ACA-SMAI

PROJECT ON MOVIE REVIEW DETECTION

PROJECT SYNOPSIS:

- The key motto of this project is to make the user predict whether a given movie is watchable or not.

- If we see any movie database like IMDB or ROTTEN TOMATOES we see each movie is given a rating and some reviews are highlighted as positive or negative or average.

- But how does they decide whether a given review is positive or negative.

- The basic idea is to read the text and select a basic set of keywords which provide a specific meaning when used in a text.

- This methodology is called SENTIMENT ANALYSIS and we will implement this in our project,

AIM: SENTIMENT ANALYSIS

- Movie Review Detection involves the algorithm of Sentiment Analysis.

- Sentiment analysis is the process of detecting positive or negative sentiment in text. It’s often used by businesses to detect sentiment in social data, gauge brand reputation, and understand customers.

- In this project we will look at the numerous movie reviews from the IMDB database and categorize it in the positive and negative sentiment.

- Our research paper states 4 basic algorithms and we will deploy those algorithms in the sentiment analysis of the movie reviews.

DATASET used:

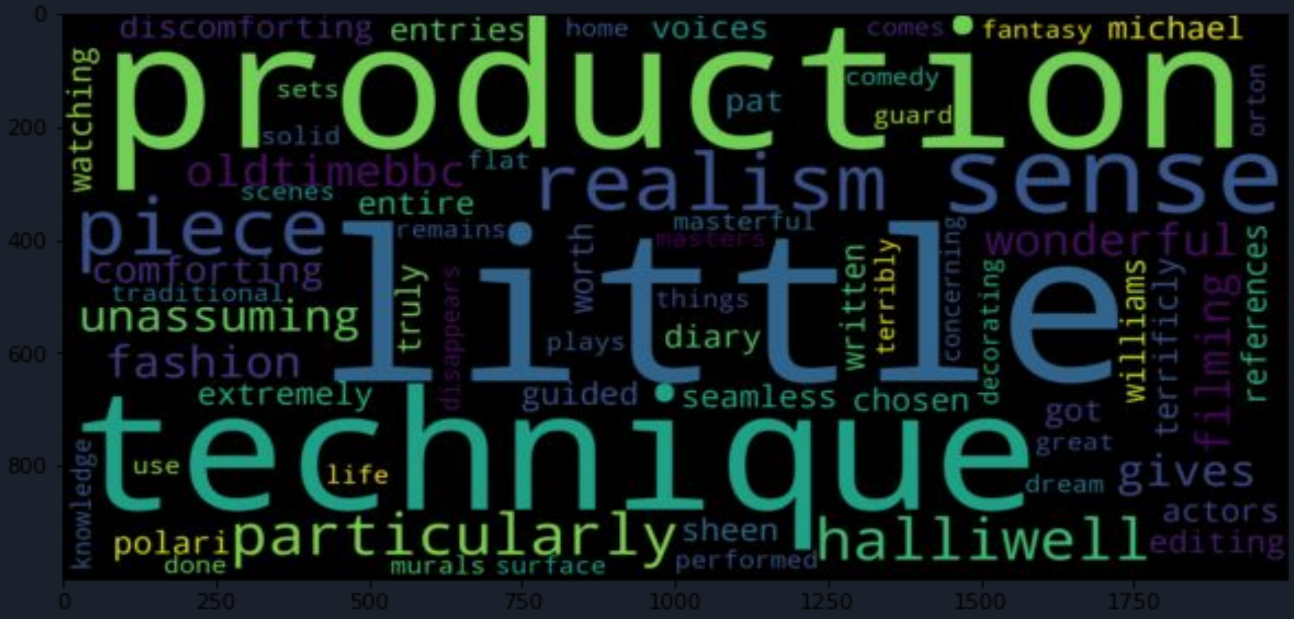
IMDB Dataset of 50K Movie Reviews

Paper dataset: The dataset contains 2000 movie review where 1000 is negative and remaining is positive.

