# AI-Powered Email Classification – Step-by-Step Report

Prepared by: Luiz Fernando Alves & Romulo Gomes da Silva

Course: Engineering and Evaluating Artificial Intelligence Systems

Instructor: Dr. Abdul Razzaq

National College of Ireland

## 1. Introduction

This report presents a step-by-step development process of an AI-based email classification system, designed to automatically categorize IT service requests. The project leverages Natural Language Processing (NLP) and Machine Learning (ML) techniques to classify emails based on their content, improving response efficiency and accuracy.

### 1.1 Problem Statement

The manual classification of support emails is both time-consuming and prone to errors, often leading to delays in responding to critical issues. This project aims to automate the process by implementing a supervised machine learning model that categorizes emails into predefined classes such as 'Problem/Fault', 'Suggestion', and 'Others'.

### 1.2 Objectives

The primary objectives of this project include:  
- Translating multilingual emails into English for standardization.  
- Cleaning and preprocessing email text to enhance ML performance.  
- Converting text data into numerical format using TF-IDF vectorization.  
- Training a Naïve Bayes classification model for automated email classification.  
- Evaluating model performance and implementing improvements where necessary.

## 2. Data Preprocessing

Data preprocessing is a critical step in machine learning, especially for NLP tasks. This stage involves transforming raw email data into a structured format that can be effectively used for training the model. The following steps were carried out:

1. \*\*Translation:\*\* Emails written in different languages were converted into English.

2. \*\*Text Cleaning:\*\* Special characters, numbers, and punctuation marks were removed.

3. \*\*Tokenization:\*\* Text was split into individual words (tokens).

4. \*\*Stopword Removal:\*\* Common but uninformative words (e.g., 'the', 'is', 'and') were removed.

5. \*\*TF-IDF Vectorization:\*\* The processed text was converted into numerical vectors.

### 2.1 Data Preprocessing Code

import pandas as pd  
import re  
import string  
import nltk  
from nltk.corpus import stopwords  
from nltk.tokenize import word\_tokenize  
from sklearn.feature\_extraction.text import TfidfVectorizer  
  
nltk.download('punkt')  
nltk.download('stopwords')  
  
# Load dataset  
file\_path = "../data/AppGallery\_cleaned.csv"  
df = pd.read\_csv(file\_path)  
  
stop\_words = set(stopwords.words('english'))  
  
def preprocess\_text(text):  
 if pd.isna(text):  
 return ""  
 text = text.lower()  
 text = re.sub(r'\d+', '', text)  
 text = text.translate(str.maketrans("", "", string.punctuation))  
 words = word\_tokenize(text)  
 words = [word for word in words if word not in stop\_words]  
 return " ".join(words)  
  
df["Processed\_Text"] = df["Ticket Summary"].apply(preprocess\_text)  
  
df.to\_csv("../data/AppGallery\_preprocessed.csv", index=False)

## 3. Model Training

For model training, we used the Naïve Bayes algorithm due to its efficiency in text classification tasks. The dataset was split into training and testing sets using an 80-20 ratio, and TF-IDF vectorization was applied to convert the text into a format suitable for ML models.

### 3.1 Model Training Code

import pandas as pd  
from sklearn.model\_selection import train\_test\_split  
from sklearn.feature\_extraction.text import TfidfVectorizer  
from sklearn.preprocessing import LabelEncoder  
from sklearn.naive\_bayes import MultinomialNB  
from sklearn.metrics import accuracy\_score, classification\_report  
  
# Load preprocessed dataset  
file\_path = "../data/AppGallery\_final\_preprocessed.csv"  
df = pd.read\_csv(file\_path)  
  
df = df.dropna(subset=["Type 2"])  
label\_encoder = LabelEncoder()  
df["Type 2 Encoded"] = label\_encoder.fit\_transform(df["Type 2"])  
  
X = df["Processed\_Text"]  
y = df["Type 2 Encoded"]  
  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)  
  
vectorizer = TfidfVectorizer(max\_features=5000)  
X\_train\_tfidf = vectorizer.fit\_transform(X\_train)  
X\_test\_tfidf = vectorizer.transform(X\_test)  
  
model = MultinomialNB()  
model.fit(X\_train\_tfidf, y\_train)  
  
y\_pred = model.predict(X\_test\_tfidf)  
  
accuracy = accuracy\_score(y\_test, y\_pred)  
print(f"Model Accuracy: {accuracy:.4f}")  
print("Classification Report:")  
print(classification\_report(y\_test, y\_pred, target\_names=label\_encoder.classes\_))

## 4. Model Evaluation

After training, the model was evaluated using accuracy and precision-recall metrics. The results provide insights into model performance across different email categories.

### \*\*Final Model Performance:\*\*

- \*\*Model Accuracy:\*\* 68%

- \*\*Category Breakdown:\*\*

- \*\*Problem/Fault:\*\* High recall (100%) but moderate precision (64%).

- \*\*Others:\*\* High precision (100%) but low recall (22%).

- \*\*Suggestion:\*\* Limited data (only 2 samples).

### 4.1 Model Evaluation Code

from sklearn.metrics import classification\_report  
  
print("Classification Report:")  
print(classification\_report(y\_test, y\_pred, target\_names=label\_encoder.classes\_))

## 5. Model Improvement Strategies

To improve the model, we explored different strategies, including data balancing, alternative models, and hyperparameter tuning.

1. \*\*Balancing the dataset\*\* to ensure fair representation of all categories.

2. \*\*Exploring alternative models\*\* such as SVM, Random Forest, and BERT.

3. \*\*Fine-tuning Naïve Bayes hyperparameters\*\* to enhance classification accuracy.

## 6. Conclusion

This project successfully implemented an AI-based email classification system, leveraging NLP and ML techniques to automate the categorization of IT service requests. The study highlighted the importance of preprocessing, model selection, and evaluation to achieve accurate results. Future work will focus on testing additional models and optimizing hyperparameters to improve performance further.