2023 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT) | 979-8-3503-3439-5/23/$31.00 ©2023 IEEE | DOI: 10.1109/CONECCT57959.2023.10234826

Text Classification using Machine Learning Algorithms on Encrypted Data

Sagarika Behera
Dept. of Computer Sc. and Engineering
CMR Institute of Technology
Bengaluru, India
sagarika.b@cmrit.ac.in

Nikhita Kolipakula
Dept. of Computer Sc. and Engineering
CMR Institute of Technology
Bengaluru, India
niko19cs@cmrit.ac.in

Swetha P
Dept. of Computer Sc. and Engineering
CMR Institute of Technology
Bengaluru, India
pswe19cs@cmrit.ac.in

Yogish M
Dept. of Computer Sc. and Engineering
CMR Institute of Technology
Bengaluru, India
myog19cs@cmrit.ac.in

Jhansi Rani Prathuri
Dept. of Computer Sc. and Engineering
CMR Institute of Technology
Bengaluru, India
jhansirani.p@gmail.com

***Abstract*—Text classification is a fundamental activity in natural language processing (NLP), that includes classifying text resources into predetermined groups. With the growing concerns over data privacy and confidentiality, there is a need to perform text classification on encrypted data to protect sensitive information. In recent years, homomorphic encryption has emerged as a powerful technique that makes computations possible on encrypted data without compromising its confidentiality. This paper provides an overview of text classification using machine learning algorithms on encrypted data. In this research work, we have taken a data set of movie reviews. The data set is cleaned by removing stop words and performing stemming, lemmatization, and tokenization. We utilized a semantic model with inverse document frequency weighting and term frequency weighting to represent the data (TF-IDF). For the classification of the data set, we have applied six machine learning algorithms Logistic Regression, Naïve Bayes, KNN, Support Vector Machine, Random Forest, and Decision Tree. The plain text is encrypted using a somewhat homomorphic encryption technique called the RSA method. The ciphertext is subjected to the machine learning algorithms mentioned above. We have compared the results for accuracy. The Random Forest algorithm was found to give 75.56% accuracy compared to other algorithms on encrypted data. In this work, we have calculated the time taken by each algorithm for execution on plain text and ciphertext. It was observed that the Random Forest method took more time and the KNN method took less time comparatively.**

***Index Terms*—Homomorphic Encryption, Naive Bayes, Random Forest, Super Vector, Logistic Regression, TF-IDF**

I. Introduction

Natural language processing (NLP) tasks frequently require classifying text documents into predetermined groups. It has many uses, including topic modeling, sentiment analysis, and spam detection. Machine learning algorithms have been widely used to perform text classification, and with the advancements in cryptography, it is now possible to perform text classification on encrypted data.

Encrypted data refers to data that has been encrypted to protect its confidentiality and privacy. In the context of machine learning, homomorphic encryption algorithms allow computations to be performed on encrypted data. Homomorphic encryption protects the confidentiality and privacy of the data by allowing computations to be performed on encrypted data without first having to decrypt it. When C. Gentry gave the idea of a fully homomorphic encryption (FHE) method [1] in his Ph.D. thesis in the year 2009, more researchers started working in this area to make it more practical. Some of the work in the field of FHE for its improvement and applications in the cloud computing environment is given in the paper [2]–[4]. To improve the performance of the FHE by bootstrapping method and using lattices given in the paper [5]–[8].

Homomorphic encryption enables computing while preserving privacy in a variety of circumstances, including private searches, encrypted databases, and the computation of data on the cloud. The operations that can be done mathematically on the ciphertext serve as a defining characteristic of the various homomorphic encryption types. As can be seen in Figure 1, Data is transformed into ciphertext using homomorphic encryption, allowing for analysis and manipulation as if the data were still in its original form.

There are various machine learning algorithms available for text classification, including logistic regression, decision trees, and support vector machines (SVMs). These algorithms are applicable to build a model that maps the features of a text document to its corresponding category. The authors of the paper [9] briefly explained how machine learning algorithms are applicable to encrypted data.

To perform text classification on encrypted data, we need to use homomorphic encryption techniques to encrypt the text documents and the model parameters. Then, we can use homomorphic operations to perform the computations required for text classification.

