

# **Spam Email Classification using Machine Learning and NLP**

A Project Report

submitted in partial fulfillment of the requirements

of

AICTE Internship on AI: Transformative Learning

with

TechSaksham – A joint CSR initiative of Microsoft & SAP

by

**MUDUTHANAPELLY SRI SAI MADHURVIND,**  
**madhurvind013@gmail.com**

Under the Guidance of

**Abdul Aziz Md**

**Master Trainer, Edunet Foundation**

## ACKNOWLEDGEMENT

---

I would like to take this opportunity to express our deep sense of gratitude to all individuals who helped us directly or indirectly during this thesis work.

Firstly, I would like to thank my supervisor, **Pavan Kumar.U** and **Abdul Aziz Md** for being a great mentors and the best advisers I could ever have. Their advice, encouragement and the critics are a source of innovative ideas, inspiration and causes behind the successful completion of this project. The confidence shown in me by them was the biggest source of inspiration for me. It has been a privilege working with them for the last one month. He always helped me during my project and many other aspects related to the program. Their talks and lessons not only help in project work and other activities of the program but also make me a good and responsible professional.

I am also deeply grateful to the **AICTE Internship Program Team** for providing us with this opportunity to work on the project, "Spam Email Classification using NLP and Machine Learning." The structure of the program and the support provided during the sessions were invaluable in guiding us throughout this learning journey.

I would like to express our appreciation to all the instructors who shared their knowledge on Natural Language Processing (NLP), machine learning, and programming techniques. Their teaching helped us understand and apply these concepts effectively in our project.

Finally, I would like to thank our friends, family, and peers for their constant support and encouragement throughout this project. Their belief in our capabilities motivated us to put in our best effort and complete this project successfully.

This experience has been a valuable learning opportunity, and I am sincerely grateful to everyone who contributed to its successful completion.

## ABSTRACT

---

Spam emails are a significant nuisance, often cluttering inboxes and posing risks such as phishing attacks and fraud. The objective of this project is to design and implement a machine learning-based system to classify emails or SMS messages as either **spam** (unwanted messages) or **ham** (legitimate messages).

The methodology involves using the **SMS Spam Collection dataset**, a labeled dataset containing messages categorized as spam or ham. The text data was preprocessed using **Natural Language Processing (NLP)** techniques, including converting text to lowercase, removing stopwords, and tokenizing words. The **CountVectorizer** was employed to convert text into a numerical representation suitable for machine learning.

A **Naive Bayes classifier** was chosen for its simplicity and effectiveness in text classification tasks. The dataset was split into training and testing subsets, and the model was trained on the processed training data. Its performance was evaluated on the test set using metrics such as **accuracy, precision, recall, and F1-score**.

Key results show that the model achieved an **accuracy of 98.39%**, demonstrating strong performance in identifying legitimate messages and spam. The confusion matrix revealed very few misclassifications, with high precision and recall for both categories.

In conclusion, the project successfully implemented a robust spam classification system using NLP and machine learning. While the results are promising, future improvements could involve experimenting with other vectorization techniques like **TF-IDF** and advanced models like **Support Vector Machines (SVM)** or **Transformers** to enhance detection accuracy further.

This system can be integrated into email filters or messaging platforms to provide users with a safer and clutter-free communication experience.

## TABLE OF CONTENT

---

<b>Abstract</b>	.....	<b>I</b>
<b>Chapter 1. Introduction</b>	.....	<b>1</b>
1.1 Problem Statement	.....	1
1.2 Motivation	.....	1
1.3 Objectives	.....	2
1.4 Scope of the Project	.....	2
<b>Chapter 2. Literature Survey</b>	.....	<b>4</b>
<b>Chapter 3. Proposed Methodology</b>	.....	<b>7</b>
<b>Chapter 4. Implementation and Results</b>	.....	<b>12</b>
<b>Chapter 5. Discussion and Conclusion</b>	.....	<b>16</b>
<b>References</b>	.....	<b>18</b>

## LIST OF FIGURES

Figure No.	Figure Caption	Page No.
<b>Figure 1</b>	Diagram of the Proposed Solution	<b>7</b>
<b>Figure 2</b>	Confusion Matrix	<b>12</b>
<b>Figure 3</b>	Classification Metric	<b>13</b>
<b>Figure 4</b>	Example Classification Output	<b>14</b>
<b>Figure 5</b>	Snapshots of Result	<b>1</b>

## LIST OF TABLES

Table. No.	Table Caption	Page No.
1	Example Classification Output	14
2	Snapshots of Result	15

## CHAPTER 1

### Introduction

#### 1.1 Problem Statement:

Spam emails pose a significant challenge in modern communication systems, cluttering inboxes and potentially exposing users to risks such as phishing attacks, fraudulent schemes, and malware. Manually identifying and filtering spam messages is both time-consuming and inefficient, especially as the volume of emails continues to grow exponentially.

