Fake News Detection Using Machine Learning Algorithms

Akshat Wadhwa
IIIT Delhi
New Delhi,India
akshat19231@iiitd.ac.in

Shruti Jha
IIIT Delhi
New Delhi, India
shruti19274@iiitd.ac.in

Tarini Sharma
IIIT Delhi
New Delhi, India
tarini19451@iiitd.ac.in

1. Introduction

With the rise in popularity of the Internet, social media platforms like Facebook and Twitter have become new sources of information for people. But these online media platforms are being manipulated by certain entities to spread distorted facts. This phenomenon is known as fake news [4]. Fake news can lead to serious problems like the spread of misinformation about vaccines during the COVID-19 pandemic that led to issues in maintaining public health [3]. In this project, we aim to use machine learning algorithms for automated classification of news articles as fake or real. We aim to explore various textual properties in natural language processing on the dataset, which we will use to train different ML models and ensemble methods and evaluate their performance to determine the best model for this learning task.

2. Related Work

In [6], the authors performed very minimal preprocessing: lowercasing the dataset and dropping stopwords. In order to train the models, a 5-fold split was also performed (80% training, 20% test). The models used are CNN, XGBoost, SVM, Naive Bayes and Random Forest. After training using the Keras API and TensorFlow module, the Accuracy, Precision, Recall and F1-Score are calculated (Figure 1). In paper [5], in addition to lowercasing and removing stopwords, empty strings and missing labels are also handled. The models used are SVM, CNN and LSTM on the metrics Accuracy, Precision and Recall (Figure 2). In [7], the preprocessing is done using word2vec (after dropping the stopwords) from the gensim library. The models used are Feedforward Neural Network, CNN with one convolution layer, CNN with multiple convolution layers and LSTM on the metrics Accuracy, Precision, Recall and F1-score. The metrics are calculated on two different aspects of this dataset: in the first part, the models are trained using the full text of each article and in the latter part models are trained on the title of each article (Figures 3 and 4).

3. Dataset and Evaluation

3.1. Dataset Information

We have used the dataset available on Kaggle [1], which has two files, train.csv (20387 training samples) and test.csv (5127 testing samples). The testing data has four attributes: id, title, author, text and the training data has the additional column of class label (0 for reliable/real news and 1 for unreliable/fake news). There is a slight imbalance in the class distribution in training data, with 10361 samples of class 0 and 7850 samples of class 1 (see Figure 5).

3.2. Pre-processing and feature extraction

Before extracting features, we remove the NaN values and data samples containing only white spaces or "\n". The entire text is converted to lowercase and English letters from a-z are kept, everything else replaced with space. Then stemming is performed to reduce the words in the dataset to their basic roots. Next we drop stop words like "an", "the" etc. [2] since they do not carry any significant meaning. Lastly, we vectorize the document by sklearn's TF-IDF Vectorizer [8]. We explored three different vectorizers for feature extraction before choosing TF-IDF because it gave best results on validation set, as seen in Figure 6. Author attribute had 1957/20800 NaN values in the train set and 500/5200 NaN values in test set. Instead of removing these, replaced NaN values with '-no author-' which helped increase the dataset size. After preprocessing, we had 5040 test samples and 20126 training samples, upon which stratified 5 fold cross validation was performed with 4 folds[16100 samples] used to train the model and 1 fold kept for validation[4026 samples].

3.3. Evaluation Metrics

Evaluation metrics used are accuracy, recall, precision and F1 score. To evaluate models, a single-number evaluation metric is needed. Since F1-score is the harmonic mean of precision and recall, we choose F1 score as a satisficing metric(threshold=0.6) and accuracy as the optimizing metric. Formulas to determine these metrics are in Figure 7.