One of the challenges of performing text classification

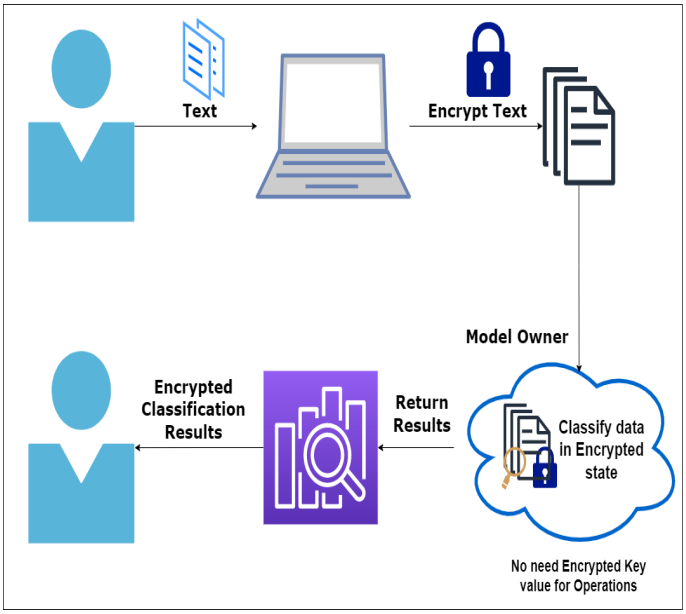

Fig. 1. Homomorphic Encryption Process

on encrypted data is the computational overhead associated with homomorphic encryption. Homomorphic operations are computationally intensive and can significantly slow down text classification. However, recent advancements in homomorphic encryption techniques have significantly reduced the computational overhead, making it possible to perform text classification on ciphertext with reasonable efficiency.

An example of natural language processing is sentiment analysis, commonly referred to as an opinion mining technique used in order to gauge the mood broadly or emotion expressed in a piece of text, such as a review, tweet, or article. The author of the paper [10] explains how these tweets or opinions can help doctors to analyze patients' behavior and mental health condition with the help of NLP.

Automatically categorizing a text's sentiment as good, negative, or neutral is the aim of sentiment analysis. This can be done using a variety of methods, including machine learning algorithms, lexicon-based approaches, and rule-based systems. The authors of the paper [11] have done a detailed analysis of fake news on COVID-19 data using various machine learning and deep learning algorithms.

To identify false information on COVID-19, a model is proposed in the paper [12]. Ten ML algorithms are used with seven methods of feature extraction for detecting the truth of the message.

Sentiment analysis has many applications, including market research, social media monitoring, and customer feedback analysis. For example, businesses can use sentiment analysis to monitor online reviews and social media posts to gain insights into customer opinions and preferences. It can also be used to identify and respond to negative feedback in real-time, improving customer satisfaction and loyalty.

In summary, text classification on encrypted data is a promising area of research that combines the benefits of machine learning and cryptography to protect the privacy and confidentiality of sensitive text data. With the advancements in homomorphic encryption techniques, it is now possible to perform text classification on encrypted data with reasonable efficiency.

Outline of the paper: Section I of the paper gives a brief introduction to text analysis, homomorphic encryption methods, and sentiment analysis. The related work in this area is described in section II. We have used different machine-learning algorithms for text analysis, TF-IDF for text mining, and RSA algorithm for an encryption operation in our work. The proposed method for our research work and the system architecture is given in section III. We have implemented it using Python and calculated the percentage of accuracy for different classification algorithms and the result is tabulated and also plotted as a graph given in section IV. Finally, we have concluded our research work with the scope for future work in section V.

II. Relevant Work

The field of sentiment analysis is expanding quickly, and researchers continue to develop new techniques and tools to improve accuracy and applicability. As the volume of online data continues to grow, sentiment analysis is likely to become an increasingly important tool for businesses and organizations looking to gain insights into customer sentiment and improve their overall performance. The authors of the paper [13] have done research on sentiment analysis for emotions along with text on airline databases. They found that the deep learning method is better than ML algorithms for classification.

The primary goal of the study [14] is to implement a privatization mechanism and text classification for word embedding to prevent leakage of information. PrivFT seeks to operate all neural network computations in an encrypted manner. BERT Embedding-based Method It makes use of fastText for this. This includes a two-layer network, an embedding layer, and bag-of-words vectors as its input. PrivFT, on the other hand, does not make use of pre-training; as a result, the classifier and embedding matrix must be modified from scratch, and training on a single dataset takes a long time.

