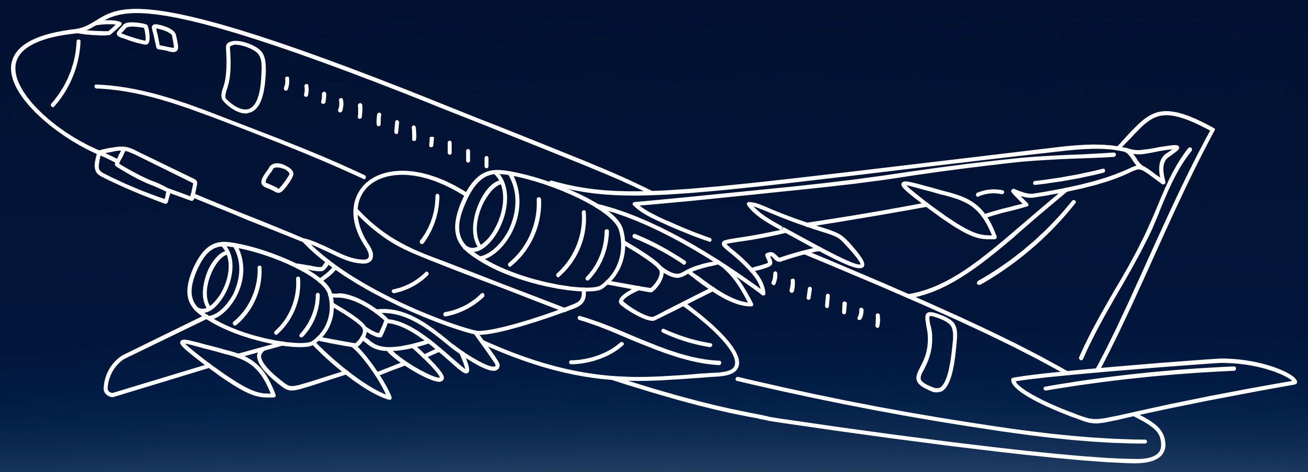
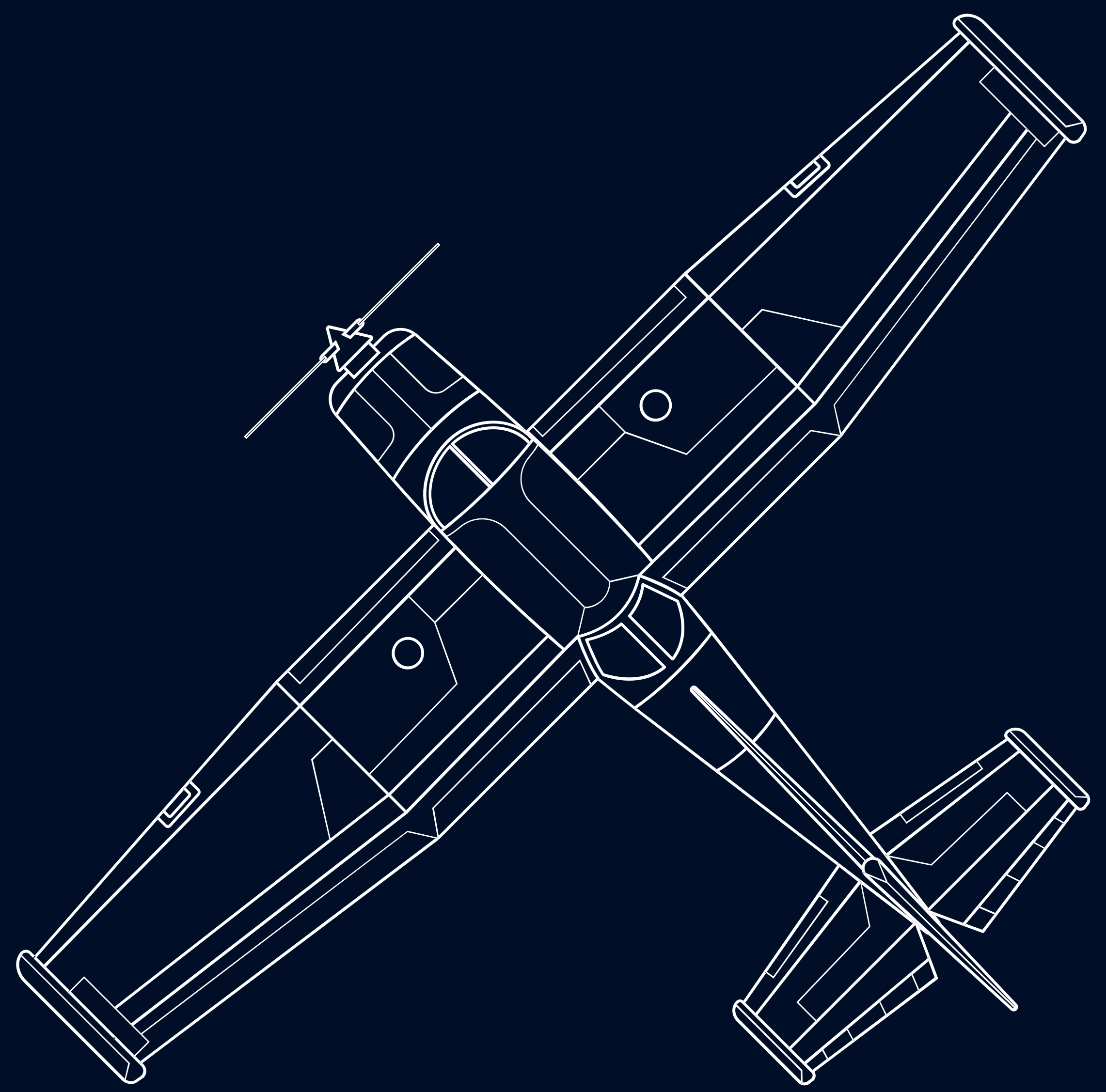
# AirConnect



