Comparitive Analysis of Text Classification Methods

(**Algorithm Track – Text Data Mining**)

Ankita Arvind Deshmukh[1]
Department of Computer Engineering
San Jose State University
San Jose, United States
ankitaarvind.deshmukh@sjsu.edu

Rutik Sanjay Sangle[2]
Department of Computer Engineering
San Jose State University
San Jose, United States
rutiksanjay.sangle@sjsu.edu

Teja Sree Goli[3]
Department of Computer Engineering
San Jose State University
San Jose, United States
tejasree.goli@sjsu.edu

***Abstract*—Users may choose whether a movie is worth their time by reading movie reviews. Users can save time by not reading all the reviews for a movie by using a summary of all the reviews to make this decision. Critics frequently publish reviews and ratings on websites that assess movies, which aids audiences in deciding whether to watch the film. Based on their reviews, sentiment analysis can infer the attitude of the critics. This project aims at classifying the user sentiments as positive and negative using various machine learning and deep learning algorithms and performing comparative performance analysis to decide which model performed better using various evaluation metrics.**

Keywords—classification, KNN, Logistic Regression, Naïve Bayes, SVM, Random Forest, LSTM, CNN

I. INTRODUCTION

Sentiment analysis aims to extract subjective data from textual reviews. The study of sentiment is strongly related to text mining and natural language processing. It can be used to assess the reviewer's perspective on particular subjects or the overall polarity of the review. A movie review's sentiment can be analyzed to determine if it is positive or negative, and this affects the movie's overall rating. Therefore, since the machine learns through training and evaluating the data, it is possible to automate the process of determining whether a review is positive or negative. To examine which machine learning and deep learning model performs better and produce more accurate results, this project seeks to rate reviews using several models. A data mining technique called classification categorizes a set of data in order to help in more accurate predictions and analysis. Sentiment analysis findings will be compared using Logistic Regression, K-Nearest Neighbors, Naive Bayes, Support Vector Machine, Random Forest, Convolutional Neural Network and LSTM. Even though the proposed work suggested binary classification only, we went ahead and performed multi-class classification to predict the rating on a scale of 1 to 10.

II. DATASET

For this project, we are using the Large Movie Review Dataset [1] which is a dataset for binary sentiment classification providing 50,000 highly polar movie reviews for training and testing purposes along with some unsupervised data. The dataset is divided into two subsets containing 25,000 examples each for training and testing. We tried to redistribute the examples as 40,000 for training and 10,000 for testing since we found the existing distribution is sub-optimal and might lead to underfitting.

III. DATA PREPROCESSING

As the first step in data pre-processing, we created a pandas data frame based on the review files in text format. These files are named with a unique identifier along with their rating. As proposed by the guidelines to use this dataset, we labelled the reviews as positive if they receive a rating >= 7 and negative if the rating <= 4. Since we are talking about the binary classification here, we do not have files with 5 <= rating >= 6 with neutral reviews. We merged the training and testing dataset including positive and negative reviews and shuffled them so that the models do not learn anything invalid based on the order of sentiments. We appended the reviews in natural language to this data frame and used them for further text-mining processes.

A. Removing Punctuations

Removal of punctuations is a commonly used pre-processing step as they do not play any significant role in data analysis. All the punctuations in the text are listed and changed to an empty string using python's string class [3].

B. Removing Stopwords

Stop words are the most frequent words used in all languages and do not offer much information to the text. Examples include articles, prepositions, pronouns, conjunctions, etc. Removal of stop words definitely reduces the dataset size and thus reduces the training time due to the fewer tokens involved in the training. NLTK (Natural Language Toolkit) libraries are used for the removal of stop words [3].

C. Stemming

Stemming is the process of producing morphological variants of a base word. For this project, we are using a stemming tool called Porter's Algorithm from the NLTK library over Snowball, Lancaster and Regexp stemming algorithms because of its speed and simplicity. Porter Stemmer employs five phases of word reduction, each with its own set of mapping rules [3].

