Phase-3 Submission Template

Student Name: MOHAMMED FAROOQ H

Register Number: 410723104045

Institution: Dhanalakshmi college of engineering

Department: Computer Science & Engineering

Date of Submission: 16-05-2025

Github Repository Link:
https://github.com/Xjod01/NM_FAROOQ

Revolutionizing customer support with an intelligent chatbot for automated assistance

1. Problem Statement

Traditional customer support systems often struggle with long response times, inconsistent service quality, and high operational costs due to their reliance on human agents for handling repetitive and routine queries. This leads to customer dissatisfaction, decreased efficiency, and scalability challenges. There is a growing need for an intelligent, always-available solution that can automate common support tasks, provide instant responses, and seamlessly escalate complex issues—ensuring improved customer experience and reduced workload on support teams.

2. Abstract

This project aims to revolutionize customer support by developing an intelligent chatbot that provides automated, real-time assistance to users. Traditional support

systems face challenges such as delayed response times, inconsistent service, and high operational costs due to heavy dependence on human agents. The objective is to create a smart, scalable solution that can handle frequently asked questions, troubleshoot common issues, and route complex queries to human representatives when needed. The chatbot leverages natural language processing (NLP) and machine learning (ML) to understand user intent, respond accurately, and continuously improve from interactions.

3. System Requirements

Specify minimum system/software requirements to run the project:

Hardware:

Intel Core i5 or Higher

Minimum 8Gb RAM

Software:

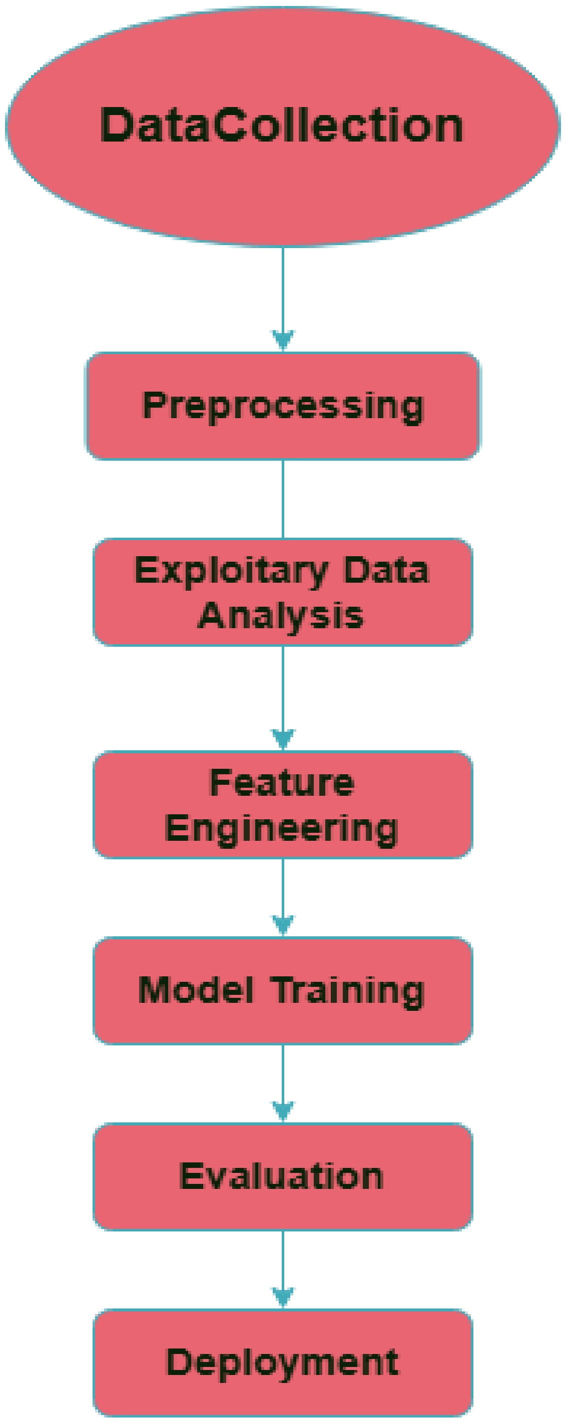
- python 3.8+
- jupyter Notebook / Google Colab
- Libraries: pandas , numpy ,Scikit – learn , tensorflow , spaCY

4. Objectives

- Enhance Customer Experience
    - Provide 24/7 instant responses to common customer queries.
    - Reduce customer wait times by at least 60% within the first 3 months of deployment.

- Automate Repetitive Support Tasks
    - Automate handling of Tier 1 support tasks (FAQs, order tracking, password resets) to reduce manual workload by 50%.
- Increase Support Team Efficiency
    - Enable live agents to focus on complex issues by routing routine queries to the chatbot.
    - Improve agent response times and satisfaction ratings by at least 20%.
- Personalize Customer Interactions
    - Use NLP and user data to deliver tailored responses that match the customer’s history and behavior.
    - Achieve a 25% increase in customer satisfaction scores (CSAT).

