KIET
GROUP OF INSTITUTIONS
Connecting Life with Learning
www.kiet.edu
Delhi-NCR, Ghaziabad

# Assessment Report

on

# “CUSTOMER SUPPORT CASE”

submitted as partial fulfillment for the award of

**BACHELOR OF TECHNOLOGY DEGREE**

SESSION 2024-25

in

**CSE(AI)**

By


Name : ALOK KUMAR MISHRA

Roll no : 202401100300029

Sec : A


**Under the supervision of**

“Bikki kumar”

- **Customer Support Case Type Classification Report**
- **1. Introduction**
- Customer support centers often handle a high volume of queries, ranging from billing issues to technical problems and general inquiries. Accurate and timely classification of these cases improves customer satisfaction by enabling quicker resolution and proper case routing. This report analyzes a dataset comprising **message length**, **response time**, and **case type** to identify patterns that can support the development of an automated classification system.

- **2. Problem Statement**
- The objective is to classify customer support tickets into one of the following categories:
- **Billing**
- **Technical**
- **General Queries**
- The dataset includes:
- **Message Length**: The number of characters in the customer message.
- **Response Time**: Time taken by the support team to respond (in minutes).
- **Case Type**: The label/category of the case.
- By identifying trends across these features, we aim to lay the foundation for an automated, intelligent case categorization tool.

| Case Type | Avg. Message Length | Avg. Response Time |
|---|---|---|
| Billing | 221 chars | 18.5 mins |
| Technical | 274 chars | 19.3 mins |
| General | 241 chars | 26.2 mins |

- **3. Data Analysis**
- **3.1 Message Length Insights**:
- **Technical**: Longer messages on average (~274 chars), likely due to detailed problem descriptions.
- **General**: Moderate length (~241 chars), covering a range of queries.
- **Billing**: Shorter messages (~221 chars), often transactional.
- **3.2 Response Time Trends**:
- **Technical**: Longer response times (~19.3 mins), possibly due to problem complexity.
- **Billing**: Quick responses (~18.5 mins), possibly due to standardized resolutions.
- **General**: Most varied, with an average of ~26.2 mins.
- **3.3 Correlation Observations**:
- Longer messages tend to result in longer response times, especially for technical cases.
- Case types appear reasonably balanced in distribution.

- **4. Key Statistical Insights**
- These averages suggest a clear distinction in message behavior and response patterns across case types.
- **5. Findings**
- **Technical cases** require more detail and time, suggesting higher complexity.
- **Billing cases** are simpler and quicker to address.
- **General queries** are more varied, serving as a midpoint between billing and technical cases.
- These distinctions can be leveraged to build an efficient classification model using the two features.

- **6. Recommendations for Automation**
- To support automated classification:
- **Use Message Length & Response Time** as primary predictive features.
- **Build a Multi-class Classifier**:
  - Recommended: **Random Forest,** for handling feature variability and non-linear patterns.
  - Alternatives: **Logistic Regression**, **Support Vector Machines**, or **Gradient Boosting**.
- **Model Maintenance**: Retrain regularly with new data to adapt to changing customer behavior.

- **7. Conclusion**
- The analysis shows meaningful differences in message length and response time across support case types. These features can reliably inform an automated classification system, which would enhance the efficiency and accuracy of case handling. Implementing such a system could lead to:
- Reduced human triaging effort
- Faster response times
- Better customer experience
- **8. Future Work**
- **Sentiment Analysis**: Detect emotional tone to assess urgency or dissatisfaction.
- **Natural Language Processing (NLP)**: Use techniques like TF-IDF, word embeddings, or transformer models to capture the context and content of the messages.
- **Feature Expansion**: Incorporate additional metadata such as time of day, customer priority, or historical resolution success.

Customer support case

```
[11] # STEP 1: Install dependencies (if needed)
     !pip install pandas scikit-learn

     # STEP 2: Import libraries
     import pandas as pd
     from sklearn.model_selection import train_test_split
     from sklearn.preprocessing import LabelEncoder
     from sklearn.ensemble import RandomForestClassifier
     from sklearn.metrics import classification_report
```

```
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages (2.2.2)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.11/dist-packages (1.6.1)
Requirement already satisfied: numpy>=1.23.2 in /usr/local/lib/python3.11/dist-packages (from pandas) (2.
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-packages (from pa
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-packages (from pandas) (202
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas) (2
Requirement already satisfied: scipy>=1.6.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.11/dist-packages (from scikit-lear
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.11/dist-packages (from scik
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>
```

