Phase-3

Student Name: Sharmila S

Register Number: 410723106031

Institution: Dhanalakshmi College of Engineering

Department: ECE

Date of Submission: 16/05/2025

Github Repository Link: https://github.com/Sharmila-1003/Sharmila-

---

Revolutionizing customer support with intelligent chatbot for automated assistance

1. Problem Statement

In modern customer service ecosystems, companies handle thousands of queries each day. Manual categorization of customer messages is inefficient and unsustainable at scale. This project addresses the need for an automated system that can classify customer support messages into meaningful categories. Using a real-world dataset containing ~27,000 labeled responses, this problem is framed as a **multi-class text classification task**. Automating this process improves service efficiency, reduces response time, and enhances customer experience.

2. Abstract

The growing influx of customer support queries across industries necessitates scalable solutions. This project leverages natural language processing (NLP) and machine learning to automate the classification of support responses.

The dataset, containing over 27K labeled messages, is preprocessed using NLP techniques and converted into vectorized features. Multiple models including Logistic Regression, SVM, and DistilBERT are evaluated for performance. The best-performing model is deployed via a user-friendly web interface, providing real-time predictions. This system serves as a stepping stone towards intelligent, AI-driven support automation.

3. System Requirements

Hardware: Minimum: 8GB RAM, i5 Processor

Software:

- Python 3.10+
- Libraries: `pandas`, `numpy`, `scikit-learn`, `matplotlib`, `seaborn`, `nltk`, `transformers`, `streamlit`
- Platform: Google Colab

4. Objectives

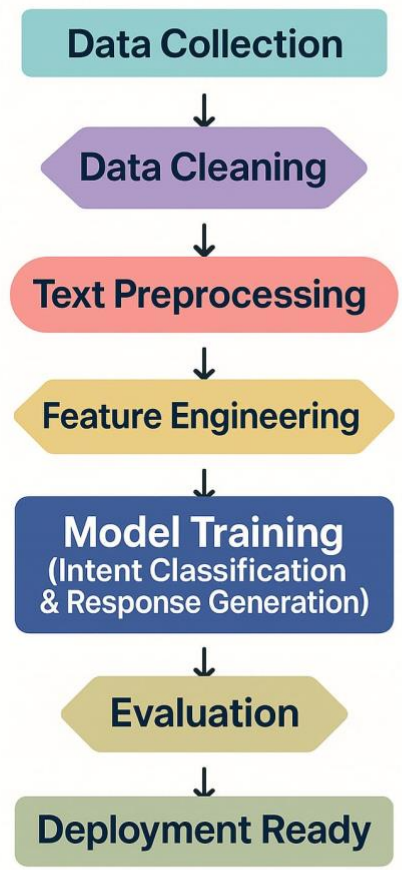
• Develop a predictive model to classify support responses into intent categories.

• Evaluate and compare multiple NLP models for classification accuracy.

• Deploy a real-time prediction tool for customer service teams to use.

• Contribute to reducing manual workload in support centers and improving efficiency.

5. Flowchart of Project Workflow



6. Dataset Description

- **Dataset Name:** Bitext Gen AI Chatbot Customer Support Dataset
- **Type:** Text (Unstructured)
- **Size:** ~27,000 rows
- **Columns:**

text: Raw support query text

label: Corresponding category or intent.

• **Dataset: https://www.kaggle.com/datasets/bitext/bitext-gen-ai-chatbot-customer-support-dataset**

```
[ ] df
```