D. TF-IDF Vectorization

Term Frequency Inverse Document Frequency or TF-IDF is useful in extracting keywords from the text. The words that received the highest scores are the ones that are most relevant to the document, making them suitable to be used as keywords. The term frequency of a word in a document is multiplied by the inverse document frequency of the word over a group of documents to determine a word's TF-IDF, which will be close to zero if the word is widely used and

appears in numerous documents. If not, it will go close to 1 [3].

E. N-Gram Analysis

Text mining and natural language processing frequently make use of text n-grams. In essence, it is a series of words that simultaneously appear in a single window. An N-gram language model predicts the probability of a given N-gram within any sequence of words in the language. A good N-gram model can predict the next word in the sentence [5]. We performed Unigram, bigram and trigram analysis for the given dataset using the n-grams function from the NLTK library.

IV. Data Visualization

To get a better understanding of our dataset we carried out some data visualization and some of the most interesting observations are shown below.

The following N-grams (unigram – Fig1, bigram – Fig2 and trigram – Fig3 respectively) show the words which are most commonly used together [7]:

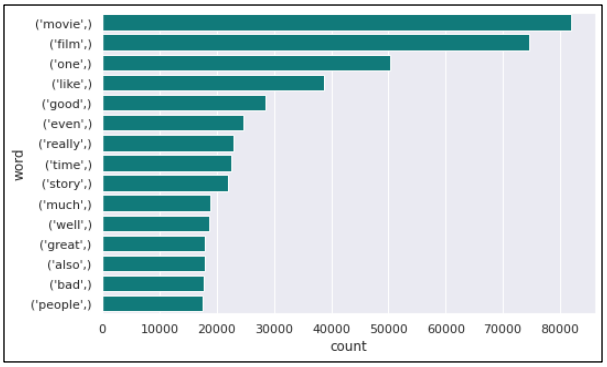

Fig1

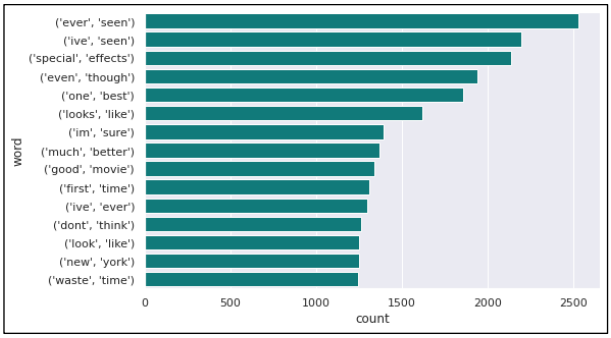

Fig2

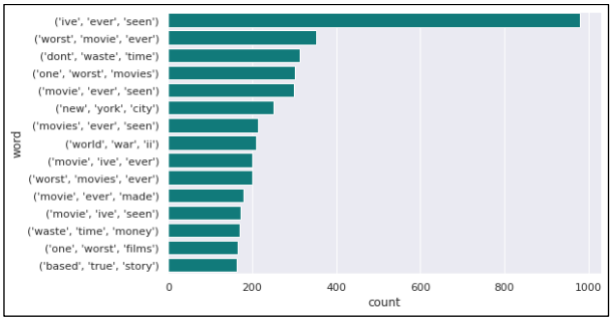

Fig3

Following word cloud shows the most frequently used keywords from the vectorized reviews (Fig4).

Fig4

The word cloud above shows the words which are the most appeared words in the reviews given by people. It seems like speed, idea, story, length or original concepts make people either hate or love the movie. It also looks like many people watch movies based on their friends' recommendations. Slashers seemed to be one of the most popular genres in the review dataset.

The boxplot below represents the distribution of review length given by critiques where the average word count is approximately 828 words (Fig5)[4].

