**ML Project Report**

Submitted for

**Machine Learning (UML501)**

Submitted by:

Arnav Gupta (102203590)

Geetesh Gupta (102203716)

Sujal Sharma (102203645)

**BE Third Year Batch – 3CO34** Submitted to:

# Mrs. Archana Singh



**Computer Science and Engineering Department**

**Thapar Institute of Engineering & Technology, Patiala**

**July-Dec 2024**

# Introduction

The concept of fake news has been in existence even before the emergence of Internet and other computational technologies. Dissemination of fake news and misleading information has always been used as a weapon to fulfil immoral objectives since ages. The advancement of Internet and web technologies has made it very easy for anyone to post anything in online platforms like blogs, comments to news articles, social media, etc. The advancement of technologies has enabled convenient access to authentic and falsified information even faster posing a real challenge. The involvement of social media replacing the traditional media has an even more catalytic effect, where both fake and authentic news are spread extremely rapidly.

The spread of such fake news has extremely negative impact on target individuals and also the society at large. Consequently, it also creates an impression among readers such that the general perception and responses towards authentic news also gets diluted hampering the balance of news ecosystem. One of the startling examples is the US 2016 presidential election wherein fake news were purposely spread through Facebook and twitter at a larger scale in comparison to authentic information.

This project involves using Google Collab which is an open-source cloud service provided by Google Inc. The data is cleaned and feature extraction is performed resulting in selection of the most significant features contributing towards detection of fake news. As part of the proposed algorithm, 70% of dataset is used for training and remaining 30% is used to test the classification model using k-fold cross validation. The extracted features are further classified using an ensemble machine learning model comprising of Decision Tree (DT) classifier, Random Forest (RF) algorithm and Extra Tree (ET) classifier.

**Literature Review**

The growth of social media is increasing exponentially, and most of the news is spread through social media instead of the standard media channels. Some of the news spread in social media is counterfeit, and this false information has a profound impact on the society, the government, political leaders, etc. Recently, several research works have been carried out using [machine learning techniques](https://www.sciencedirect.com/topics/computer-science/machine-learning-technique" \o "Learn more about machine learning techniques from ScienceDirect's AI-generated Topic Pages) to identify fake news.

Several studies have contributed to fake news detection using diverse datasets and methodologies. For multilingual detection, [16] explored features like Word2Vec and bag-of-words with classifiers such as SVM and Random Forest (RF), achieving moderate accuracy. [17] proposed early detection using semantic, structural, and sentiment features, attaining 91% testing accuracy on Italian Facebook data. Deep learning applications include [18], where CNNs achieved 90% accuracy on Twitter and Weibo data, and [20], where BiLSTM classified tweets with 86.12% accuracy. However, these studies often used limited datasets or lacked multimodal feature integration.

