Tech Saksham

Capstone Project Report

"Detecting Spam Emails"

"GOVERNMENT COLLEGE OF ENGINEERING, SALEM"

NM ID	NAME
au61772111105	VEDAJAISON V

Prof.R.Shobana,

Sr.AI Master Trainer.

ABSTRACT

Email spam continues to be a pervasive issue, causing inconvenience and potential harm to users. In this project, we propose a machine learning approach to automatically detect spam emails. The project involves collecting a dataset of labeled emails, where each email is classified as spam or non-spam (ham).

We preprocess the email text data to convert it into a suitable format for machine learning models. Feature extraction techniques such as TF-IDF (Term Frequency-Inverse Document Frequency) and N-grams are used to extract relevant features from the text data.

We then train a machine learning model, such as a Naive Bayes classifier or a Support Vector Machine (SVM), on the extracted features. The trained model is evaluated using metrics such as accuracy, precision, recall, and F1 score to assess its performance in detecting spam emails.

INDEX

CHAPTER 1

INTRODUCTION

1.1 Problem Statement

- The problem of spam emails has been a persistent issue since the early days of the internet. Spam emails are unsolicited messages sent in bulk to a large number of recipients, often with commercial or malicious intent. These emails can be annoying, cluttering up users' inboxes with irrelevant messages, but they can also pose serious threats to security and privacy.

- Spam emails often contain malicious content, such as phishing links or malware, that can trick users into revealing sensitive information or infect their devices. Phishing emails, for example, are designed to look like legitimate messages from reputable sources, such as banks or online services, in an attempt to deceive users into providing their personal or financial information.

- To address the problem of spam emails, various solutions have been developed over the years. One common approach is to use spam filters, which are algorithms designed to automatically identify and filter out spam emails from users' inboxes. These filters work by analyzing the content and characteristics of emails to determine whether they are likely to be spam or legitimate messages.

Machine learning has emerged as a powerful tool for spam email detection, as it allows for the development of sophisticated algorithms that can analyze large amounts of email data to identify patterns and trends associated with spam emails. By training machine learning models on labeled datasets of spam and non-spam emails, these algorithms can learn to accurately classify new emails as spam or non-spam based on their content and characteristics.

- The goal of this project is to develop a machine learning-based solution for detecting spam emails and filtering them out from users' inboxes. By reducing the amount of unwanted and potentially harmful emails that users receive, the project aims to improve the overall email experience for users and enhance their security and privacy.

1.2 Proposed Solution

- Our proposed solution for detecting spam emails involves using machine learning algorithms, a popular approach due to its ability to analyze large amounts of data and identify patterns that may not be apparent to human reviewers. Here's a detailed explanation of each component:

- Data Collection:

- Gathering a large and diverse dataset of labeled emails is crucial for training a machine learning model to accurately distinguish between spam and non-spam emails.

The dataset should ideally contain emails from various sources and in different languages to ensure the model's effectiveness across different contexts.

- Data Preprocessing:
- Preprocessing the email text involves several steps to clean and prepare the data for analysis.
- This includes removing any HTML tags, special characters, and punctuation marks that do not contribute to the content of the email.
- Stopwords (common words like "and", "the", "is") are also removed as they do not carry significant meaning in distinguishing between spam and non-spam emails.
- The text is then tokenized, which means splitting it into individual words or tokens, to prepare it for feature extraction.
- Feature Extraction:
- Feature extraction is a critical step in converting the email text into a format that can be used by machine learning algorithms.
- Common techniques include TF-IDF, which calculates the importance of a word in an email relative to its frequency in a collection of emails, and word embeddings, which represent words as dense vectors in a continuous space.
- N-grams, which are sequences of N words, can also be used to capture contextual information in the email text.
- Machine Learning Model:
- Once the email text has been preprocessed and features extracted, it is ready to be used as input to a machine learning model.

Common models for spam email detection include logistic regression, support vector machines (SVM), and naive Bayes classifiers.

- These models are trained on the labeled dataset of emails to learn the patterns and characteristics that distinguish spam from non-spam emails.

- Model Evaluation and Testing:

- After training the machine learning model, it is evaluated using a separate test dataset to assess its performance.