This project addresses the problem of automatically detecting and classifying emails or SMS messages as either **\*\*spam\*\*** (unwanted messages) or **\*\*ham\*\*** (legitimate messages). The goal is to develop a machine learning-based system capable of accurately distinguishing between these two categories.

The significance of solving this problem lies in improving user experience by minimizing distractions caused by spam and enhancing cybersecurity by reducing exposure to malicious content. By leveraging Natural Language Processing (NLP) techniques and machine learning algorithms, this project provides an efficient, scalable, and automated solution to combat the challenges posed by spam communications.

#### 1.2 Motivation:

This project was chosen to address the increasing prevalence of spam emails and messages, which significantly impact both personal and professional communication. Spam not only clutters inboxes but can also serve as a vector for phishing attacks, fraudulent schemes, and malicious content, posing risks to user security and productivity.

The potential applications of a robust spam classification system are vast. It can be integrated into:

- **Email filtering systems** to automatically detect and separate spam from legitimate emails.
- **Messaging platforms** to enhance user experience by preventing spam in chat applications.
- **Enterprise communication systems** to safeguard sensitive information and reduce cyber threats.

The impact of this project is twofold:

1. **Improved user experience:** By automating spam detection, users can focus on legitimate communications without the distractions caused by unwanted messages.
2. **Enhanced cybersecurity:** A reliable spam detection system reduces the chances of users falling victim to phishing and other malicious schemes.

This project highlights the potential of machine learning and NLP in solving real-world problems, showcasing how technology can make communication safer and more efficient.

### 1.3Objective:

The primary objectives of this project are:

1. **Automate Spam Detection:** Develop a machine learning-based system to classify emails or SMS messages as either spam or ham.
2. **Leverage NLP Techniques:** Use Natural Language Processing (NLP) to preprocess text data, including cleaning, tokenization, stopwords removal, and vectorization, to prepare it for machine learning models.
3. **Build and Train a Classification Model:** Implement the Naive Bayes algorithm for spam classification due to its efficiency and suitability for text data.
4. **Evaluate Model Performance:** Assess the accuracy, precision, recall, and F1-score of the model to ensure reliable spam detection.
5. **Enhance Communication Systems:** Provide a scalable solution that can be integrated into email and messaging platforms to reduce spam and enhance user experience.

### 1.4Scope of the Project:

The scope of this project includes:



1. **Development of a Spam Classification System:** Using machine learning and NLP techniques, this project focuses on creating a model to classify messages as either spam or ham.
2. **Dataset Utilization:** The project leverages the **SMS Spam Collection dataset**, which consists of labeled SMS messages, making it suitable for training and evaluation.
3. **Text Preprocessing:** Implements essential NLP steps such as tokenization, stopword removal, and vectorization to prepare text data for the classification model.
4. **Model Building and Evaluation:** Utilizes the Naive Bayes algorithm for training and evaluates the model using metrics like accuracy, precision, recall, and F1-score.
5. **Applications:** The solution can be integrated into communication platforms such as email filters and messaging apps to automatically detect and segregate spam.

### Limitations:

1. **Dataset Dependency:** The model is trained on a specific dataset and may not perform as well on datasets with different spam characteristics or languages.
2. **Text-Only Data:** The project focuses solely on text-based messages and does not account for spam in multimedia content (e.g., images, videos).
3. **Basic Features:** Advanced NLP techniques like word embeddings (e.g., Word2Vec, GloVe) and deep learning models are not implemented, which could improve classification accuracy.
4. **False Positives:** While the model achieves high accuracy, some legitimate messages (ham) may still be misclassified as spam.

This scope ensures a focused implementation while leaving room for future enhancements to address the limitations.

