STOCK SENTIMENT ANALYSIS USING NEWS HEADLINES

A Project Report Submitted

in Partial Fulfilment of the Requirements

for the Degree of

MASTER OF SCIENCE

in

Mathematics and Computing

by

Shibasish Shaw

(Roll No. 212123053)

to the

DEPARTMENT OF MATHEMATICS

INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI

GUWAHATI - 781039, INDIA

April 2023

CERTIFICATE

This is to certify that the work contained in this report entitled **"Stock Sentiment Analysis using News Headlines"** submitted by **Shibasish Shaw** (**Roll No: 212123053**) to Department of Mathematics, Indian Institute of Technology Guwahati towards the requirement of the course **MA699 Project** has been carried out by him under my supervision.

It is also certified that this report is a survey work based on the references in the bibliography.

Guwahati - 781 039	(Prof. Rajen Kumar Sinha)
April 2023	Project Supervisor

ABSTRACT

The main aim of the project is to demonstrate how the price of stocks fluctuate as a result of human sentiment using Machine Learning algorithms. We have talked about various ML models that happen to be more efficient in the context of text classification problems. Also, we have demonstrated how the various algorithms perform in comparison to others.
The Turnitin similarity is 21 %.

Contents

List of Figures vi

List of Tables viii

1 Introduction 1

1.1 On Sentiment Analysis 1

1.2 How is Sentiment Relevant in the Stock Market? 2

1.3 How can Machine Learning be used for Sentiment Analysis? . 2

2 The Dataset 3

2.1 Description of the Data 3

2.2 Data Preprocessing 4

2.3 Exploring the Data 7

2.4 Converting the Textual Data into Vectors 10

2.4.1 Why choose Bag of Words? 10

2.4.2 How does BoW work? 11

3 Applying Random Forest 12

3.1 How does Random Forest work? 12

3.2 The Experiment 15

3.2.1 With Removing Stopwords and Lemmatization 16

3.2.2 With Lemmatization only 17

3.2.3 Without Removing Stopwords and Lemmatization . . . 18

4 Applying Naïve Bayes **20**

4.1 How does Naïve Bayes work? 20

4.2 The Experiment 21

5 Applying CNN **23**

5.1 How does CNN work? 23

5.2 The Experiment 25

6 Conclusion **28**

Bibliography **30**

List of Figures

2.1 Top 5 rows of the Dataset 4

2.2 The Dataset is More or Less Balanced 8

2.3 Output vs The Number of Negative Words 9

2.4 Sentiment Distribution of News Headlines using VaderSentiment 10

3.1 How Bagging Works 14

3.2 Random Forest Example 15

3.3 The Confusion Matrix using RF with Removing Stopwords and Lemmatization 16

3.4 The Confusion Matrix using RF with Lemmatization only . . 17

3.5 The Confusion Matrix using RF without Removing Stopwords and Lemmatization 18

4.1 The Confusion Matrix using Naïve Bayes 22

5.1 Basic CNN Architecture 23

5.2 Max Pooling 25

5.3 The Confusion Matrix using CNN 26

6.1 Confusion Matrices 29

6.2 Comparing the Performances 29

List of Tables

3.1 Classification Report using RF with Removing Stopwords and Lemmatization . . . 17

3.2 Classification Report using RF with Lemmatization only . . . 18

3.3 Classification Report using RF without Removing Stopwords and Lemmatization . . . 19

4.1 Classification Report using Naïve Bayes . . . 22

5.1 Classification Report using CNN . . . 27

Chapter 1

Introduction

This project basically demonstrates how price of a particular stock fluctuates in the share market due to human sentiment. The following Machine Learning models have been used to do the same

- Random Forest
- Naïve Bayes

Also, the following Deep Learning model has been used

- Convolutional Neural Networks (CNN)

1.1 On Sentiment Analysis

Sentiment analysis, also referred to as opinion mining, is an approach to natural language processing (NLP) that identifies the emotional tone behind a body of text.

The main objective of Sentiment Analysis is to determine whether the sentiment of the relevant masses is positive or negative or neutral towards a particular topic or subject matter.

1.2 How is Sentiment Relevant in the Stock Market?

There are various reasons for the fluctuations in a stocks price. One of them happens to be Sentiment. Sentiment is basically formed as a result of news regarding the company.

As an example, an earnings report that reveals significant profit, the launch of a new product, missed targets, or the death or departure of a key figure could all lead to swings in demand and share prices.

1.3 How can Machine Learning be used for Sentiment Analysis?

Machine Learning becomes relevant when we want to predict what is going to happen on the basis of the data we have from the past.

We basically train the models on how the stock price changed as a result of several news headlines, and ask it to predict what might happen the future.

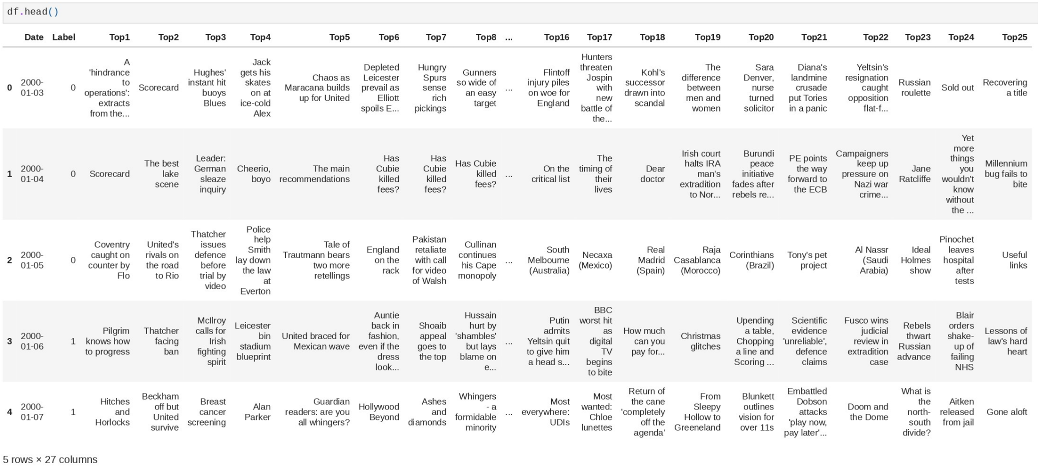
Chapter 2

The Dataset

The dataset has been taken from Kaggle. [9]

2.1 Description of the Data

- Data ranges from 2008 to 2016 and the data from 2000 to 2008 was scrapped from Yahoo finance.
- There are 25 columns of top news headlines for each day in the data frame.
- Class 1- the stock price increased.
- Class 0- the stock price stayed the same or decreased.



```
df.head()
```