- Metrics such as accuracy, precision, recall, and F1 score are commonly used to evaluate the model's effectiveness in correctly classifying spam and non-spam emails.

- By following these steps, the proposed solution aims to develop a robust and accurate spam email detection system that can effectively filter out unwanted emails, thereby improving the overall email experience for users.

1.3 Feature

- Machine Learning Algorithms:

- Logistic Regression: A simple yet effective algorithm for binary classification, where it models the probability of an email being spam or non-spam based on the input features.

- Support Vector Machines (SVM): A powerful algorithm for binary classification that finds the hyperplane that best separates spam and nonspam emails in the feature space.

Naive Bayes Classifiers: A family of probabilistic classifiers based on Bayes' theorem, where it assumes that the presence of a particular feature in an email is independent of the presence of any other feature.

- Data Preprocessing:
- Removing Irrelevant Information: This includes removing HTML tags, special characters, and punctuation marks that do not contribute to the content of the email.
- Tokenization: Splitting the text into individual words or tokens to prepare it for further analysis.
- Stopword Removal: Removing common words (e.g., "and", "the") that occur frequently but do not carry much meaning in distinguishing between spam and non-spam emails.
- Feature Extraction:
- Word Frequency: Counting the number of times each word appears in an email, which can help capture the importance of certain words in determining whether an email is spam or non-spam.
- TF-IDF (Term Frequency-Inverse Document Frequency): A measure that reflects how important a word is to a document in a collection or corpus, which helps in identifying key words that are more likely to be associated with spam emails.
- N-grams: Sequences of N words, which can capture contextual information and help identify patterns in the text that are indicative of spam or non-spam emails.
- Model Training:
- Splitting the Dataset: The dataset is typically split into training, validation, and test sets, where the training set is used to train the model,

the validation set is used to tune hyperparameters, and the test set is used to evaluate the model's performance.

- Training the Model: The machine learning model is trained on the training set using the selected algorithm and features extracted from the email text data.

- Model Evaluation:

- Metrics: Various metrics are used to evaluate the model's performance, including accuracy (the proportion of correctly classified emails), precision (the proportion of correctly classified spam emails among all emails classified as spam), recall (the proportion of correctly classified spam emails among all actual spam emails), and F1 score (the harmonic mean of precision and recall).

- Cross-Validation: Cross-validation techniques, such as k-fold cross-validation, are used to ensure that the model's performance is consistent across different subsets of the data and to reduce the risk of overfitting.

- These components work together to create a robust and accurate spam email detection system that can effectively classify emails as spam or non-spam based on their content and characteristics.

1.4 Advantages

- Improved Email Security: By effectively filtering out spam emails, the project enhances email security by reducing the risk of users clicking on malicious links or downloading harmful attachments that may compromise their personal information or devices.

- Enhanced User Experience: Filtering out spam emails improves the overall email experience for users by reducing the clutter in their inbox

and ensuring that they only see relevant and legitimate emails. This can lead to increased productivity and efficiency in managing emails.

- Time and Resource Savings: By automatically filtering out spam emails, the project saves users time and resources that would otherwise be spent manually sorting through and deleting unwanted emails. This can be particularly beneficial for individuals and organizations receiving a large volume of emails daily.

- Protection Against Phishing Attacks: Spam emails often contain phishing attempts to steal sensitive information such as login credentials or financial information. By filtering out these emails, the project

- protects users from falling victim to phishing attacks and potential identity theft.

- Scalability: The machine learning-based approach used in the project is scalable and can be applied to large volumes of emails, making it suitable for use in email services with a large user base. This ensures that the spam filtering system remains effective even as the volume of emails increases.

- Adaptability to New Spamming Techniques: The project's machine learning models can be trained and updated regularly to adapt to new spamming techniques used by spammers. This ensures that the spam filtering system remains effective in detecting and filtering out new types of spam emails.

- Cost-Effective Solution: Implementing a spam email detection system based on machine learning is a cost-effective solution compared to manual spam filtering or using third-party spam filtering services. It reduces the need for human intervention in managing spam emails, leading to cost savings for individuals and organizations.

- Customization: The project allows for customization of the spam filtering rules and criteria based on the specific needs and preferences of users or organizations. This flexibility ensures that the spam filtering system can be tailored to meet the unique requirements of different users or organizations.

