# 🌟 Tamizhan Skills SE RISE Internship – Machine Learning & AI

## 📝 Task 1: Email Spam Detection using Naive Bayes

**👤 Name: Prachee**
**🏫 College: CRSSIET, Jhajjar**
**🎓 B.Tech CSE (AI & ML), 2023–2027**

## 📄 Project Summary

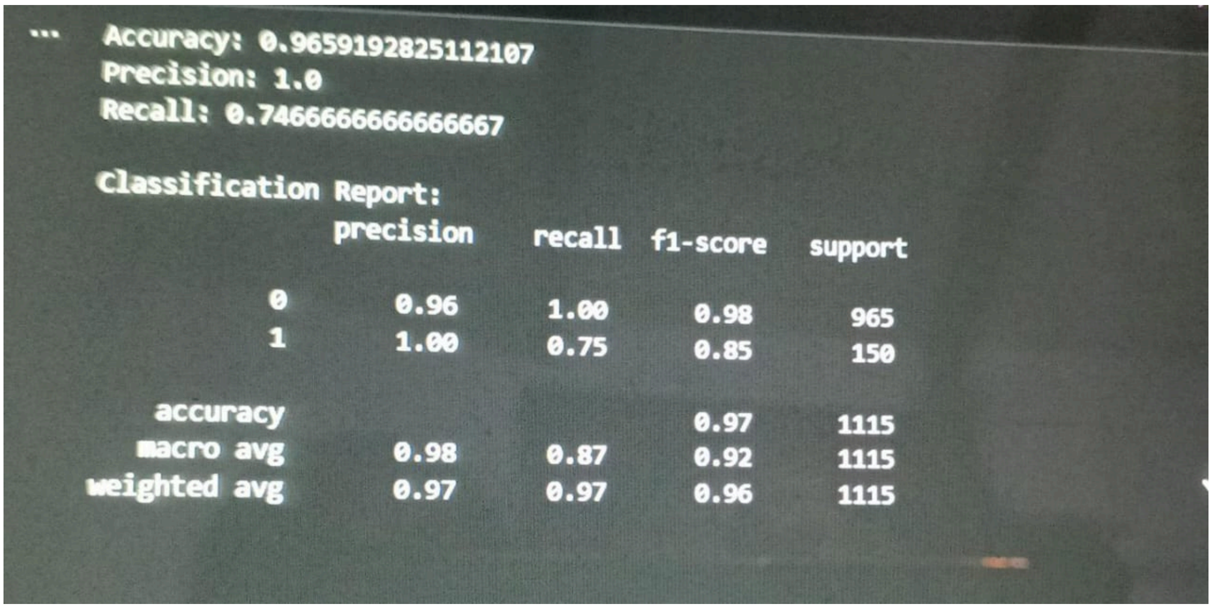
**>Objective:**
To classify emails as "Spam" or "Not Spam" using a supervised machine learning algorithm like Naive Bayes or SVM.

**>Tools Used:**
Google Colab
Python
Scikit-learn (sklearn)
Natural Language Toolkit (nltk)
Email dataset (CSV)

**>Approach:**
1. Loaded and cleaned the email dataset
2. Preprocessed text using NLP techniques like tokenization, stopword removal, and stemming
3. Converted text into numerical format using TF-IDF or CountVectorizer
4. Trained a Naive Bayes classifier or SVM on the data
5. Evaluated performance using accuracy, precision, recall, and confusion matrix

**>Result:**
The final model accurately detects spam emails with good accuracy and can be used for email filtering systems in real-world applications.

## 📊 Model Evaluation Screenshot:



```
...  Accuracy: 0.9659192825112107
     Precision: 1.0
     Recall: 0.7466666666666667

     Classification Report:
                   precision    recall  f1-score   support

                0       0.96      1.00      0.98       965
                1       1.00      0.75      0.85       150

         accuracy                           0.97      1115
        macro avg       0.98      0.87      0.92      1115
     weighted avg       0.97      0.97      0.96      1115
```

## 🧠 Code Used (Google Colab):

- **Import libraries**

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import string
import nltk
from nltk.corpus import stopwords
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, precision_score,
recall_score, classification_report

nltk.download('stopwords')
```

- **Load and clean dataset**

```
# Load the CSV file
data = pd.read_csv('spam.csv', encoding='latin-1')
```

```
# Keep only important columns
data = data[['v1', 'v2']]
data.columns = ['label', 'text']

# Display first 5 rows
data.head()

stop_words = set(stopwords.words('english'))

def preprocess_text(text):
    text = text.lower()  # Lowercase
    text = ''.join([char for char in text if char not in
string.punctuation])  # Remove punctuation
    words = text.split()
    words = [word for word in words if word not in stop_words]  #
Remove stopwords
    return ' '.join(words)

data['clean_text'] = data['text'].apply(preprocess_text)

# Show cleaned data
data[['text', 'clean_text']].head()
```

- **Preprocess text**

```
vectorizer = TfidfVectorizer()
X = vectorizer.fit_transform(data['clean_text'])

# Convert labels: ham = 0, spam = 1
y = data['label'].map({'ham': 0, 'spam': 1})

# Split into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)
```

- **Train model**

```
model = MultinomialNB()
model.fit(X_train, y_train)

y_pred = model.predict(X_test)
```

- **Evaluate model**

```
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("\nClassification Report:\n", classification_report(y_test, y_pred))
```

✨**This model can be extended using deep learning (LSTM) or deployed using Streamlit for live spam detection.**