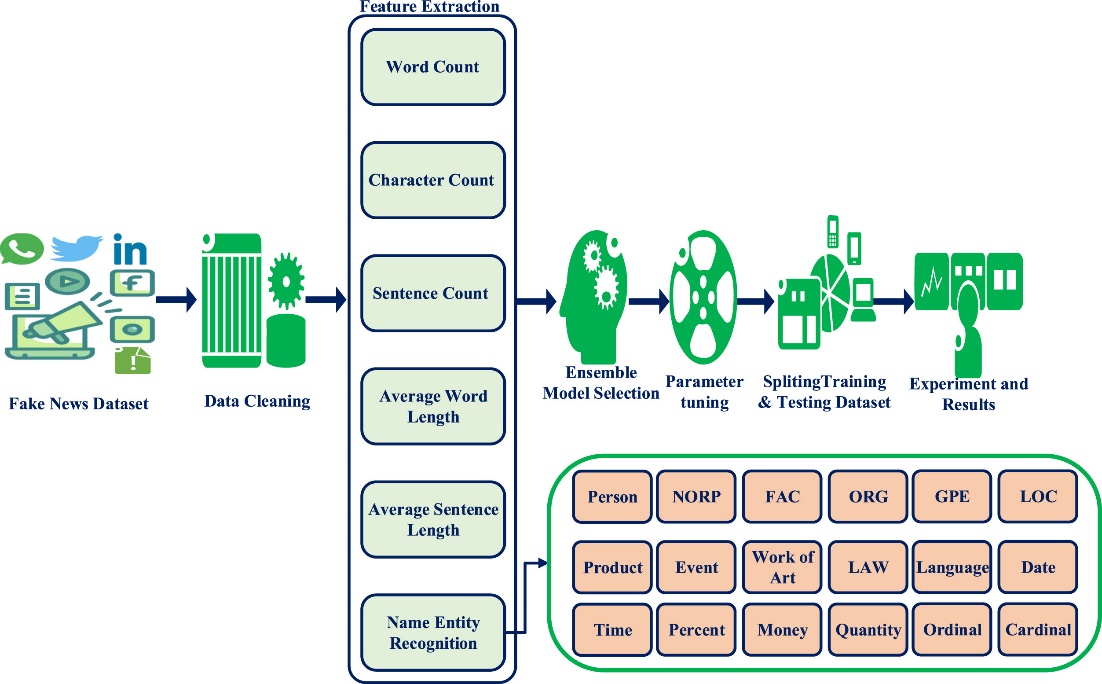
Notable frameworks include a supervised RF model ([19]), achieving 85% accuracy on US election news, and matrix-tensor factorization with XGBoost and DNN ([24]), achieving ~88% accuracy on PolitiFact and BuzzFeed datasets. Neural models like DSSM with LSTM ([23]) achieved 99% training accuracy but lacked robust semantic feature validation. Despite advancements, limitations in feature diversity and scalability persist, indicating potential for integrating text, image, and contextual features for improved accuracy in fake news detection.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.No.** | **Dataset** | **Contributions** | **Accuracy** | **Challenges** |
| 1. | TwitterBR, FakeBrCorpus, FakeNewsData1, FakeOrRealNews, btvlifestyle | Multilingual fake news detection with SVM, RF | |  | | --- | | 79% |  |  | | --- | |  | | |  | | --- | | Limited accuracy |  |  | | --- | |  | |
| |  | | --- | | 2. |  |  | | --- | |  | | Italian Facebook (300K official news, 50K false) | Classifier with semantic, structural, and sentiment features | |  | | --- | | 91% |  |  | | --- | |  | | Did not identify factors affecting accuracy |
| |  | | --- | | 3. |  |  | | --- | |  | | Twitter, Weibo | CNN with positive and unlabeled samples | |  | | --- | | 90% |  |  | | --- | |  | | |  | | --- | | Small dataset |  |  | | --- | |  | |
| |  | | --- | | 4. |  |  | | --- | |  | | US election news (2282 articles) | |  | | --- | | Added features for classifier training |  |  | | --- | |  | | |  | | --- | | 85% |  |  | | --- | |  | | |  | | --- | | Limited dataset |  |  | | --- | |  | |
| 5. | |  | | --- | | Twitter (5800 tweets) |  |  | | --- | |  | | |  | | --- | | BiLSTM-CNN to classify tweets |  |  | | --- | |  | | |  | | --- | | 86.12% |  |  | | --- | |  | | |  | | --- | | Only text-based features used |  |  | | --- | |  | |
| 6. | |  | | --- | | Liar dataset |  |  | | --- | |  | | |  | | --- | | DSSM-LSTM with semantic features |  |  | | --- | |  | | |  | | --- | | 99% |  |  | | --- | |  | | Lack of feature extraction for semantic validity |
| 7. | |  | | --- | | BuzzFeed, PolitiFact |  |  | | --- | |  | | |  | | --- | | Matrix-tensor factorization with XGBoost, DNN |  |  | | --- | |  | | |  | | --- | | 85.86% |  |  | | --- | |  | | |  | | --- | | Limited feature extraction for content and context |  |  | | --- | |  | |

**Methodology**

The architecture includes phases like data preprocessing, feature selection, ensemble model selection and model training. Key steps include:

* Using popular fake news datasets
* Preprocessing to remove noise
* Applying machine learning models like ensemble decision tree, random forest, and extra trees classifier.
* Evaluating performance using various metrics



**1. Data preprocessing:**

The dataset was taken from Kaggle which contains true news articles that were been taken from reliable sources and fake news articles that were taken from sites that had flagged those articles.

