Cryptography And Cyber Security
Course Code: R1UC505C

Research Papper

For

BACHELOR OF

ENGINEERING & TECHNOLOGY

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

GALGOTIAS UNIVERSITY, GREATER NOIDA

UTTAR PRADESH

NAME : ABHISHEK KUMAR

ADM.NO : 22SCSE1012568

SEMESTER : V

NAME : AVISHANT KUMAR

ADM.NO : 22SCSE1012569

SEMESTER : V

Email Spam Detection Using Machine Learning Algorithms

Abstract – *Email spam is a growing issue with the rise of internet users, often exploited for phishing, fraud, and spreading malicious links. Spammers create fake profiles and impersonate genuine users to target unaware individuals. This project uses machine learning techniques to detect fraudulent spam emails by applying and evaluating various algorithms, selecting the one with the best precision and accuracy.*

Keywords: Machine learning, Naïve Bayes, support vector machine, nearest neighbour, random forest, bagging, boosting, neural networks.

I. INTRODUCTION

Email spam refers to unsolicited emails sent without the recipient's consent, often for advertising, phishing, or malicious purposes. Spam wastes storage, time, and network resources. While automatic email filtering is effective, spammers have adapted to bypass these systems, making detection challenging. Early techniques like blacklisting and whitelisting domains have limitations, as spammers frequently change domains. Spam detection now relies on methods like text analysis, community-based techniques, and domain filtering. Machine learning approaches, especially Naïve Bayes, are widely used for effective spam filtering. "Ham" refers to non-spam emails, as coined by Spam Bayes in 2001.

Keywords: Machine learning, Naïve Bayes, text analysis, blacklists, whitelists, spam, ham, spam detection.

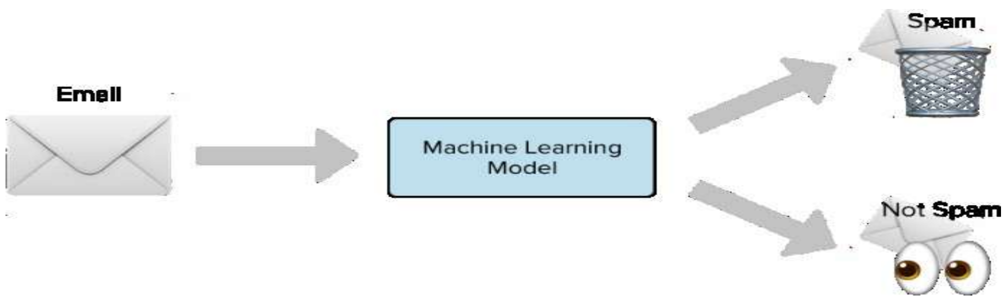


Fig 1. Classification in Spam and Non-spam

A set of pre-classified email samples is utilized as training data in machine learning techniques, which are more effective. Numerous algorithms in machine learning techniques can be applied to email screening. "Naïve Bayes, support vector machines, neural networks, K-nearest neighbor, random forests, etc." are some of these algorithms.

<u>II. LITERATURE REVIEW</u>

Several studies have applied machine learning to email spam detection. A. Karim et al. reviewed AI and ML methods for this purpose. K. Agarwal, T. Kumar, and Harisinghaney et al. (2014) used image and text datasets with KNN, Naïve Bayes, and Reverse DBSCAN algorithms, employing OCR for text recognition, though it performed suboptimally. Mohamad & Selamat (2015) utilized a hybrid approach combining TF-IDF and Rough Geometry for feature selection.

Data Set:

This model uses email datasets from sources like Kaggle, sklearn, and custom-generated sets. The Kaggle spam dataset ("spam.csv") with 5573 entries and other datasets with 574, 1001, and 956 lines in text format were used for training and evaluation.

III. METHODOLOGY

A. Data Preprocessing:

- **Purpose:** Converts raw data (images, text, audio, etc.) into a format machines can process (1s and 0s).
- **Steps:**
    1. **Data Cleaning:** Handle missing values, smooth noisy data, remove outliers, and resolve inconsistencies.
    2. **Data Integration:** Combine multiple datasets or files.
    3. **Data Transformation:** Perform normalization and aggregation.
    4. **Data Reduction:** Summarize data while preserving analytical value.

Key Techniques:

1. **Stop Words:** Common words (e.g., "the," "is") are removed to focus on meaningful terms.
2. **Tokenization:** Breaks text into tokens (words or symbols) for further processing.
3. **Bag of Words (BOW):** Creates a vocabulary of unique words in a dataset for feature extraction in ML models.