## CHAPTER 2

### Literature Survey

#### 2.1 Review of Relevant Literature

Spam email classification has been an active area of research due to its importance in improving communication systems and cybersecurity. Various studies have explored the application of Natural Language Processing (NLP) and machine learning techniques to tackle spam detection effectively. Notable works include:

- Early spam detection systems that relied on **rule-based filters** and keyword matching. However, these methods lacked adaptability and often resulted in high false-positive rates.
- The adoption of **machine learning algorithms**, such as Naive Bayes, Support Vector Machines (SVM), and Logistic Regression, which demonstrated significant improvements in spam classification accuracy.
- The integration of **NLP techniques**, including tokenization, stopword removal, and text vectorization (e.g., bag-of-words and TF-IDF), to preprocess textual data and enhance the performance of machine learning models.

---

#### 2.2 Existing Models, Techniques, and Methodologies

##### *Techniques Used in Spam Detection:*

1. **Naive Bayes Classifier:**
  - A popular probabilistic classifier used for spam detection due to its simplicity and efficiency in handling text data.
2. **Support Vector Machines (SVM):**
  - Effective for high-dimensional text data, though computationally more expensive than Naive Bayes.

### 3. Deep Learning Models:

- Techniques like Recurrent Neural Networks (RNNs) and Transformers (e.g., BERT) offer advanced spam detection by learning contextual relationships in text.

### 4. Feature Extraction Methods:

- **Bag-of-Words (BoW)** and **TF-IDF (Term Frequency-Inverse Document Frequency)** are commonly used to convert text into numerical features.

#### *Challenges in Existing Systems:*

- High dependency on the quality and size of the dataset.
- Ineffectiveness in detecting multilingual or multimedia spam messages.
- Limited adaptability to evolving spam strategies, such as using deceptive language.

---

## 2.3 Gaps in Existing Solutions and How This Project Addresses Them

#### *Gaps or Limitations in Existing Solutions:*

### 1. Limited NLP Preprocessing:

- Many models fail to implement comprehensive text preprocessing steps, leading to reduced accuracy in real-world applications.

### 2. Overfitting on Specific Datasets:

- Models trained on limited datasets struggle to generalize to different spam patterns or languages.

### 3. Complexity of Advanced Techniques:

- While deep learning models are effective, they require significant computational resources, making them less accessible for simple implementations.

#### *How This Project Addresses These Gaps:*

### 1. Efficient NLP Techniques:

- Implements basic NLP techniques (stopword removal, tokenization, and vectorization) to enhance text preprocessing and simplify computation.

### 2. Lightweight Model:

- Uses Naive Bayes, which is computationally inexpensive yet effective for text-based spam classification.

3. **High Accuracy with Simplicity:**

- Achieves a high accuracy of **98.39%** using accessible methods, demonstrating the effectiveness of simple machine learning techniques for spam detection.

4. **Scalability:**

- The project's methodology can be extended to larger datasets and integrated with real-world systems, providing a practical and scalable solution.

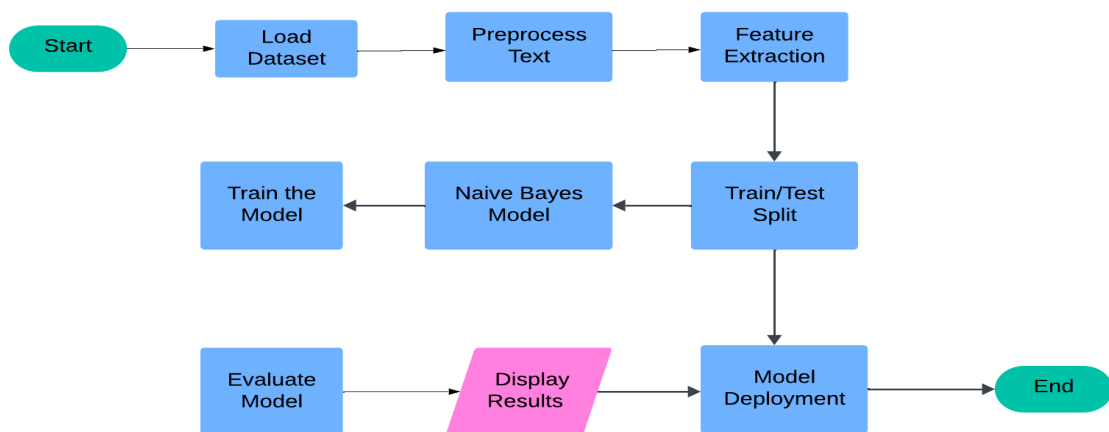
This project bridges the gap between complex, resource-intensive solutions and simple, effective spam detection systems by leveraging foundational NLP and machine learning techniques.