4. Analysis and Progress

4.1. EDA

We checked the separability of training data by reducing the dimensions of data to 2 using t-SNE and plotting the scatterplot (Figure 8). We see that the data is not very well separated and there are quite a few samples of class 0 overlapping with samples of class 1. We also plotted the word clouds for the reliable news articles and the fake news articles as shown in Figures 9 and 10. There are some general words like “one”, “time”, “many” present in both real and fake news in equal weightage. To decide the number of features to extract in the tf-idf vectorizer, we plotted the mean dev set (for 5-fold) accuracy with Naive Bayes model and got the best result for 3000 features (Figure 11).

4.2. Models and Learning methods

We have explored 6 ML models as of now: Naives bayes (baseline), Logistic Regression [learning methods used: LBFGS (variant of Newton’s method using approximated Hessian matrix) and SGD], Passive-Aggressive Classifier, SVM [learning methods used: Sequential Minimal Optimisation (SMO) and SGD], Decision Tree and SGD based Multi-layer Perceptron(MLP). We used stratified k-fold cross validation (k=5) while training all models and selected the model with best dev set accuracy while satisfying threshold of 0.6 on f1 score.

4.2.1 Hyperparameter Tuning

For hyperparameter tuning, we used Bayesian Optimization implemented in sklearn to get the best hyperparameters for each model (see Figure 12). Since we did not have enough RAM to fit the whole dataset, we used PCA decomposition to reduce the data dimensionality to 100 dimensions.

4.3. Challenges

Overfitting was detected using learning curves. (see fig. 13, 14, 15 and 16) Naive Bayes has a high bias of about 10% and a low variance of about 2%. The model is underfitting and so we use advanced models. Logistic regression[LBFGS] has a low bias of 0.01%(assuming desired performance to be 100% accuracy) and has a variance of 6%. Since the test and dev set distributions are the same, we infer that the model is overfitting the dev set. Logistic regression[SGD] has a bias of 2% and variance of 3.4%. Passive Aggressive model has bias of 1.3% and variance of 4.7%. To reduce overfitting, we tried to add regularization(L2 regularization for logistic regression, early stopping in SGD). We also tried decreasing the number of features which decreased the dev set accuracy as shown in EDA). Reducing the number of features to 100 increased bias by 14.%

4.4. Covariate shift check between validation set and test set

Preprocessed the train(and test) set, added a column with targetDataset=0(and 1) and dropped the classLabel from train dataset. Concatenated and shuffled the 2 datasets and used a RF classifier to classify combined dataset based on targetDataset. An ROC value of 0.505 was observed which implies that the datasets do not have much covariate shift.

5. Results and Interpretation

5.1. Results

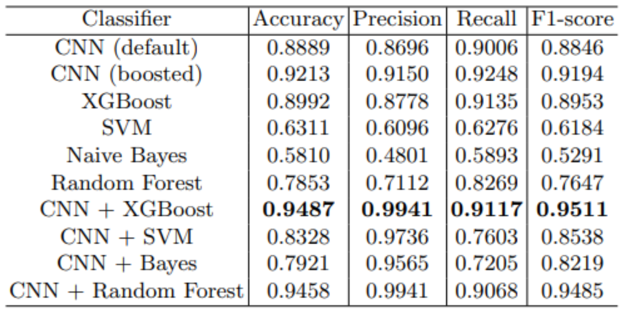
Figures 17 and 18 contain the value of the metrics mentioned in the section 3 for the dev and test set. Gaussian Naive Bayes is underfitting as can be inferred from the learning curve and relatively low training, dev and test accuracy for Naive Bayes. Other models are overfitting as test and dev distribution does not have covariate shift and dev set accuracy is relatively higher with lower test set accuracy.

5.2. Interpretation

Looking at dev set samples which are mislabelled as real, it was noticed that the dev set contains some key words(essential for a human classifying the text as fake or real) not present in the train set. Eg, in the validation sample with text “pokemon go players are inadvertently stopping people committing suicide in japan“, “pokemon” and “inadvertently” are not present in the features extracted using TF-IDF vectorizer. This implies we need to increase training dataset size and make it more diverse. Performance of gaussian naive bayes on the test set is comparable to paper [6] (difference in accuracy 0.2%). Performance of SVM on test set is comparable to papers [5] and [6] with difference in accuracy of about 0.69% and 5.8%.