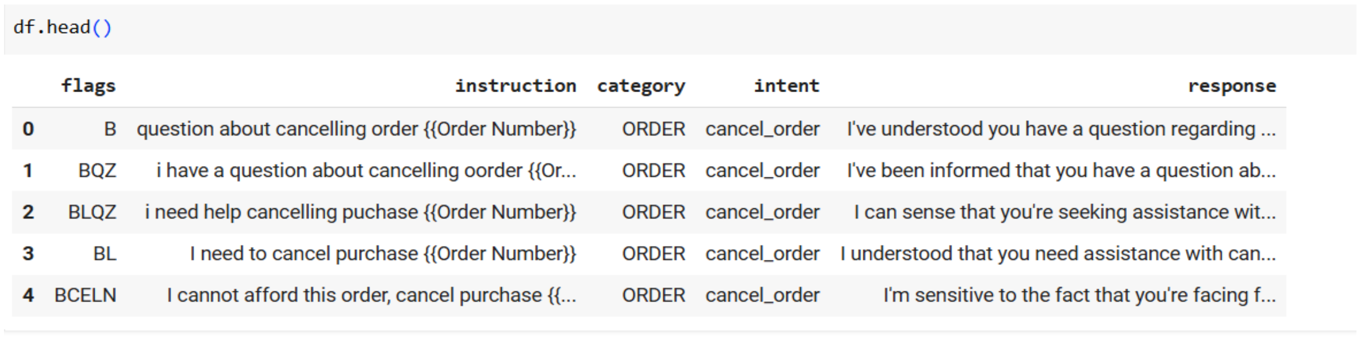
5. Flowchart of Project Workflow



6. Dataset Description

Source: Kaggle-Telco Coustomer Churn.

Type: Public

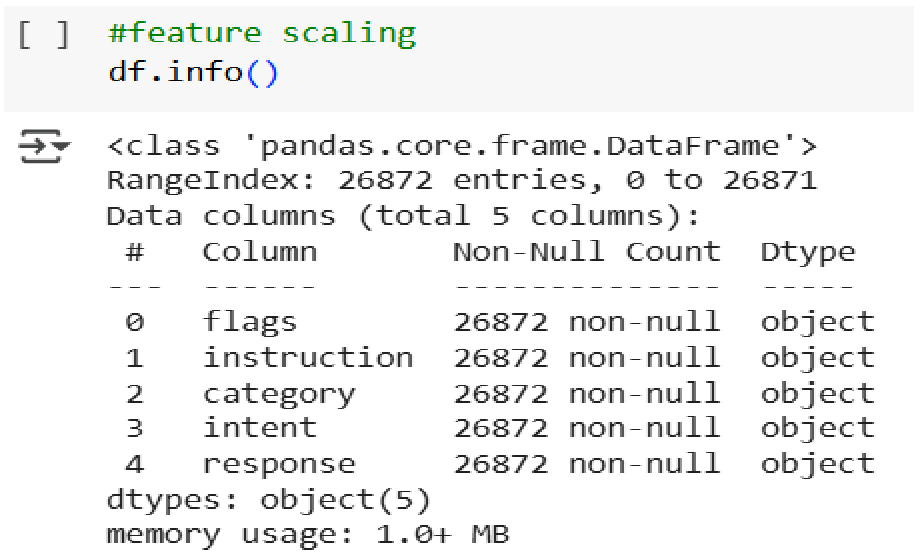


```
df.head()
```

	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 understand 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...

7. Data Preprocessing

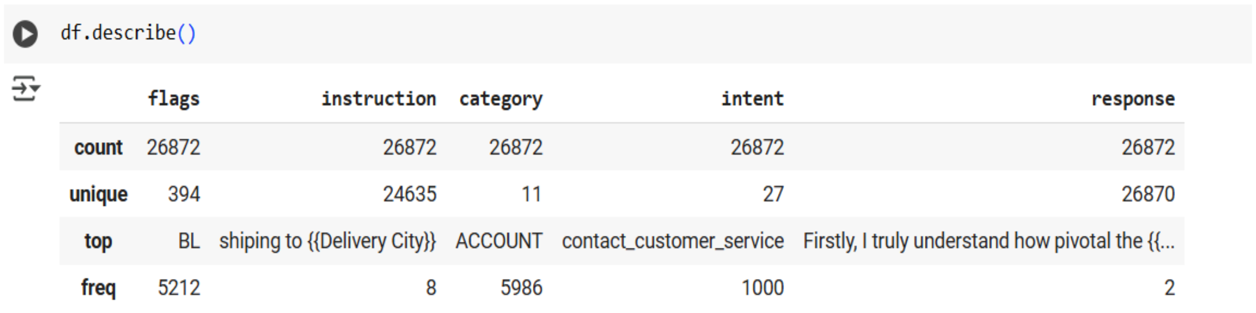
- Handle missing values, duplicates, outliers
- Feature encoding and scaling
- Show before/after transformation screenshots



```
[ ]  #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
```

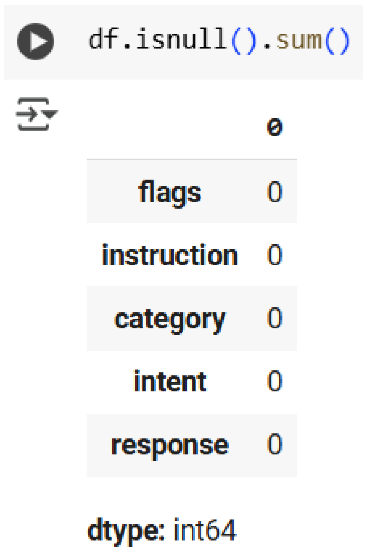
df.describe()



```
df.describe()
```

	flags	instruction	category	intent	response
count	26872	26872	26872	26872	26872
unique	394	24635	11	27	26870
top	BL	shiping to {{Delivery City}}	ACCOUNT	contact_customer_service	Firstly, I truly understand how pivotal the {{...
freq	5212	8	5986	1000	2

df.isnull().sum()