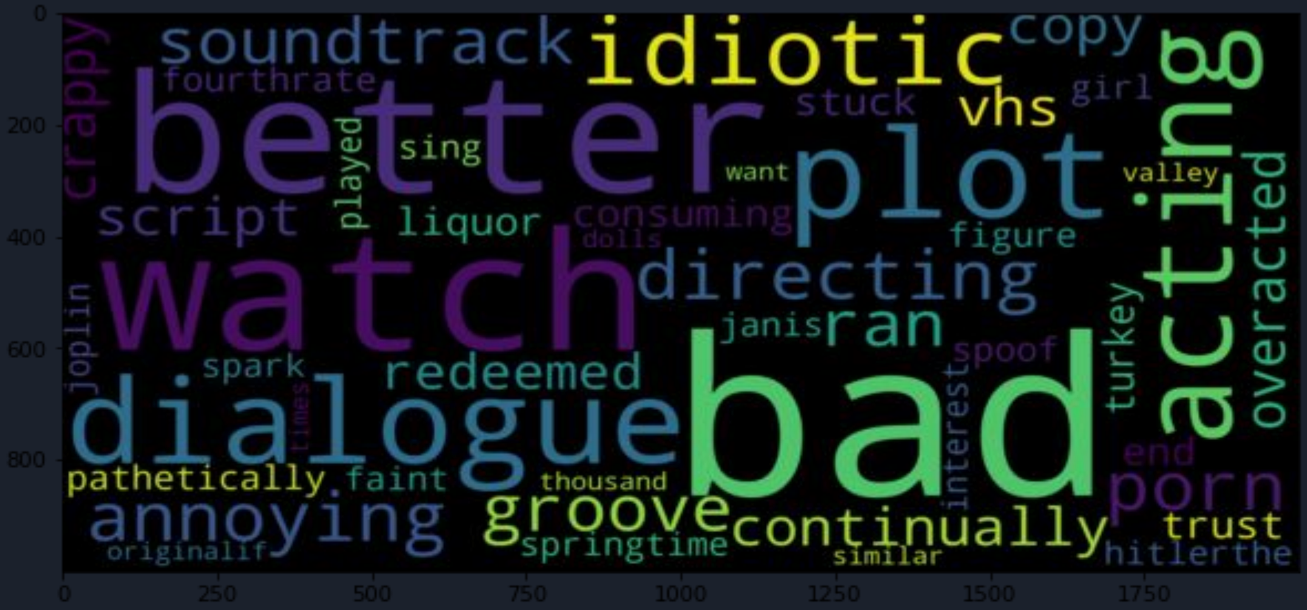
Movie Review 2000 dataset,

https://github.com/riyadatik/Sentiment-Analysis-on-Movie-Review-Data/blob/master/Data%20set.xlsx, 2019.

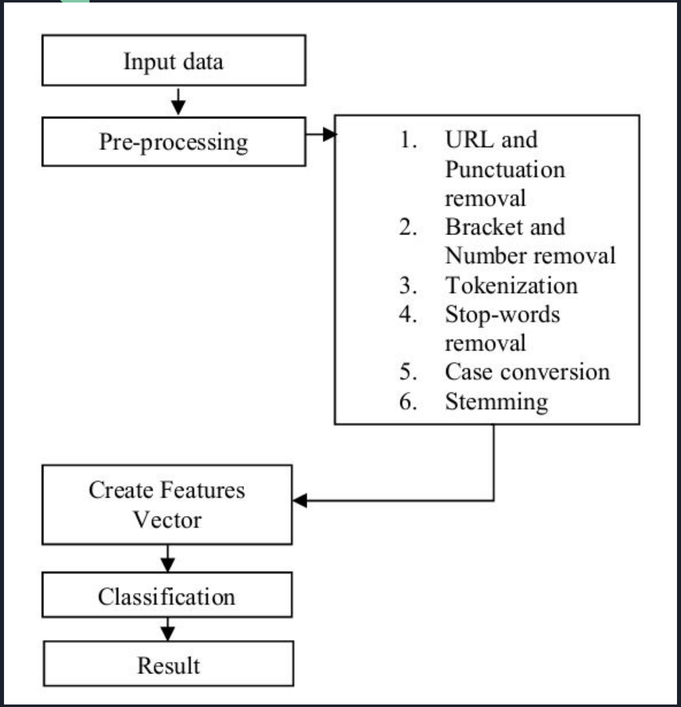
Word Cloud of Positive Review:



Word Cloud of Negative Review:



Theory:



PREPROCESSING THE DATA:

- Removing html strips and noise text
- Removing special characters
- Text stemming
- Removing stopwords

- **Step 1**:The first thing we are doing is checking whether any punctuation mark in reviews is an emoticon or not. If it is an emoticon, we are replacing it with the sentiment associated with that emoticon.

