**Source code:** Revolutionizing Customer Support with an Intelligent for Automated Assistance

Import libraries

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt from sklearn.model\_selection

import train\_test\_split from sklearn.feature\_extraction.text

import TfidfVectorizer from sklearn.linear\_model

import LogisticRegression from sklearn.pipeline import Pipeline from sklearn.metrics

import classification\_report, accuracy\_score, confusion\_matrix import gradio as gr

Load dataset

df = pd.read\_csv("Historical\_ticket\_data.csv") df.columns = df.columns.str.strip()

Exploratory Data Analysis (EDA)

plt.figure(figsize=(15, 10)) plt.subplot(2, 2, 1) sns.countplot(data=df, y='Issue Category', order=df['Issue Category'].value\_counts().index) plt.title('Issue Category Distribution') plt.subplot(2, 2, 2) sns.countplot(data=df, x='Sentiment', order=df['Sentiment'].value\_counts().index) plt.title('Sentiment Distribution') plt.subplot(2, 2, 3) sns.countplot(data=df, x='Priority', order=df['Priority'].value\_counts().index) plt.title('Priority Distribution') plt.subplot(2, 2, 4) sns.countplot(data=df, x='Resolution Status', order=df['Resolution Status'].value\_counts().index) plt.title('Resolution Status Distribution') plt.tight\_layout() plt.show()

Data Processing

X = df['Issue Category'] + " " + df['Sentiment'] + " " + df['Priority'] y = df['Solution'] X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

Model Building

model = Pipeline([ ('tfidf', TfidfVectorizer()), ('clf', LogisticRegression(max\_iter=1000)) ]) model.fit(X\_train, y\_train)

Model Evaluation

y\_pred = model.predict(X\_test) print("\nClassification Report:\n", classification\_report(y\_test, y\_pred)) print("\nAccuracy Score:", accuracy\_score(y\_test, y\_pred)) print("\nConfusion Matrix:\n", confusion\_matrix(y\_test, y\_pred))

Gradio Chatbot Interface

def chatbot(issue, sentiment, priority): input\_text = f"{issue} {sentiment} {priority}" solution = model.predict([input\_text])[0] return f"Suggested Solution: {solution}"

demo = gr.Interface( fn=chatbot, inputs=[ gr.Textbox(label="Issue Category"), gr.Textbox(label="Sentiment"), gr.Dropdown(choices=["Low", "Medium", "High"], label="Priority") ], outputs="text", title="AI Customer Support Chatbot", description="Enter issue details to get an automated solution." )

demo.launch()