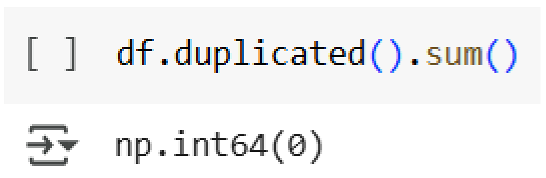


```
df.isnull().sum()
```

	0
flags	0
instruction	0
category	0
intent	0
response	0

dtype: int64

df.duplicated().sum()



```
[ ]  df.duplicated().sum()
```

```
np.int64(0)
```

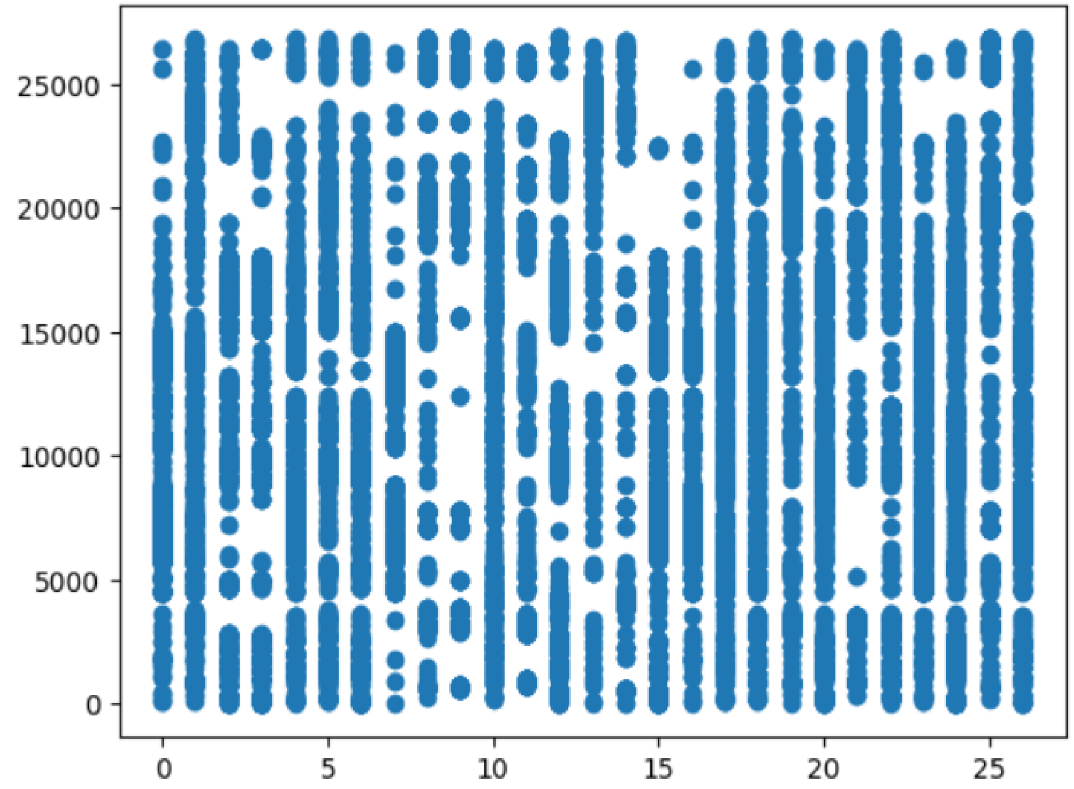
8. Exploratory Data Analysis (EDA)

- Use visual tools like histograms, boxplots, heatmaps
- Reveal correlations, trends, patterns

- Write down key takeaways and insights
- Include screenshots of visualizations

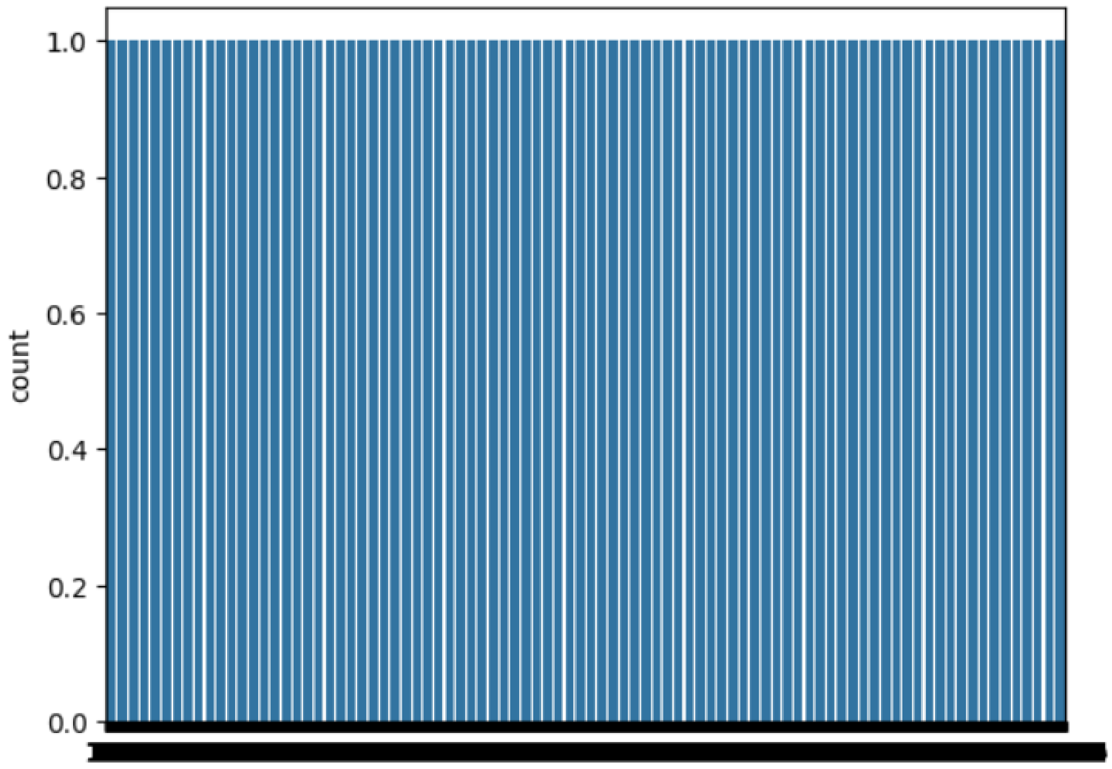
```
[ ]  #bivariate analysis
     plt.scatter(df['intent'],df['response'])
```