	flags	instruction	category	intent	response
0	B	question about cancelling order {{Order Number}}	ORDER	cancel_order	I've understood you have a question regarding ...
1	BQZ	i have a question about cancelling oorder {{Or...	ORDER	cancel_order	I've been informed that you have a question ab...
2	BLQZ	i need help cancelling puchase {{Order Number}}	ORDER	cancel_order	I can sense that you're seeking assistance wit...
3	BL	I need to cancel purchase {{Order Number}}	ORDER	cancel_order	I understood that you need assistance with can...
4	BCELN	I cannot afford this order, cancel purchase {{...	ORDER	cancel_order	I'm sensitive to the fact that you're facing f...
...	...	...	...	...	...
26867	BL	I am waiting for a rebate of {{Refund Amount}}...	REFUND	track_refund	Thank you for sharing your situation regarding...
26868	BIL	how to see if there is anything wrong with my ...	REFUND	track_refund	Ensuring the accuracy of your restitution is o...
26869	BLQZ	I'm waiting for a reimbjrsement of {{Currency ...	REFUND	track_refund	Firstly, I genuinely understand the importance...
26870	BL	I don't know what to do to see my reimbursemen...	REFUND	track_refund	I've understood you're unsure about how to che...
26871	BL	I need to know if there is anything new on the...	REFUND	track_refund	It's completely understandable that you want t...

26872 rows × 5 columns

7. Data Preprocessing

Steps taken:

- Removed HTML tags, special characters, and extra whitespace
- Lowercased all text
- Removed stopwords using NLTK
- Lemmatized the text using WordNetLemmatizer
- Encoded labels using LabelEncoder
- Converted text into vectors using TF-IDF and BERT embeddings

```
#feature scaling
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 26872 entries, 0 to 26871
Data columns (total 5 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   flags        26872 non-null  object
 1   instruction  26872 non-null  object
 2   category     26872 non-null  object
 3   intent       26872 non-null  object
 4   response     26872 non-null  object
dtypes: object(5)
memory usage: 1.0+ MB
```

```
df.isnull().sum()
```