Paper [15] discusses automatically extracting information and categorizing articles based on suitable topics. Text Classification Algorithms like K- Nearest Neighbors, Naive Bayes, and Support Vector Machine were used. The results of each classification algorithm were compared. This paper also describes the steps involved in article classification. KNN performs better with fewer features. When the number of features increases it leads to an increase in dimension, thus leading to the problem of overfitting.

By comparing our system's performance on a sentiment classification job on the IMDb movie review dataset to a plaintext implementation, Paper [16] demonstrates that RNNs can be used over encrypted data without suffering any loss in accuracy. is a completely connected layer that has SoftMax activation on two units. Keras is used to do the training, which

results in 86.47% accuracy on the test data that hasn't been seen. The encrypted data is accurate to the same degree.

The method and approach used in Paper [17] to homomorphically encrypt the heart disease medical dataset and apply the KNN machine learning algorithm to the encrypted dataset are presented. Homomorphic encryption can be effectively utilized to protect the privacy of data in the cloud since it allows processing to happen while the data is protected.

III. Proposed Method

The architecture of the suggested method to safely store data in the cloud is thoroughly described in this section. The suggested technique provides a practical way to protect cloud-based data. Machine learning algorithms can be applied to the encrypted dataset for classification.

A. System Architecture

The system architecture is shown in figure 2 for the suggested model. A movie review dataset is taken as the input for this proposed model. First, the data set is cleaned by eliminating stop words and performing stemming, lemmatization, and tokenization. Later, the data is classified using different classification algorithms such as Naïve Bayes classifier, K Nearest Neighbor, Decision Tree, Logistic Regression, Random Forest, and Support Vector machine. The output is stored as a result1. Then an encryption algorithm like RSA is applied to the plain text and ciphertext is produced. This ciphertext is stored in another CSV file (the ciphertext is in the form of numbers). All the classification algorithms are also used on the encrypted data, and this outcome is stored as result 2. Now, both the outputs stored in result 1 and result 2 are compared and visualized.

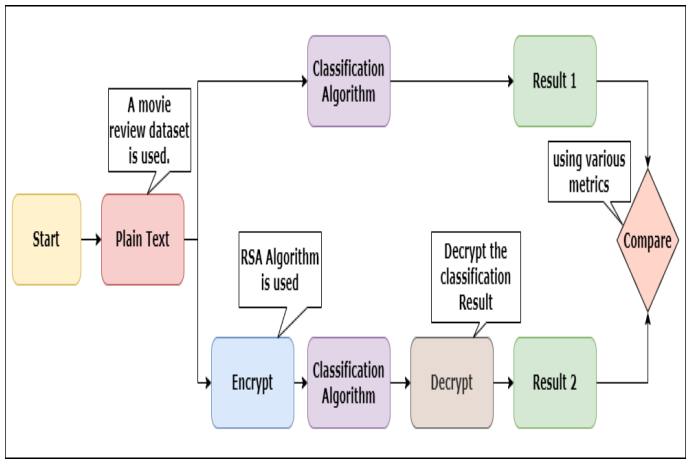

Fig. 2. Design Architecture for the Proposed Model

B. Methodology of the Proposed System

The suggested solution is based on the use of homomorphic encryption, where we encrypt a selected dataset using the RSA technique because it has homomorphic features and the results are stored in a file. Here, we make use of the Kaggle dataset of movie reviews.

The classification algorithms are then used on the encrypted data that has been stored to produce predictions using the input features as a variety of factors and the target variable as a binary variable.

The output is now saved in a file, and the RSA technique is used to decode it. The dataset is taken from Kaggle, and Python is used to implement both the RSA and classification algorithms. In order to retrieve the output and save it in a file, we next apply the same classification method to the original data without encrypting it. Two findings have been obtained; at this point, we compare them and visualize the same. The data flow diagram is shown in figure 3.