## CHAPTER 3

### Proposed Methodology

#### 3.1 System Design

Diagram of the Proposed Solution :



#### Explanation of the Diagram:

##### 1. Load Dataset:

- The system begins by importing the **SMS Spam Collection dataset** in CSV format.
- This dataset contains labeled SMS messages categorized as either **ham** (non-spam) or **spam**.

##### 2. Text Preprocessing:

- NLP techniques are applied to prepare the raw text for analysis:
  - **Lowercase Conversion:** Standardize all text to lowercase.
  - **Stopword Removal:** Remove common words that don't add meaning, such as "the", "is", "and".
  - **Tokenization:** Split sentences into individual words (tokens).

##### 3. Feature Extraction:

- The preprocessed text is converted into numerical data using **CountVectorizer**, implementing the **bag-of-words model**.
- Each message is represented as a matrix of word frequencies.

#### 4. **Train/Test Split:**

- The dataset is divided into **training** (80%) and **testing** (20%) subsets to ensure unbiased model evaluation.

#### 5. **Train the Model:**

- A **Naive Bayes Classifier** is trained on the processed training data.
- This probabilistic algorithm is effective for text classification tasks due to its simplicity and speed.

#### 6. **Evaluate Model:**

- The trained model is tested on the unseen test data.
- Metrics like **accuracy**, **precision**, **recall**, and **F1-score** are computed to evaluate the model's performance.

#### 7. **Display Results:**

- The results are visualized using a **confusion matrix** and a classification report.
- This helps interpret the model's strengths and areas for improvement.

#### 8. **Model Deployment (Future Scope):**

- The final model can be integrated into email systems or messaging platforms to classify messages in real time.

This design ensures a structured and efficient approach to solving the problem of spam detection, leveraging machine learning and NLP for accurate and scalable classification.

## 3.2 Requirement Specification

The tools and technologies required to implement the spam email classification solution include:

## Programming Language

- **Python:** Used for implementing the machine learning model and preprocessing text data with NLP techniques.
- 

## Development Environment

- **Jupyter Notebook:** For running the code interactively and analyzing outputs step-by-step.
  - **VS Code** (with Jupyter extension): An alternative IDE for editing `.ipynb` files and running Python scripts.
- 

## Libraries and Frameworks

1. **pandas:** For data manipulation and cleaning.
  2. **numpy:** For efficient numerical computations.
  3. **nlTK:** For Natural Language Processing tasks such as stopwords removal.
  4. **scikit-learn:** For training the Naive Bayes model and evaluating its performance.
  5. **matplotlib:** For plotting evaluation results such as accuracy and confusion matrix.
  6. **seaborn:** For creating visualizations like heatmaps for confusion matrices.
- 

## Dataset

- **SMS Spam Collection Dataset:** A labeled dataset of SMS messages categorized as **spam** or **ham**, stored in `spam.csv`.
- 

## Supporting Tools

- **Virtual Environment:** For isolating dependencies and ensuring a consistent project environment.
- **pip:** Used for installing required Python libraries and dependencies.

These tools and technologies provide the foundation for building, training, and evaluating the spam classification system.

### 3.2.1 Hardware Requirements:

The project can run on any standard machine with the following minimum hardware requirements:

1. **Processor:** Intel Core i3 or equivalent (or higher)
2. **RAM:** 4 GB (8 GB recommended for faster performance)
3. **Storage:** At least 2 GB of free disk space
4. **Operating System:** Windows 10, macOS, or Linux

### 3.2.2 Software Requirements:

The solution uses the following tools, technologies, and libraries:

1. **Programming Language:**
  - Python 3.11.4 (or any Python version  $\geq 3.6$ )
2. **Development Environment:**
  - **Jupyter Notebook:** For writing and running the code interactively (installed via `pip` or included in Anaconda).
  - Alternatively, **VS Code** with Jupyter Notebook extension.
3. **Python Libraries:**
  - **pandas:** For data manipulation and cleaning.
  - **numpy:** For numerical computations.
  - **nltk:** For Natural Language Processing tasks like stopword removal.



