# PPML PROJECT

**By Mohith V-220901057**

## Abstract:

**Detecting whether a message is spam or not using python programming for machine learning.**

## Overview:

**The provided Python program implements a simple text classification system to distinguish between spam and ham (non-spam) messages using the Naive Bayes algorithm. This system preprocesses text data, converts it into numerical features using CountVectorizer, and trains a Multinomial Naive Bayes classifier to predict whether a message is spam or not. The program then evaluates its performance using accuracy and provides a function to predict new messages.**

## About the abstract:

**In the modern world, the internet is inundated with vast amounts of text data daily, ranging from emails and social media posts to advertisements and messages. Among these, spam messages—unsolicited or irrelevant messages, often sent for malicious purposes—pose a significant problem for individuals and businesses alike. Spam detection, the task of distinguishing spam messages**

**from legitimate ones, is an important challenge in natural language processing (NLP). One of the simplest yet effective techniques to solve this problem is the Naive Bayes classifier, a probabilistic model that has been successfully applied to text classification tasks. This essay explores a Python program that implements a spam detection system using Naive Bayes and outlines the essential steps involved in creating and evaluating such a model.**

# Program:

```
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
messages = [
'Free money now!!!', 'Hi, how are you?', 'Limited time offer, buy now!',
'Let me know your thoughts', 'Get paid to work from home',
'Meeting tomorrow at 10 AM', 'You won a lottery', 'Please attend the interview at 9 AM',
'Huge discount on products', 'Let’s catch up soon!'
]
labels = [1, 0, 1, 0, 1, 0, 1, 0, 1, 0]
```

```
X_train, X_test, y_train, y_test = train_test_split(messages, labels, test_size=0.3, random_state=42)

vectorizer = CountVectorizer()

X_train_vectorized = vectorizer.fit_transform(X_train)

classifier = MultinomialNB()

classifier.fit(X_train_vectorized, y_train)

X_test_vectorized = vectorizer.transform(X_test)

y_pred = classifier.predict(X_test_vectorized)

accuracy = accuracy_score(y_test, y_pred)

print(f"Accuracy: {accuracy * 100:.2f}%")

def predict_spam(message):

message_vectorized = vectorizer.transform([message])

prediction = classifier.predict(message_vectorized)

return "Spam" if prediction[0] == 1 else "Ham"

test_message = "Congratulations, you have won a free vacation!"

print(f"The message is: {predict_spam(test_message)}")
```

# Algorithm used in the program:

**Explanation of Multinomial Naive Bayes:**

**Naive Bayes is a family of probabilistic algorithms based on Bayes' Theorem, which applies the concept of conditional probability to classify data points. The key assumption of**

**Naive Bayes classifiers is that the features (in this case, words in the messages) are conditionally independent, given the class label (spam or ham). Despite this "naive" assumption often not holding true in practice, Naive Bayes classifiers tend to perform well, especially for text classification tasks.**

**The Multinomial Naive Bayes is used when features are discrete counts, such as word frequencies or counts of specific terms. It is particularly effective for problems like spam detection, where we deal with word counts or frequencies in text data.**

**In the program, the Multinomial Naive Bayes model is implemented using the MultinomialNB class from the sklearn.naive_bayes library, which:**

1. **Learns from the training data by calculating the probability of each word given the class label (spam or ham).**
2. **Uses these probabilities to classify new, unseen messages based on the frequency of words in the message.**

**By treating each word occurrence as an independent feature, the model calculates the likelihood that a message belongs to a particular class (spam or ham) based on the presence and frequency of specific words, and selects the class with the highest probability.**

**Thus, the Multinomial Naive Bayes algorithm is used for text classification, specifically for distinguishing spam messages from legitimate ones.**

## Advantages:

**Simplicity and Efficiency**

**Fast Training and Prediction**

**Good Performance with Text Data**

**Works Well with High Dimensionality**

**Handles Multiclass Classification**

**Effective with Small Datasets**

**Easy to Interpret**

**Scalability**

**No Need for Extensive Feature Engineering**

**Good Baseline Model**

## Conclusion:

**This program demonstrates how to build a basic text classification system using Naive Bayes for distinguishing between spam and ham messages. While the dataset is small, the model's accuracy can be improved with a larger and more diverse dataset, as well as by tuning hyperparameters or using more advanced feature extraction techniques. This type of system could be further extended to handle larger-scale email or SMS filtering systems.**