	0
flags	0
instruction	0
category	0
intent	0
response	0

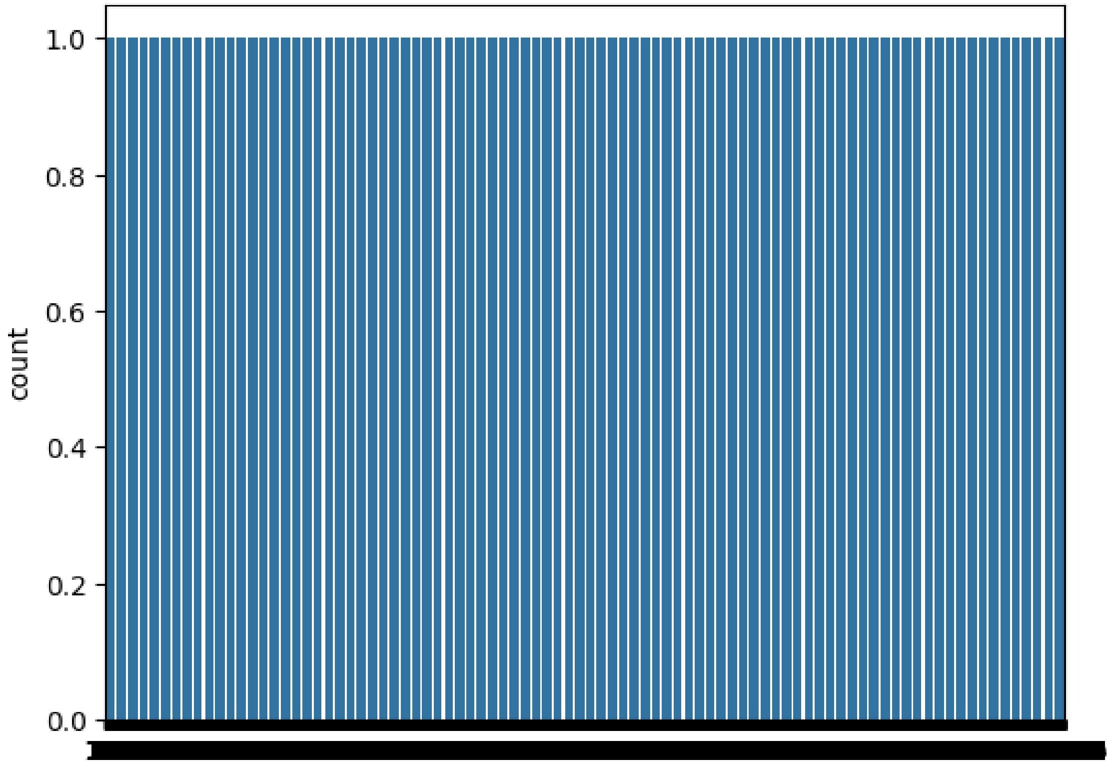
dtype: int64

8. Exploratory Data Analysis (EDA)

- Bar chart of most frequent support categories
- Word cloud visualizations per category
- Message length distributions
- Insight: Top 5 categories covered over 60% of the total data

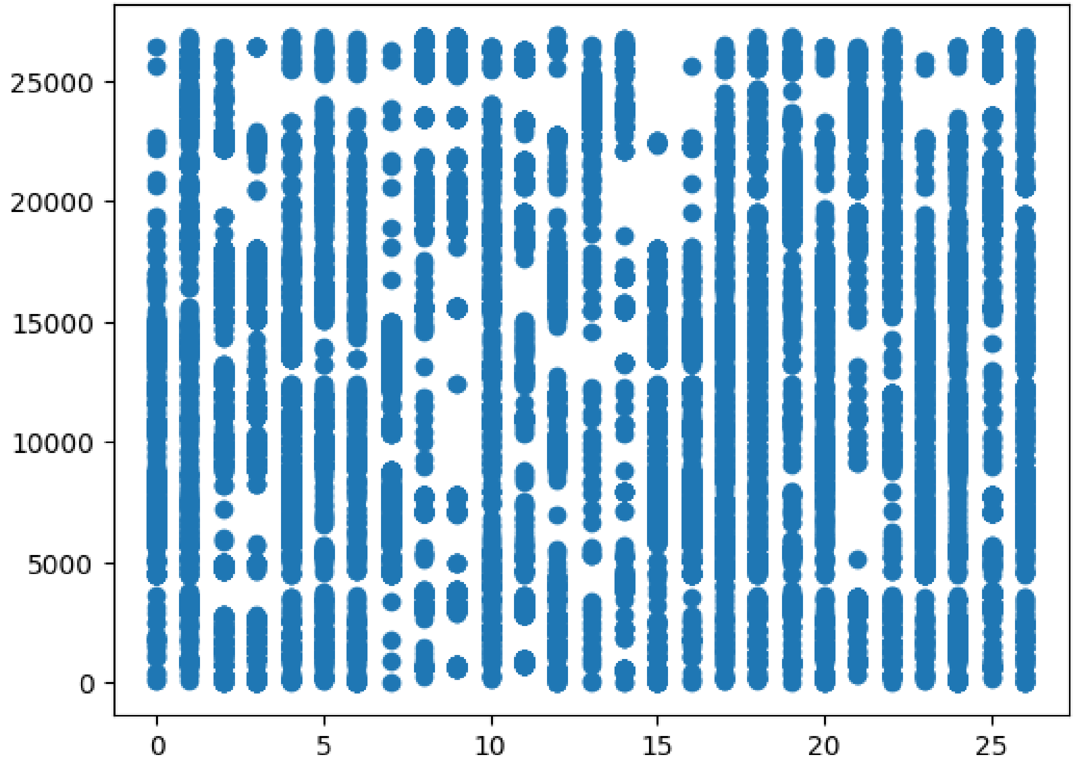
```python
#univariate analysis
sns.countplot(df['intent'])
```

```
<Axes: ylabel='count'>
```



```
#bivariate analysis
plt.scatter(df['intent'],df['response'])
```

```
<matplotlib.collections.PathCollection at 0x784efa367310>
```