- **Step 2**: Removing tag   </br> which is not necessary, replacing the emoticons with words and then we are removing the unnecessary punctuation marks. Then we are splitting,converting the sentences to lowercase, lemmatizing it and then combining the words to form sentences.

Step 3:

- Bag of Words is a method to extract features from text documents. These features can be used for training machine learning algorithms. It creates a vocabulary of all the unique words occurring in all the documents in the training set

- Convert in into bag of words. The Bag of Words model learns a vocabulary from all of the documents, then models each document by counting the number of times each word appears. It is used to convert text documents to numerical vectors called bag of words.

Step 4:

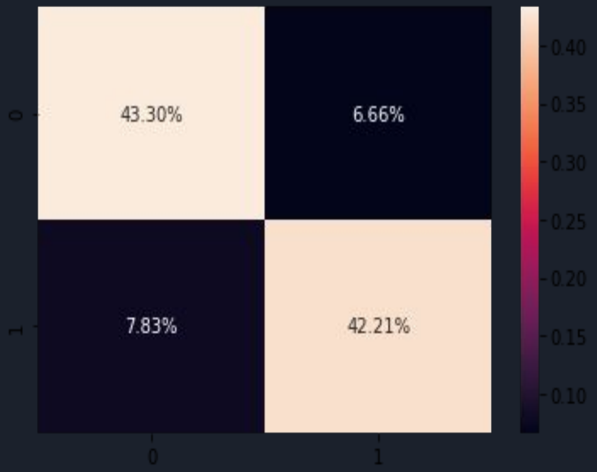
- We will now have numeric training features from the Bag of Words and the original sentiment labels for each feature vector, so we can now train our model and find out how it performs.
- Labeling the sentiment text: 0 or 1.

Comparing several models

Models to be used: We’ve used bag of words model and word2vec model

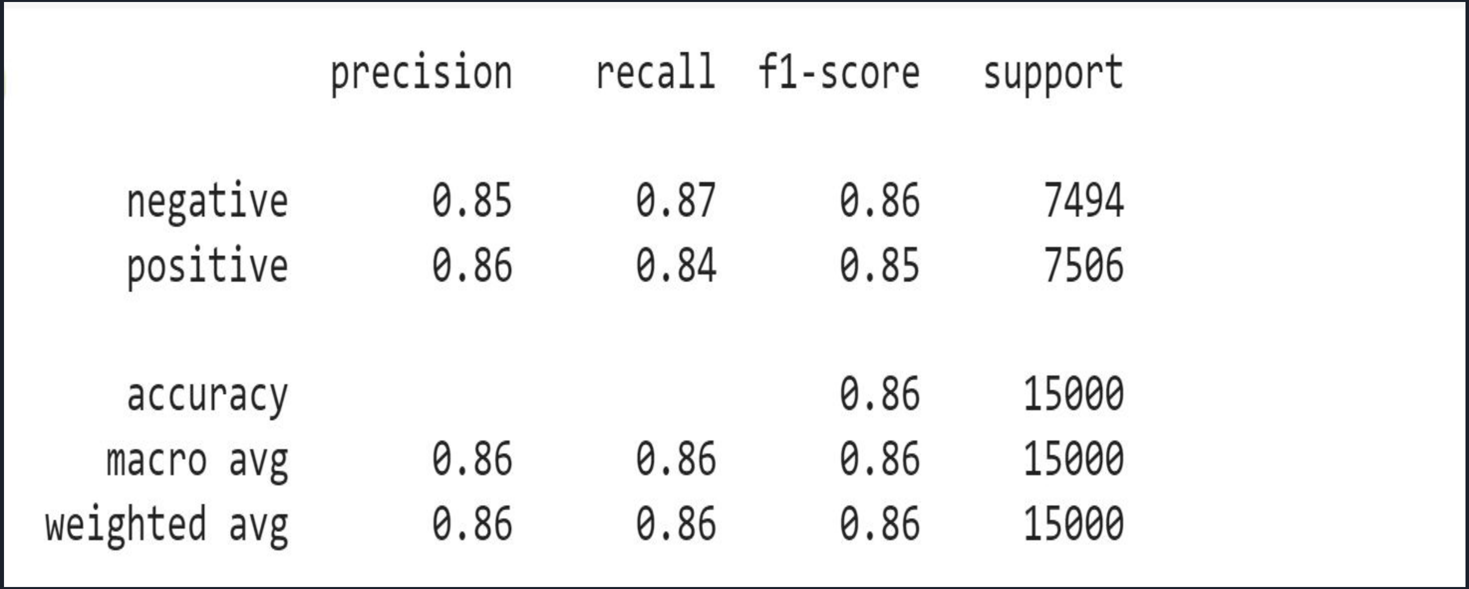
- Multinomial Naive bayes
- SVM
- Maximum Entropy
- Decision tree
- Gaussian Naive Bayes
- Convolutional Neural Network (not in research paper)