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



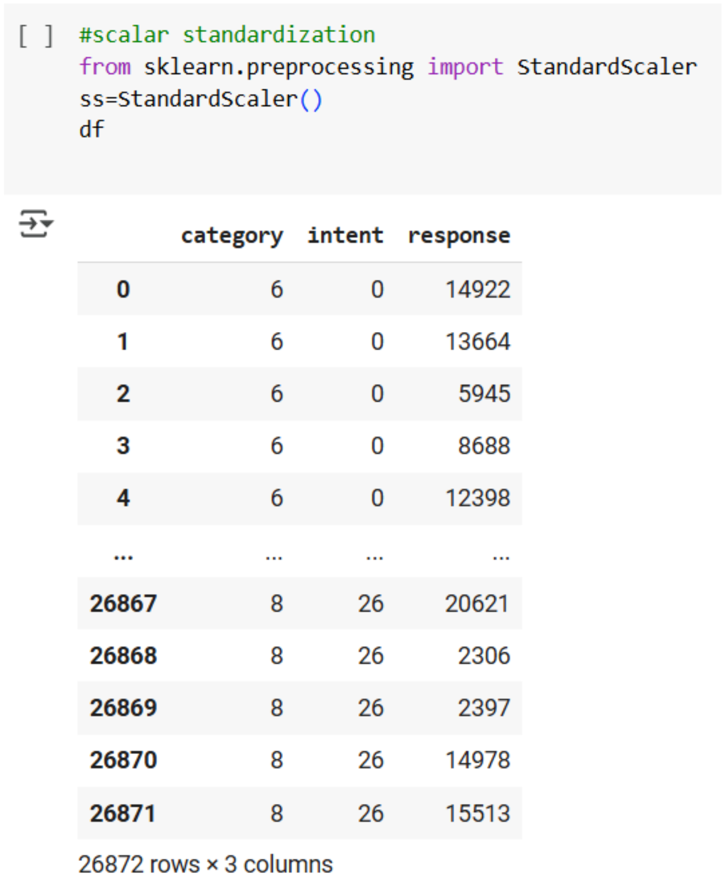
```
#univariate analysis
sns.countplot(df['intent'])
```

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



9. Feature Engineering

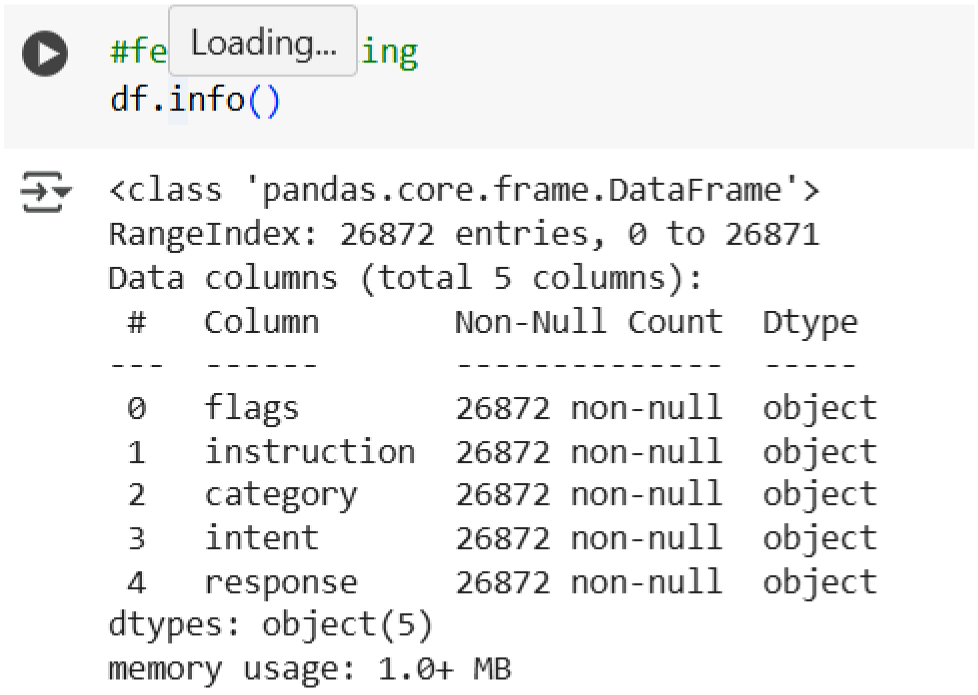
- New feature creation
- Feature selection
- Transformation techniques
- Explain why and how features impact your model



```
[ ] #scalar standardization
    from sklearn.preprocessing import StandardScaler
    ss=StandardScaler()
    df
```

	category	intent	response
0	6	0	14922
1	6	0	13664
2	6	0	5945
3	6	0	8688
4	6	0	12398
...	...	...	...
26867	8	26	20621
26868	8	26	2306
26869	8	26	2397
26870	8	26	14978
26871	8	26	15513