	Date	Label	Top1	Top2	Top3	Top4	Top5	Top6	Top7	Top8	...	Top16	Top17	Top18	Top19	Top20	Top21	Top22	Top23	Top24	Top25
0	2000-01-03	0	A 'hindrance to operations': extracts from the...	Scorecard	Hughes' instant hit buoys Blues	Jack gets his skates on at ice-cold Alex	Chaos as Maracana builds up for United	Depleted Leicester prevail as Elliott spoils E...	Hungry Spurs sense rich pickings	Gunners so wide of an easy target	...	Flintoff injury piles on woe for England	Hunters threaten Jospin with new battle of the...	Kohl's successor drawn into scandal	The difference between men and women	Sara Denver, nurse turned solicitor	Diana's landmine crusade put Tories in a panic	Yeltsin's resignation caught opposition flat-f...	Russian roulette	Sold out	Recovering a title
1	2000-01-04	0	Scorecard	The best lake scene	Leader: German sleaze inquiry	Cheerio, boyo	The main recommendations	Has Cubie killed fees?	Has Cubie killed fees?	Has Cubie killed fees?	...	On the critical list	The timing of their lives	Dear doctor	Irish court halts IRA man's extradition to Nor...	Burundi peace initiative fades after rebels re...	PE points the way forward to the ECB	Campaigners keep up pressure on Nazi war crime...	Jane Ratcliffe	Yet more things you wouldn't know without the ...	Millennium bug fails to bite
2	2000-01-05	0	Coventry caught on counter by Flo	United's rivals on the road to Rio	Thatcher issues defence before trial by video	Police help Smith lay down the law at Everton	Tale of Trautmann bears two more retellings	England on the rack	Pakistan retaliate with call for video of Walsh	Cullinan continues his Cape monopoly	...	South Melbourne (Australia)	Necaxa (Mexico)	Real Madrid (Spain)	Raja Casablanca (Morocco)	Corinthians (Brazil)	Tony's pet project	Al Nassr (Saudi Arabia)	Ideal Holmes show	Pinochet leaves hospital after tests	Useful links
3	2000-01-06	1	Pilgrim knows how to progress	Thatcher facing ban	McIlroy calls for Irish fighting spirit	Leicester bin stadium blueprint	United braced for Mexican wave	Auntie back in fashion, even if the dress look...	Shoaib appeal goes to the top	Hussain hurt by 'shambles' but lays blame on e...	...	Putin admits Yeltsin quit to give him a head s...	BBC worst hit as digital TV begins to bite	How much can you pay for...	Christmas glitches	Upending a table, Chopping a line and Scoring ...	Scientific evidence 'unreliable', defence claims	Fusco wins judicial review in extradition case	Rebels thwart Russian advance	Blair orders shake-up of failing NHS	Lessons of law's hard heart
4	2000-01-07	1	Hitches and Horlocks	Beckham off but United survive	Breast cancer screening	Alan Parker	Guardian readers: are you all whingers?	Hollywood Beyond	Ashes and diamonds	Whingers - a formidable minority	...	Most everywhere: UDIs	Most wanted: Chloe lunettes	Return of the cane 'completely off the agenda'	From Sleepy Hollow to Greeneland	Blunkett outlines vision for over 11s	Embattled Dobson attacks 'play now, pay later'...	Doom and the Dome	What is the north-south divide?	Aitken released from jail	Gone aloft

5 rows × 27 columns

Figure 2.1: Top 5 rows of the Dataset

As we can see, the dataset has 27 columns. The columns titled "Label"is indicative of the stock price movement. It is followed by the top 25 news headlines of the corresponding date. There are 4101 rows in total.

2.2 Data Preprocessing

Now, firstly we will check how many null values are present in each of the columns.

We see that the last three columns are having 1, 3 and 3 null values respectively. But, in order to apply the models, we will combine all the news headlines in a single sentence corresponding to a particular date. That is why the few null values that are appearing would not really matter in practice. We have done the following operations on the data:

- We have split the dataset into two parts, train (before 2014) and test (after 2014). We will train our models with the first part and then test the accuracy of the models with the later one.

- Then, We have constructed a new dataset, called “data”, that stores only the news headlines, for the train dataset.
- Next, we have replaced all of the punctuation (i.e. anything apart from “a”to “z”and “A”to “Z”.
- Now, we make a list “headlines”, where each of the elements of it comprise of the 25 corresponding news headlines.

  This is how headlines[675] looks like:

  millar sorry for vuelta protest verdict on the season the heroes and zeros and hopes for next year fa ups its stake in new wembley to m fletcher poised to sign new contract this is personal battle of the bulge my new media new media diary television s dating game media monkey news real race row victory local radio for local people getting broadband off the launch pad my media itv looks to plan b the man who lured dawn airey to sky explains how he did it and why round up spurs must see that keane is not a front runner ipswich on road to riches via lips of matt holland al fayed the spy chief and a festering year old feud boston s up european round up west ham manchester city manchester united tottenham hotspur
- Now, some of the words can be avoided in sentiment analysis as they might not play any part in deciding the sentiment of any text. Such words can be found in the list of **Stopwords** in the **NLTK** library. The words are [’i’, ’me’, ’my’, ’myself’, ’we’, ’our’, ’ours’, ’ourselves’, ’you’,

”you’re”, ”you’ve”, ”you’ll”, ”you’d”, ’your’, ’yours’, ’yourself’, ’yourselves’, ’he’, ’him’, ’his’, ’himself’, ’she’, ”she’s”, ’her’, ’hers’, ’herself’, ’it’, ”it’s”, ’its’, ’itself’, ’they’, ’them’, ’their’, ’theirs’, ’themselves’, ’what’, ’which’, ’who’, ’whom’, ’this’, ’that’, ”that’ll”, ’these’, ’those’, ’am’, ’is’, ’are’, ’was’, ’were’, ’be’, ’been’, ’being’, ’have’, ’has’, ’had’, ’having’, ’do’, ’does’, ’did’, ’doing’, ’a’, ’an’, ’the’, ’and’, ’but’, ’if’, ’or’, ’because’, ’as’, ’until’, ’while’, ’of’, ’at’, ’by’, ’for’, ’with’, ’about’, ’against’, ’between’, ’into’, ’through’, ’during’, ’before’, ’after’, ’above’, ’below’, ’to’, ’from’, ’up’, ’down’, ’in’, ’out’, ’on’, ’off’, ’over’, ’under’, ’again’, ’further’, ’then’, ’once’, ’here’, ’there’, ’when’, ’where’, ’why’, ’how’, ’all’, ’any’, ’both’, ’each’, ’few’, ’more’, ’most’, ’other’, ’some’, ’such’, ’no’, ’nor’, ’not’, ’only’, ’own’, ’same’, ’so’, ’than’, ’too’, ’very’, ’s’, ’t’, ’can’, ’will’, ’just’, ’don’, ”don’t”, ’should’, ”should’ve”, ’now’, ’d’, ’ll’, ’m’, ’o’, ’re’, ’ve’, ’y’, ’ain’, ’aren’, ”aren’t”, ’couldn’, ”couldn’t”, ’didn’, ”didn’t”, ’doesn’, ”doesn’t”, ’hadn’, ”hadn’t”, ’hasn’, ”hasn’t”, ’haven’, ”haven’t”, ’isn’, ”isn’t”, ’ma’, ’mightn’, ”mightn’t”, ’mustn’, ”mustn’t”, ’needn’, ”needn’t”, ’shan’, ”shan’t”, ’shouldn’, ”shouldn’t”, ’wasn’, ”wasn’t”, ’weren’, ”weren’t”, ’won’, ”won’t”, ’wouldn’, ”wouldn’t”]