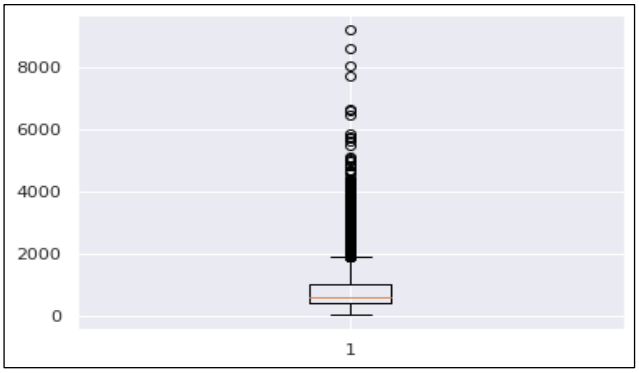

Fig5

V. Methods

For the binary classification task, we applied the following algorithms:

A. Logistic regression

A statistical model called logistic regression employs a logistic function to predict binary classes. Here, the dependent and independent variables in the logistic model have values of either 0 or 1, and the logistic function or logistic curve is a sigmoid curve. As a result, logistic regression is a popular and practical regression technique for predicting binary classes, or two classes. The link between one dependent variable and independent factors is estimated using the logistic model. The logistic function is the function that converts log odds into probability.

B. K-Nearest Neighbors

A non-parametric supervised machine learning approach called K-Nearest Neighbor (KNN) is utilized for classification and regression. The item is categorized for the purpose of text categorization based on votes from the object's k closest neighbors. The class of the item is determined by the class with the greatest number of neighbor votes. A vote is given to the k nearest neighbors, which is

determined using many distance measurements, including Minkowski, Euclidean, and others.

C. *Naïve Bayes*

A group of supervised learning algorithms known as naive Bayes methods utilize Bayes' theorem with the "naive" assumption that each pair of features is conditionally independent given the value of the class variable. In comparison to more complex techniques, naive Bayes learners and classifiers can operate at lightning speeds. Each distribution can be individually estimated as a one-dimensional distribution due to the decoupling of the class conditional feature distributions. This in turn aids in resolving issues brought on by the "curse of dimensionality."

D. *Support Vector Machine*

A straightforward supervised machine approach called the Support Vector Machine (SVM) is employed for regression and classification tasks. The data is interpreted geometrically by SVM. It is a binary classifier by default. To maximize the distance between the two categories, it maps the data points in space. SVM searches for an N-1 dimensional hyperplane to divide its data points, which are N-dimensional vectors. The term for this is a linear classifier. This criterion could be satisfied by many hyperplanes. The hyperplane that provides the greatest margin, or distance, between the two groups is therefore the best hyperplane. It is known as the largest margin hyperplane as a result.

E. *Random Forest*

A supervised machine learning technique known as the Random Forest or Random Decision Forest is used for classification, regression, and other tasks using decision trees. A randomly chosen portion of the training data is used by the Random Forest classifier to generate a collection of decision trees. It simply consists of a collection of decision trees from a randomly chosen subset of the training set, which are subsequently used to decide the final prediction.

F. *Convolutional Neural Network*

The CNN model uses a multi-layered feed-forward neural network that is created by sequentially stacking many hidden layers on top of one another. CNN can observe and pick up on hierarchical aspects because to this design. The pooling layer, convolutional layer, and fully connected layer are the three layers that make up CNN's network structure. In order to extract the feature variables, data convolution is used. In order to learn patterns at a certain point in a sentence, 1D convolution layers are used. These layers are then used to recognize patterns at other positions. A mask is used in this phase to both improve and improve upon the convolution layer's output. Each feature map patch is subjected to Max-Pooling in order to extract the maximum value and discard the rest. As a result, the inputs to the subsequent layer are reduced. The final output's word count has reduced by 50% thanks to the max-pooling layer. There are three levels in use. sub-matrices of order 2 make up the matrix.