Multinomial Naive Bayes:



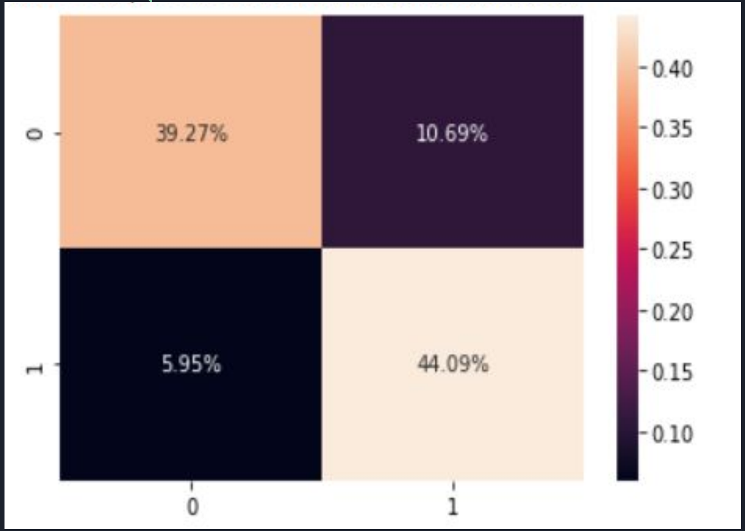
- Multinomial Naive Bayes is a probabilistic learning method.
- The algorithm is based on the Bayes theorem and predicts the tag of a text such as a piece of email or newspaper article.
- It calculates the probability of each tag for a given sample and then gives the tag with the highest probability as output.
- The Accuracy obtained on Validation data set using MNB is 85.50%.

Multinomial Naive Bayes Classification Report:



```
              precision    recall  f1-score   support

    negative       0.85      0.87      0.86      7494
    positive       0.86      0.84      0.85      7506

    accuracy                           0.86     15000
   macro avg       0.86      0.86      0.86     15000
weighted avg       0.86      0.86      0.86     15000
```

Support Vector Machine (SVM)

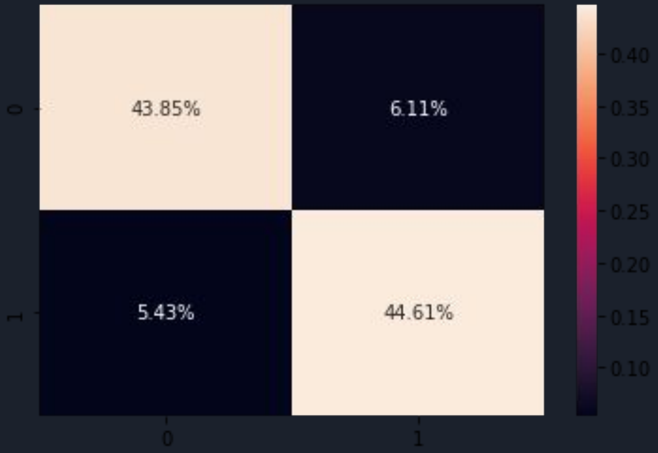


- We’ve used the polynomial kernel with C = 0.01.
- SVM performs classification by finding the hyper-plane that differentiate the classes we plotted in n-dimensional space.
- The SVM performed fairly well since the data was balanced. Accuracy was 83.36.
- From the heatmap we can see that 39.27% times the SVM predicted correctly on the positive sentiment data.
- 44.09% times it predicted correctly when the sentiment was negative.