B. Classic Classifiers:

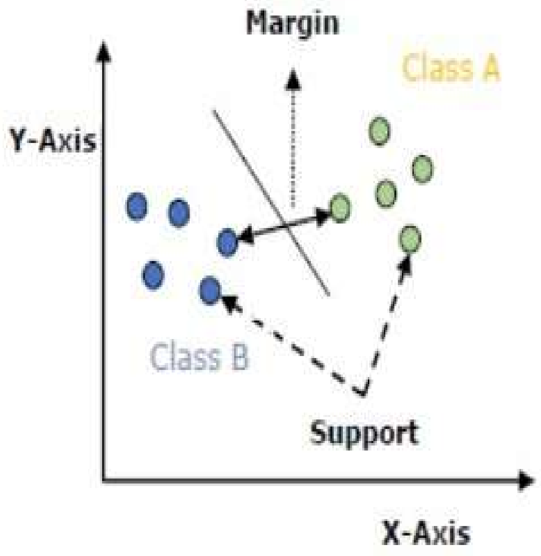
- **Classification:** A two-step process: learning (model training) and prediction.
- **Naïve Bayes:**
    - A supervised learning algorithm based on Bayes' theorem.
    - Assumes feature independence and calculates word probabilities to classify emails as spam or ham.
    - Effective for spam filtering, offering high accuracy by identifying frequently occurring words in spam emails.

$$\mathrm{P(A|B)} = \frac{P(B|A)P(A)}{P(B)} \qquad (1)$$

$$\mathrm{P(B)} = \sum_{y} P(B|A)P(A) \qquad (2)$$

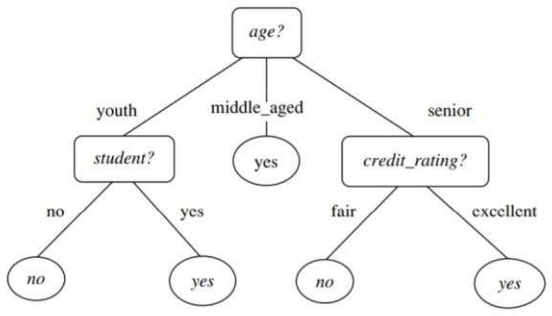
2. Support Vector Machine (SVM):

- A popular supervised learning algorithm for classification tasks.
- Creates a decision boundary (hyperplane) to separate data into classes.
- In 2D, the hyperplane is a line dividing the plane into two parts, each representing a class.



3. Decision Tree:

- A flowchart-like structure for classification.
  - **Internal Node:** Attribute test.
  - **Branch:** Test outcome.
  - **Leaf Node:** Class label.
- Root node is the top node.
- Tree pruning removes noisy branches to improve accuracy.
- Entropy is calculated to measure attribute splits.



Entropy using the frequency table of one attribute:

$$E(S) = \sum_{i=1}^{c} - p_i \log_2 p_i \quad (3)$$

Entropy using the frequency table of two attributes:

$$E(T,X) = \sum_{c \in X} P(c)E(c)$$

4. K-Nearest Neighbour (KNN):

- A supervised, lazy learning algorithm.
- Classifies new data points based on similarity (e.g., Euclidean distance).
- Does not learn but uses stored data for predictions.

C. Ensemble Learning Methods:

- Combines multiple base models to improve predictions.
    - **Sequential:** Models built sequentially (boosting).
    - **Parallel:** Models built in parallel (bagging).

1. **Random Forest Classifier:**
    - Ensemble of decision trees built with random sampling and feature subsets.
    - Reduces correlation and improves generalization.
2. **Bagging (Bootstrap Aggregating):**
    - Fits base classifiers on random subsets of data and combines predictions via voting/averaging.
    - Reduces variance and overfitting.
3. **Boosting and AdaBoost Classifier:**
    - Builds a strong classifier by correcting errors of weak classifiers.
    - AdaBoost, a binary classification algorithm, iteratively improves predictions.

IV. Algorithms

1. **Dataset Handling:**
    - Insert the dataset for training or testing.
    - Check dataset encoding:
        - If supported, proceed.
        - If unsupported, convert to a supported encoding and retry.
    - Choose to "Train," "Test," or "Compare" models.
2. **Training Phase:**
    - Select a classifier and check for duplicates or missing values.
    - Perform hyperparameter tuning and feature transformation.
    - Train the model, save it, and display results.
3. **Testing Phase:**
    - Load the saved model and features.

- Test the dataset using the loaded values and show results.