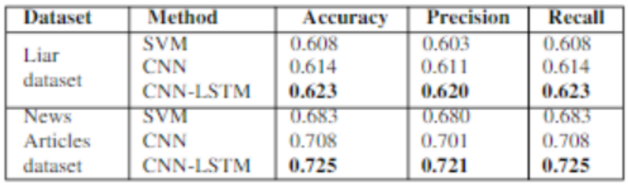
6. Future Work

As there exists a slight imbalance in class labels, we would use weighted accuracy as an evaluation metric in future in place of accuracy. To overcome overfitting, we plan to augment the training data using heuristics. Further error analysis may be required to improve models’ performance. We would also identify drifting features and drop them for better performance on test set and add the ‘polarity’ feature, detected using sentiment analysis in the fake and real news text. In the future, we plan to use Doc2vec for vectorizing the textual data as we found that it had better test accuracy(84%) compared to TF-IDF vectorizer(58%) on the baseline model. We also need to complete analysis of ensemble approaches mentioned in the proposal: Random Forest and XGBoost and LSTM and plan to explore Quasi-Newton learning method for MLP model. Work distributed as in Figure 19.



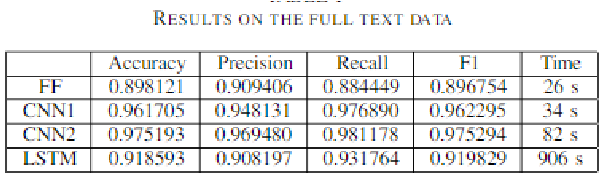
Classifier	Accuracy	Precision	Recall	F1-score
CNN (default)	0.8889	0.8696	0.9006	0.8846
CNN (boosted)	0.9213	0.9150	0.9248	0.9194
XGBoost	0.8992	0.8778	0.9135	0.8953
SVM	0.6311	0.6096	0.6276	0.6184
Naive Bayes	0.5810	0.4801	0.5893	0.5291
Random Forest	0.7853	0.7112	0.8269	0.7647
CNN + XGBoost	**0.9487**	**0.9941**	**0.9117**	**0.9511**
CNN + SVM	0.8328	0.9736	0.7603	0.8538
CNN + Bayes	0.7921	0.9565	0.7205	0.8219
CNN + Random Forest	0.9458	0.9941	0.9068	0.9485

Figure 1. Evaluation Metrics results in referenced paper [6]



Dataset	Method	Accuracy	Precision	Recall
Liar dataset	SVM	0.608	0.603	0.608
	CNN	0.614	0.611	0.614
	CNN-LSTM	**0.623**	**0.620**	**0.623**
News Articles dataset	SVM	0.683	0.680	0.683
	CNN	0.708	0.701	0.708
	CNN-LSTM	**0.725**	**0.721**	**0.725**

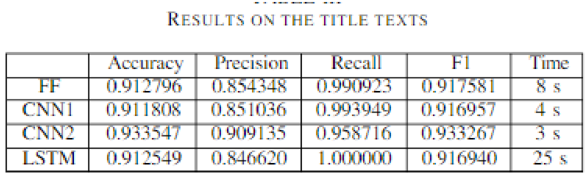
Figure 2. Evaluation Metrics results in the referenced paper [5]



RESULTS ON THE FULL TEXT DATA

	Accuracy	Precision	Recall	F1	Time
FF	0.898121	0.909406	0.884449	0.896754	26 s
CNN1	0.961705	0.948131	0.976890	0.962295	34 s
CNN2	0.975193	0.969480	0.981178	0.975294	82 s
LSTM	0.918593	0.908197	0.931764	0.919829	906 s

Figure 3. Metrics of the models trained only on text field as given in the referenced paper [7]