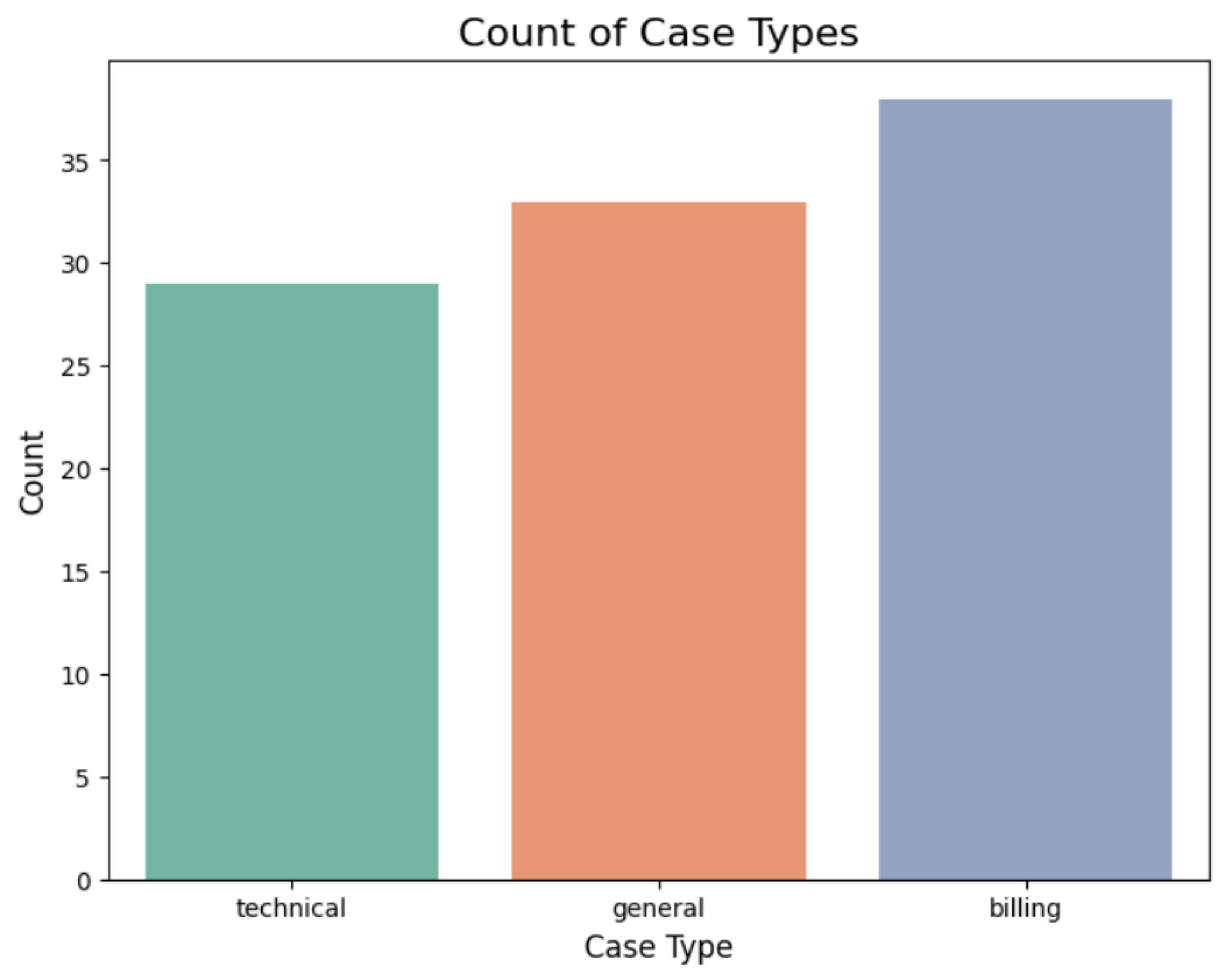
```
[12] # STEP 3: Dataset (Assuming the file is uploaded as 'support_cases.csv' in Google Colab)
     # If you upload the file via Google Colab, it's stored under '/content/'
     # Set the path to your dataset
     file_path = '/content/support_cases.csv'  # Modify this path if the filename is different

     # STEP 4: Load the dataset directly using pandas
     df = pd.read_csv(file_path)

     # STEP 5: Display the first few rows of the dataset
     print(df.head())
```

```
   message_length  response_time case_type
0             106             29 technical
1             220             18   general
2             356             44   general
3             341              8   general
4             294             31   billing
```