G. *Long Short-Term Memory*

The LSTM has a sequential model, just like the Recurrent Neural Network (RNN). RNN and LSTM vary in that the former features a feature known as "cell memory" that provides additional signal information from one time step to the next. The vanishing gradient issue is addressed by LSTM employing the gate mechanism. The LSTM is made up of the Forget Gate f (a NN with a sigmoid as the activation function), the Candidate Layer g (a NN with a tanh as the activation function), the Input Gate I (a NN with a sigmoid as the activation function), the Output Gate O (a NN with a sigmoid as the activation function), the Hidden State H (a vector), and the Memory State C (vector) [6].

VI. EVALUATION

Building machine learning models is based on the premise of constructive feedback. A model is created, metrics are used to provide feedback, modifications are made using hyperparameter tuning, and the process is repeated until the desired accuracy is reached. The effectiveness of a model is explained by evaluation metrics. The ability of evaluation metrics to distinguish between different model outcomes is a crucial feature. For the performance analysis of our project, we used multiple evaluation metrics such as accuracy score, precision, recall, F-1 score, confusion matrix and AUC curve. This section presents the confusion matrix and AUC curve for each algorithm used for binary classification. The task of comparing the other parameters we utilized for evaluation is addressed in the subsequent sections. An N X N matrix, where N is the number of expected classes, is a confusion matrix. Recall, precision, specificity, accuracy, and—most importantly—AUC-ROC curves may all be measured with great success with this tool. The True Positive Rate (TPR) and False Positive Rate (FPR) are represented on the y-axis and x-axis, respectively, of the ROC curve. An excellent model will have an AUC near to 1, which denotes a high level of separability.