RESULTS ON THE TITLE TEXTS

	Accuracy	Precision	Recall	F1	Time
FF	0.912796	0.854348	0.990923	0.917581	8 s
CNN1	0.911808	0.851036	0.993949	0.916957	4 s
CNN2	0.933547	0.909135	0.958716	0.933267	3 s
LSTM	0.912549	0.846620	1.000000	0.916940	25 s

Figure 4. Metrics of the models trained on the title field as given in the referenced paper [7]

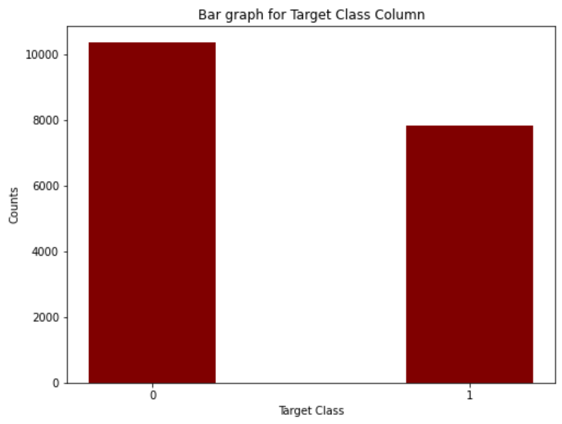
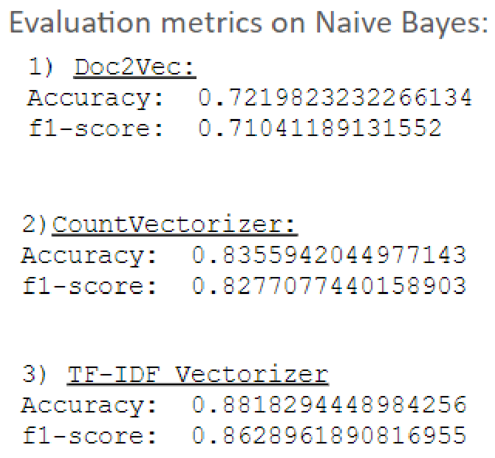


Figure 5. Bar Graph for Target class distribution (Class 0 is real news and Class 1 is fake news)



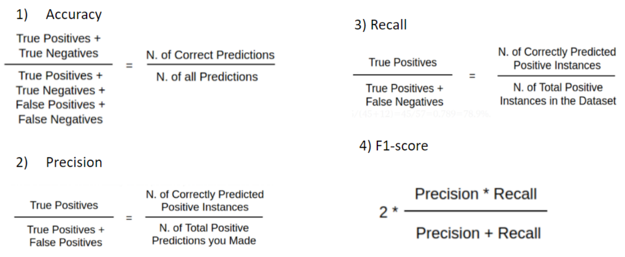
Evaluation metrics on Naive Bayes:

```
1) Doc2Vec:
Accuracy:  0.7219823232266134
f1-score:  0.71041189131552

2)CountVectorizer:
Accuracy:  0.8355942044977143
f1-score:  0.8277077440158903

3) TF-IDF Vectorizer
Accuracy:  0.8818294448984256
f1-score:  0.8628961890816955
```

Figure 6. Validation accuracy using Baseline model of the three different vectorizers explored for feature extraction.



1) Accuracy

$$\frac{\text{True Positives + True Negatives}}{\text{True Positives + True Negatives + False Positives + False Negatives}} = \frac{\text{N. of Correct Predictions}}{\text{N. of all Predictions}}$$

2) Precision

$$\frac{\text{True Positives}}{\text{True Positives + False Positives}} = \frac{\text{N. of Correctly Predicted Positive Instances}}{\text{N. of Total Positive Predictions you Made}}$$

3) Recall

$$\frac{\text{True Positives}}{\text{True Positives + False Negatives}} = \frac{\text{N. of Correctly Predicted Positive Instances}}{\text{N. of Total Positive Instances in the Dataset}}$$

4) F1-score

$$2 * \frac{\text{Precision * Recall}}{\text{Precision + Recall}}$$

Figure 7. Formulae for the evaluation metrics used by us.