26872 rows × 3 columns

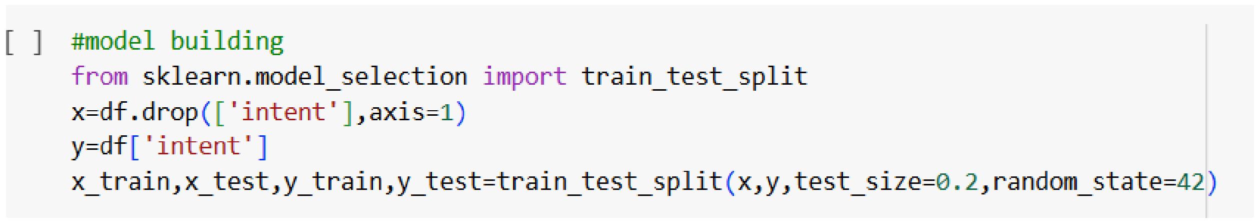


```
#fe Loading... ing
df.info()
```

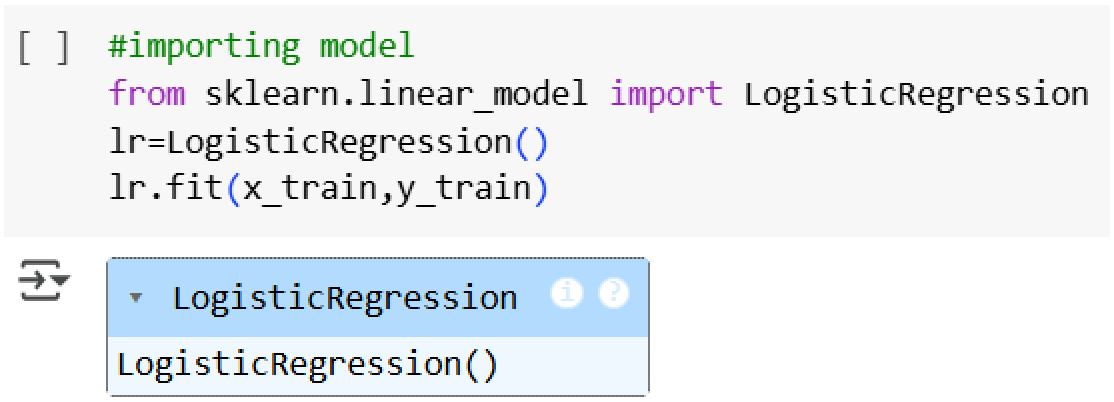
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 26872 entries, 0 to 26871
Data columns (total 5 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   flags        26872 non-null  object
 1   instruction  26872 non-null  object
 2   category     26872 non-null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

10. Model Building

- Try multiple models (baseline and advanced)
- Explain why those models were chosen
- Include screenshots of model training outputs

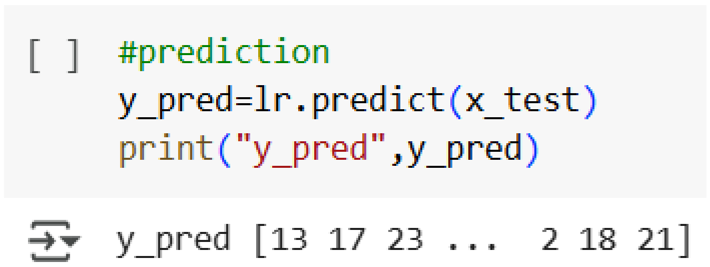


```
[ ] #model building
    from sklearn.model_selection import train_test_split
    x=df.drop(['intent'],axis=1)
    y=df['intent']
    x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=42)
```



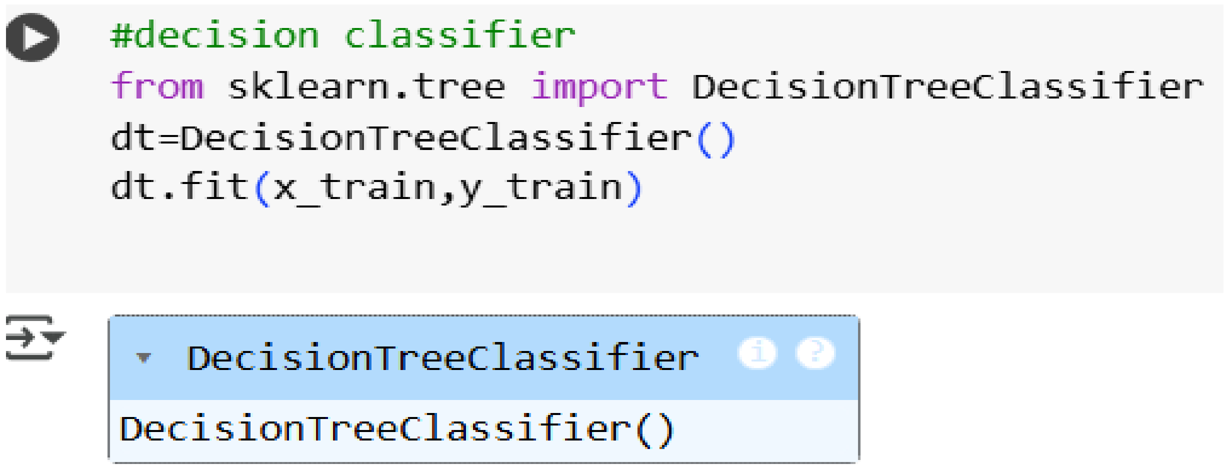
```
[ ] #importing model
    from sklearn.linear_model import LogisticRegression
    lr=LogisticRegression()
    lr.fit(x_train,y_train)
```

```
▾ LogisticRegression
LogisticRegression()
```



```
[ ] #prediction
    y_pred=lr.predict(x_test)
    print("y_pred",y_pred)
```

```
y_pred [13 17 23 ...  2 18 21]
```