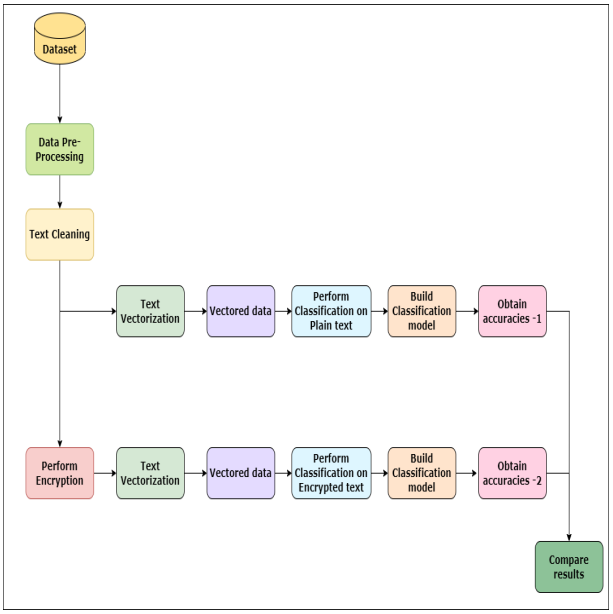

Fig. 3. Data Flow Diagram

IV. Implementation and Result

The work includes employing a dataset of movie reviews, as the project's input. The dataset is collected from the Kaggle website. The dataset has 2 columns which are sentiment and review. The sentiment contains sentences explaining the user's feedback or comment which is text data and the review column contains binary values i.e 0 or 1, where 1 represents a positive review while the value 0 represents a negative review. The data set is cleaned by removing stop words and performing stemming, lemmatization, and tokenization. The results are obtained and saved as a result 1 when the data is classified later using various classifiers. The plain text is then subjected to an encryption scheme like RSA, and ciphertext is produced. Another CSV file is used to store the ciphertext (the ciphertext is in the form of numbers). On the encrypted data again various classification algorithms were applied, and the outcome is saved as result 2. Both the results are compared and analyzed.

We consider the same movie review dataset. We mainly focus on encrypting the Sentiment part of the movie review dataset. To perform encryption we take each sentence, then

we extract each word from the sentence. Now further every alphabet is taken and converted into its ASCII code which is then encrypted. Next, each of these encrypted alphabets is combined together to create an encrypted word, the encrypted words are further combined to form an encrypted sentence.

Since our model is a binary classifier, where 1 represents positive feedback and 0 represents negative feedback, we have calculated the Recall, Precision, F1 score, and AUC (Area Under the ROC Curve) value to measure the performance of the model. This metric value is tabulated and shown in table I.

TABLE I
MACHINE LEARNING-BASED CLASSIFICATION RESULT

Algorithm	Precision (Without Encryption)	Precision (With Encryption)	Recall (Without Encryption)	Recall (With Encryption)	F1 Score (Without Encryption)	F1 Score(With Encryption)	AUC (Without Encryption)	AUC (With Encryption)
Logistic Regression	0.81	0.63	0.77	0.53	0.72	0.45	0.86	0.65
Random Forest	0.88	0.85	0.83	0.71	0.8	0.7	0.88	0.67
SVM	0.76	0.56	0.77	0.56	0.72	0.56	0.89	0.7
Naive Bayes	0.77	0.28	0.67	0.5	0.57	0.36	0.87	0.68
Decision Tree	0.75	0.7	0.76	0.7	0.72	0.7	0.76	0.7
KNN	0.66	0.43	0.62	0.45	0.47	0.43	0.79	0.49

A. Algorithms

The algorithms that the model uses are described in this section. In order to fulfill the numerous functional and non-functional needs of the suggested scheme, the model uses these methods.

Several techniques, including Logistic Regression, Random Forest, Naive Bayes, Decision Tree, K Nearest Neighbor, and Support Vector Machine for Comparison, are used to achieve the proposed system.

The steps involved in all these ML algorithms for the classification of data are explained in this section. Algorithm 1 explains the steps involved in Logistic Regression for calculating the accuracy score. Similarly, algorithm 2 shows the steps for the Random Forest method, algorithm 3 explains the Naive Bayes method, algorithm 4 shows the steps for the Decision tree method, algorithm 5 is for Support Vector Machine, and algorithm 6 is for KNN classifier to calculate the accuracy score.

B. Result

All six machine learning algorithms are applied to the plain text and to the cipher text for classification. Table II shows the accuracy in %age for encrypted and plain text data.

On the movie review dataset, we applied various machine learning algorithms on the original dataset and on the encrypted dataset. Table II shows the accuracy in percentage for different algorithms. The graphical representation of the accuracy in %age is shown in figure 4. We calculated the percentage of accuracy difference of each algorithm. It is observed that the Decision Tree algorithm gives less difference. The difference between accuracy without encryption and accuracy with encryption is 5.01% in the Decision Tree algorithm. We observed that accuracy for all the Classifiers on encrypted data decreased significantly compared to classification on plain data. The highest accuracy for classification on plain text and on the encrypted text was observed by the Random Forest Classifier.

