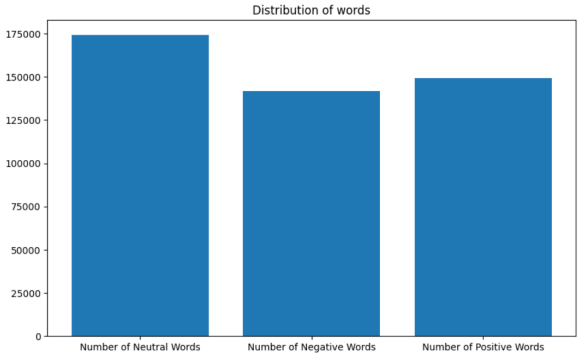


Figure 2: Distribution of words in text in train data

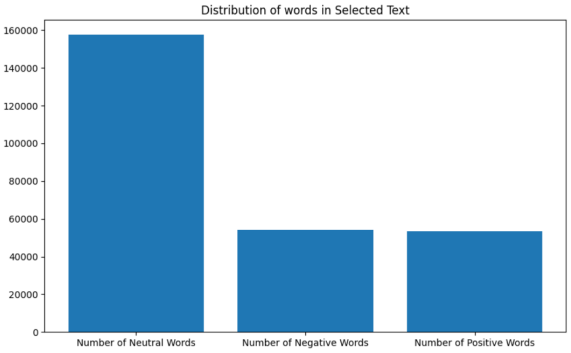


Figure 3: Distribution of words in selected text in train data

- **Naive Bayes and Multinomial NB from Scratch**
  The Training set is very small for the amount of words i.e. the variance present in the Training and Test Data set is very high compared to the number of sentences.

- **Vanilla NB**
  It poorly classifies the text as due to high probability of common words being in Neutral set. The NB Classifies most of the data into Neutral classifier and produces Accuracy of 0.5758 on Text and 0.4204 on selected text .

- **Vanilla NB with Changes**
  The classification greatly improves for both the text and selected text and having a much better spread of the variance among all the three classes and produces Accuracy of 0.6423 on Text and 0.6428 on selected text .

- **Multinomial NB**
  The Classification has good variance and good accuracy on text but fails on the selected text due to the uneven nature of the Selected Text. The Accuracy produced are 0.6392 on Text and 0.5432 on selected Text and produces accuracy of 0.6247 with sklearn.

- **Gaussian NB**
  It fails very bad in classification with a very Low accuracy of 0.3842 and almost uni-classification of Text to Positive class.