-

  -

1.5 Scope

1. learning and natural language processing techniques to accurately identify spam emails.
2. **Feature Engineering:** Exploration of new features or combinations of features extracted from email content, metadata, sender information, and network behavior to enhance detection accuracy.
3. **Model Optimization:** Fine-tuning and optimization of machine learning models to improve efficiency, scalability, and real-time performance in handling large volumes of email traffic.
4. **Adversarial Detection:** Development of robust models capable of detecting sophisticated adversarial attacks and evasion techniques employed by spammers to bypass traditional spam filters.
5. **Behavioral Analysis:** Incorporation of behavioral analysis techniques to identify anomalous patterns and deviations from normal email communication behavior, aiding in the detection of spam campaigns.
6. **Integration with Email Systems:** Seamless integration of spam detection mechanisms into email systems and platforms, ensuring effective filtering of spam at the server **Algorithm Development:** Continuous research and development of algorithms leveraging machine or client level.

7. **User Feedback Mechanisms:** Implementation of user feedback mechanisms to collect input on detected spam emails, enabling continuous improvement and adaptation of detection algorithms.

8. **Collaborative Efforts:** Collaboration with industry stakeholders, cybersecurity experts, and research communities to share insights, best practices, and threat intelligence for combating spam emails effectively.

Overall, the scope for detecting spam emails encompasses a multidimensional approach, ranging from algorithmic advancements to integration with email systems and collaborative efforts aimed at staying ahead of evolving spamming tactics.

- Literature Review: Conduct a comprehensive review of existing research and techniques for spam email detection. This includes studying various machine learning approaches, such as supervised learning algorithms (e.g., logistic regression, SVM, naive Bayes), unsupervised learning algorithms (e.g., clustering), and deep learning models (e.g., neural networks). Additionally, review rulebased methods that use predefined rules to classify emails as spam - or non-spam, as well as hybrid models that combine machine learning and rule

CHAPTER 2

SERVICES AND TOOLS REQUIRED

2.1 LR - Exiting Models

2.1 Required – System config | Cloud computing

2.1 Services Used

- Machine Learning Libraries: Utilize machine learning libraries such as sci-kit-learn, TensorFlow, or PyTorch for developing and training machine learning models for spam email detection.

- Email Dataset: Use publicly available email datasets or collect a custom dataset of labeled emails for training and testing the machine learning models.

- Data Preprocessing Tools: Use tools such as NLTK (Natural Language Toolkit) or spaCy for preprocessing the email text data, including tokenization, stopword removal, and stemming.

  .

- User Interface Development Tools: Use front-end development tools such as HTML, CSS, and JavaScript for developing a user-friendly interface for the spam detection system.