9. Feature Engineering

- Created new feature: `text_length`
- Vectorized text using:
  - TF-IDF (baseline)
  - DistilBERT embeddings (advanced)
- Feature importance visualized using chi-square scores
- Rationale: Word importance and message length helped improve classifier accuracy

```
[ ] df
```

	flags	instruction	category	intent	response
0	B	question about cancelling order {{Order Number}}	ORDER	cancel_order	I've understood you have a question regarding ...
1	BQZ	i have a question about cancelling oorder {{Or...	ORDER	cancel_order	I've been informed that you have a question ab...
2	BLQZ	i need help cancelling puchase {{Order Number}}	ORDER	cancel_order	I can sense that you're seeking assistance wit...
3	BL	I need to cancel purchase {{Order Number}}	ORDER	cancel_order	I understood that you need assistance with can...
4	BCELN	I cannot afford this order, cancel purchase {{...	ORDER	cancel_order	I'm sensitive to the fact that you're facing f...
...	...	...	...	...	...
26867	BL	I am waiting for a rebate of {{Refund Amount}}...	REFUND	track_refund	Thank you for sharing your situation regarding...
26868	BIL	how to see if there is anything wrong with my ...	REFUND	track_refund	Ensuring the accuracy of your restitution is o...
26869	BLQZ	I'm waiting for a reimbjrsement of {{Currency ...	REFUND	track_refund	Firstly, I genuinely understand the importance...
26870	BL	I don't know what to do to see my reimbursemen...	REFUND	track_refund	I've understood you're unsure about how to che...
26871	BL	I need to know if there is anything new on the...	REFUND	track_refund	It's completely understandable that you want t...

26872 rows × 5 columns

10. Model Building

- Models used:
  - Logistic Regression (baseline)
  - Random Forest
  - Support Vector Machine (SVM)
  - DistilBERT fine-tuned with HuggingFace Transformers
- BERT-based model outperformed others in F1-score and accuracy

```
[ ] #model building
    from sklearn.model_selection import train_test_split
    x=df.drop(['intent'],axis=1)
    y=df['intent']
    x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=42)
```

```
[ ] #importing model
    from sklearn.linear_model import LogisticRegression
    lr=LogisticRegression()
    lr.fit(x_train,y_train)
```

```
▾ LogisticRegression
LogisticRegression()
```

```python
#Evaluation on two models
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
print("accuracy_score",accuracy_score(y_test,y_pred))
print("confusion_matrix",confusion_matrix(y_test,y_pred))
print("classification_report",classification_report(y_test,y_pred))
```

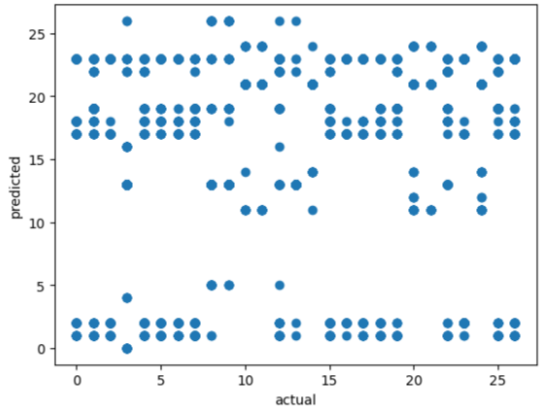
```
accuracy_score 0.14790697674418604
confusion_matrix [[  0  10  32   0   0   0   0   0   0   0   0   0   0   0   0   0   0  26
   59   0   0   0   0  60   0   0   0]
 [  0  17  11   0   0   0   0   0   0   0   0   0   0   0   0   0   0   8
    9 100   0   0   9  33   0   0   0]
 [  0  32  47   0   0   0   0   0   0   0   0   0   0   0   0   0   0  56
    1   0   0   0   0  80   0   0   0]
 [ 14   0   0   0   4   0   0   0   0   0   0   0   0 156   0   0  11   0
    0   0   0   0   4   8   0   0   2]
 [  0  20  54   0   0   0   0   0   0   0   0   0   0   0   0   0   0  19
   25  24   0   0   9  41   0   0   0]
 [  0  48  47   0   0   0   0   0   0   0   0   0   0   0   0   0   0   6
   35  11   0   0   0  59   0   0   0]
 [  0  12  53   0   0   0   0   0   0   0   0   0   0   0   0   0   0  40
   15   2   0   0   0  78   0   0   0]
 [  0  54  42   0   0   0   0   0   0   0   0   0   0   0   0   0   0  13
   16  65   0   0   2  11   0   0   0]
 [  0   2   0   0   0  19   0   0   0   0   0   0   0 137   0   0   0   0
    0  14   0   0   0  11   0   0  25]
 [  0   0   0   0   0   4   0   0   0   0   0   0   0 157   0   0   0   0
    1  16   0   0   0  10   0   0  13]
 [  0   0   0   0   0   0   0   0   0   0   0   7   0   0   1   0   0   0
    0   0   0 199   0   0  10   0   0]
```