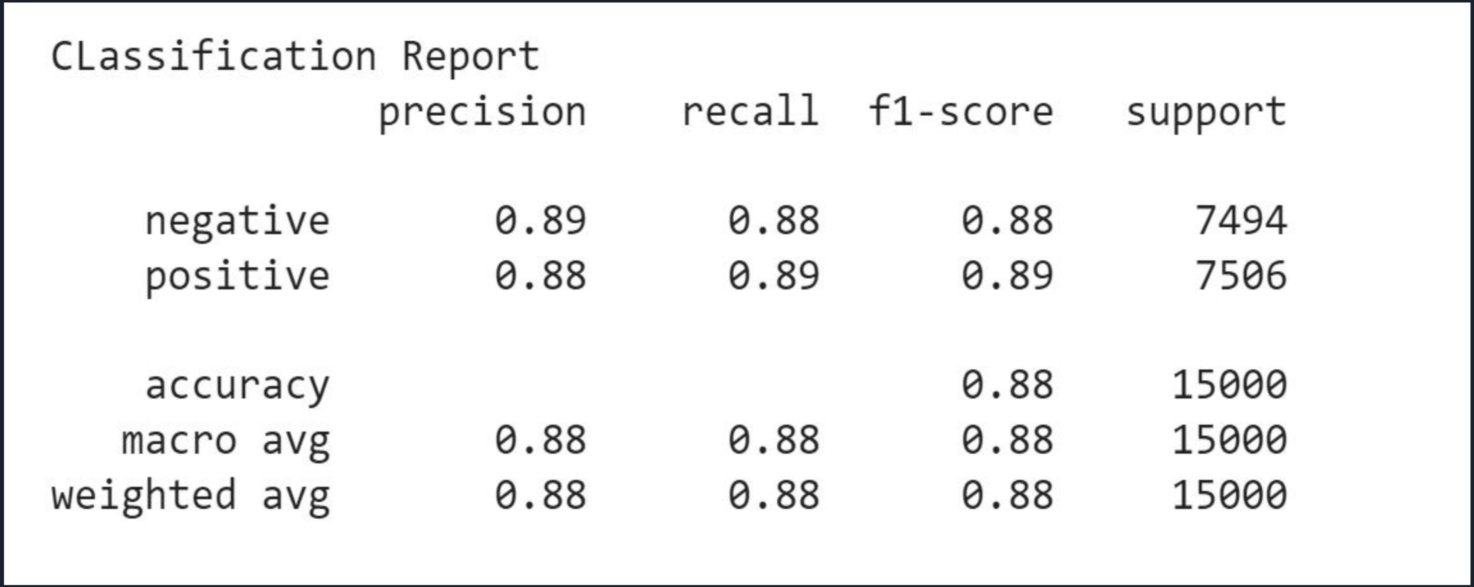
Maximum Entropy

- Maximum Entropy Text classification means: start with least informative weights (priors) and optimize to find weights that maximize the likelihood of the data, the P(D). Essentially, it's the EM algorithm.

- The Accuracy obtained on Validation data set using ME is 88.46%.

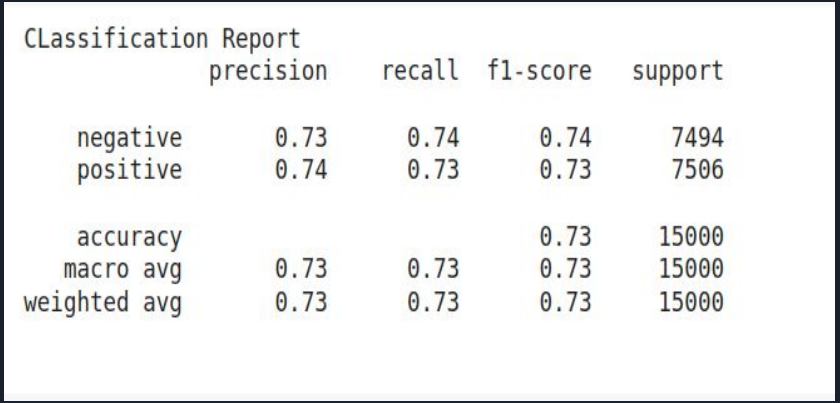


Maximum Entropy Classification Report:



```
CLassification Report
              precision    recall  f1-score   support

    negative       0.89      0.88      0.88      7494
    positive       0.88      0.89      0.89      7506

    accuracy                           0.88     15000
   macro avg       0.88      0.88      0.88     15000
weighted avg       0.88      0.88      0.88     15000
```