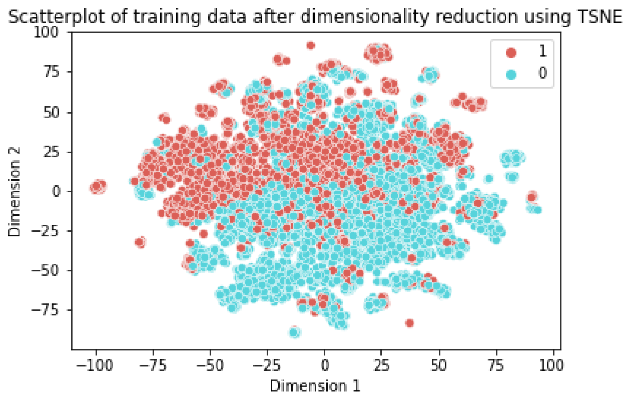


Figure 8. Scatterplot of the training data using t-SNE to check separability of data.

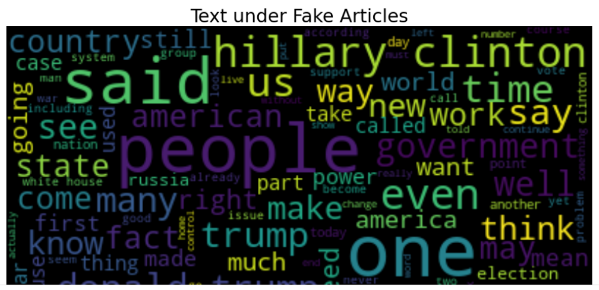


Figure 9. Word cloud for text of Fake articles

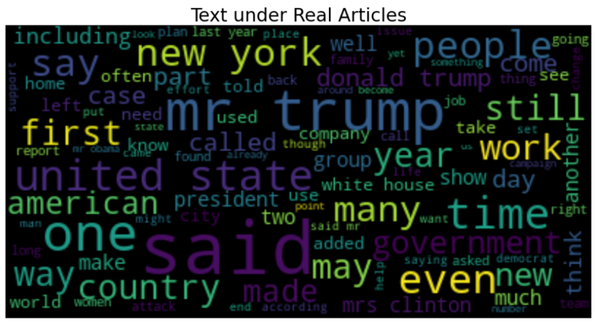


Figure 10. Word cloud for text of Real articles

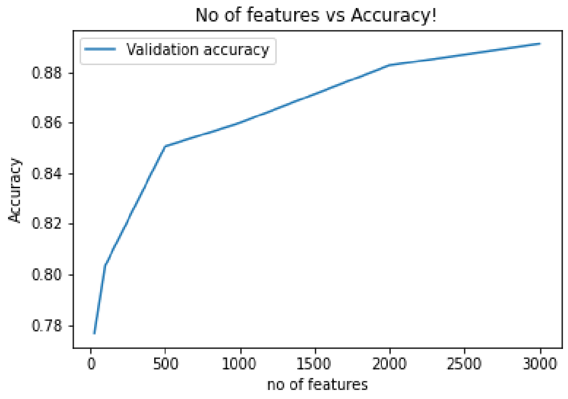
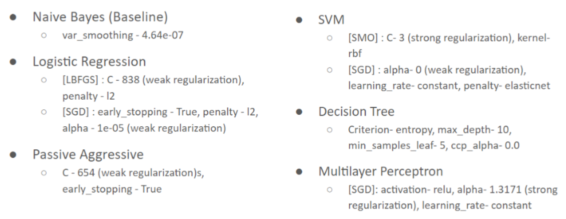


Figure 11. Number of features(in tf-idf vectorization of training data) Vs Mean Accuracy of Naive Bayes model(on validation set)