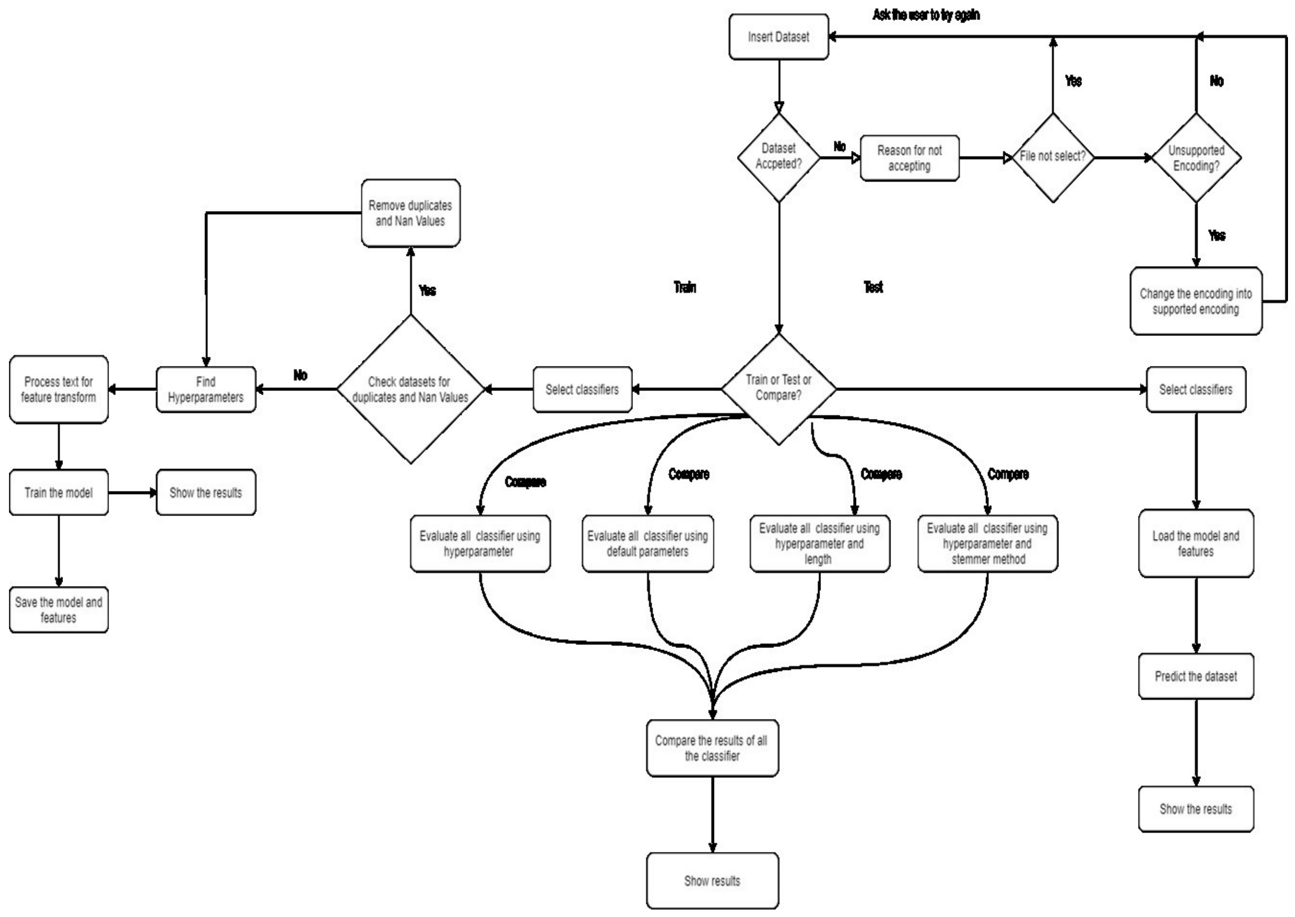
4. **Comparison Phase:**
    - Compare all classifiers on the dataset and display results.

A. Implementation:

- **Platform:** Visual Studio Code.
- **Dataset:** Sourced from Kaggle; split into 70% training and 30% testing.
- **Preprocessing:** Remove duplicates, null values, punctuation, and stop words.
- **Feature Transformation:** Clean text is used to create a vocabulary via fit and transform.
- **Hyperparameter Tuning:** Finds optimal classifier parameters.
- **Model Training:** Trains the model using sklearn classifiers, saves state, and features for future testing.

B. Flowchart:

- Represents the sequential process of data input, training, testing, and comparison.