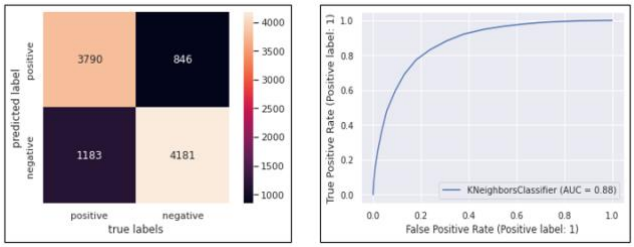

Fig5 – KNN

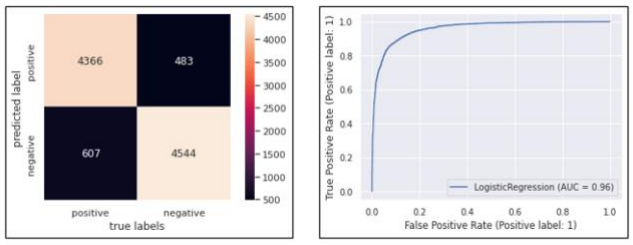

Fig6 – Logistic Regression

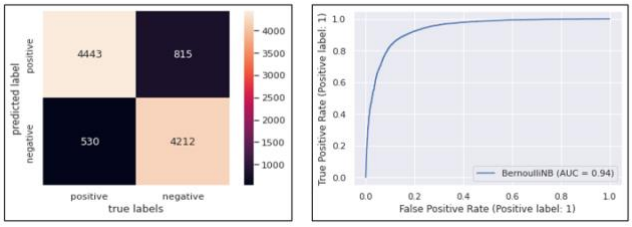

Fig7 – Naïve Bayes

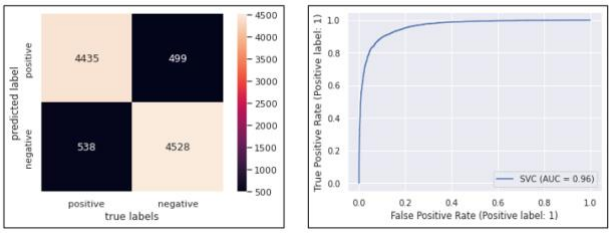

Fig8 – SVM

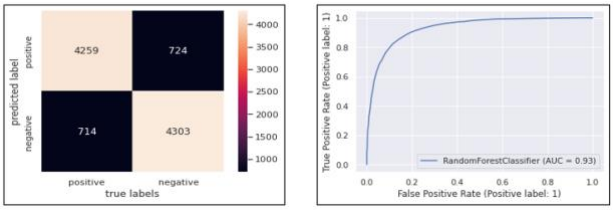

Fig9 – Random Forest

VII. COMPARISON

Since every evaluation metric offers different kind of pros and cons for a classification algorithm, we have tried to compare and display them in tabular (Fig10) as well as graphical format below. An evaluation metric called accuracy allows you to quantify the total number of correct predictions made by a model. The accuracy calculation is as follows:

$$Accuracy = \frac{TrueNegatives + TruePositive}{TruePositive + FalsePositive + TrueNegative + FalseNegative}$$

Algorithm	Accuracy	Precision	Recall	F1-Score
Naive Bayes	0.86	0.88	0.84	0.86
Random Forest	0.85	0.86	0.85	0.86
KNN	0.79	0.78	0.83	0.80
Logistic Regression	0.89	0.88	0.90	0.89
SVM	0.89	0.89	0.90	0.89

Fig10 – Comparison

The accuracy comparison graph in Fig11 shows that the SVM Classifier has the highest accuracy of 0.89 among all the state-of-the-art models. Random Forest, Logistic Regression and Naïve Bayes models have the accuracy in the similar ranges while KNN has the lowest accuracy of 0.79. The accuracy metric helps in understanding if a model is being trained properly.

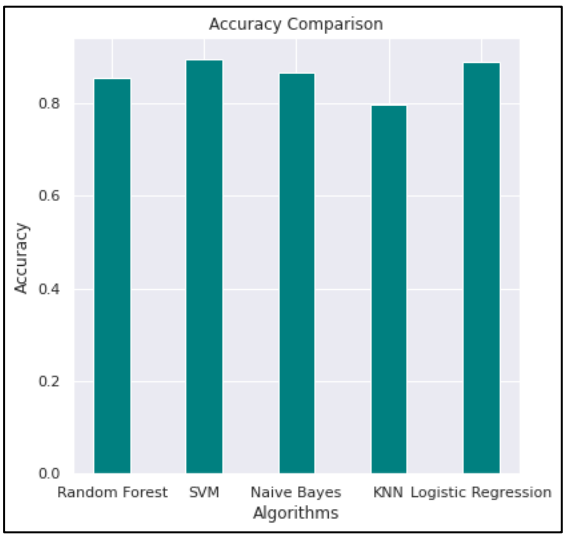

Fig11 – Accuracy Comparison

The bar graph of Fig12 shows the precision value comparison of different machine learning models that have been implemented.

$$precision = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

The Precision metric represents the accurate positive predictions among all the positive predictions made.

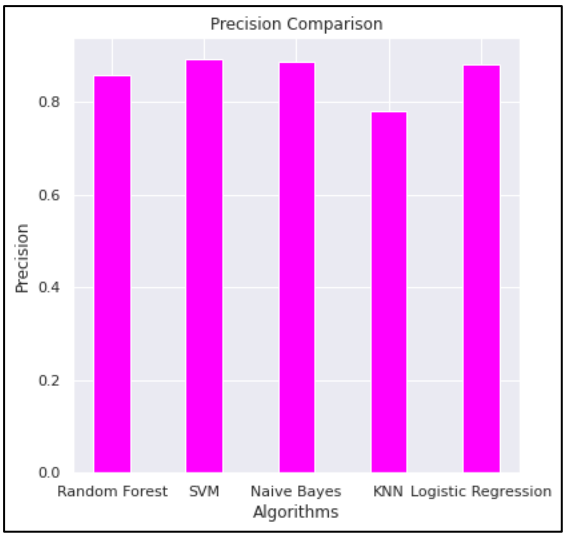

Fig12 – Precision Comparison

The Recall is a metric that is used to measure the model's ability to predict the positive samples.

$$recall = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

The below Fig13 shows the recall metric comparison of all the implemented machine learning.

