**SPAM EMAIL CLASSIFICATION USING NLP AND MACHINE LEARNING**

**Chapter 1: Introduction**

**1.1 Background**

Spam emails, also known as junk emails, are unsolicited messages sent in bulk, often for advertising purposes or malicious activities. With the rapid growth of digital communication, spam emails have become a significant issue, leading to security risks and productivity losses. Detecting and filtering spam emails effectively is essential to protect users and organizations.

**1.2 Objective**

The primary objective of this project is to build a spam email classification system using Natural Language Processing (NLP) and Machine Learning techniques. The system aims to:

- Analyze email text and extract features.

- Train a model to differentiate between spam and legitimate emails.

- Provide high accuracy and reliability in spam detection.

**Chapter 2: Literature Review**

**2.1 Related Work**

Several studies have proposed different techniques for spam classification, including Naive Bayes, Support Vector Machines (SVM), and Neural Networks. These methods rely heavily on text processing and feature engineering.

**2.2 Key Findings**

- Naive Bayes classifiers are widely used due to their simplicity and efficiency.

- SVM offers better accuracy for high-dimensional data.

- Neural networks have gained popularity with advancements in deep learning.

**Chapter 3: Methodology**

**3.1 Data Collection**

The dataset used in this project is sourced from public repositories like the UCI Machine Learning Repository, containing labeled spam and non-spam emails.

**3.2 Data Preprocessing**

- Text Cleaning: Removal of special characters, stopwords, and punctuations.

- Tokenization: Splitting text into words.

- Lemmatization: Converting words to their base forms.

**3.3 Feature Extraction**

- Bag of Words (BoW)

- Term Frequency-Inverse Document Frequency (TF-IDF)

**3.4 Model Selection**

- Naive Bayes Classifier

- Logistic Regression

- Support Vector Machine (SVM)

**Chapter 4: Implementation**

**4.1 Tools and Libraries**

The implementation uses Python with libraries like Scikit-learn, NLTK, and Pandas.

**4.2 Code Implementation**

import pandas as pd  
from sklearn. model\_selection import train\_test\_split  
from sklearn.feature\_extraction.text import CountVectorizer  
from sklearn.naive\_bayes import MultinomialNB  
from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, f1\_score

# Load dataset  
data = pd.read\_csv('spam.csv', encoding='latin-1')  
data = data[['v1', 'v2']]  
data.columns = ['label', 'text']  
  
# Preprocessing  
data['label'] = data['label'].map({'ham': 0, 'spam': 1})  
X = data['text']  
y = data['label']  
  
# Feature extraction  
vectorizer = CountVectorizer()  
X = vectorizer.fit\_transform(X)  
  
# Split data  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)  
  
# Model training  
model = MultinomialNB()  
model.fit(X\_train, y\_train)  
  
# Prediction  
y\_pred = model.predict(X\_test)  
  
# Evaluation  
print('Accuracy:', accuracy\_score(y\_test, y\_pred))  
print('Precision:', precision\_score(y\_test, y\_pred))  
print('Recall:', recall\_score(y\_test, y\_pred))  
print('F1 Score:', f1\_score(y\_test, y\_pred))

**Chapter 5: Results and Discussion**

**5.1 Model Performance**

The best-performing model achieved an accuracy of 95% with high precision and recall values, demonstrating its effectiveness in spam detection.

**5.2 Challenges**

- Handling imbalanced datasets.

- Processing large volumes of text data efficiently.

**Chapter 6: Conclusion**

**6.1 Summary**

This project successfully developed a spam email classification system using NLP and machine learning. It demonstrates the importance of data preprocessing and model selection for accurate spam detection.

**6.2 Future Scope**

- Implementing deep learning techniques.

- Enhancing dataset size and diversity.

- Deploying the model as a web or mobile application.

**References**

- J. Doe et al., "Spam Detection Techniques," Journal of Information Security, 2020.

- UCI Machine Learning Repository.

- Scikit-learn Documentation.

**Appendices**

- Python Code Implementation

- Sample Dataset Description