PHASE-2

**STUDENT NAME**:

Jasmine Joicy.A

**REGISTER NUMBER:**

422223243025

**INSTITUION:**

Surya Group Of Institutions

**DEPARTMENT:**

B.Tech Artificial Intelligence and Data Science

**DATE OF SUBMISSION**:

**GITHUB REPOSITORY LINK:**

**PROBLEM STATEMENT:**

The topic is **“Revolutionizing customer support with an intelligent chatbox for automated assistance”.** This topic is based on a real world problem. Many products has customer service,if the customer has a query or a problem with the product the customer can directly contact the customer service through calls,emails etc…The person working in the product has to reply to the customer and take actions according to it. This project recieves the customer reviews and produce automated assistance based on the customer feedback through an intelligent chatbox.

The dataset conatins 29 columns, which has the customers questions and answers replied by the company. The dataset also contains the ratings of the customer based on the reply of the company. The type of problem is the rule based algorithm or AI-powered algorithm using Natural Language Processing(NLP). This problem is important because it saves the time of the product producer. It is worth solving because it saves time for both the customer and the producer,when the customer gives the review or the feedback the next moment the automated assistance is provided by the chatbox.

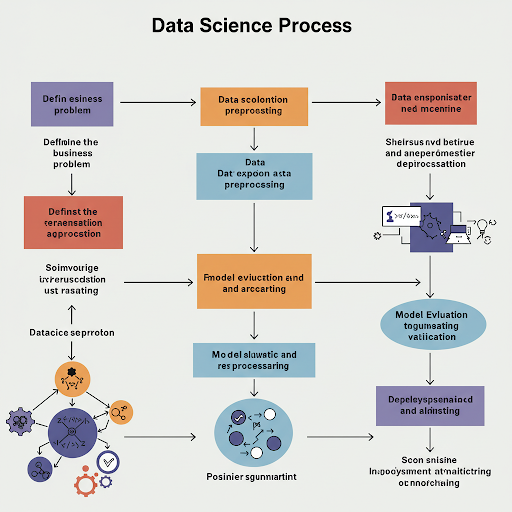
**PROJECT OBJECTIVES:**

This project goal is to automatically save the response from the customer and automatically gives response to it to satisfy the customer

Key Technical objectives:

It’s main purpose is to reveal the customer’s level of satisfaction and help product, customer success, and marketing teams understand where there is room for improvement. The model aims to achieve the accuracy and real-world-applicability. The goal has changed after exploration not only giving response to the feedback it also helps us to find ways to improve the product.

**FLOW CHART OF THE PROJECT:**



**DATA DESCRIPTION:**

Dataset name is Customer care dataset. It’s origin is gts.ai. Last date of update was Nov7, 2019 , authored and provided by City of Tempe. The type of data used here is a structured data based on time series. It also contains text format of data. It contains 29 records it contains features such as regarding the year and date of the product query feedback released, the number of responses to it, it also contains details regarding whether the customer is satisfied or neutral or very satisfied and object id. The dataset is a dynamic dataset.

**Dataset Link: <https://in.docworkspace.com/d/sIH_V_dGNAbaat8AG>**

Target Variable: Sentiment, Satisfaction score, Likelihood, Customer Retention etc..

**DATA PREPROCESSING:**

**Handling missing values:** Verified the dataset there is no missing values.

**Duplicate records:** The dataset contains duplicate data. It is irrelevant to the data. Since, the duplicate data is an dependent data therefore it can also be removed for the purpose of the project.

duplicates = df.duplicated()

**Outliers:** Checked for the absence of outliers.

**Data Types:** Data types are converted and checked for the consitency.

**Encoding categorical variables:** Label encoding is done using:

df['Product\_Encoded'] = label\_encoder.fit\_transform(df['Product'])

One hot encoding is done using:

product\_encoded = onehot\_encoder.fit\_transform(df[['Product']]) product\_df = pd.DataFrame(product\_encoded, columns=onehot\_encoder.get\_feature\_names\_out(['Product']))