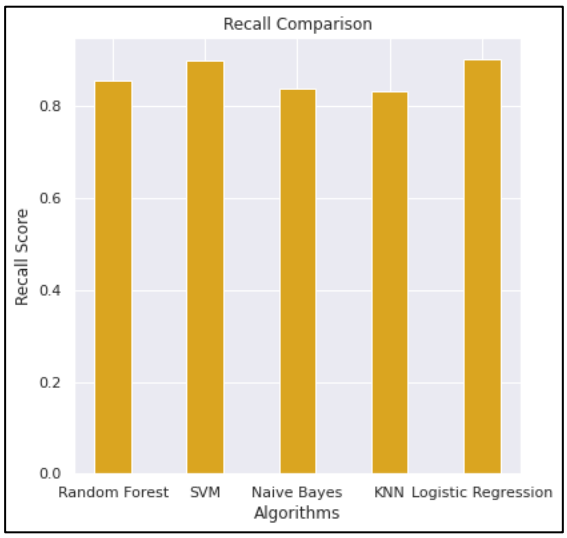

Fig13 – Recall Comparison

The F1-score metric is used to evaluate the performance of the model. The score is obtained by calculating the harmonic mean of recall and precision. Higher the value of F1-score, the better is the model's performance.

$$F1\ Score = 2 \times \frac{recall \times precision}{recall + precision}$$

The below figure (Fig14) shows the F1-score comparison of 5 different machine learning models. It can be observed that SVM and logistic regression have the highest F1 score while KNN classifier has the lowest.

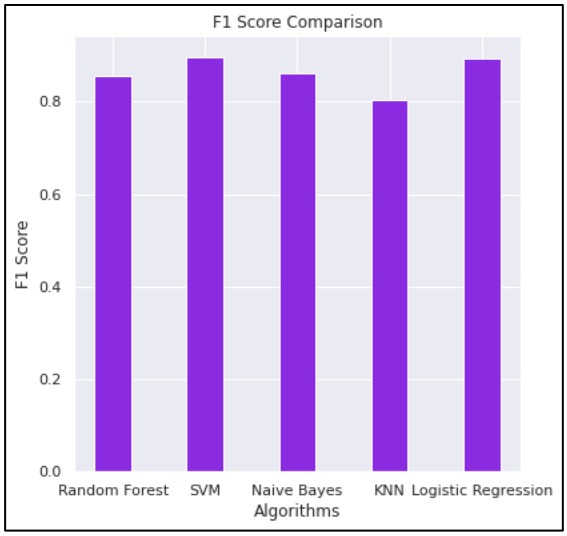

Fig14 – F-1 Score Comparison

VIII. Additional Work

In addition to analyzing the implemented state-of-the-art machine learning models by comparing different error metrics, we have also tried comparing the training and prediction times of the models to observe if any correlation exists between the training time and performance of the models. The comparison graph can be seen in the below graph (Fig15).

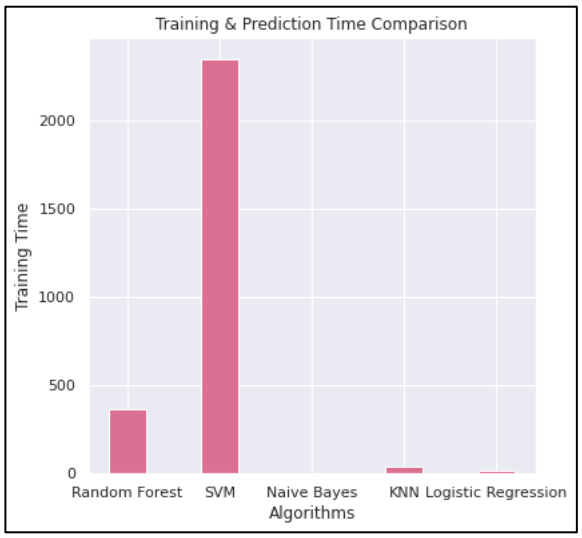

Fig15 – Speed Comparison

From the graph above, it can be clearly observed that SVM takes too long to train compared to all the other algorithms especially the Naïve bayes classifier which took only 0.15 seconds to train and predict.