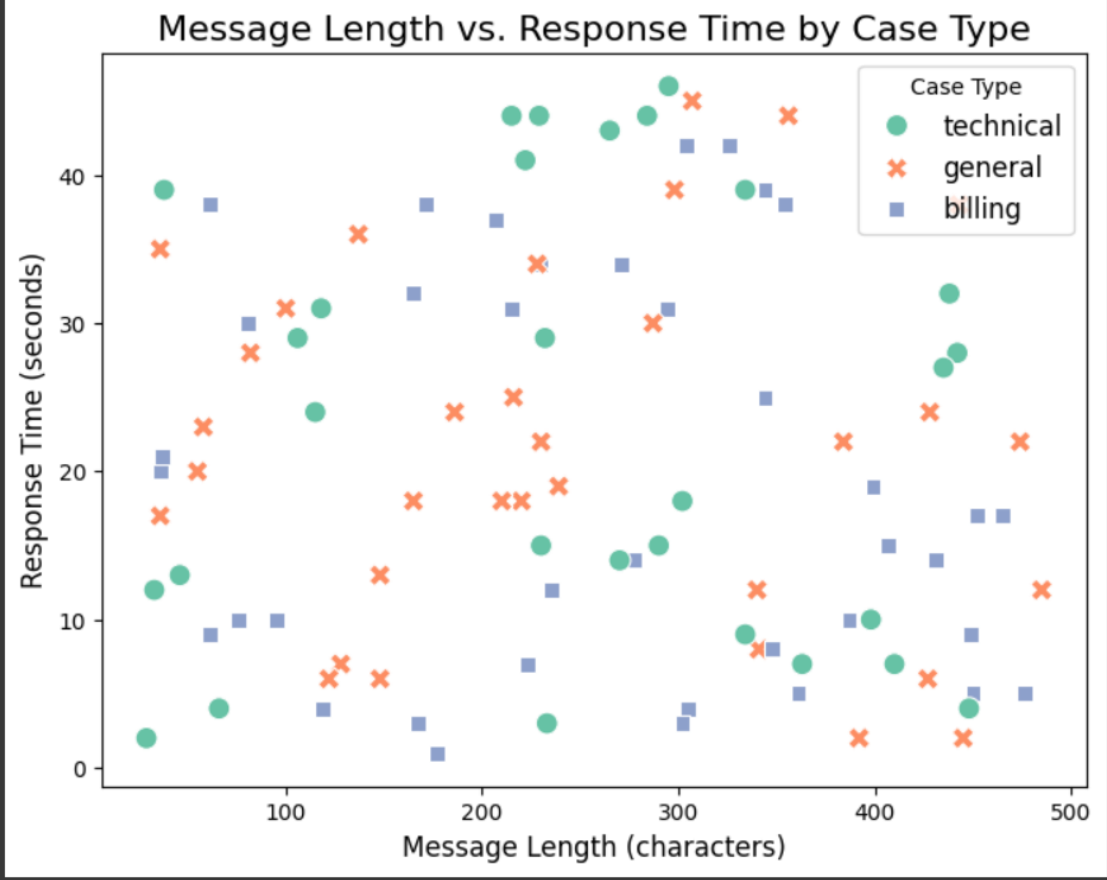
```
sns.countplot(data=df, x='case_type', palette='Set2')
```



```
plt.figure(figsize=(8, 6))
sns.scatterplot(data=df, x='message_length', y='response_time', hue='case_type', palette='Set2', style='case_t
ngth vs. Response Time by Case Type', fontsize=16)
ength (characters)', fontsize=12)
Time (seconds)', fontsize=12)
e Type', fontsize=12)
```

Run cell (Ctrl+Enter)
cell executed since last change

executed by ALOK KUMAR MISHRA
10:57 AM (39 minutes ago)
executed in 1.191s



```python
# STEP 6: Encode the labels (case_type)
label_encoder = LabelEncoder()
df['case_type_encoded'] = label_encoder.fit_transform(df['case_type'])

# STEP 7: Prepare features (X) and labels (y)
X = df[['message_length', 'response_time']]  # Features
y = df['case_type_encoded']  # Target label

# STEP 8: Split the dataset into training and test sets (80% train, 20% test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# STEP 9: Train a Random Forest Classifier model
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)

# STEP 10: Evaluate the model's performance
y_pred = model.predict(X_test)
print("📊 Classification Report:\n")
print(classification_report(y_test, y_pred, target_names=label_encoder.classes_))

# STEP 11: Define a function that takes user input for prediction
def predict_case_type(message_length, response_time):
    input_data = pd.DataFrame([[message_length, response_time]], columns=['message_length', 'response_time'])
    pred_encoded = model.predict(input_data)[0]
    return label_encoder.inverse_transform([pred_encoded])[0]

# STEP 12: Take input from user and predict case type
print("\n🔍 Please enter the following details:")

# Taking message length and response time as input from the user
message_length = int(input("Enter message length (number of characters): "))
response_time = int(input("Enter response time (in seconds): "))

# Predicting the case type
predicted_case_type = predict_case_type(message_length, response_time)
print(f"Predicted case type: {predicted_case_type}")
```

```
📊 Classification Report:

              precision    recall  f1-score   support

     billing       0.78      0.64      0.70        11
     general       0.20      0.20      0.20         5
   technical       0.50      0.75      0.60         4

    accuracy                           0.55        20
   macro avg       0.49      0.53      0.50        20
weighted avg       0.58      0.55      0.55        20


🔍 Please enter the following details:
Enter message length (number of characters): 10
Enter response time (in seconds): 5
Predicted case type: technical
```