- Also, we want to apply **Lemmatization** on the textual data. What it basically does, is reduce each of the words to its base form. For example, “am”, “are”, and “is”are all reduced to “be”. This happens so that the infected words can be grouped together in a single form. The analysis might become easier and more proficient this way.

- After applying both of these techniques on our data, the data is supposed to become much more suitable for being converted into vectors and training the models efficiently. Here is the same sentence, that we encountered before after the preprocessing, headline[675]: millar sorry vuelta protest verdict season hero zero hope next year fa ups stake new wembley fletcher poised sign new contract personal battle bulge new medium new medium diary television dating game medium monkey news real race row victory local radio local people getting broadband launch pad medium itv look plan b man lured dawn airey sky explains round spur must see keane front runner ipswich road rich via lip matt holland al fayed spy chief festering year old feud boston european round west ham manchester city manchester united tottenham hotspur

- **We will perform the experiment with and without applying these two text preprocessing techniques and see which is more suitable.**

2.3 Exploring the Data

Now, let us have a look about how the price movements are distributed.

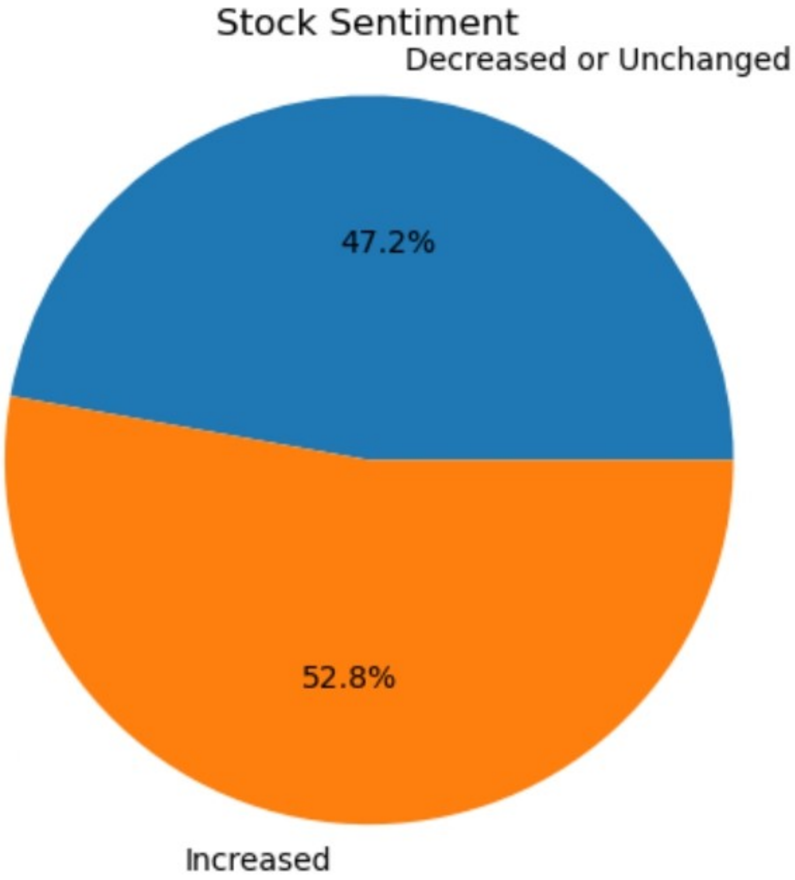


Figure 2.2: The Dataset is More or Less Balanced

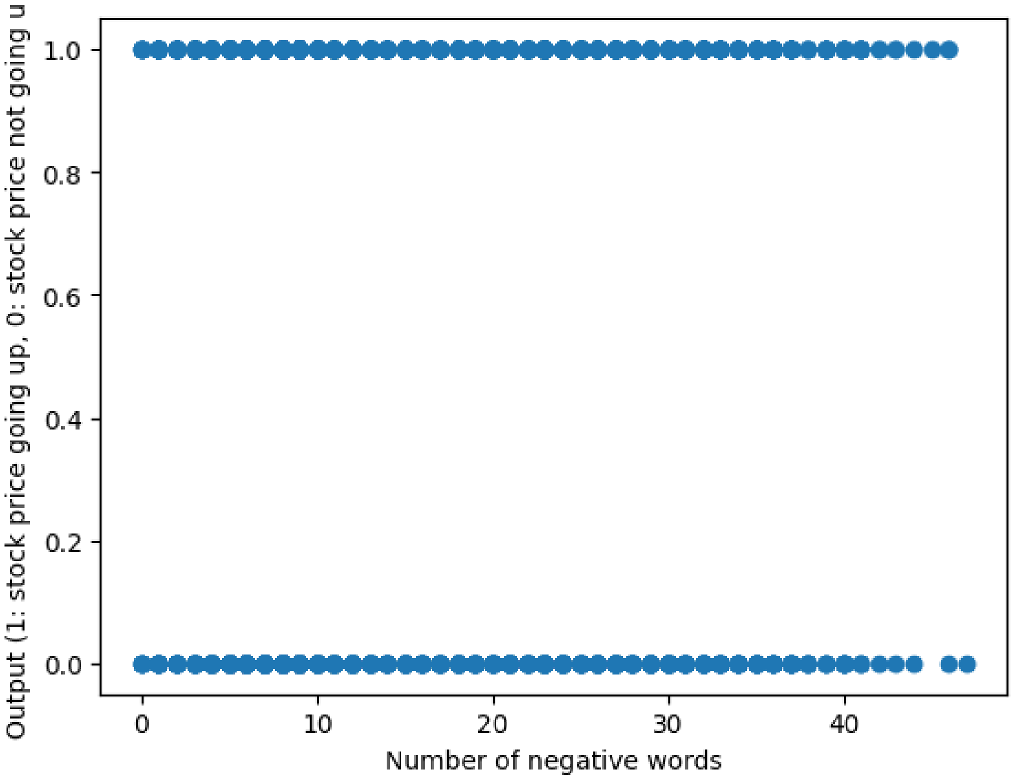


Figure 2.3: Output vs The Number of Negative Words

From the above figure, it can be concluded that the sentiment of a sentence depends on other factors apart from the negative word count.