Multiclass Classification

Multiclass classification consists of more than two classes unlike general binary classification models. In multiclass classification, we train a classifier model using our training data and use this classifier for classifying the new examples. We have implemented Random Forest, SVM and Multinomial Naïve Bayes models for the multiclass classification and compared the accuracies of these models as shown in the below figure (Fig16).

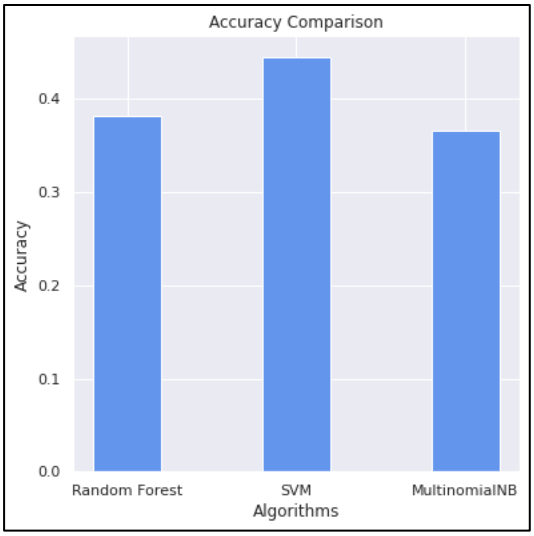

Fig16 – Accuracy Comparison

Neural Networks Comparison

In order to create a simple convolutional neural network model, we attempted using CNN, which contains a convolutional layer to extract information from a larger piece of text. By sequentially stacking numerous hidden

layers on top of one another, the CNN model creates its multi-layered feed forward neural network. In this way, hierarchical features can be observed and learned by CNN. For the model fitting, 10 epochs are used with a batch size of 256. The pooling layer, convolutional layer, and fully connected layer are the three layers that make up CNN's network structure. In order to learn patterns at a certain point in a sentence, 1D convolution layers are used. These layers are then used to recognize patterns at other positions. Number of layers used is 3 and the matrix is divided into sub-matrices of order 2. In the implemented LSTM model, LSTM layer was applied on the output resulting from the max-pooling layer [7].
The model summary of CNN and LSTM is shown below in fig16 and fig17 respectively.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 150, 50)           9182200

 conv1d (Conv1D)             (None, 150, 32)           4832

 max_pooling1d (MaxPooling1D  (None, 75, 32)           0
 )

 conv1d_1 (Conv1D)           (None, 75, 32)            3104

 max_pooling1d_1 (MaxPooling  (None, 37, 32)           0
 1D)

 conv1d_2 (Conv1D)           (None, 37, 32)            3104

 max_pooling1d_2 (MaxPooling  (None, 18, 32)           0
 1D)

 dense (Dense)               (None, 18, 1)             33

=================================================================
Total params: 9,193,273
Trainable params: 9,193,273
Non-trainable params: 0
_________________________________________________________________
None
```

Fig16 – CNN Model Summary

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_1 (Embedding)     (None, 150, 50)           9182200

 conv1d_3 (Conv1D)           (None, 150, 32)           4832

 max_pooling1d_3 (MaxPooling  (None, 75, 32)           0
 1D)

 conv1d_4 (Conv1D)           (None, 75, 32)            3104

 max_pooling1d_4 (MaxPooling  (None, 37, 32)           0
 1D)

 conv1d_5 (Conv1D)           (None, 37, 32)            3104

 max_pooling1d_5 (MaxPooling  (None, 18, 32)           0
 1D)

 lstm (LSTM)                 (None, 100)               53200

 dense_1 (Dense)             (None, 1)                 101

=================================================================
Total params: 9,246,541
Trainable params: 9,246,541
Non-trainable params: 0
_________________________________________________________________
None
```

Fig17 – LSTM Model Summary

After implementing the deep learning models, we compared their performance against each other as shown in the below figure (Fig18).