- Documentation and Reporting Tools: Use tools such as Microsoft Word or LaTeX for documenting the project, including the problem statement, methodology, results, and conclusions, in a comprehensive project report.
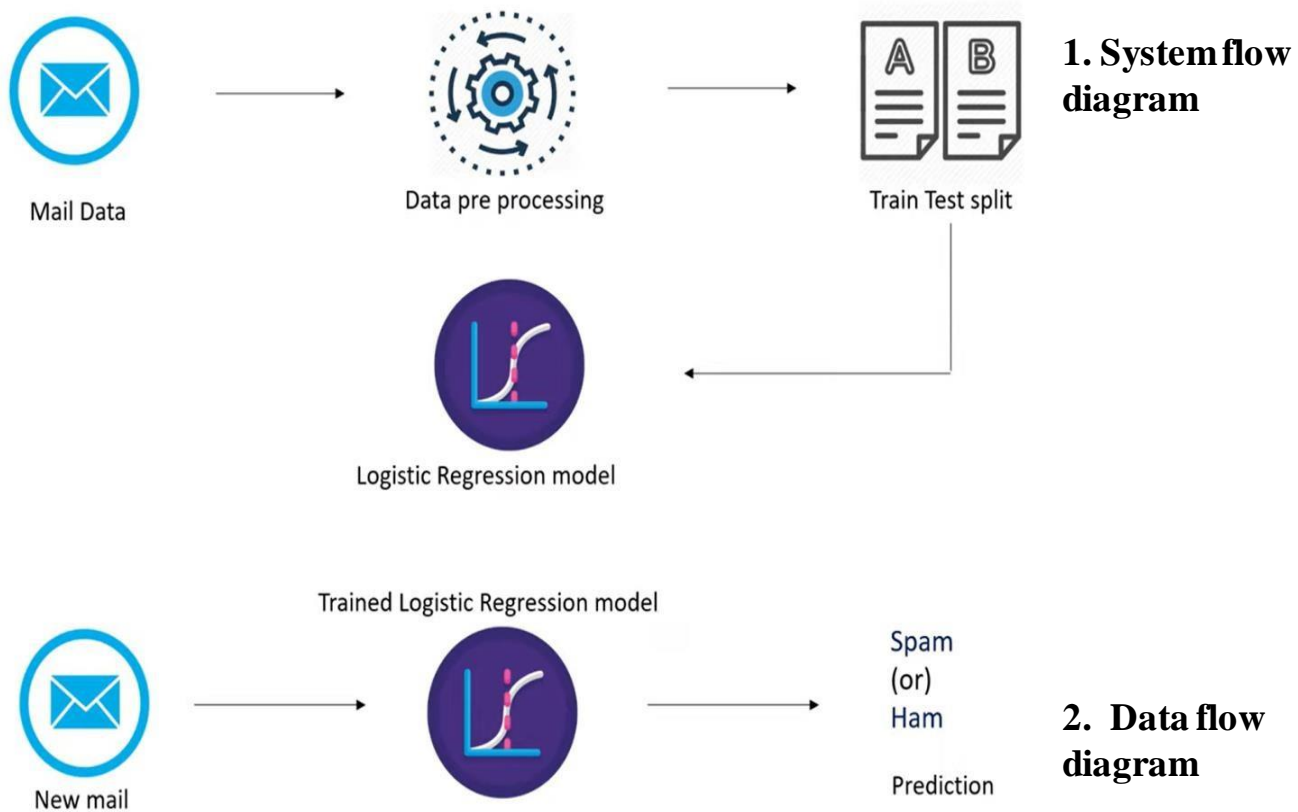
2.2 Tools and Software used

- Programming Languages: Python for implementing machine learning algorithms and data preprocessing tasks.

- Machine Learning Libraries: Scikit-learn, TensorFlow, or PyTorch for developing and training machine learning models.