We have calculated the sentiment score of each of the headlines (without the preprocessing), using **SentimentIntensityAnalyzer()**, which we have imported from **nltk.sentiment.vader**. Here is the sentiment distribution

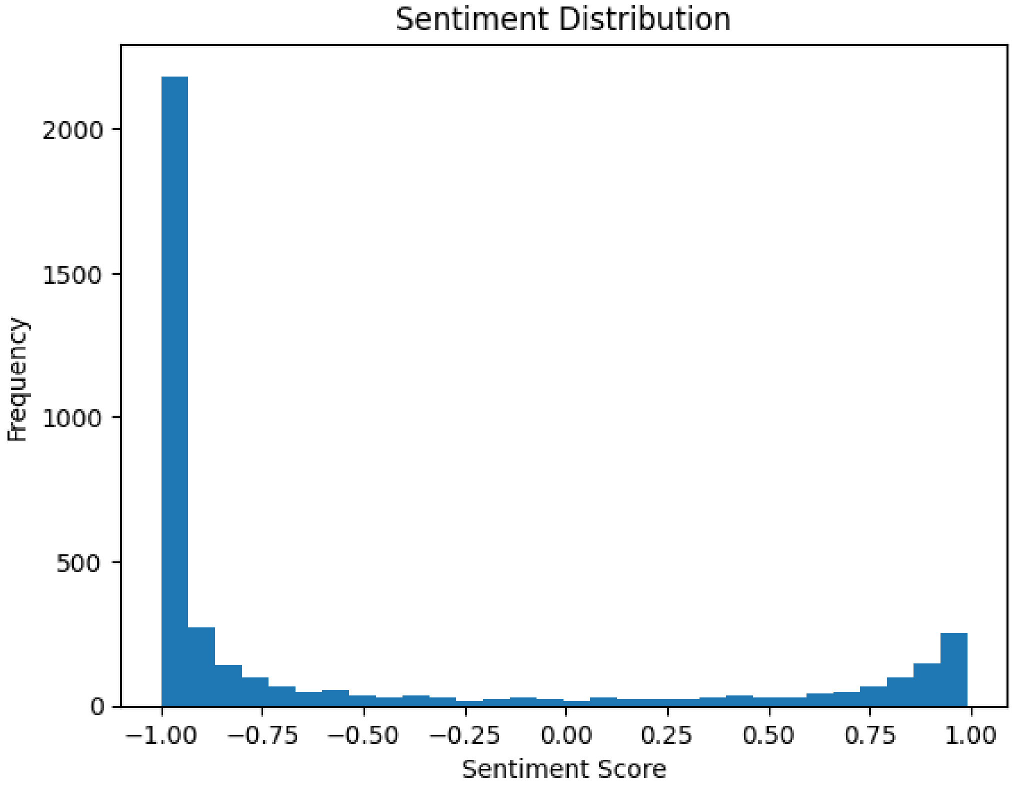


Figure 2.4: Sentiment Distribution of News Headlines using VaderSentiment

2.4 Converting the Textual Data into Vectors

The objective is to convert the textual input data into vectors. We will train our models with these vectors.

2.4.1 Why choose Bag of Words?

We are going to use **Bag of Words** for the conversion. The reasons why we decided to go with this over **Word2Vec** and **TF-IDF** are

- Word2Vec is much more complex and requires a much larger dataset in order to build the Vocabulary, but there is not a very big dataset needed for BoW. That is why it should be better to stick with BoW.

- TF-IDF on the other hand, puts more emphasis on the words that are more frequent throughout the dictionary. But, a negative word is a negative word at the end of the day. All of them will contribute just equally to the output. That is why BoW is appropriate.

2.4.2 How does BoW work?

BoW basically works in the following manner:

- **Tokenization:** In this part, all of the sentences are broken down into words or tokens.

- **Vocabulary Creation:** After the first part we are left with numerous words. Now, BoW identifies each of the unique words and assigns an ID to it.

- **Document Representation:** Each of the documents are represented as vectors. Where each of the entries of the vector are basically the word count of the word in the vocabulary that is at that particular index. This means that the size of each vector is the number of words in the vocabulary or the dictionary.

- **Normalization:** In this part the vectors are normalized to compensate for the fact that some of the documents are going to be longer than the others.

Chapter 3

Applying Random Forest

In this chapter, we are going to apply Random Forest on the dataset, with and without the preprocessing.

3.1 How does Random Forest work?

[7] In order to understand Random Forest, we need to understand Decision Trees first. In Decision Tree, we divide the data points using several questions into smaller subsets that are easier to model. We use **entropy** or **Gini Impurity** to select the best feature and the threshold for splitting. We keep on making splits unit we reach a **leaf node**. Once the model has been trained, we can make predictions for new data by following the tree branches to some leaf node.

In Random Forest, we use the method of **Bagging**. Bagging works in the following steps:

- Firstly, we draw a number of samples, say n from the dataset. This n must not be greater than the size of the data set.

- Then, we train a model on the sample collected. This process is repeated any number of times with different models. Generally, more the number of models, better it is. But, using an excessive number of models can lead to overfitting.

- In the final step, for the new dataset, we train each of the models and record their outputs.

- For Regression problems, we take the average of the outputs from the models and for Classification problems, we take the majority of the outputs of the models.

- Here is a demonstration of how Bagging works: [10]