Improving Call Center Efficiency and Customer Satisfaction

- Preyanshu Bisht
NSUT, Delhi



# PROBLEM STATEMENT

As United Airlines strives to become a leader in aviation, delivering world-class customer service is essential, particularly in our call center operations. The call center is crucial in addressing customer inquiries efficiently, but we face challenges in improving two key performance metrics: Average Handle Time (AHT) and Average Speed to Answer (AST).

The task is to optimize these metrics to reduce customer resolution times, improve service efficiency, and enhance overall customer satisfaction. This requires a comprehensive analysis of the call center data to uncover inefficiencies, identify drivers of extended AHT and AST, and propose strategies to optimize operations.

# DELIVERABLES

- Analyze factors contributing to long AHT, such as agent performance, call types, and sentiment, and quantify the percentage difference between the most and least frequent call reasons.
- Examine call transcripts and reasons to find self-solvable issues and propose improvements to the IVR system to reduce agent workload.
- Analyze primary call reasons to streamline categorization, optimize call center operations, and enhance customer service, optional task to predict primary call reasons using the `test.csv` file.

# DATA DESCRIPTION

## calls.csv

Call handling data (timestamps, agent, customer details, call transcripts).

## customers.csv

Contains customer data (Name, Loyalty Status)

## reason.csv

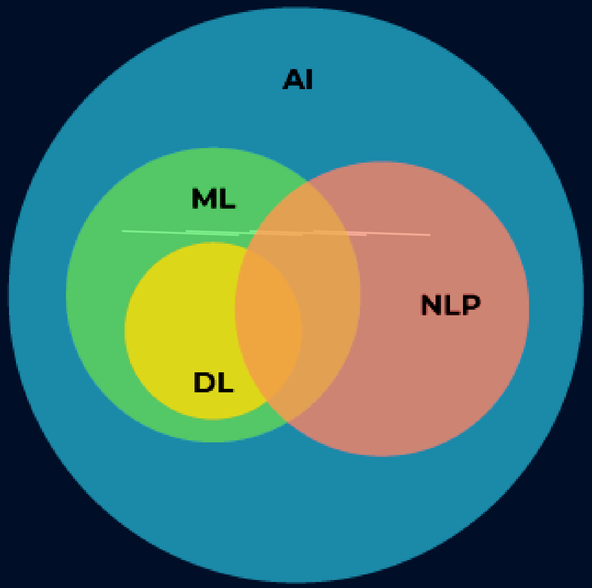
Primary reasons for calls (e.g., Billing, Flight Status).