- Naive Bayes (Baseline)
    - var_smoothing - 4.64e-07
- Logistic Regression
    - [LBFGS] : C - 838 (weak regularization), penalty - l2
    - [SGD] : early_stopping - True, penalty - l2, alpha - 1e-05 (weak regularization)
- Passive Aggressive
    - C - 654 (weak regularization)s, early_stopping - True
- SVM
    - [SMO] : C- 3 (strong regularization), kernel-rbf
    - [SGD] : alpha- 0 (weak regularization), learning_rate- constant, penalty- elasticnet
- Decision Tree
    - Criterion- entropy, max_depth- 10, min_samples_leaf- 5, ccp_alpha- 0.0
- Multilayer Perceptron
    - [SGD]: activation- relu, alpha- 1.3171 (strong regularization), learning_rate- constant

Figure 12. Best Hyperparameters obtained using Bayesian Optimisation Hyperparameter tuning of models

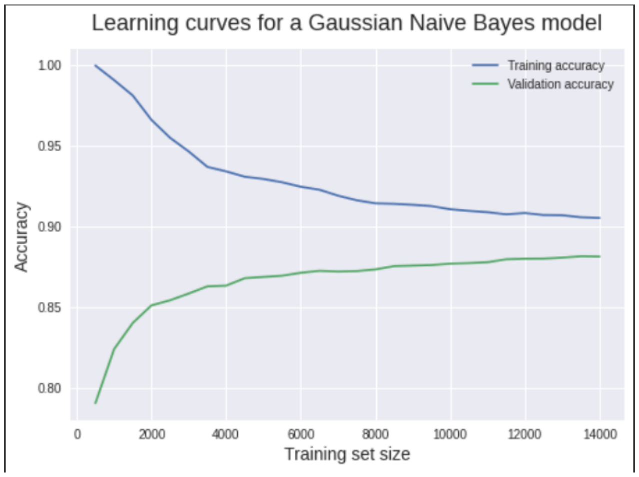


Figure 13. Learning Curve for Naive Bayes

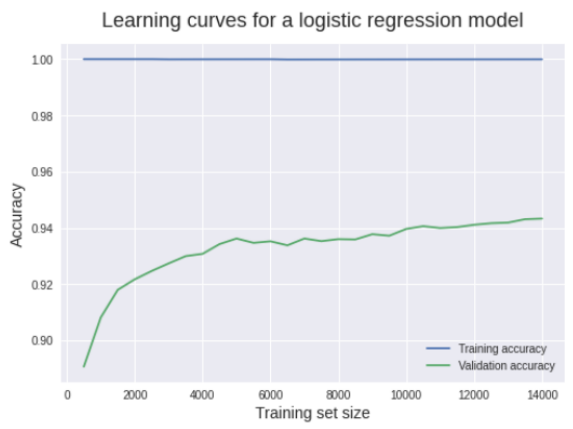


Figure 14. Learning Curve for Logistic Regression

Evaluation Metrics for various models on Validation set (with kfold, k=5)

Model	Accuracy	Precision	Recall	f1-score
Naives Bayes (baseline)	0.8716086925	0.8779999609	0.853373114	0.86548016
Logistic Regression [LBFGS]	0.9399281811	0.9385383964	0.9372614	0.937880176
Logistic Regression [SGD]	0.94618877914	0.94137416739	0.947838194	0.944586786
Passive-Aggressive	0.9370463416	0.9354244069	0.934489221	0.934921898
SVM [SMO]	0.9581634283	0.9544631254	0.959337906	0.95687750
SVM [SGD]	0.9492198597	0.94809468932	0.947015943	0.94750381
Decision Tree	0.8811488430	0.8606924783	0.900195530	0.87996108
Multi-layer Perceptron	0.9421639097	0.9408151943	0.939623113	0.94019892

Figure 17. Performance of our models on validation set (5-fold)