In this work, we have calculated the time required by each of the machine-learning methods for the classification of the plain text and the ciphertext. The result is tabulated and presented in table III.

Algorithm 1 Logistic Regression Algorithm

Input: $X_{train}, X_{test}, Y_{train}, Y_{test}$
Output: Accuracy Score

1: Import LogisticRegression into the sklearn.linear model package.
2: LogisticRegression() assigned to the lr variable in order to categorize the dataset.
3: Send the X_{train} and Y_{train} training data to the selected algorithm to fit it.
4: The following step is to create predictions based on the test results from X_{test}.
5: The accuracy score is calculated by importing the accuracy score from sklearn. metrics. Passing the anticipated data and testing data Y_{test}, the accuracy score will be determined.
6: Depending on the accuracy score, we can experiment with changing the algorithm to raise the accuracy score.
7: **return** Accuracy Score

Algorithm 2 Random Forest Algorithm

Input: $X_{train}, X_{test}, Y_{train}, Y_{test}$
Output: Accuracy Score

1: Import the RandomForestClassifier classifier from sklearn.ensemble
2: In order to categorize the dataset, we set max accuracy to 0.
3: Send the training data X_{train} and Y_{train} to the loop that will suit the chosen method with an x range of 2000.
4: The following step is to create predictions based on the test results from X_{test}.
5: The accuracy score is calculated by importing the accuracy score from sklearn. metrics. Passing the anticipated data and testing data Y_{test}, the accuracy score will be determined.
6: Depending on the accuracy score, we can experiment with changing the algorithm to raise the accuracy score.
7: **return** Accuracy Score

Algorithm 3 Naive Bayes Algorithm

Input: $X_{train}, X_{test}, Y_{train}, Y_{test}$
Output: Accuracy Score

1: Add GaussianNB to your program using sklearn.naive Bayes
2: GaussianNB() assigned to the nb variable in order to categorize the dataset.
3: Send the X_{train} and Y_{train} training data to the selected algorithm to fit it.
4: The following step is to create predictions based on the test results from X_{test}.
5: The accuracy score is calculated by importing the accuracy score from sklearn. metrics. Passing the anticipated data and testing data Y_{test}, the accuracy score will be determined.
6: Depending on the accuracy score, we can experiment with changing the algorithm to raise the accuracy score.
7: **return** Accuracy Score

Algorithm 4 Decission Tree Algorithm

Input: $X_{train}, X_{test}, Y_{train}, Y_{test}$
Output: Accuracy Score

1: Import the DecisionTreeClassifier classifier from sklearn.tree
2: In order to categorize the dataset, max accuracy is set to be 0.
3: Send the training data X_{train} and Y_{train} to the loop that will suit the chosen algorithm with an x range of 200.
4: The following step is to create predictions based on the test results from X_{test}.
5: The accuracy score is calculated by importing the accuracy score from sklearn. metrics. Passing the anticipated data and testing data Y_{test}, the accuracy score will be determined.
6: Depending on the accuracy score, we can experiment with changing the algorithm to raise the accuracy score.
7: **return** Accuracy Score

Algorithm 5 Support Vector Machine Algorithm

Input: $X_{train}, X_{test}, Y_{train}, Y_{test}$
Output: Accuracy Score

1: Import the svm file from Sklearn
2: To categorize the dataset, the kernel is set to linear and assign svm.SVC() to sv.
3: Send the X_{train} and Y_{train} training data to the selected algorithm to fit it.
4: The following step is to create predictions based on the test results from X_{test}.
5: The accuracy score is calculated by importing the accuracy score from sklearn. metrics. Passing the anticipated data and testing data Y_{test}, the accuracy score will be determined.
6: Depending on the accuracy score, we can experiment with changing the algorithm to raise the accuracy score.
7: **return** Accuracy Score

Algorithm 6 K-Nearest Neighbor (KNN) Algorithm

Input: $X_{train}, X_{test}, Y_{train}, Y_{test}$
Output: Accuracy Score

1: Import KNeighbhorsClassifier from the neighborhood. sklearn.
2: To categorize the dataset, we set k neighbors to be 7.
3: Send the X_{train} and Y_{train} training data to the selected algorithm to fit it.
4: The following step is to create predictions based on the test results from X_{test}.
5: The accuracy score is calculated by importing the accuracy score from sklearn. metrics. Passing the anticipated data and testing data Y_{test}, the accuracy score will be determined.
6: Depending on the accuracy score, we can experiment with changing the algorithm to raise the accuracy score.
7: **return** Accuracy Score