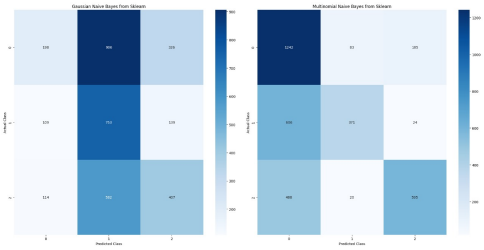


Figure 4: Multinomial and Gaussian Bayes from sklearn

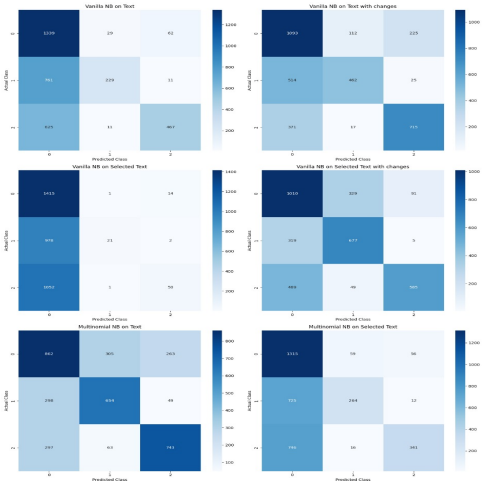
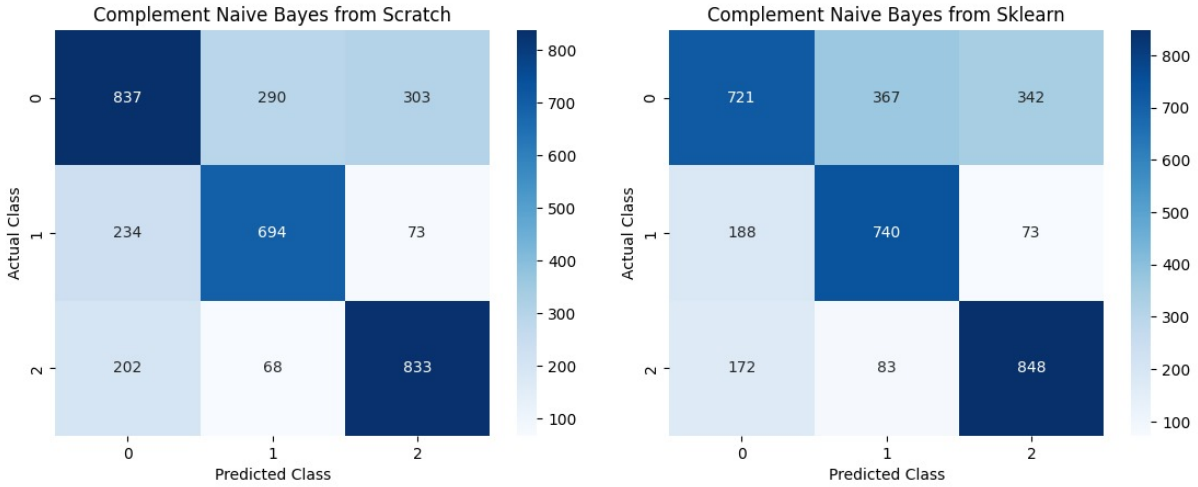


Figure 5: Multinomial and Vanilla Bayes from sklearn

- **Complement Naive Bayes**
  The Complement Naive Bayes produced best results out of all the Naive Bayes as it is specifically designed for the data sets where there is a mismatch of size in the classes. The accuracy of implementation from scratch was 0.6689 and of implementation through sklearn was 0.6533. The Sklearn version had much better classification for positive and negative class but fell short in correctly classifying Neutral class massively.



(a) (b)

Figure 6

- **Decision tree**
  The decision tree classifier when applied from sklearn library has accuracy 0.54. While it performs better than random guessing, its performance is modest,.

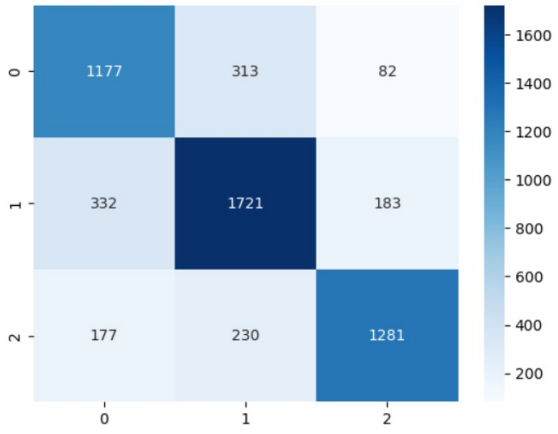


Figure 7: Decision Tree from sklearn

- **Random Forest**
  The random forest has an accuracy of 0.527 when applied from sklearn library. Despite its ensemble nature, it performs marginally better than chance. This suggests that the model may not effectively capture the complexities of tweet sentiment.

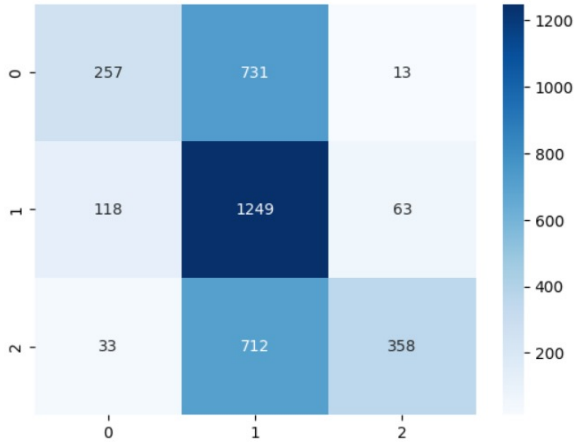


Figure 8: Random Forest from sklearn

- **Support Vector Machine**
  The Support Vector Machine has an accuracy of 0.70 because the performance indicates it captures underlying patterns well, balancing variance effectively.