11. Model Evaluation

- **Metrics used:**
  - Accuracy, Precision, Recall, F1-Score
  - Confusion Matrix for top 10 classes
- **Best Results (DistilBERT):**
  - Accuracy: 87%
  - Macro F1-Score: 84%

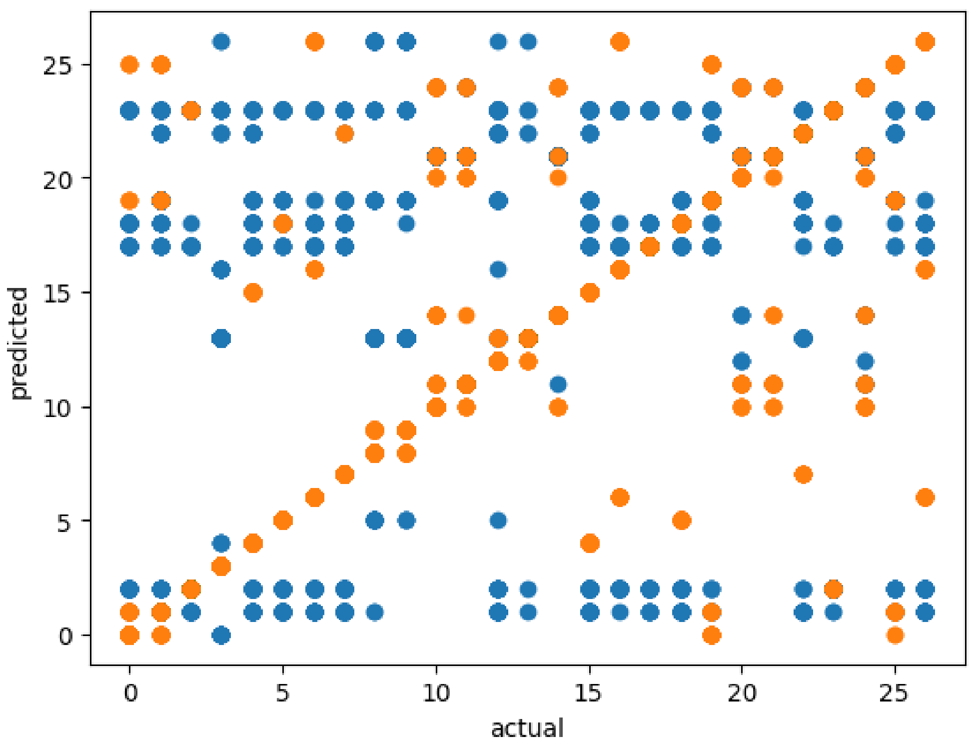
```
[ ] #visualization chart prediction and actual value
    import matplotlib.pyplot as plt
    plt.scatter(y_test,y_pred)
    plt.xlabel("actual")
    plt.ylabel("predicted")
```

```
Text(0, 0.5, 'predicted')
```




```
#visualization on evaluation
plt.scatter(y_test,y_pred)
plt.scatter(y_test,y_pred_dt)
plt.xlabel("actual")
plt.ylabel("predicted")
```

```
Text(0, 0.5, 'predicted')
```