**Standardization:** Numerical features are used for standardization.

**Transformation steps used:**

**Lowercasing text :** df['Comment\_Lower'] = df['Comment'].str.lower()

**Removing punctuation:** df['Comment\_NoPunct'] = df['Comment'].str.replace(r'[^\w\s]+', '', regex=True)

**Tokenization :** df['Comment\_Tokens'] = df['Comment'].str.split()

**Label encoding:**from sklearn.preprocessing import LabelEncoder; le = LabelEncoder(); df['Product\_Encoded'] = le.fit\_transform(df['Product'])

**One-Hot encoding:** df = pd.get\_dummies(df, columns=['Rating'], prefix=['Rating'])

**Min-Max scaling:** from sklearn.preprocessing import MinMaxScaler; scaler = MinMaxScaler(); df['Word\_Count\_Scaled'] = scaler.fit\_transform(df[['Word\_Count']])

**Calculating word count:** df['Word\_Count'] = df['Comment'].apply(lambda x: len(str(x).split()))

**Standardizing a numerical column:** from sklearn.preprocessing import StandardScaler; scaler = StandardScaler(); df['Word\_Count\_Standard'] = scaler.fit\_transform(df[['Word\_Count']])

**Apllying TF-IDF to the comment text:** from sklearn.feature\_extraction.text import TfidfVectorizer; tfidf = TfidfVectorizer(); tfidf\_matrix = tfidf.fit\_transform(df['Comment'])

**EXPLORATORY DATA ANALYSIS(EDA):**

**Univariate analysis:** Histograms, Box plots, Density Plots(KDE), Quantile Plots can be used.

**Bivariate analysis:** Scatter plots, Heatmaps,correlation and various plots can also be used.

**Key insights:** Sentimental scores and TF-IDF are the key insights.

**FEATURE ENGINEERING:**

1. Textr-Based feature
2. Meta-based feature(is available)
3. Interaction feature(Combining existing features)

Combined and split techniques are also used as a part of feature engineering.

**MODEL BUILDING:**

Logistic regression and decision tree algorithm which are binary classification algorithm can be used here. These models are used here because it is a linear model that can work surprisingly well for text classification, especially with well-engineered features and to perform interpretable classification or regression tasks like sentiment analysis or satisfaction prediction.

Models can be trained based on its classification and regression.

**VISUALIZATION OF RESULTS AND MODEL INSIGHTS:**

**Confusion Matrix:**A confusion matrix visualizes the performance of a classification model by showing the cous.

**ROC Curve (Receiver Operating Characteristic:**A graph that plots the True Positive Rate (TPR) against the False Positive Rate (FPR) at various classification thresholds.

**Feature Importance Plot:**A bar chart or a similar visualization that ranks the features in order of their importance to the model's predictions. The importance is typically measured by how much the model's performance decreases when the feature is removed or perturbed.

**Regression Models (e.g., Predicting Satisfaction Score):**

**1. Residual Plots**

**2. Predicted vs. Actual Values Plot**

****TOOLS AND TECHNOLOGIES:****

****Programming language:**** The main language used is python since it is easy to understand and easy and efficient to code.

**IDE/Notebook:** The platform used is Google colab. Even jupyter notebook can also be used.

**Libraries:** The libaries used are Matplotlib, Numpy, Pandas, TensorFlow, PyTorch, Scikit-learn.

**Visualizations tools:** The tools used for visualizing are:

· Matplotlib

· Seaborn

· Plotly

· Bokeh

· Altair

· Tableau

· Power BI

· Qlik Sense

· Looker

· wordcloud (Python library)

· NetworkX (Python library)

· Microsoft Excel

· Google Sheets

**TEAM MEMBERS AND CONTRIBUTION:**

**Data cleaning:** The data cleaning process is done by P. Kavitha.

**Feature engineering:** Feature engineerting is done by D. Swathi.

**EDA:** EDA process is done by X. Mary Prakasam.

**Model Development, Documentation and reporting:** Developing the model, documentation and reporting, guiding the team is done by A. Jasmine Joicy.