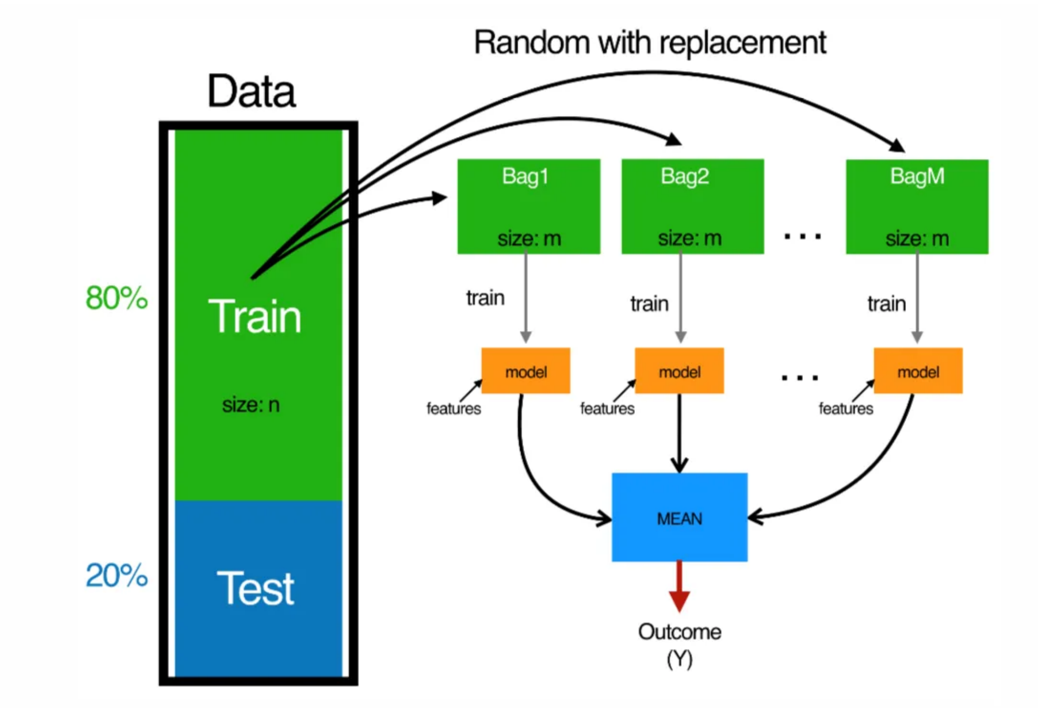


Figure 3.1: How Bagging Works

Now, let us understand Random Forest. In this ensemble method, we use Decision Tree models for all of the constituents. Also, it is a little different from vanilla bagging. In addition to performing **Row Sampling**, we also perform something called **Feature Sampling**. In Feature Sampling, we take a subset of the set of features for the training. We take different subsets for each of the models.

Why do we do Feature Sampling? In the list of the features, there might be a strong predictor, and some moderate predictors. Now, if we take all of the features, all of the models will use the strong predictor in the first split, as the strong predictor will be having more entropy or Gini impurity. This will result in all the Trees being similar, which will ultimately lead to higher correlation and high variance.

After training each of the models, just like in the case of vanilla bagging, we take the majority of the outputs.

Decision Trees are highly sensitive to a small variation in the input data, by using Random Forest, we can get rid of such troubles.

Here is an example of how Random Forest works [10]

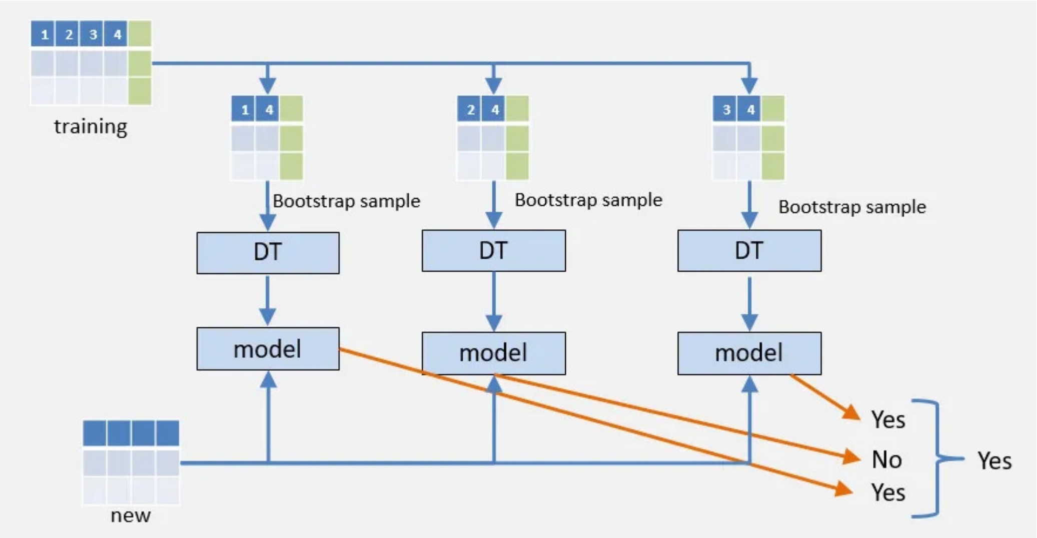


Figure 3.2: Random Forest Example

3.2 The Experiment

We have used the scikit-learn library's RandomForestClassifier to train a classification model on the given training dataset and corresponding labels for each document. The model is based on a random forest algorithm with 200 decision trees and the 'entropy' criterion for node splitting.

We have trained the models in the following ways:

- With Removing Stopwords and Lemmatization
- With Lemmatization only
- Without Removing Stopwords and Lemmatization

3.2.1 With Removing Stopwords and Lemmatization

The performance metrics are given by:

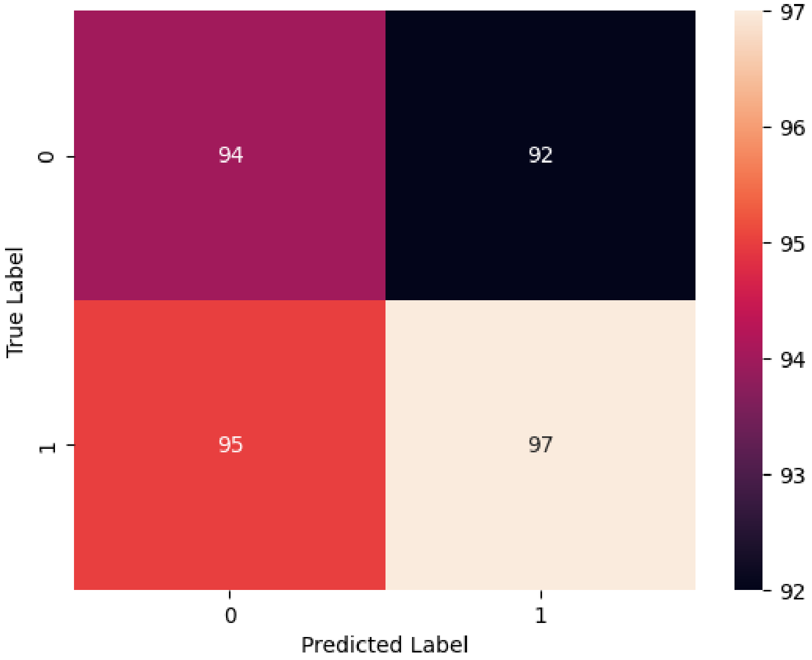


Figure 3.3: The Confusion Matrix using RF with Removing Stopwords and Lemmatization

The accuracy is 0.51 with the support of 378.

	precision	**recall**	**f1-score**	**support**
0	0.50	0.51	0.50	186
1	0.51	0.51	0.51	192
macro avg	0.51	0.51	0.51	378
weighted avg	0.51	0.51	0.51	378

Table 3.1: Classification Report using RF with Removing Stopwords and Lemmatization

3.2.2 With Lemmatization only

The performance metrics are given by:

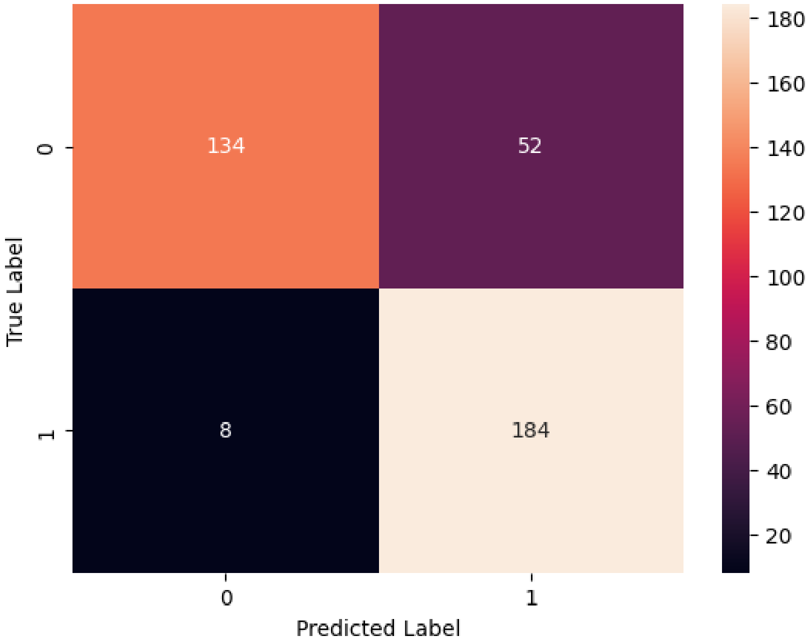


Figure 3.4: The Confusion Matrix using RF with Lemmatization only

The accuracy is 0.84 with the support of 378.

	precision	recall	f1-score	support
0	0.94	0.72	0.82	186
1	0.78	0.96	0.86	192
macro avg	0.86	0.84	0.84	378
weighted avg	0.86	0.84	0.84	378

Table 3.2: Classification Report using RF with Lemmatization only

3.2.3 Without Removing Stopwords and Lemmatization

The performance metrics are given by:

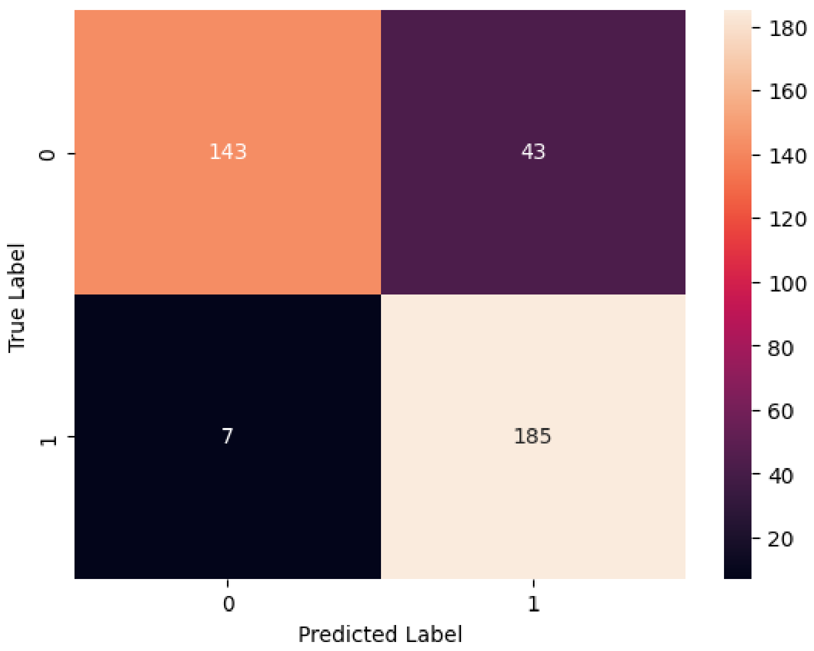


Figure 3.5: The Confusion Matrix using RF without Removing Stopwords and Lemmatization

The accuracy is 0.87 with the support of 378.

	precision	**recall**	**f1-score**	**support**
0	0.95	0.77	0.85	186
1	0.81	0.96	0.88	192
macro avg	0.88	0.87	0.87	378
weighted avg	0.88	0.87	0.87	378

Table 3.3: Classification Report using RF without Removing Stopwords and Lemmatization

We are **not** going to apply such preprocessing techniques further as they fetch a much lower accuracy rate.

Chapter 4

Applying Naïve Bayes

4.1 How does Naïve Bayes work?

[8]

- **Data Preprocessing:** The first step is to preprocess the text data to convert it into a set of numerical features that can be used for classification. The result of this step is a set of features for each news article, such as the frequency of each word in the article.

- **Training:** Once we have preprocessed the data, we can train the Naïve Bayes model on a labeled dataset. During training, the algorithm calculates the prior probability of each class (for example, sports or politics) based on the frequency of each class in the training data. It also calculates the likelihood of each feature (i.e., the probability of a word occurring in a news article given the class label) based on the frequency of each word in the training data for each class. To calculate the likelihood, Naïve Bayes assumes that the features are conditionally independent given the class label, which means that the presence or absence of one feature doesn't affect the probability of another feature occurring given the class label.

- **Prediction:** To classify a new news article, the Naïve Bayes algorithm calculates the posterior probability of each class given the observed features using Bayes' theorem. Specifically, it multiplies the prior probability of each class by the likelihood of each feature given that class, and normalizes the result so that the probabilities sum to 1. The class with the highest posterior probability is then predicted as the output.

4.2 The Experiment

We have used the Multinomial Naïve Bayes algorithm from the scikit-learn library to train the classification model. The fitted model is trained on the training dataset and the corresponding labels.

We have performed this experiment without removing Stopwords and Lemmatization. The performance metrics are given by:

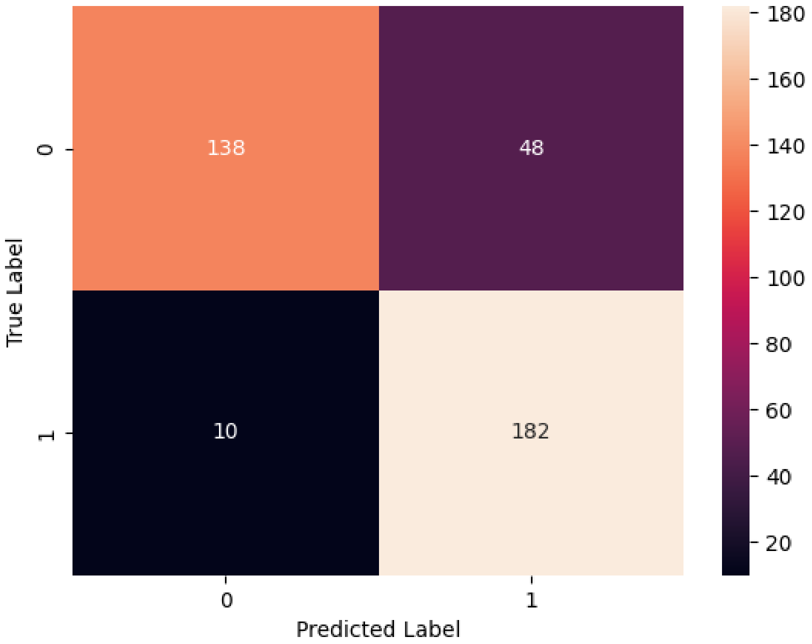


Figure 4.1: The Confusion Matrix using Naïve Bayes

The accuracy is 0.85 with the support of 378.