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

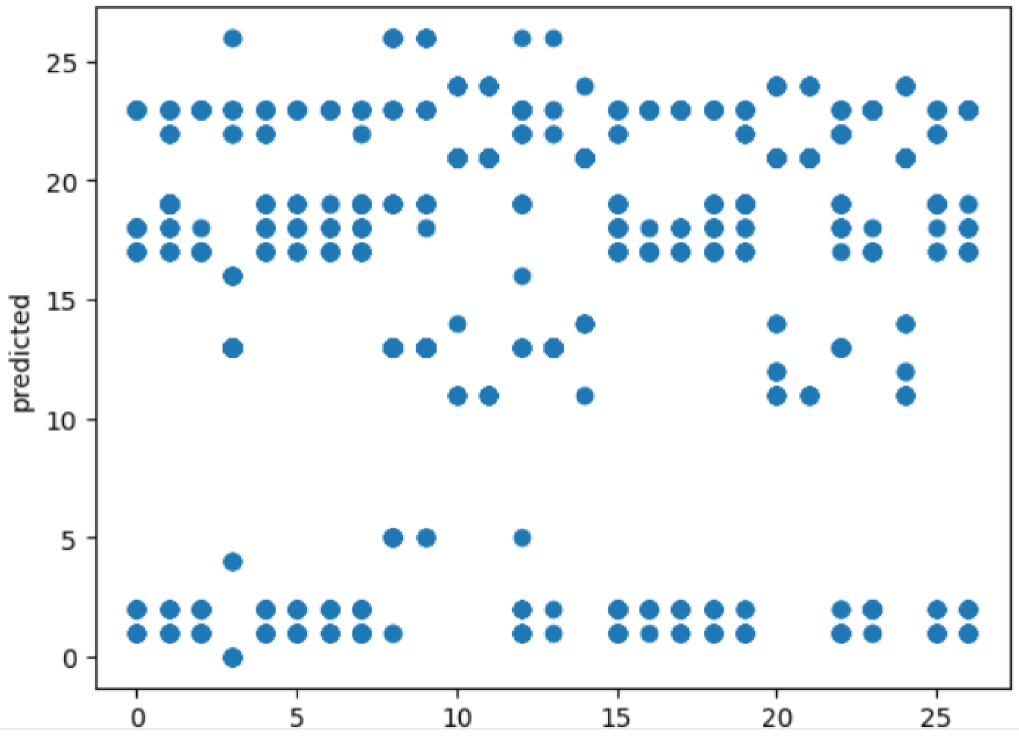
```
DecisionTreeClassifier
DecisionTreeClassifier()
```

11. Model Evaluation

- Show evaluation metrics: accuracy, F1-score, ROC, RMSE, etc.
- Visuals: Confusion matrix, ROC curve, etc.
- Error analysis or model comparison table
- Include all screenshots of outputs

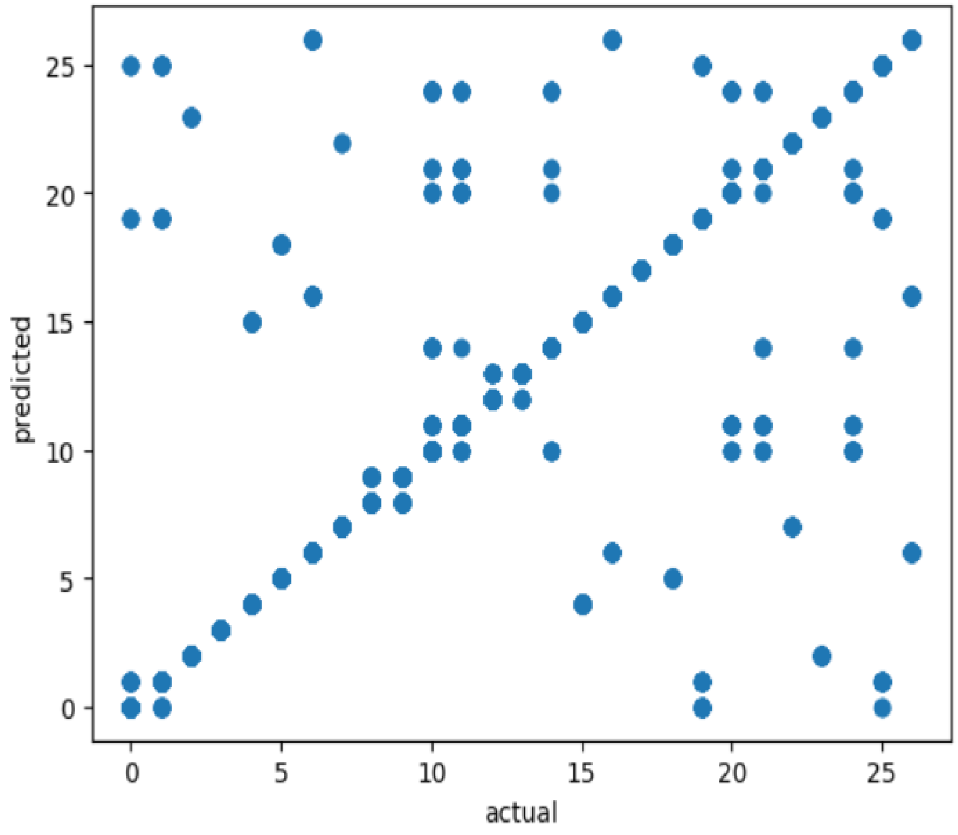
```
[ ] #visualization chart prediction and actual value
    import matplotlib.pyplot as plt
    plt.scatter(y_test,y_pred)
    plt.xlabel("actual")
    plt.ylabel("predicted")
```