Decision Tree



```
CLassification Report
              precision    recall  f1-score   support

    negative       0.73      0.74      0.74      7494
    positive       0.74      0.73      0.73      7506

    accuracy                           0.73     15000
   macro avg       0.73      0.73      0.73     15000
weighted avg       0.73      0.73      0.73     15000
```

- Decision tree needs several key nodes, while it's hard to find "several key tokens" for text classification, and random forest works bad for high sparse dimensions.

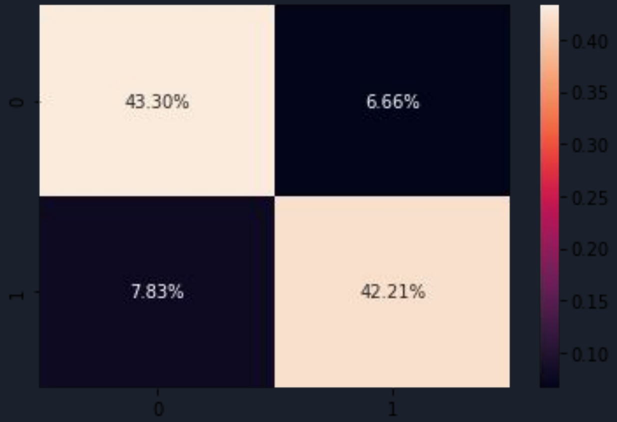
- Accuracy for decision tree achieved is 73%

Gaussian Naive Bayes:

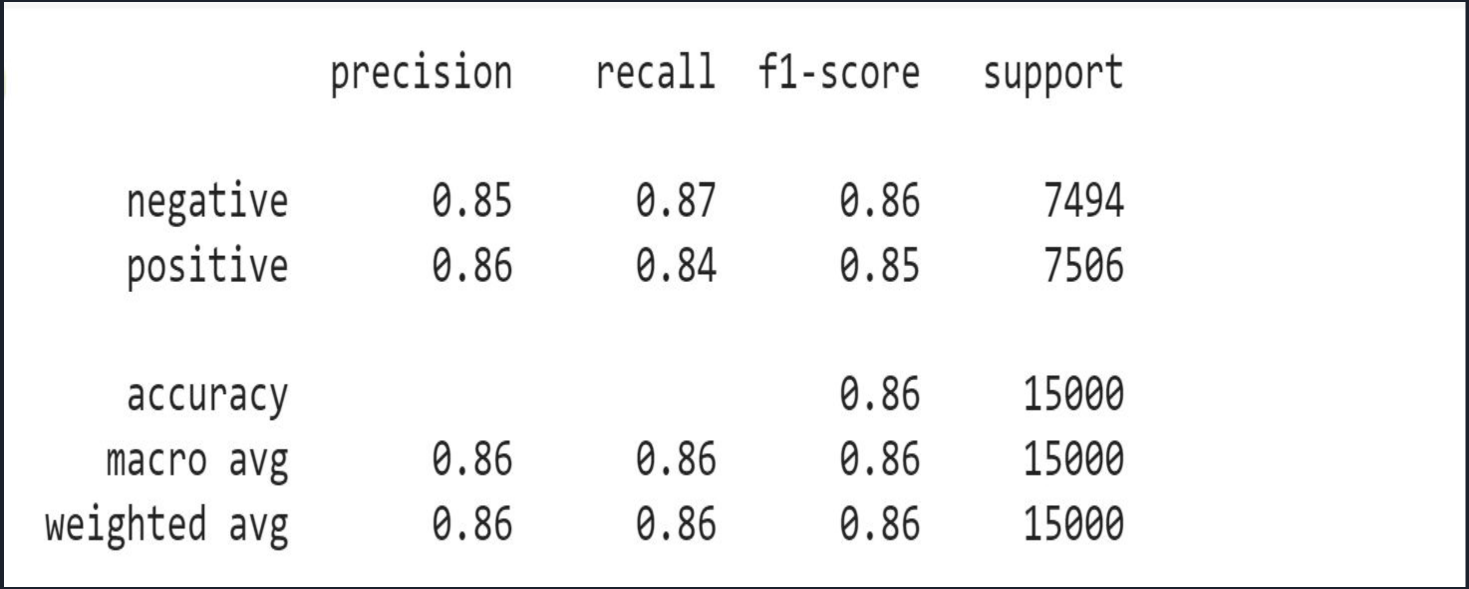
- Binomial Naive Bayes uses the binomial theorem
  while assuming that there is no relationship
  between different features.