TABLE II
ACCURACY COMPARISON RESULT

Algorithm	Accuracy in %age (Plain Text)	Accuracy in %age (Cipher Text)
Logistic Regression	80.6	60.0
Random Forest	86.57	75.56
SVM	77.61	57.78
Naive Bayes	73.13	57.78
Decision Tree	76.12	71.11
KNN	67.16	48.89

C. Discussions

The work includes employing a dataset of movie reviews, as the project's input. This dataset is collected from the Kaggle website. The data set is cleaned by removing stop words and performing stemming, lemmatization, and tokenization. The results are obtained and saved as a result 1 when the data is classified later using various classifiers. The plain text is then subjected to an encryption scheme like RSA, and ciphertext is produced.

Another CSV file is used to store the ciphertext (the ciphertext is a string of numbers). On the encrypted data again various classification algorithms were applied, and the outcome is saved as result 2. Both the results are compared and analyzed.

It is observed that the Random Forest algorithm gives better

TABLE III
TIME REQUIRED FOR EXECUTION

Algorithm	Execution Time (Plain Text) in msec	Execution Time (Cipher Text) in msec
Logistic Regression	41.65	31.72
Random Forest	198.42	169.3
SVM	337.82	40.6
Naive Bayes	20.61	3.81
Decision Tree	32.16	17.6
KNN	6.84	2.22

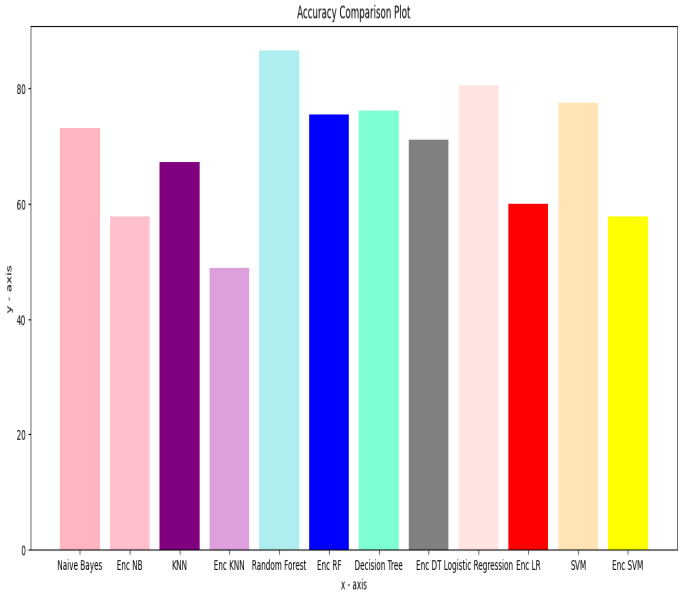

Fig. 4. Accuracy Comparison of the Model for Different classification algorithms

accuracy compared to the other classification algorithms but it took more time for execution. It gives 86.57% accuracy on plain text and 75.56% accuracy on ciphertext. But it took 198.42 msec to execute on the plaintext and 169.3 msec to execute on the ciphertext.

Since our model is a binary classifier model, where 0 represents negative feedback and 1 represents positive feedback on movie reviews, we have calculated the performance metric which includes precision, recall, F1 score, and AUC (Area Under the ROC Curve) values for both plain text and ciphertext. We found that the Random Forest method gives better results compared to other ML classifiers.

V. Conclusion and Future Work

This project focused on text classification using Natural Language Processing (NLP) techniques on encrypted data. The goal was to explore the feasibility and effectiveness of applying machine learning algorithms to encrypted text while maintaining the privacy and security of the data. Through the experimentation process, it was found that the Random Forest classifier yielded promising results for both plain text and encrypted text classification. The model achieved an impressive accuracy of 86.57% for plain text and 75.56% for encrypted text. These results indicate that it is possible to perform accurate text classification even when dealing with encrypted data. The success of the Random Forest classifier can be attributed to its ability to handle high-dimensional data and capture complex relationships between features. It demonstrates the potential of machine learning algorithms in analyzing encrypted text, thus offering a practical solution for protecting sensitive information while still enabling meaningful analysis.