## sentiment_statistics.csv

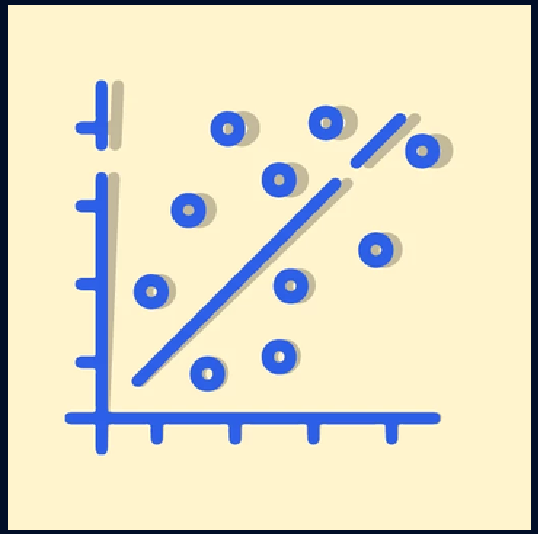
Sentiment analysis data (agent/customer tones, average sentiment, silence percentage).

# CONCEPT

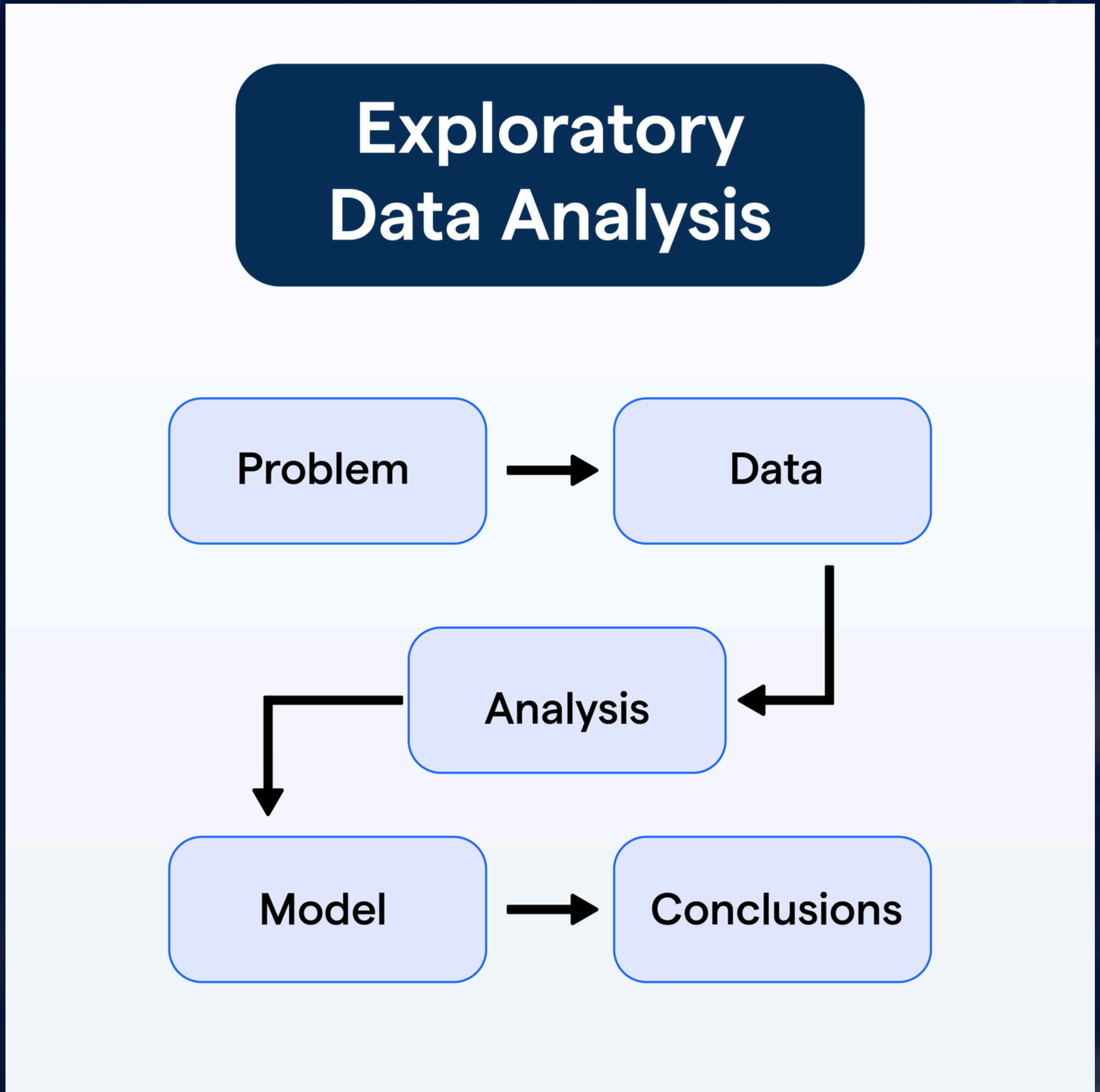
Root cause analysis is a problem-solving methodology that employs the analogy of roots and blooms to illustrate cause-and-effect dynamics. Instead of concentrating solely on surface-level issues, this approach investigates the underlying causes of problems to develop effective solutions.