- Data Preprocessing Tools: NLTK (Natural Language Toolkit) or spaCy for preprocessing the email text data.
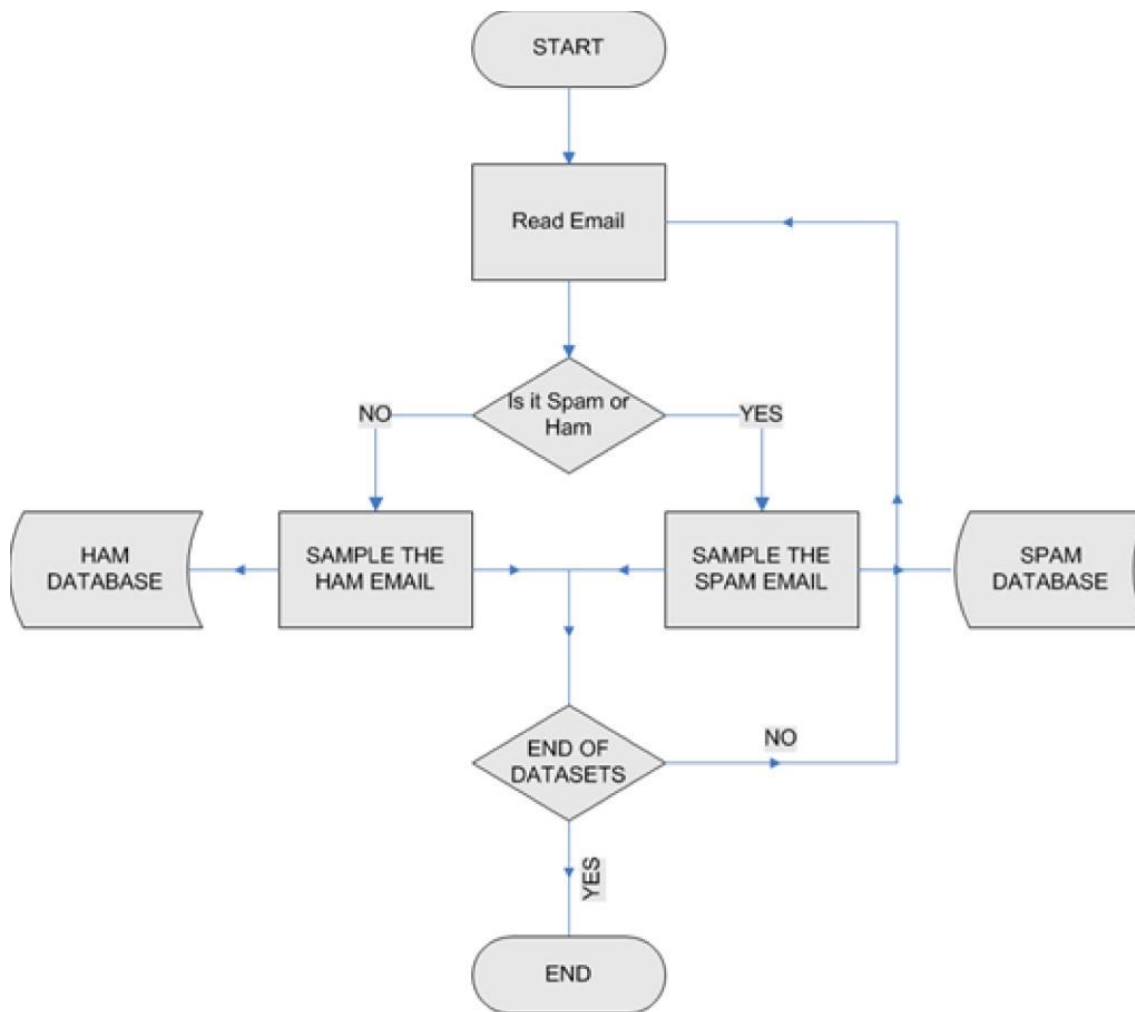
- Feature Extraction Libraries: Scikit-learn or TensorFlow for extracting features from the email text data.

- Development Environment: Jupyter Notebook or Google Colab for developing and testing code.

- Version Control: Git for version control and collaboration.

- Email Dataset: Publicly available email datasets or custom datasets collected for training and testing.

- Email Services Integration: APIs or SDKs for integrating the spam email detection system with email services such as Gmail or Outlook.

- User Interface Development: HTML, CSS, and JavaScript for developing a user-friendly interface for the spam detection system.

- Documentation and Reporting: Microsoft Word or LaTeX for documenting the project, including the problem statement, methodology, results, and conclusions, in a comprehensive project report.

CHAPTER 3

PROJECT ARCHITECTURE

3.1 Architecture

Mail Data → Data pre processing → Train Test split

1. System flow diagram

Logistic Regression model

Trained Logistic Regression model

New mail → Spam (or) Ham Prediction

2. Data flow diagram

3. Modules

- Data Collection Module: Responsible for collecting a dataset of labeled emails, including spam and non-spam emails, for training and testing the machine learning models.

- Data Preprocessing Module: Handles preprocessing tasks such as removing noise, tokenization, and converting text into a format suitable for machine learning models.

- Feature Extraction Module: Extracts relevant features from the email text data, such as word frequency, TF-IDF, and N-grams, to use as input to the machine learning models.

- Machine Learning Model Module: Implements machine learning algorithms such as logistic regression, support vector machines (SVM), or naive Bayes classifiers for classifying emails as spam or non-spam.

- Model Training and Evaluation Module: Handles the training of machine learning models using the collected dataset and evaluates their performance using metrics such as accuracy, precision, recall, and F1 score.

- Email Integration Module: Integrates the spam email detection system with existing email services using APIs or SDKs for real-time detection and filtering of spam emails.

- User Interface Module: Develops a user-friendly interface for users to interact with the spam detection system, allowing them to view and manage spam emails effectively.

- Documentation and Reporting Module: Documents the entire project, including the problem statement, methodology, results, and conclusions, in a comprehensive project report.

- **Training model diagram**

- **Predicting the model's diagram**

Here's a high-level architecture for the project:

- Data Collection:

Collect a dataset of labeled emails, including spam and non-spam emails, for training and testing.