- It uses the formula:
  Posterior = likelihood*proposition/evidence

- Gaussian Naive Bayes uses gaussian distribution to classify the model hence we first send the data in dense form.



Classification Report of Gaussian Naive Bayes:

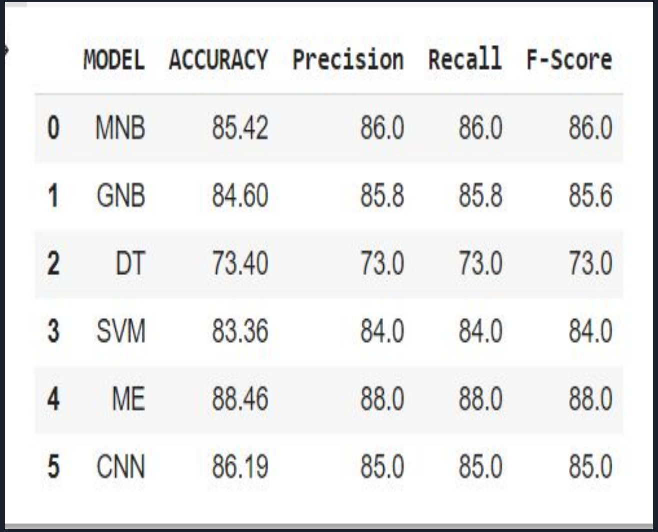


```
              precision    recall  f1-score   support

    negative       0.85      0.87      0.86      7494
    positive       0.86      0.84      0.85      7506

    accuracy                           0.86     15000
   macro avg       0.86      0.86      0.86     15000
weighted avg       0.86      0.86      0.86     15000
```

Convolutional Neural Network

- This model was not in the Research paper.
- We implemented a one layer convolutional neural network.
- Like a two-dimensional convolutional layer, a one-dimensional convolutional layer uses a one-dimensional cross-correlation operation.
- The text was handled using the word2vec model.
- In one-dimensional cross-correlation model, the convolutional window starts from leftmost side of input array and slides input array from left to right successively.

Comparing models(Using Bag of Words):

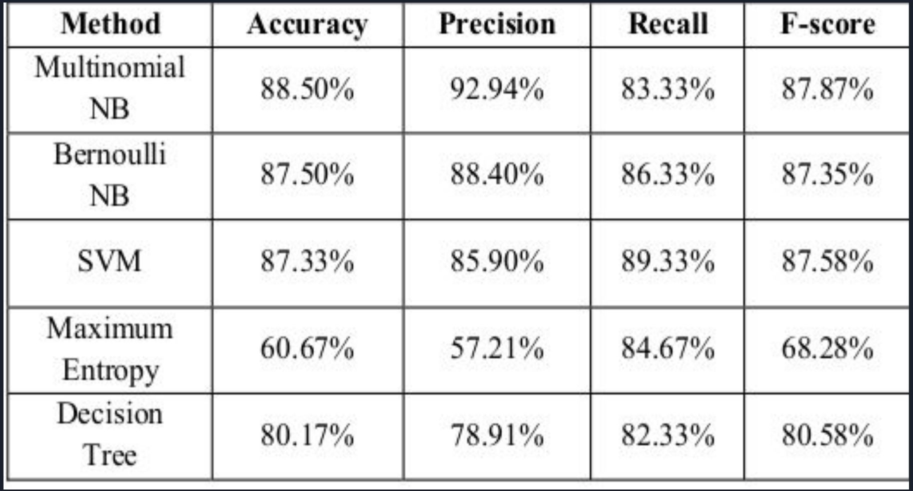


	MODEL	ACCURACY	Precision	Recall	F-Score
0	MNB	85.42	86.0	86.0	86.0
1	GNB	84.60	85.8	85.8	85.6
2	DT	73.40	73.0	73.0	73.0
3	SVM	83.36	84.0	84.0	84.0
4	ME	88.46	88.0	88.0	88.0
5	CNN	86.19	85.0	85.0	85.0