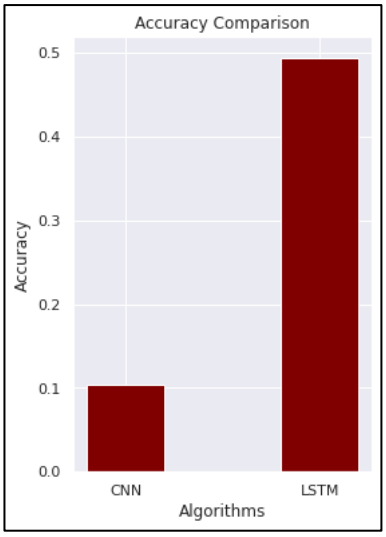

Fig18 – Accuracy Comparison

Although we expected our deep learning models to outperform the more conventional machine learning models, this was not the case for us. With a greater understanding of the hyperparameter, we believe the performance may have been enhanced.

IX. CONTRIBUTIONS

Ankita Arvind Deshmukh[1]

- Performed data preprocessing, exploratory data analysis and data transformation, stemming and TF-IDF vectorization
- Trained and tuned SVM and Random Forest classifiers for binary and multiclass classification
- Created visualization for training time and multiclass performance
- Contributed to training CNN and LSTM models

Rutik Sanjay Sangle[2]

- Carried out data preprocessing which includes stop words and punctuation removal
- Trained and tuned Naive Bayes for binary and multiclass classification models
- Created visualizations for comparison of binary classification algorithms that helped finding basic insights from the dataset

Teja Sree Goli[3]

- Implemented N-gram analysis to identify words which are frequently used together
- Trained and tuned KNN and Logistic Regression classifier for binary and multiclass classification
- Trained deep Learning models such as CNN and LSTM

X. CONCLUSION

Sentiment analysis seeks to extract meaningful information from assessments of texts. Natural language processing and text mining both have a close connection to the study of sentiment. It can be used to evaluate the reviewer's viewpoint on certain topics or the review's overall polarity. In this project, we attempted to conduct a comparative study of the performance of the various traditional machine learning models for sentiment analysis of movie reviews. We concluded experimentally that SVM and logistic regression classifier predicted the results with better accuracy than other models like Random Forest, KNN and Naïve Bayes. The proposed SVM classifier gave an accuracy of 0.89 with 0.96 F1-score. We extended the study by also implementing multiclass classification models and neural network models and evaluated their performance.

REFERENCES

[1] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts, "Learning Word Vectors for Sentiment Analysis", The 49th Annual Meeting of the Association for Computational Linguistics (ACL 2011).

[2] E. B. Solovyeva and A. Abdullah, "Comparison of Different Machine Learning Approaches to Text Classification," *2022 Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus)*, 2022, pp. 1427-1430, doi: 10.1109/ElConRus54750.2022.9755806.

[3] "Sentiment Analysis for Movie reviews", medium.com, https://medium.com/analytics-vidhya/sentiment-analysis-for-movie-reviews-791be2a58297

[4] "Predict sentiment of movie reviews using Deep Learning", machinelearningmastery.com, https://machinelearningmastery.com/predict-sentiment-movie-reviews-using-deep-learning/

[5] "N-gram language modeling with NLTK", geeksforgeeks.com, https://www.geeksforgeeks.org/n-gram-language-modelling-with-nltk/

[6] "Text LSTM", algoritmaonline.com, https://algoritmaonline.com/text-lstm/

[7] "Movie Review Analysis", github.com, https://github.com/072arushi/Movie_review_analysis