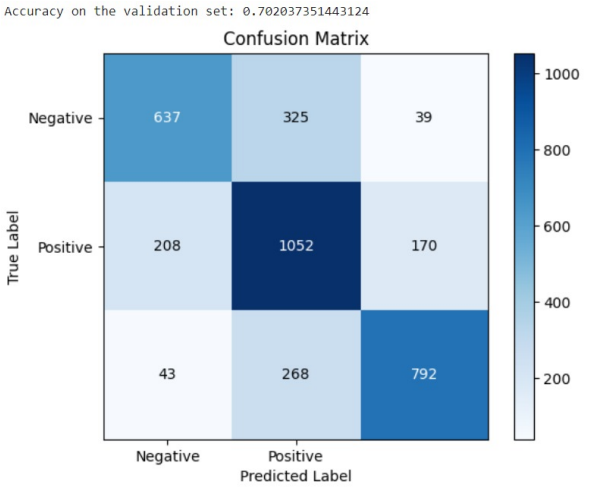


Figure 9: Support Vector Machines from sklearn

- **Logistic Regression**
  The Logistic regression is primarily a binary classifier but can be extended to fit multiple classes. The accuracy of the model when tested on the training data is 0.5947. The large number of Predicted Positive label can be accounted as the if we visualize the graph of the regression with three classes the range of probabilities which lie in the middle region is considerably higher than the other two regions.

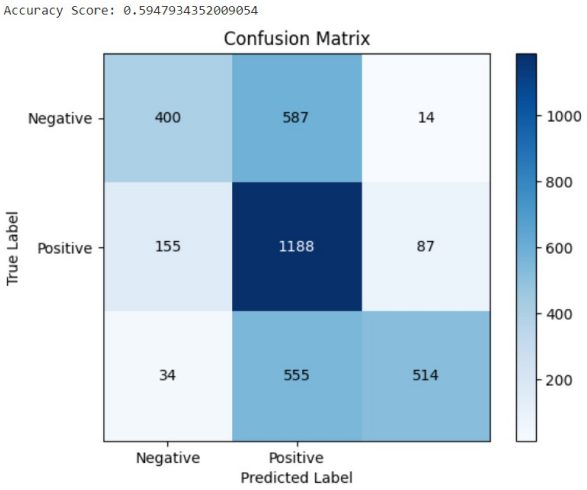


Figure 10: Logistic Regression from sklearn

4 Summary

All the models produced accuracy results in the mid 60's to low 70's which shows the gradual convergence of testing accuracies with different models. An observation which can be made from the confusion matrix plot of all the prediction shows that most of the models had no problem in differentiating between Positive and Negative sentiment but a large amount of inaccuracies came from either misclassification of neutral sentiment or from categorization into neutral sentiment. This again highlights the vast amount of emotions categorized as neutral and the big overlap of neutral and other two classes while there is no to little overlap between negative and positive sentiment and most of the models if given a choice could easily identify out of the two.
To Summarize the entire process the best model for the given training and testing data set was SVM. A tabulated accuracy for various models has been shown below:
multirow booktabs

Table 1: Accuracy of Machine Learning Models

Model	**Accuracy (scratch) (%)**	**Accuracy (sklearn) (%)**
Vanilla NB	57.58	64.23
Multinomial NB	63.92	62.47
Gaussian NB	-	38.42
Random Forest	-	52.74
Decision Tree	-	54
Complement Naive Bayes	66.89	65.33
Support Vector Machine	-	70
Logistic Regression	-	59.4

References

[1] Machine learning techniques for sentiment analysis: A review. In *researchgate*. URL `https://www.researchgate.net/profile/Shabib-Aftab-2/publication/317284281_Machine_Learning_Techniques_for_Sentiment_Analysis_A_Review/links/59302d6ba6fdcc89e78431ec/Machine-Learning-Techniques-for-Sentiment-Analysis-A-Review.pdf`.

[2] Application of machine learning techniques to sentiment analysis. In *IEEE*. URL `https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7912076`.

A Contribution of each member

1. **Aditya Mundhara**: Implemented Multinomial naive bayes and logistic regression using scikit library. Prepared project page and report
2. **Anubhav Tandon**: Implemented decision tree, random forest and cleaned the data. Prepared video recording and slides.
3. **Sidharth Shanu**: Implemented naive bayes and Multinomial naive bayes from scratch. Prepared report.
4. **Sushrut Barmate**: Implemented Support Vector Machine (SVM). Prepared the video recording and slides.
5. **Tanmay Parashar**: Implemented Complement naive bayes