	precision	**recall**	**f1-score**	**support**
0	0.93	0.74	0.83	186
1	0.79	0.95	0.86	192
macro avg	0.86	0.84	0.84	378
weighted avg	0.86	0.85	0.84	378

Table 4.1: Classification Report using Naïve Bayes

Chapter 5

Applying CNN

5.1 How does CNN work?

CNN is a little different than ANN. At each layer, it tries to find some pattern or some useful information of the data. Here is the basic CNN architecture [3]

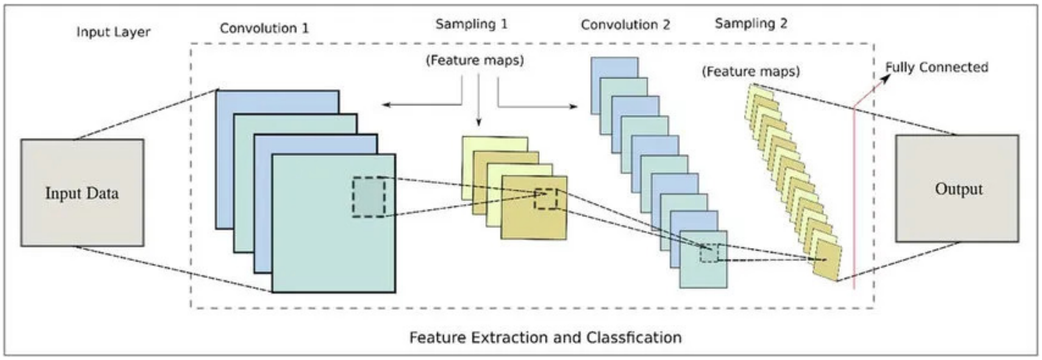


Figure 5.1: Basic CNN Architecture

CNN works in the following manner [6]

- **Input Layer:** Firstly, it converts the text data into a sparse matrix of term frequency counts for each word in the corpus.
- **Convolution Layer:** Next, a set of filters or kernels are applied on the input matrix in order to facilitate feature extraction, the filters slide over the input matrix and perform a dot product between their weights and a small window of the input matrix. This generates a feature map, which highlights local features of the matrix. There are some parameters that we need to know in this process, they are, Stride: Size of the step filter moves every instance of time. Filter count: Number of filters we want to use. Padding: Now, we generally add padding surrounding the inputs so that feature map does not shrink.
- **Activation Function Layer:** In order to introduce non-linearity, this layer passes an activation function, for example, ReLU over the output.
- **Pooling Layer:** A pooling layer is used in between the convolutional layers, it reduces dimensional complexity. Applying pooling layers ensures that the model learns from the data and not memorize it. The chances of overfitting are reduced. One example is of max pooling layer. It finds the maximum of the pool and sends it to the next layer as we can see in the figure below.[3]

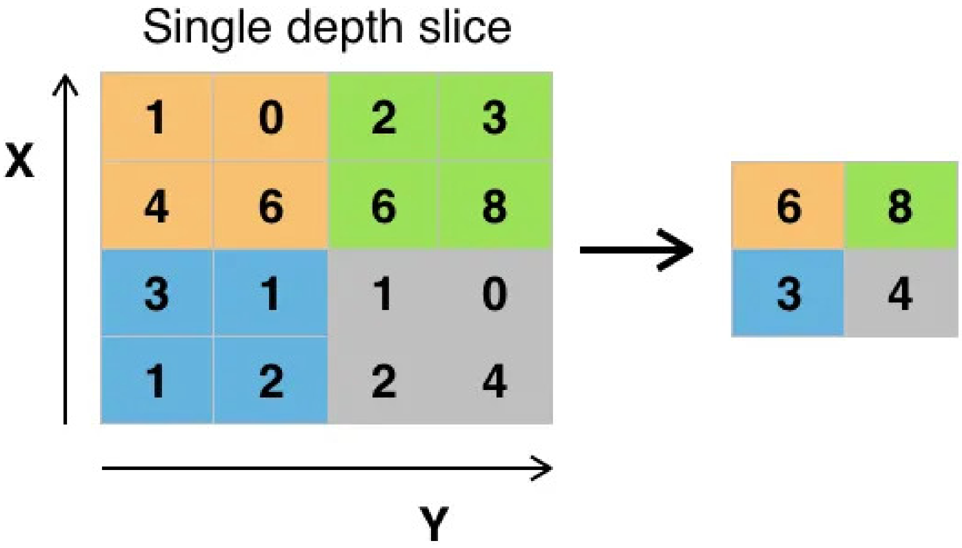


Figure 5.2: Max Pooling

- **Fully Connected Layer:** It flattens the output into a 1-D vector.
- **Output Layer:** The output layer produces the final output of the network, which is typically a probability distribution over the different classes in the classification task.

5.2 The Experiment

We have used Keras' Tokenizer API to tokenize the input headlines (without preprocessing) and convert them into sequences of integers. The tokenizer is configured to keep the 5000 most common words and is fitted on the training dataset. The sequences of integers are then padded with zeros or truncated to ensure that they all have a fixed length of 100.

We have defined and trained a CNN model using Keras API with an embedding layer followed by two 1D convolutional layers with **ReLU** activation and pooling layers, a dropout layer, and a dense layer with a **sigmoid** activation function for binary classification. The model is trained using the **binary cross-entropy** loss function and the **Adam optimize**r on the given training data for 10 epochs with a batch size of 64.