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



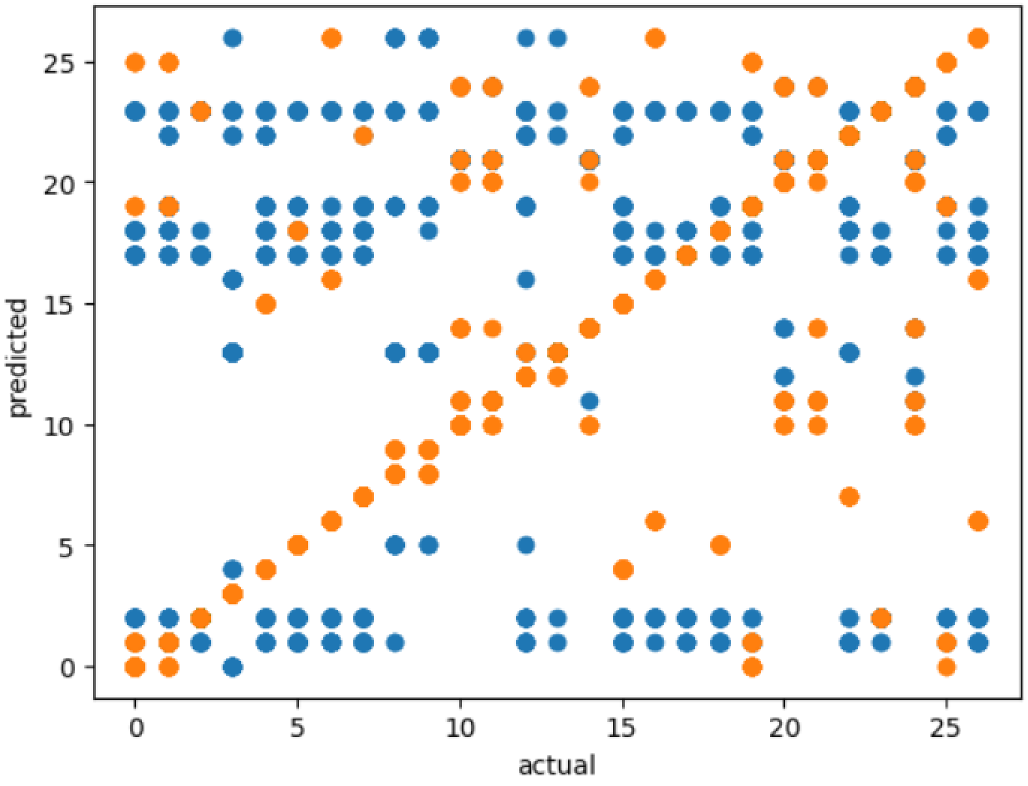
```
[ ]  #visualization on prediction
     plt.scatter(y_test,y_pred_dt)
     plt.xlabel("actual")
     plt.ylabel("predicted")
```

```
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

- Deploy using a free platform:
- Streamlit Cloud
- Gradio + Hugging Face Spaces
- Flask API on Render or Deta
- Include:
- Deployment method

- Public link
- UI Screenshot
- Sample prediction output

13. Source code

- import pandas as pd
- from sklearn.model_selection import train_test_split
- from sklearn.preprocessing import LabelEncoder, StandardScaler
- from sklearn.ensemble import RandomForestClassifier
- from sklearn.metrics import classification_report, confusion_matrix
- import matplotlib.pyplot as plt
- import seaborn as sns
- df = pd.read_csv('/content/Customer-Churn-Records.csv')
- df.head( )
- df.info( )
- df.describe( )
- df.isnull( ).sum( )
- df.drop_duplicates( )
- df.drop_duplicates( ).sum( )
- df.drop(['RowNumber', 'CustomerId', 'Surname'], axis=1, inplace=True)
- #Histogram chart

- df.hist(figsize=(10,10))
- plt.show()
- #Bivariate analysis
- sns.pairplot(df)
- plt.show()
- #Feature engineering
- for col in ['Geography', 'Gender', 'Card Type']:
- le = LabelEncoder()
- df[col] = le.fit_transform(df[col])
- df
- #Scalar standardization
- scaler = StandardScaler()
- df_scaled = scaler.fit_transform(df)
- df
- #Label encoding and onehot encoding
- df_encoded = pd.get_dummies(df, columns=['Geography', 'Gender', 'Card Type'])
- df
- #Model building
- X = df.drop('Exited', axis=1)
- y = df['Exited']
- #import model

- from sklearn.model_selection import train_test_split
- from sklearn.ensemble import RandomForestClassifier
- from sklearn.metrics import classification_report, confusion_matrix
- x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
- random_state=42)
- from sklearn.linear_model import LogisticRegression
- model = LogisticRegression()
- model.fit(x_train, y_train)
- #Prediction
- y_pred = model.predict(x_test)
- print("y_prediction", y_pred)
- #Random forest classifier
- model = RandomForestClassifier(n_estimators=100, random_state=42)
- model.fit(x_train, y_train)
- y_random_prediction = model.predict(x_test)