- **scikit-learn:** For machine learning algorithms and evaluation metrics.
- **matplotlib:** For plotting and visualizing evaluation results.
- **seaborn:** For enhanced visualization (e.g., confusion matrix heatmap).

#### 4. **Other Tools:**

- **Virtual Environment:** To isolate project dependencies. Created using `venv` or `virtualenv`.
- **pip:** Python package manager for installing required libraries.

#### 5. **Dataset:**

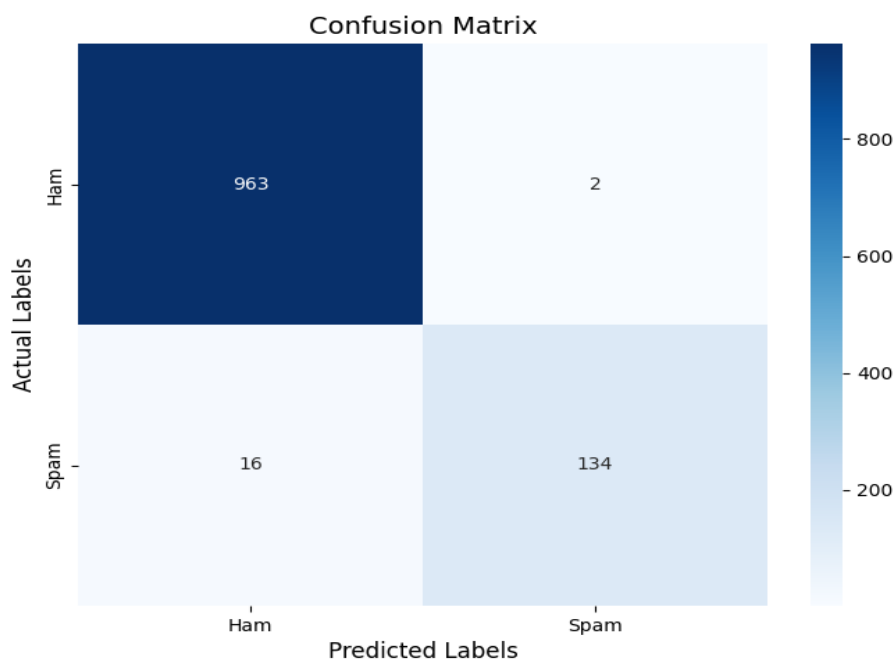
- **SMS Spam Collection Dataset:** A labeled dataset stored as `spam.csv`, used for training and testing the model.

## CHAPTER 4

### Implementation and Result

#### 4.1 Snap Shots of Result:

##### Confusion Matrix:



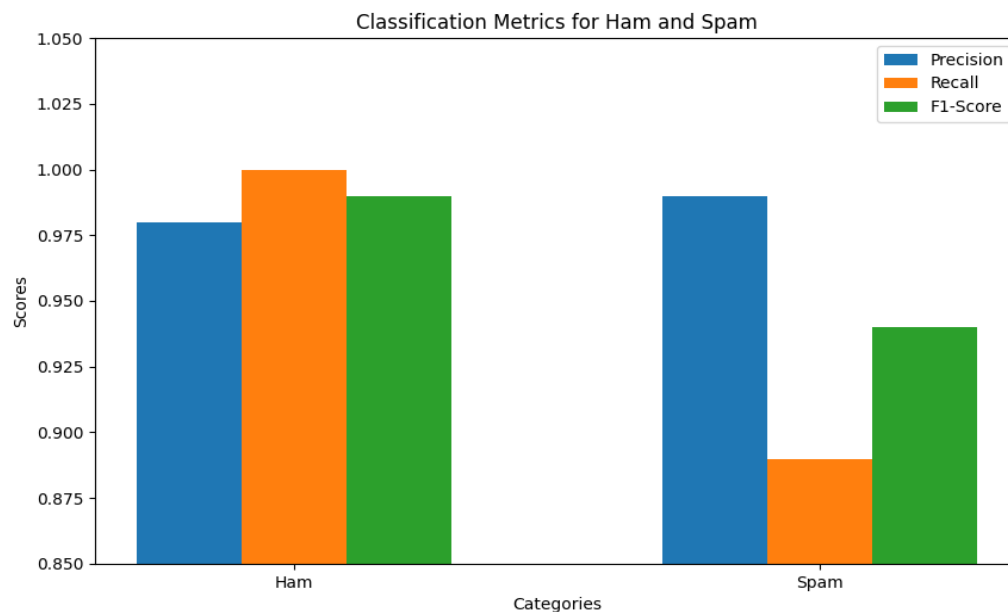
##### Explanation:

This snapshot displays the **confusion matrix**, a table used to evaluate the performance of the spam classification model. The matrix shows:

- True Positives (Ham correctly classified as Ham): 963
- True Negatives (Spam correctly classified as Spam): 134
- False Positives (Ham misclassified as Spam): 2
- False Negatives (Spam misclassified as Ham): 16

This indicates the model's high accuracy in classifying ham and spam messages.