The performance metrics are given by:

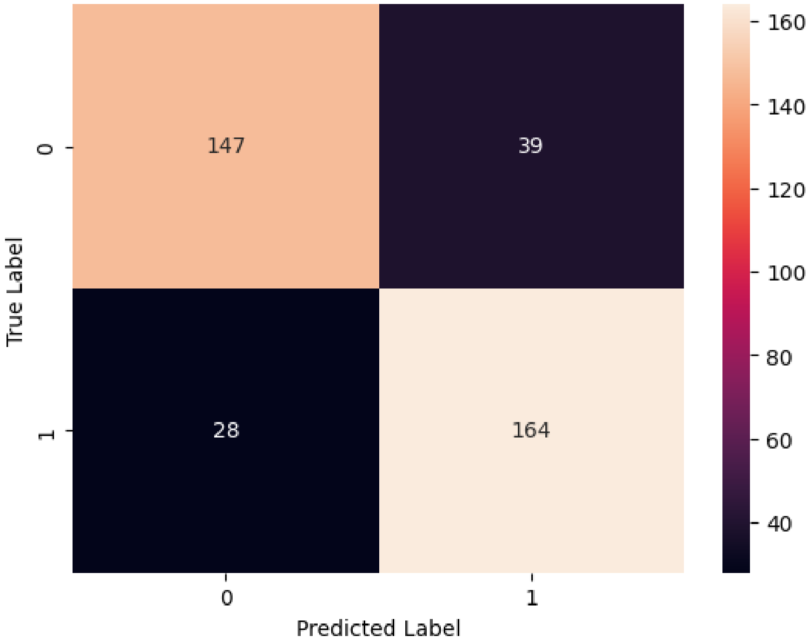


Figure 5.3: The Confusion Matrix using CNN

The accuracy is 0.82 with the support of 378.

	precision	**recall**	**f1-score**	**support**
0	0.84	0.79	0.81	186
1	0.81	0.85	0.83	192
macro avg	0.82	0.82	0.82	378
weighted avg	0.82	0.82	0.82	378

Table 5.1: Classification Report using CNN

Chapter 6

Conclusion

The following conclusions can be made from the observations

- Standard textual data cleaning procedures might not always work fine for all types of sentiment analysis.
- A lot of keywords, that are much relevant to the corresponding sentiment might get deleted when removing the Stopwords. Therefore, one always has to be careful with the removal.
- The process of Stemming or Lemmatization, even though used for increasing the efficiency and the accuracy, might actually turn out to be really troubling. As, words with different contexts might get mixed together resulting in lower accuracy.

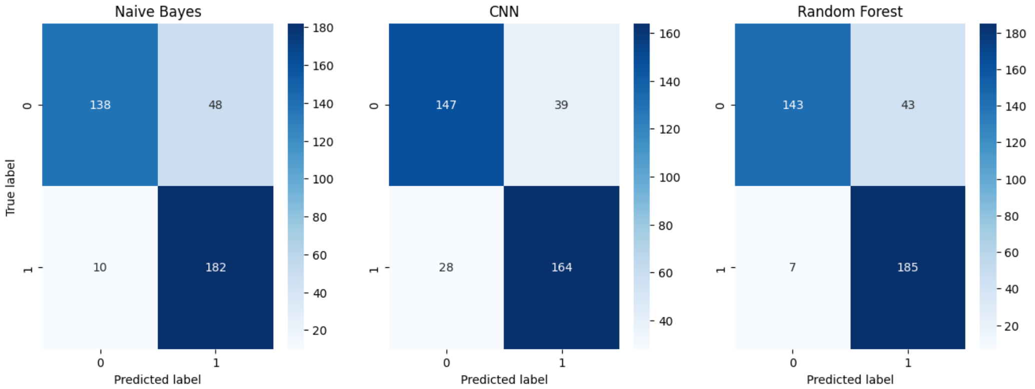


Figure 6.1: Confusion Matrices

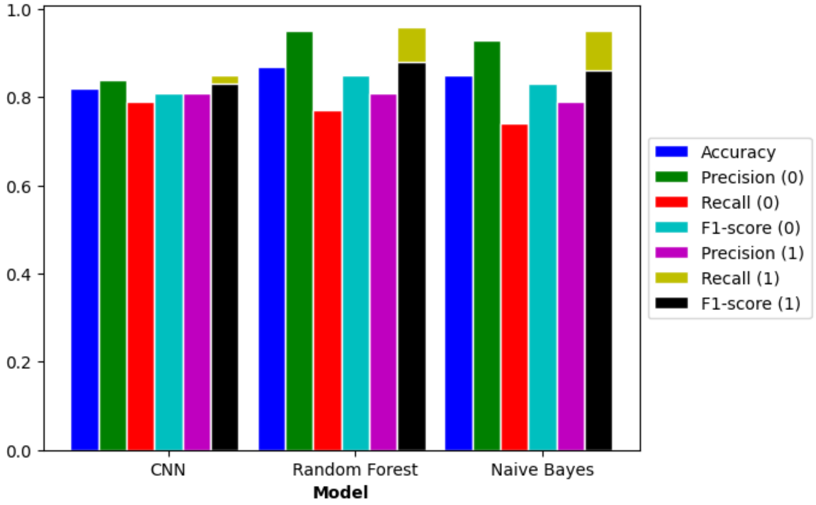


Figure 6.2: Comparing the Performances

- Naïve Bayes, CNN, Random Forest, all of these algorithms work pretty well in terms of text classification. When it comes to comparison, in this use case, Random Forest worked better than Naïve Bayes which worked better than CNN.

Bibliography

[1] K. Andrews and B. Rajiv. On some applications of eigenvalues of toeplitz matrices. *Journal of Mathematical Analysis and Applications*, 56(2):237–239, 2007.

[2] C. C. Chang. Algebraic analysis of many valued logics. *Transactions of American Mathematical Society*, 88:467–490, 1958.

[3] Vijay Choubey. Text classification using cnn. `https://medium.com/voice-tech-podcast/text-classification-using-cnn-9ade8155dfb9`, July 2020.

[4] Brunella Gerla. Automata over MV-algebras. In *ISMVL '04: Proceedings of the 34th International Symposium on Multiple-Valued Logic*, pages 49–54, Washington, DC, USA, 2004. IEEE Computer Society.

[5] G.H. Golub and C.F. Van Loan. *Matrix Computations*. Second Edition. The John Kopkins University Press, 1989.

[6] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Convolutional networks. In *Deep Learning*, chapter 9, pages 267–313. MIT Press, Cambridge, MA, 1st edition, 2016.

[7] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. Random forests. In *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, chapter 15, pages 587–604. Springer, New York, 2nd edition, 2009.

[8] Kevin P. Murphy. Naïve bayes. In *Machine Learning: A Probabilistic Perspective*, chapter 3, pages 77–94. MIT Press, Cambridge, MA, 1st edition, 2012.

[9] Rohit Sharma. Stock sentiment analysis using news headlines. `https://www.kaggle.com/code/rohit0906/stock-sentiment-analysis-using-news-headlines`, May 2021. Accessed on April 22, 2023.

[10] Harshdeep Singh. Understanding random forests. `https://medium.com/@harshdeepsingh_35448/understanding-random-forests-aa0ccecdbbbb`, March 2019.