Ai lab project

# Semester Project

**Group Members:  
Hammad Ul Hasan 058.**

**Muhammad Hammad 211-102.**

**Course** **Title**: Artificial Intelligence Lab

**Instructor:** Ayeza Ghazanfar

**Semester**: Spring 2024

**Deadline:** 04 June 2024

**Group Members**: Two

**Scenario:**

A customer support company wants to improve its email categorization system to automatically sort incoming emails into appropriate categories such as “Billing”, “Technical Support”, “Feedback”. The company has a large dataset of labelled emails that can be used to train a machine learning model. They aim to develop a Naïve Bayes classifier to automate the email categorization process, enhancing efficiency and response times.

**Problem Statement:**

The company is currently manually categorizing incoming emails, which is time consuming and prone to errors. This manual leads to delays in addressing customer queries and reduces overall productivity. The goal is to develop a Naïve Bayes classifier that can automatically categorize emails with high accuracy, thereby improving efficiency and customer satisfaction.

**Tasks:**

1. **Data Collection and Preprocessing:**

* Load the provided dataset of labelled emails.

1. **Text Preprocessing:**

* Preprocess the email text to prepare it for feature extraction.

1. **Feature Extraction:**

* Convert the pre-processed text into numerical features.

1. **Model Development:**

* Develop and train the Naïve Bayes classifier.

1. **Model Evaluation:**
   * Evaluate the performance of the trained classifier.
2. **Model Optimization:**

* Fine-tune the classifier to improve its performance.

1. **Documentation and Reporting:**

* Document the entire tasks and report the results.

**Deliverables:**

1. **Pre-processed Dataset:**
   * Provided email dataset of given scenario.
2. **Feature Extraction Code:**
   * Python files used to convert the data.
3. **Trained Naïve Bayes Model:**
   * The final trained Naïve Bayes saved in a proper format.
4. **Model Evaluation Report:**
   * Detailing of the model’s performance including visualizations.
5. **Documentation:**

* A detailed document including all the deliverables and attached screenshots as well.

**Note**

The effort of more than one project members should be reflected in the report.

**Plagiarism Policy**

Students will get a F grade if plagiarism is detected.

**Marks Distribution**

|  |  |  |
| --- | --- | --- |
| SNo. | Task | Marks |
| 1 | Report Writing | 40% |
| 2 | Functionality/Demo | 60% |
| Total | | 100% |

**Timeline**

Deadline: 04 June 2024

**Project Vivas: 4th June 2024**

***CODE:***import pandas as pd

from sklearn.feature\_extraction.text import TfidfVectorizer

from sklearn.model\_selection import train\_test\_split

from sklearn.naive\_bayes import MultinomialNB

from sklearn.metrics import classification\_report, accuracy\_score

from sklearn.model\_selection import GridSearchCV

import joblib

df = pd.read\_csv('/content/drive/MyDrive/customer\_support\_tickets.csv')

vectorizer = TfidfVectorizer(stop\_words='english', lowercase=True)

X = vectorizer.fit\_transform(df['Ticket Priority'])

y = df['Ticket Status']

print(y.value\_counts())

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

nb\_classifier = MultinomialNB()

nb\_classifier.fit(X\_train, y\_train)

y\_pred = nb\_classifier.predict(X\_test)

print("Accuracy:", accuracy\_score(y\_test, y\_pred))

print("\nClassification Report:\n", classification\_report(y\_test, y\_pred, zero\_division=1))

param\_grid = {'alpha': [0.1, 0.5, 1.0, 5.0, 10.0]}

grid\_search = GridSearchCV(MultinomialNB(), param\_grid, cv=5, scoring='accuracy')

grid\_search.fit(X\_train, y\_train)

print("Best Parameters:", grid\_search.best\_params\_)

optimized\_nb\_classifier = grid\_search.best\_estimator\_

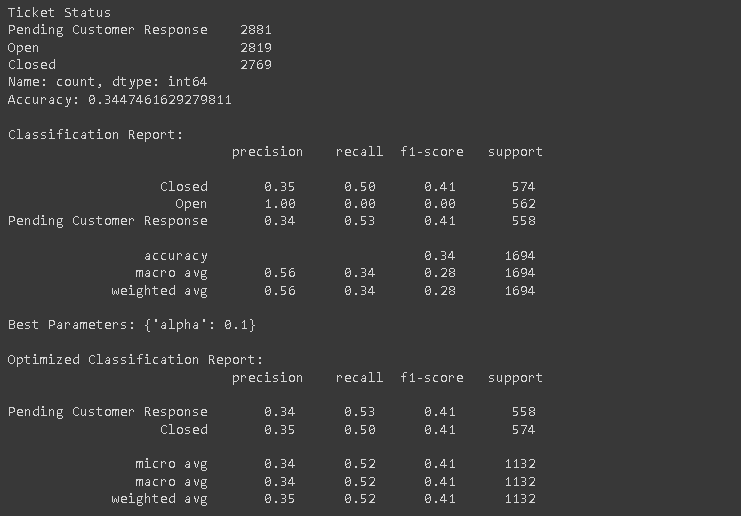
y\_pred\_optimized = optimized\_nb\_classifier.predict(X\_test)

labels = list(set(y\_pred\_optimized))

print("\nOptimized Classification Report:\n", classification\_report(y\_test, y\_pred\_optimized, zero\_division=1, labels=labels))

joblib.dump(optimized\_nb\_classifier, 'naive\_bayes\_classifier.pkl')

report = classification\_report(y\_test, y\_pred\_optimized, output\_dict=True)

***OUTPUT:  
***

**Conclusion:**

We have managed to make it more efficient to 34.47%.