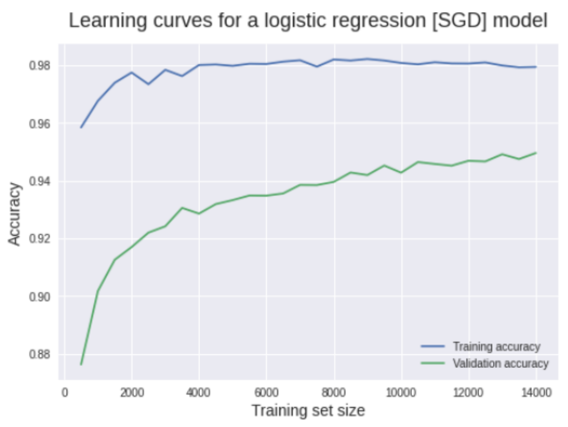


Figure 15. Learning Curve for Logistic regression [SGD]

Evaluation Metrics for various models on Test set

Model	Accuracy	Precision	Recall	f1-score
Naives Bayes (baseline)	0.5791666666	0.6262408286	0.536215816	0.57774238
Logistic Regression [LBFGS]	0.61825396825	0.66130363036	0.592387287	0.624951267
Logistic Regression [SGD]	0.62023809523	0.6629629629	0.59534368	0.627336448
Passive-Aggressive	0.61567460317	0.65908150599	0.588691796	0.621901229
SVM [SMO]	0.6242063492	0.6654720779	0.603399852	0.63291719
SVM [SGD]	0.6233333334	0.6660064965	0.598669623	0.63053485
Decision Tree	0.6087301587	0.6442861410	0.605617147	0.62435031
Multi-layer Perceptron	0.6186904761	0.6620633430	0.591943828	0.62503422

Figure 18. Performance of our models on the test set.

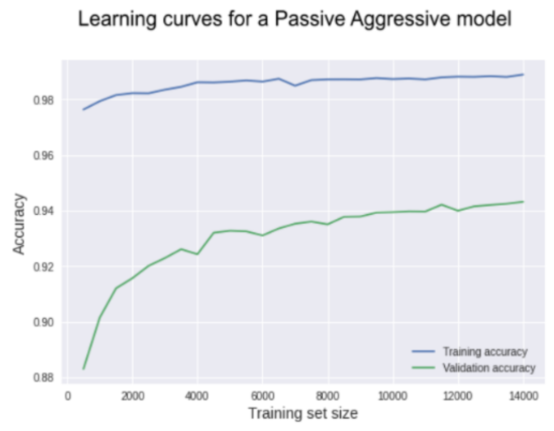


Figure 16. Learning Curve for the Passive Aggressive model

Future Tasks	Team Member Assigned
Augmenting training dataset	Tarini,Shruti
Further Error analysis	Everyone
Dropping drift features	Tarini
Adding polarity feature in dataset	Akshat
Explore Doc2vec vectorization	Akshat
Ensemble approaches	Tarini, Shruti
Quasi-Newton Learning technique for MLP model	Shruti
Advanced RNN model (LSTM)	Akshat

Figure 19. Future Tasks distribution amongst the team

References

[1] Dataset Available Here (Click). 1

[2] Pre-processing in NLP (Click). 1

[3] A. Salma et al. B. Zapan, B. Sajib. Effects of misinformation on covid-19 individual responses and recommendations for resilience of disastrous consequences of misinformation, 2020. Volume 8, ISSN 2590-0617. 1

[4] Y. Benkler et al. D. M. J. Lazer, M. A. Baum. The science of fake news, 2018. Science, vol. 359, no. 6380, pp. 1094–1096. 1

[5] Zineb Giordano Silvia. Drif, Ahlem Ferhat Hamida. Fake news detection method based on text-features, 2019. 1, 3

[6] Anna Marciniak Martyna Tarczewska and Agata Gie lczyk. Fake or real? the novel approach to detecting online disinformation based on multi ml classifiers. 2020. 1, 3

[7] Viera Maslej Krešňáková, Martin Sarnovsky, and Peter Butka. Deep learning methods for fake news detection. 11 2019. 1, 3

[8] Ray Oshikawa, Jing Qian, and William Yang Wang. A survey on natural language processing for fake news detection. *arXiv preprint arXiv:1811.00770*, 2018. 1