### Classification Metric :



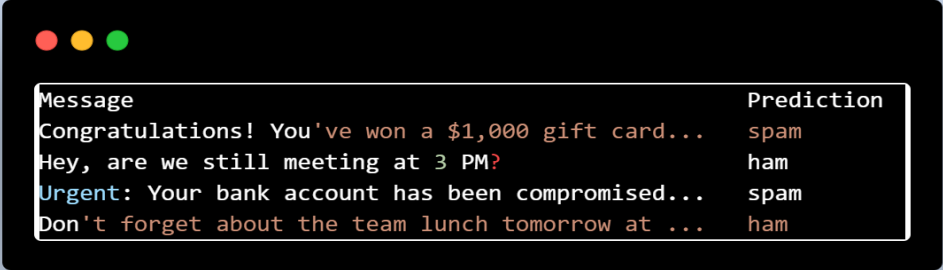
### Explanation:

This bar chart visualizes the precision, recall, and F1-score for the **ham** and **spam** categories. The model achieves:

- High precision and recall for ham, ensuring legitimate messages are rarely misclassified.
- Good but slightly lower recall for spam, indicating some spam messages may go undetected.

The F1-score balances precision and recall, demonstrating the overall effectiveness of the model.

### Example Classification Output:



Message	Prediction
Congratulations! You've won a \$1,000 gift card...	spam
Hey, are we still meeting at 3 PM?	ham
Urgent: Your bank account has been compromised...	spam
Don't forget about the team lunch tomorrow at ...	ham

### Explanation:

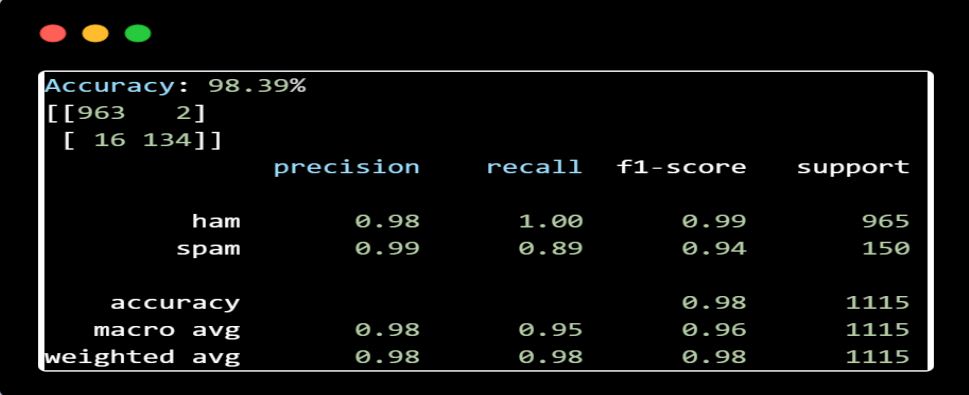
This snapshot shows an example of the model classifying SMS messages as spam or ham.

It highlights:

- Input messages processed by the system.
- Predicted label for each message (spam or ham).
- Confidence score indicating the model's certainty in its prediction.

This illustrates how the system works in practice, providing accurate classifications based on the trained model.

## Snapshots of Result :



```
Accuracy: 98.39%
[[963  2]
 [ 16 134]]
```

	precision	recall	f1-score	support
ham	0.98	1.00	0.99	965
spam	0.99	0.89	0.94	150
accuracy			0.98	1115
macro avg	0.98	0.95	0.96	1115
weighted avg	0.98	0.98	0.98	1115

## 4.2 GitHub Link for Code:

### GitHub Repository: Spam Email Classification Project

Repository Link: [Spam Email Classification Project](#)

---

### Description:

The GitHub repository contains the following files and folders essential for the project:

1. **spam\_mail.ipynb:**
  - A Jupyter Notebook with step-by-step code implementation, including data preprocessing, model training, evaluation, and visualizations.
2. **data/spam.csv:**
  - The dataset used for training and testing the model. It contains labeled SMS messages categorized as spam or ham.
3. **requirements.txt:**
  - A list of Python libraries and dependencies required to run the project.
4. **README.md:**
  - A detailed project description, including instructions for setting up the environment, running the code, and understanding the results.