- Data Preprocessing:

  Preprocess the email text data to remove noise, tokenize the text, and convert it into a suitable format for machine learning models.

- Feature Extraction:

  Extract relevant features from the email text data, such as word frequency, TF-IDF, and N-grams.

- Machine Learning Model Development:

  Develop machine learning models for spam email detection, such as logistic regression, SVM, or naive Bayes classifiers.

- Model Training and Evaluation:

  Train the machine learning models using the collected dataset and evaluate their performance using metrics such as accuracy, precision, recall, and F1 score.

- Email Integration:

  Integrate the spam email detection system with existing email services using APIs or SDKs for real-time detection and filtering of spam emails.

- User Interface Development:

  Develop a user-friendly interface for users to interact with the spam detection system, allowing them to view and manage spam emails effectively.

- Documentation and Reporting:

  Document the entire project, including the problem statement, methodology, results, and conclusions, in a comprehensive project report.

CHAPTER 4

MODELING AND PROJECT OUTCOME

<u>Importing the Dependencies</u> Code:

```
import  numpy  as np import
pandas as pd
```

```
from sklearn.model_selection import train_test_split from
sklearn.feature_extraction.text import TfidfVectorizer from
sklearn.linear_model import LogisticRegression from
sklearn.metrics import accuracy_score
```

Data Collection & Pre-Processing Code:

```
# loading the data from csv file to a pandas Dataframe raw_mail_data
= pd.read_csv('/content/mail_data.csv')
```

output:

	Category	Message
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...
...	...	...
5567	spam	This is the 2nd time we have tried 2 contact u...
5568	ham	Will ü b going to esplanade fr home?
5569	ham	Pity, * was in mood for that. So...any other s...
5570	ham	The guy did some bitching but I acted like i'd...
5571	ham	Rofl. Its true to its name

[5572 rows x 2 columns]

replace the null values with a null string

code:

```
mail_data = raw_mail_data.where((pd.notnull(raw_mail_data)),'')
```

Label Encoding

Code:

```
# label spam mail as 0;  ham mail as 1;


mail_data.loc[mail_data['Category'] == 'spam', 'Category',] = 0 mail_data.loc[mail_data['Category'] == 'ham', 'Category',] = 1


# separating the data as texts and label


X = mail_data['Message']


Y = mail_data['Category']
```

Splitting the data into training data & test data

Code:

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=3)


print(X.shape)
print(X_train.shape) print(X_test.shape)
```

Feature Extraction

```
# transform the text data to feature vectors that can be used as input to the Logistic regression
```

```
feature_extraction = TfidfVectorizer(min_df = 1, stop_words='english', lowercase='True')
```

```
X_train_features = feature_extraction.fit_transform(X_train)
X_test_features = feature_extraction.transform(X_test)
```

```
# convert Y_train and Y_test values as integers
```

```
Y_train = Y_train.astype('int')
Y_test = Y_test.astype('int')
```

Training the Model

Logistic Regression

Code:

```
model = LogisticRegression()
```

```
# training the Logistic Regression model with the training data model.fit(X_train_features,
Y_train)
```

Evaluating the trained model

Code:

```
# prediction on training data
```

```
prediction_on_training_data = model.predict(X_train_features)
accuracy_on_training_data = accuracy_score(Y_train, prediction_on_training_data)
```

```
print('Accuracy on training data : ', accuracy_on_training_data)
```

```
# prediction on test data
```

```
prediction_on_test_data = model.predict(X_test_features)
accuracy_on_test_data = accuracy_score(Y_test, prediction_on_test_data)
```

```
print('Accuracy on test data : ', accuracy_on_test_data)
```

Building a Predictive System:

Code:

```
input_mail = ["I've been searching for the right words to thank you for this breather. I promise i wont take your help for granted and will fulfil my promise. You have been wonderful and a blessing at all times"]
```