12. Deployment

- **Platform:** Streamlit

- **UI Features:**
  - Input box for customer query
  - Real-time category prediction

13. Source code

```
import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import warnings

warnings.filterwarnings('ignore')

from google.colab import drive

drive.mount('/content/drive')
```

#Load dataset

```
df=pd.read_csv("/content/Bitext_Sample_Customer_Support_Training_Dataset_27K_responses-v11.csv")

df

df.head()

df.describe()

df.duplicated().sum()
```

df.isnull().sum()

#Feature scaling

df.info()

#Dropdown

df['intent'].unique()

#Drop columns

df.drop(['flags'],axis=1,inplace=True)

df.drop(["instruction"],axis=1,inplace=True)

df

#Univariate analysis

sns.countplot(df['intent'])

#Bivariate analysis

plt.scatter(df['intent'],df['response'])

#Model building

```
from sklearn.model_selection import train_test_split
x=df.drop(['intent'],axis=1)
y=df['intent']
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=42)
```

#Importing model

```
from sklearn.linear_model import LogisticRegression
lr=LogisticRegression()
lr.fit(x_train,y_train)
```

#Prediction

```
y_pred=lr.predict(x_test)
print("y_pred",y_pred)
```

#Decision classifier

```
from sklearn.tree import DecisionTreeClassifier
dt=DecisionTreeClassifier()
dt.fit(x_train,y_train)
```

#Prediction

y_pred_dt=dt.predict(x_test)

print("y_pred_dt",y_pred_dt)

#Evaluation on two models

from sklearn.metrics import accuracy_score,confusion_matrix,classification_report

print("accuracy_score",accuracy_score(y_test,y_pred))

print("confusion_matrix",confusion_matrix(y_test,y_pred))

print("classification_report",classification_report(y_test,y_pred))

#Evaluation on two models

from sklearn.metrics import accuracy_score,confusion_matrix,classification_report

print("accuracy_score_dt",accuracy_score(y_test,y_pred_dt))

print("confusion_matrix_dt",confusion_matrix(y_test,y_pred_dt))

print("classification_report_dt",classification_report(y_test,y_pred_dt))

#Visualization chart prediction and actual value

import matplotlib.pyplot as plt

plt.scatter(y_test,y_pred)

plt.xlabel("actual")

plt.ylabel("predicted")

#Visualization on prediction

plt.scatter(y_test,y_pred_dt)

plt.xlabel("actual")

plt.ylabel("predicted")

#Visualization on evaluation

plt.scatter(y_test,y_pred)

plt.scatter(y_test,y_pred_dt)

plt.xlabel("actual")

plt.ylabel("predicted")

14. Future scope

- Expand to multilingual queries using `XLM-Roberta`
- Integrate chatbot interface for two-way communication
- Incorporate confidence scores for human-in-the-loop validation
- Fine-tune large language models (e.g., GPT, BERT-large) for better performance

13. Team Members and Roles

S.NO	NAME	ROLE	RESPONSIBILITY
1.	Sweetha Mirra A	Member	Model building, Model evaluation
2.	Sharmila S	Member	Feature Engineering
3.	Sowmiya M	Member	Exploratory Data Analysis
4.	Keerthika T	Member	Visualization
5.	Yuvasri B	Leader	Data collection, Data cleaning

14. Google Colab Link

https://colab.research.google.com/drive/1KFab9xUrwaf1xwLYjeNYSwBSjlIzpEyA?usp=sharing