The future work that needs to be accomplished is to change the encryption technique. The current model is using a partial Homomorphic technique to encrypt the data.

So, we are planning to implement a Fully Homomorphic Encryption (FHE) scheme like BGV and CKKS. The accuracy will be compared for both schemes.

References

[1] C. Gentry, "A fully homomorphic encryption scheme," PhD thesis, Stanford University, 2009, [Online]. Available: crypto.stanford.edu/craig.

[2] V. Vaikuntanathan, "Computing blindfolded: New developments in fully homomorphic encryption," in *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science,* IEEE, pp. 5–16, 2011

[3] M.Tebaa, S. El Hajji, and A. El Ghazi, "Homomorphic encryption method applied to Cloud Computing," *2012 National Days of Network Security and Systems,* IEEE, pp. 86–89, 2012.

[4] M. Alkharji, H. Liu, and C. Washington, "Homomorphic encryption algorithms and schemes for secure computations in the cloud," *Proceedings of 2016 International Conference on Secure Computing and Technology*, 2016

[5] C. Gentry, S. Halevi, and N.P Smart, "Better bootstrapping in fully homomorphic encryption," *International Workshop on Public Key Cryptography*, Springer, pp. 1–16, 2012.

[6] M.Naehrig, K. Lauter,and V. Vaikuntanathan, "Can homomorphic encryption be practical?," *Proceedings of the 3rd ACM workshop on Cloud computing security workshop*, pp. 113–124, 2011.

[7] C. Gentry, "Fully homomorphic encryption using ideal lattices," *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pp. 169–178, 2009.

[8] Z.Brakerski and V. Vaikuntanathan, "Efficient fully homomorphic encryp¬tion from (standard) lwe,"*SIAM Journal on Computing*, vol. 43, no. 2, pp. 831–871, 2014.

[9] S. Behera, J.R, Prathuri, "Application of homomorphic encryption in machine learning," *2020 2nd PhD Colloquium on Ethically Driven Innovation and Technology for Society (PhD EDITS),*IEEE, pp. 1–2, 2020.

[10] Rajput, Adil, "Natural language processing, sentiment analysis, and clinical analytics," *Innovation in health informatics*, Elsevier, pp. 79–97, 2020,

[11] Bangyal, Waqas Haider, Qasim, Rukhma, Rehman, Najeeb Ur, Ahmad, Zeeshan, Dar, Hafsa, Rukhsar, Laiqa, Aman, Zahra, and Ahmad, Jamil, "Detection of fake news text classification on COVID-19 using deep learning approaches,"*Computational and mathematical methods in medicine*, Hindawi Limited, pp. 1–14, 2021

[12] Elhadad, Mohamed K and Li, Kin Fun, and Gebali, Fayez, "Detecting misleading information on COVID-19,"*Ieee Access*, IEEE, Vol. 8, pp. 165201–165215, 2020.

[13] Ullah, Mohammad Aman, Marium, Syeda Maliha, Begum, Shamim Ara, and Dipa, Nibadita Saha, "An algorithm and method for sentiment analysis using the text and emoticon,"*ICT Express*, Elsevier, Vol. 6, No. 4, pp.357–360, 2020.

[14] Lee, Garam, Kim, Minsoo, Park, Jai Hyun, Hwang, Seung-won, and Cheon, Jung Hee, "Privacy-Preserving Text Classification on BERT Embeddings with Homomorphic Encryption,"*arXiv preprint arXiv:2210.02574*,2022.

[15] Dien, Tran Thanh, Loc, Bui Huu, and Thai-Nghe, Nguyen, "Article classification using natural language processing and machine learning,"*2019 International Conference on Advanced Computing and Applications (ACOMP)*, IEEE, pp. 78–84, 2019.

[16] Podschwadt, Robert, and Takabi, Daniel, "Classification of Encrypted Word Embeddings using Recurrent Neural Networks,"*PrivateNLP@ WSDM*, pp. 27–31, 2020.

[17] Behera, Sagarika, Rekha, B, Pandey, Pragya, Vidya, B, and Prathuri, Jhansi Rani, "Preserving the Privacy of Medical Data using Homomorphic Encryption and Prediction of Heart Disease using K-Nearest Neighbor,"*2022 IEEE International Conference on Data Science and Information System (ICDSIS)*, IEEE, pp. 1–6, 2022.