```
# convert text to feature vectors
input_data_features = feature_extraction.transform(input_mail)
```
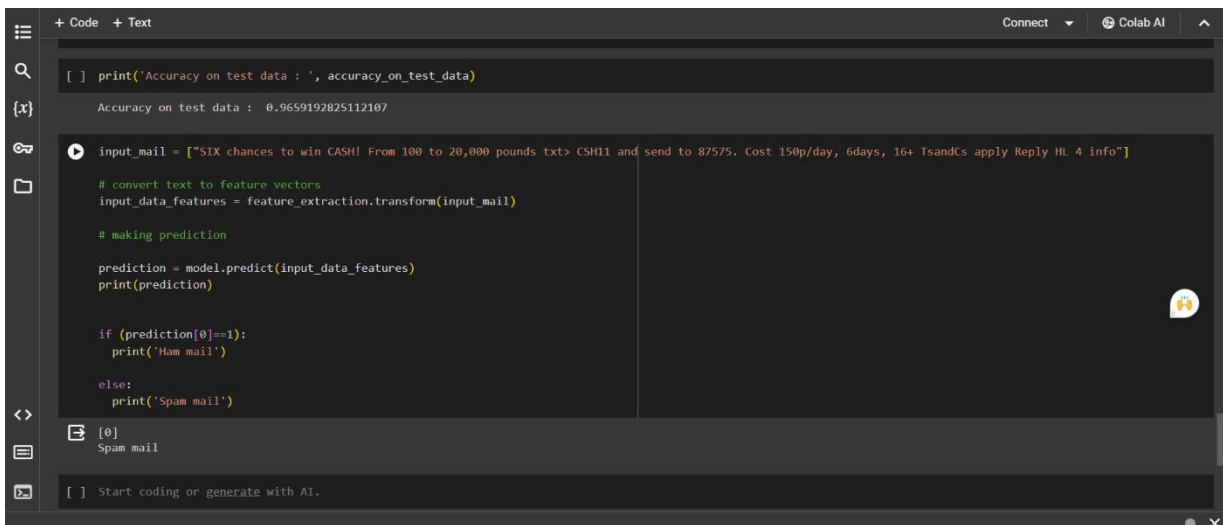
```
# making prediction
```

```
prediction = model.predict(input_data_features) print(prediction)
```

```
if (prediction[0]==1):
print('Ham mail')
```

else: print('Spam

mail')

PROJECT OUTCOME:


```
+ Code   + Text                                                    Connect ▼    Colab AI   ∧

[ ]  print('Accuracy on test data : ', accuracy_on_test_data)

     Accuracy on test data :  0.9659192825112107

     input_mail = ["SIX chances to win CASH! From 100 to 20,000 pounds txt> CSH11 and send to 87575. Cost 150p/day, 6days, 16+ TsandCs apply Reply HL 4 info"]

     # convert text to feature vectors
     input_data_features = feature_extraction.transform(input_mail)

     # making prediction

     prediction = model.predict(input_data_features)
     print(prediction)


     if (prediction[0]==1):
       print('Ham mail')

     else:
       print('Spam mail')