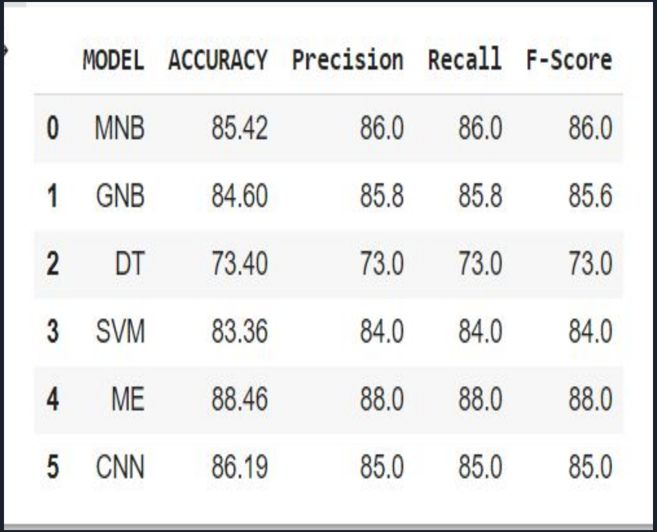
- We used to bag of words model to compute accuracies between 5 models presented in the given table.
- We found out that ME performed the best with an overall accuracy of **85.09%.**
- We further tested the word2vec model on the given models. Additionally we ran the CNN.

Comparing Models(BOW):

Comparing research paper vs our results

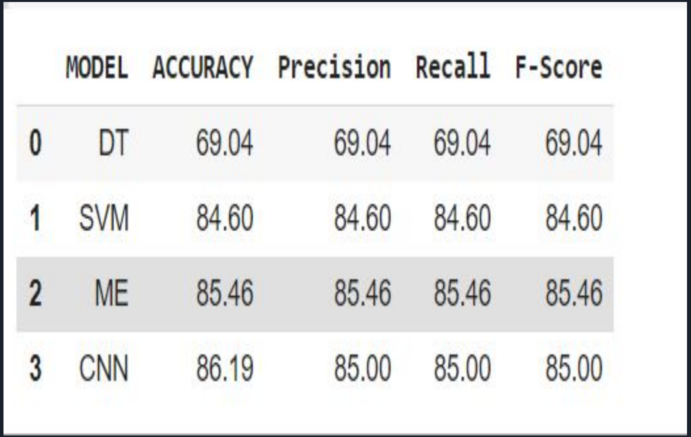


Method	Accuracy	Precision	Recall	F-score
Multinomial NB	88.50%	92.94%	83.33%	87.87%
Bernoulli NB	87.50%	88.40%	86.33%	87.35%
SVM	87.33%	85.90%	89.33%	87.58%
Maximum Entropy	60.67%	57.21%	84.67%	68.28%
Decision Tree	80.17%	78.91%	82.33%	80.58%



	MODEL	ACCURACY	Precision	Recall	F-Score
0	MNB	85.42	86.0	86.0	86.0
1	GNB	84.60	85.8	85.8	85.6
2	DT	73.40	73.0	73.0	73.0
3	SVM	83.36	84.0	84.0	84.0
4	ME	88.46	88.0	88.0	88.0
5	CNN	86.19	85.0	85.0	85.0

Accuracy Summary using Word2vec:



	MODEL	ACCURACY	Precision	Recall	F-Score
0	DT	69.04	69.04	69.04	69.04
1	SVM	84.60	84.60	84.60	84.60
2	ME	85.46	85.46	85.46	85.46
3	CNN	86.19	85.00	85.00	85.00

- Word2Vec is a two layer neural network which converts the text into a set of vectors that represents the word. This vector is now used by the model.
- Since the vector element can be negative Word2Vec doesn’t work on Gaussian Naive Bayes and Multinomial Naive Bayes as both are probabilistic models

Individual Contribution

Member Name	Topics
Agrima Singh	Decision Tree, SVM
Ashutosh Ranjan	CNN, GNB
Chirag Shilwant	MNB, Maximum Entropy