The datasets were preprocessed to remove noises like punctuations, HTML tags, digits and converting into lowercase. Key preprocessing steps included:

1. The real and fake news datasets were combined

2. Data was shuffled and the indexing of the data was corrected

3. Useless headings were dropped

4. HTML tags, punctuations, newline characters and digits were removed.

5. Text was converted to lowercase.

6. Tokenization and Stemming were performed.

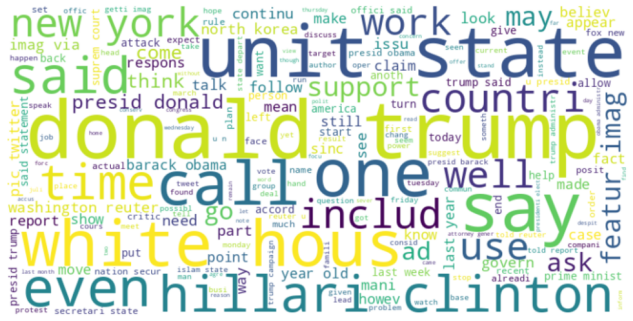
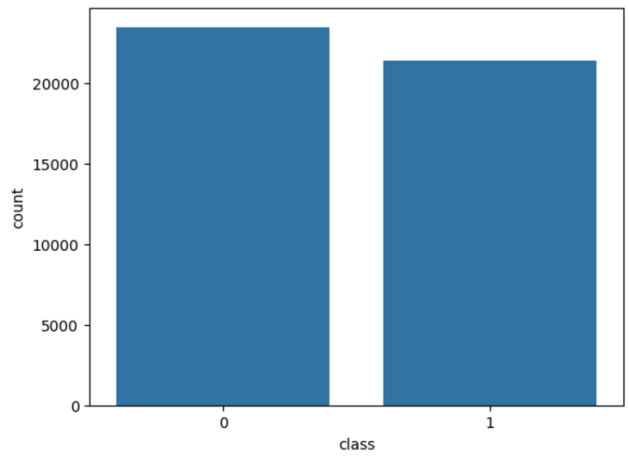
7. Stopwords removal process was also executed to remove any extra noise.

8. Feature Extraction was done.

9. Vectorization was done that transformed the data into vectors, which are then used to improve efficiency and performance.

**2.Data visualization:**

We visualized the datasets using word clouds, pie charts, emotion graphs, and frequency bar graphs to understand their structure. Figure below shows the word cloud and bar graph.

**3.Ensemble model selection**

An ensemble approach combines predictions from multiple machine learning algorithms to improve accuracy. We selected three supervised algorithms: Decision Tree Classifier (DT), Extra Trees Classifier (ET), and Random Forest Classifier (RF).

**4.Parameter tuning & training**

After completing the preprocessing steps, we split the datasets into training and testing sets. 20% was assigned for the testing data and the remaining 80% was trained. Various combinations were tested to find the best values. For evaluation, we used accuracy, precision, recall, and F-score metrics.

***Accuracy*** is the estimate of total number of correctly classified instances and is calculated using

(Tp+Tn)/(Tp+Fp+Fn+Tn)

***Precision*** is the percentage of relevant instances obtained from the total number of instances and is computed using

Tp/(Tp+Fp)

***Recall*** refers the percentage of relevant instances retrieved from the total number of relevant instances and computed using

Tp/(Tp+Fn)

***F1 Measure*** is the harmonic average of precision and recall given by:

F1-Score = (2\*Precision\*Recall)/(Precision+Recall)

Here Tp ,Fn ,Fp ,Tn represent the number of true positives, false negatives, false positives and true negatives respectively.

**Result**

The experimentation on the dataset was performed in Google Collab, a free GPU based cloud service offered by Google Inc. The programming language used for the experimentation is Python. 80% of the records are used to train the classifiers and the remaining 20% of the records are used for testing the classifiers. To classify the features extracted from the dataset, Decision Tree (DT) classifier, Random Forest algorithm and Extra Trees(ET) were used.

# Model Performance:

Decision Tree Performance metrics:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-Score | Support |
| 0 | 1.0 | 1.0 | 1.0 | 4696 |
| 1 | 1.0 | 1.0 | 1.0 | 4285 |
| Accuracy |  |  | 1.0 | 8980 |
| Macro Avg | 1.0 | 1.0 | 1.0 | 8980 |
| Weighted Avg | 1.0 | 1.0 | 1.0 | 8980 |

Accuracy: 99.7%

# Confusion Matrix

# 