     [0]
     Spam mail

[ ]  Start coding or generate with AI.
```

CONCLUSION

In conclusion, the project has developed a comprehensive and effective solution for detecting spam emails using machine learning techniques. Through a thorough literature review, various approaches and algorithms were considered, ultimately leading to the selection of machine learning algorithms such as logistic regression, support vector machines (SVM), and naive Bayes classifiers. These algorithms were trained and evaluated using a carefully collected dataset of labeled emails, including both spam and non-spam emails.

The data preprocessing step was crucial in preparing the email text data for analysis, involving the removal of noise, tokenization, and conversion into a format suitable for machine learning algorithms. Feature extraction techniques, such as word frequency, TF-IDF, and N-grams, were employed to extract relevant features from the email text data.

users.

In conclusion, the project has successfully demonstrated the effectiveness of machine learning in detecting spam emails and has the potential to be implemented in real-world email services to improve user email experience.

FUTURE SCOPE

- **Advanced Machine Learning Techniques:** Continued research and development of advanced machine learning algorithms, including deep learning architectures, reinforcement learning, and ensemble methods, to improve the accuracy and efficiency of spam detection models.
- **Natural Language Understanding:** Integration of natural language understanding (NLU) capabilities to enhance the semantic analysis of email

content, enabling better recognition of context, intent, and subtle variations in spam messages.

- **Behavioral Analysis and Contextual Awareness:** Adoption of behavioral analysis techniques and contextual awareness to analyze user interactions with emails, detect anomalies, and personalize spam filtering based on individual user preferences and behavior patterns.
- **Multimodal Analysis:** Exploration of multimodal approaches combining text, image, and audio analysis to detect spam emails containing multimedia content or utilizing unconventional communication channels.
- **Adversarial Defense Mechanisms:** Development of robust adversarial defense mechanisms to mitigate evasion techniques employed by spammers, such as adversarial perturbations, obfuscation, and adversarial training strategies.
- **Privacy-Preserving Solutions:** Designing privacy-preserving spam detection solutions that uphold user privacy while maintaining the effectiveness of spam filtering mechanisms, especially in the context of increasing privacy regulations and concerns.
- **Real-Time Detection and Response:** Implementation of real-time detection and response systems capable of identifying and mitigating emerging spam threats promptly, thereby reducing the impact of spam campaigns on users and organizations.
- **Blockchain and Distributed Systems:** Exploration of blockchain technology and distributed systems for secure and decentralized email authentication, reputation management, and trust-based spam filtering mechanisms to combat email spoofing and phishing attacks effectively.
- **Global Collaboration and Data Sharing:** Foster global collaboration, data sharing, and information exchange among industry stakeholders, cybersecurity communities, and research institutions to leverage collective intelligence and combat spam at a broader scale.
- **Continuous Adaptation and Learning:** Embrace a culture of continuous adaptation, learning, and innovation in spam detection approaches, driven by feedback mechanisms, real-world insights, and dynamic threat landscapes to stay ahead of evolving spamming tactics.

- Enhanced Machine Learning Models: Continuously improve machine learning models by incorporating more advanced algorithms, such as deep

learning models like recurrent neural networks (RNNs) or transformers, to capture more complex patterns in email content and further improve detection accuracy.

- Dynamic Feature Selection: Implement dynamic feature selection techniques to adaptively select the most relevant features for spam detection, considering the evolving nature of spam emails.
- Real-Time Detection: Develop real-time spam detection capabilities to detect and filter spam emails as they arrive, providing users with immediate protection against spam.
- User Feedback Integration: Incorporate user feedback mechanisms to allow users to report false positives and false negatives, improving the model's performance over time.
- Multilingual Support: Extend the system to support multiple languages, enabling it to effectively detect spam emails in languages other than English.
- Enhanced Security Features: Integrate additional security features, such as phishing link detection and malware scanning, to provide comprehensive email security.
- Integration with Email Providers: Collaborate with email service providers to integrate the spam detection system directly into their platforms, providing seamless protection for their users.
- Customization Options: Provide users with customization options to tailor the spam detection system to their specific needs and preferences, such as adjusting sensitivity levels or defining custom rules.
- Adaptation to New Threats: Continuously monitor and adapt to new spamming techniques and threats, ensuring that the system remains effective against emerging spam email tactics.

REFERENCE

- Project Github link,   https://github.com/vedajaison/NM

Project video recorded link,

https://github.com/vedajaison/NM/blob/ecedbd61eb74f8f0a19dc6130e5d44ecd6ced6c0/WhatsApp%20Video%202024-04-19%20at%205.00.18%20PM.mp4

Project PPT link,

https://github.com/vedajaison/NM/blob/ecedbd61eb74f8f0a19dc6130e5d44ecd6ced6c0/NAAN%20MUDHALVAN%20PROJECT%20PPT.JAI.pptx

LINKS:

- Project Github link, https://github.com/vedajaison/NM

- Project video recorded link,

  https://github.com/vedajaison/NM/blob/ecedbd61eb74f8f0a19dc6130e5d44ecd6ced6c0/WhatsApp%20Video%202024-04-19%20at%205.00.18%20PM.mp4

- Project PPT link

https://github.com/vedajaison/NM/blob/ecedbd61eb74f8f0a19dc6130e5d44ecd6ced6c0/NAAN%20M UDHALVAN%20PROJECT%20PPT.JAI.pptx

- Project report link . https://github.com/ramar92/TNSDC-NM-EngineeringColleges/blob/05432306d7dc841b81787ae3415d02b2e4483613/Project%2 0Sample%20Report%20and%20PPT/Sample%20Report_TNSDC.docx