- print("y_prediction", y_random_prediction)
- # Evaluate
- y_pred = model.predict(x_test)
- print("Classification Report:\n", classification_report(y_test, y_pred))
- print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
- y_random_prediction = model.predict(x_test)

- print("Classification Report:\n", classification_report(y_test,
- y_random_prediction))
- print("Confusion Matrix:\n", confusion_matrix(y_test, y_random_prediction))
- #Visualize prediction and actual value
- plt.figure(figsize=(10, 6))
- plt.scatter(y_test, y_pred, alpha=0.5, label='Predicted')
- plt.scatter(y_test, y_random_prediction, alpha=0.5, label='Random Predicted')
- plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], linestyle='--',
- color='red', label='Perfect Prediction')
- plt.xlabel('Actual Values')
- plt.ylabel('Predicted Values')
- plt.title('Actual vs. Predicted Values')
- plt.legend()
- plt.show()
- #Histogram chart random forest and logistic regression
- plt.figure(figsize=(10, 6))
- plt.hist(y_pred, bins=20, alpha=0.5, label='Logistic Regression')
- plt.hist(y_random_prediction, bins=20, alpha=0.5, label='Random Forest')
- plt.xlabel('Predicted Values')
- plt.ylabel('Frequency')

- plt.title('Histogram of Predicted Values')
- plt.legend()
- plt.show()

14. Future scope

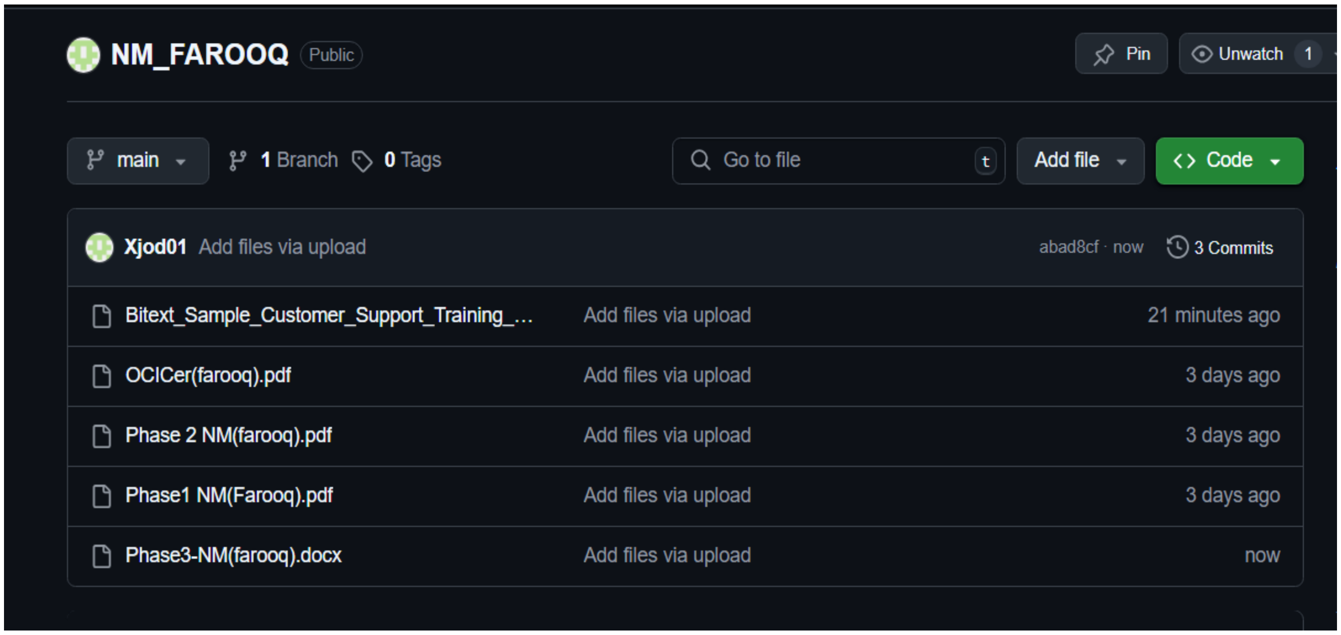
- Integrate with live CRM systems for real-time predictions
- Add NLP to analyze customer feedback sentiment
- Implement customer segmentation for targeted retention strategies

13. Team Members and Roles

NAME	ROLE	RESPONSIBILITY
HARIHARAN K	LEADER	Data Collection and Cleaning
MAGESH L	MEMBER	Data Visualization and Interpretation
MOHAMMED FAROOQ H	MEMBER	Exploratory Data Analysis
HARISH JAYARAJ R	MEMBER	Model Evaluation

ABDUL AZEEZ M	MEMBER	Model Building

GITHUB SCREENSHOT



Xjod01 Add files via upload abad8cf · now 3 Commits

Bitext_Sample_Customer_Support_Training_…	Add files via upload	21 minutes ago
OCICer(farooq).pdf	Add files via upload	3 days ago
Phase 2 NM(farooq).pdf	Add files via upload	3 days ago
Phase1 NM(Farooq).pdf	Add files via upload	3 days ago
Phase3-NM(farooq).docx	Add files via upload	now

COLAB LINK :

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