Ask the user to try again
Insert Dataset
Dataset Accpeted?
No
Reason for not accepting
File not select?
Yes
Unsupported Encoding?
No
Yes
Change the encoding into supported encoding
Remove duplicates and Nan Values
Yes
Train
Test
Process text for feature transform
Find Hyperparameters
No
Check datasets for duplicates and Nan Values
Select classifiers
Train or Test or Compare?
Select classifiers
Train the model
Show the results
Save the model and features
Compare
Compare
Compare
Compare
Evaluate all classifier using hyperparameter
Evaluate all classifier using default parameters
Evaluate all classifier using hyperparameter and length
Evaluate all classifier using hyperparameter and stemmer method
Load the model and features
Predict the dataset
Show the results
Compare the results of all the classifier
Show results

V. Result

The model was trained using multiple classifiers to compare accuracy. Results classify data as "spam" or "ham" and are presented in graphs and tables for clarity.

- **Dataset:**
  - Training: "spam.csv" (from Kaggle).
  - Testing: "emails.csv" (unseen data).

After formatting, the paper is styled for presentation using the prescribed template.

TABLE I. COMP ARISION TABLE

	Classifiers	*Score 1*	*Score 2*	*Score 3*	*Score 4*
1	Support Vector Classifier	0.81	0.92	0.95	0.92
2	K-Nearest Neighbour	0.92	0.88	0.87	0.88
3	Naïve Bayes	0.87	0.98	0.98	0.98
4	Decision Tree	0.94	0.95	0.93	0.95
5	Random Forest	0.90	0.92	0.92	0.92
6	AdaBoost Classifier	0.95	0.94	0.95	0.94
7	Bagging Classifier	0.94	0.94	0.95	0.94

a. score 1: using default parameters

b. score 2: using hy perparameter tuning

c. score 3: using stemmer and hy perparameter tuning

d. score 4: using length, stemmer and hy perparameter tuning

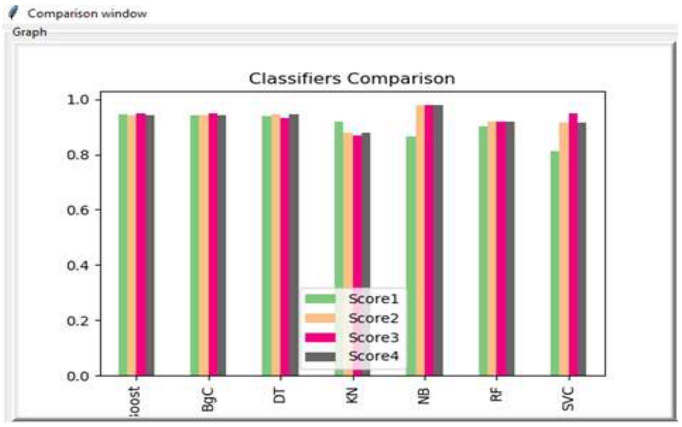


Fig.5 Comparison of all algorithms

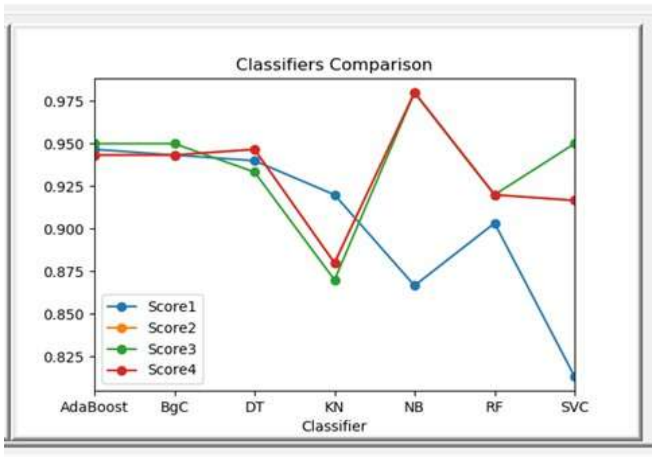


Fig.6. Comparison Graph

VI. Conclusion

The Multinomial Naïve Bayes classifier shows the best results but faces limitations due to class-conditional independence, leading to some misclassification. Ensemble methods effectively improve predictions by combining multiple classifiers.

Key Insights:

- The model efficiently detects spam based on email content, not domain names.
- Limited by the size of the corpus used for testing.

Future Improvements:

- Incorporate filtering based on trusted and verified domain names.
- Enhance spam email classification for broader applicability.
- Support large-scale organizations in differentiating desired emails.

References:

A detailed list of references is provided to support the study and findings.

1. A. K. Ameen and B. Kaya, "Spam detection in online social networks by deep learning," 2018 International Conference on Artificial Intelligence and Data Processing (IDAP), Malatya, Turkey, 2018, pp. 1-4.

2. Diren, D.D., Boran, S., Selvi, I.H., & Hati, T. (2019). Root Cause Detection with an Ensemble Machine Learning Approach in the Multivariate Manufacturing Process.