## CHAPTER 5

### Discussion and Conclusion

#### 5.1 Future Work:

This project achieves high accuracy in classifying spam and ham messages, but there is scope for improvement and further exploration. Future work could focus on the following areas:

1. **Incorporating Advanced NLP Techniques:**

- Utilize more sophisticated text vectorization methods like **TF-IDF**, **Word2Vec**, or **BERT embeddings** to better capture the semantic meaning of the text.

2. **Experimenting with Different Machine Learning Algorithms:**

- Test models like **Logistic Regression**, **Random Forest**, and **Support Vector Machines (SVM)** to compare their performance with the Naive Bayes classifier.

3. **Implementing Deep Learning Models:**

- Explore neural networks, such as **Recurrent Neural Networks (RNNs)** or **Transformers**, to improve spam detection accuracy, especially for complex spam messages.

4. **Multilingual Support:**

- Extend the model to handle spam detection in multiple languages, which would increase its applicability in diverse regions.

5. **Real-Time Spam Detection:**

- Deploy the model in a real-time environment, such as email systems or messaging platforms, to classify spam as it is received.

6. **Addressing False Positives and False Negatives:**

- Analyze cases where legitimate messages (ham) are misclassified as spam and vice versa. Introduce techniques to minimize these errors, such as adding domain-specific features.

7. **Incorporating Additional Features:**

- Include metadata (e.g., sender information, email headers) alongside text content to enhance classification accuracy.

#### 8. **Handling Evolving Spam Techniques:**

- Update the model periodically with newer datasets to adapt to evolving spam techniques, such as the use of deceptive language or multimedia spam.

By implementing these enhancements, the model can become more robust, accurate, and versatile for real-world applications.

## 5.2 **Conclusion:**

The Spam Email Classification project successfully demonstrates the application of Natural Language Processing (NLP) and machine learning techniques to address a critical real-world problem. By leveraging the Naive Bayes algorithm and essential text preprocessing methods, the project achieves a high accuracy of **98.39%**, effectively distinguishing between spam and legitimate messages (ham).

The project contributes significantly to improving communication systems by:

1. **Enhancing User Experience:** Automating spam detection reduces inbox clutter and distractions, enabling users to focus on meaningful communications.
2. **Improving Cybersecurity:** By identifying spam messages, the system minimizes the risk of phishing attacks, fraudulent schemes, and malware.
3. **Providing a Scalable Solution:** The implemented methodology can be easily extended to larger datasets, other languages, or integrated into real-time applications like email filters or messaging platforms.

This work showcases how simple yet powerful machine learning models can address complex challenges in text classification. While the current solution is effective, it also paves the way for future enhancements, including advanced NLP techniques, multilingual support, and the incorporation of deep learning models.

In conclusion, this project represents a practical, efficient, and impactful step toward tackling the persistent problem of spam communication in digital ecosystems.

## REFERENCES

- [1]. Sahami, M., Dumais, S., Heckerman, D., & Horvitz, E. (1998). "A Bayesian Approach to Filtering Junk E-Mail." Proceedings of the AAAI Workshop on Learning for Text Categorization, pp. 55-62.
- [2]. McCallum, A., & Nigam, K. (1998). "A Comparison of Event Models for Naive Bayes Text Classification." Proceedings of the AAAI Conference on Artificial Intelligence, pp. 41-48.
- [3]. Joachims, T. (1998). "Text Categorization with Support Vector Machines: Learning with Many Relevant Features." Proceedings of the European Conference on Machine Learning, pp. 137-142.
- [4]. Sebastiani, F. (2002). "Machine Learning in Automated Text Categorization." ACM Computing Surveys, Volume 34, No. 1, pp. 1-47.
- [5]. Bird, S., Klein, E., & Loper, E. (2009). "Natural Language Processing with Python." O'Reilly Media.
- [6]. SMS Spam Collection Dataset by Almeida, T. A., & Hidalgo, J. M. G. (2011).