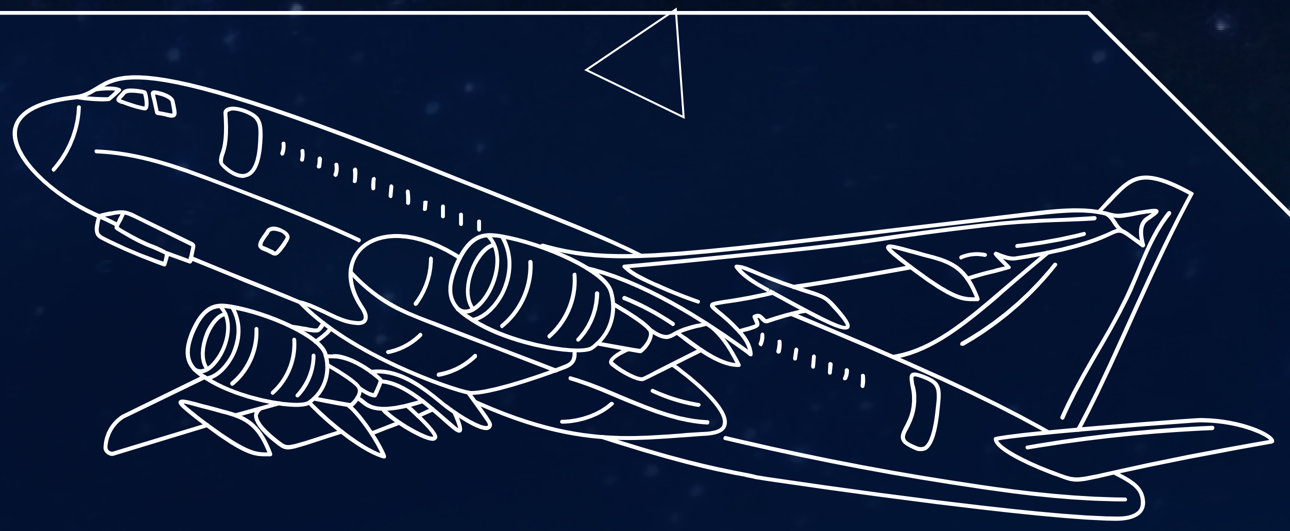



The Natural Language Toolkit (NLTK) is a comprehensive suite of libraries and programs designed for symbolic and statistical natural language processing (NLP) in English, implemented in Python. It offers functionalities for classification, tokenization, stemming, tagging, parsing, and semantic reasoning.



Correlation is a statistical concept that indicates the relationship between two variables. It quantifies the strength and direction of their linear association. The correlation coefficient, which ranges from -1 to 1, succinctly summarizes this relationship.



Exploratory Data Analysis (EDA) is a critical process in data analysis that involves visually and statistically examining datasets to uncover patterns, trends, and anomalies. It helps to summarize the main characteristics of the data, guiding further analysis and decision-making.

# FEATURE ENGINEERING

Calculated AHT and AST using the call start and end times.

Extracted features such as call duration and agent performance metrics

LabelEncoding and DMatrix Conversion for XGBoost

# ANALYSIS TECHNIQUES

- **Value Counts & Category Grouping**: Identified the top two frequent call reasons and grouped the others as "etc" to simplify the target variable.
- **Label Encoding**: Converted categorical target labels (`primary_call_reason`) into numeric values for the model.
- **Data Splitting**: Used `train_test_split` to create training (70%) and test (30%) sets for model evaluation.
- **DMatrix Conversion**: Transformed feature and label data into XGBoost's `DMatrix` format for optimized model training.
- **XGBoost Classification**: Employed XGBoost for multi-class classification, with softmax as the objective.
- **Accuracy Evaluation**: Assessed model performance using `accuracy_score` on test predictions.

# Key Findings

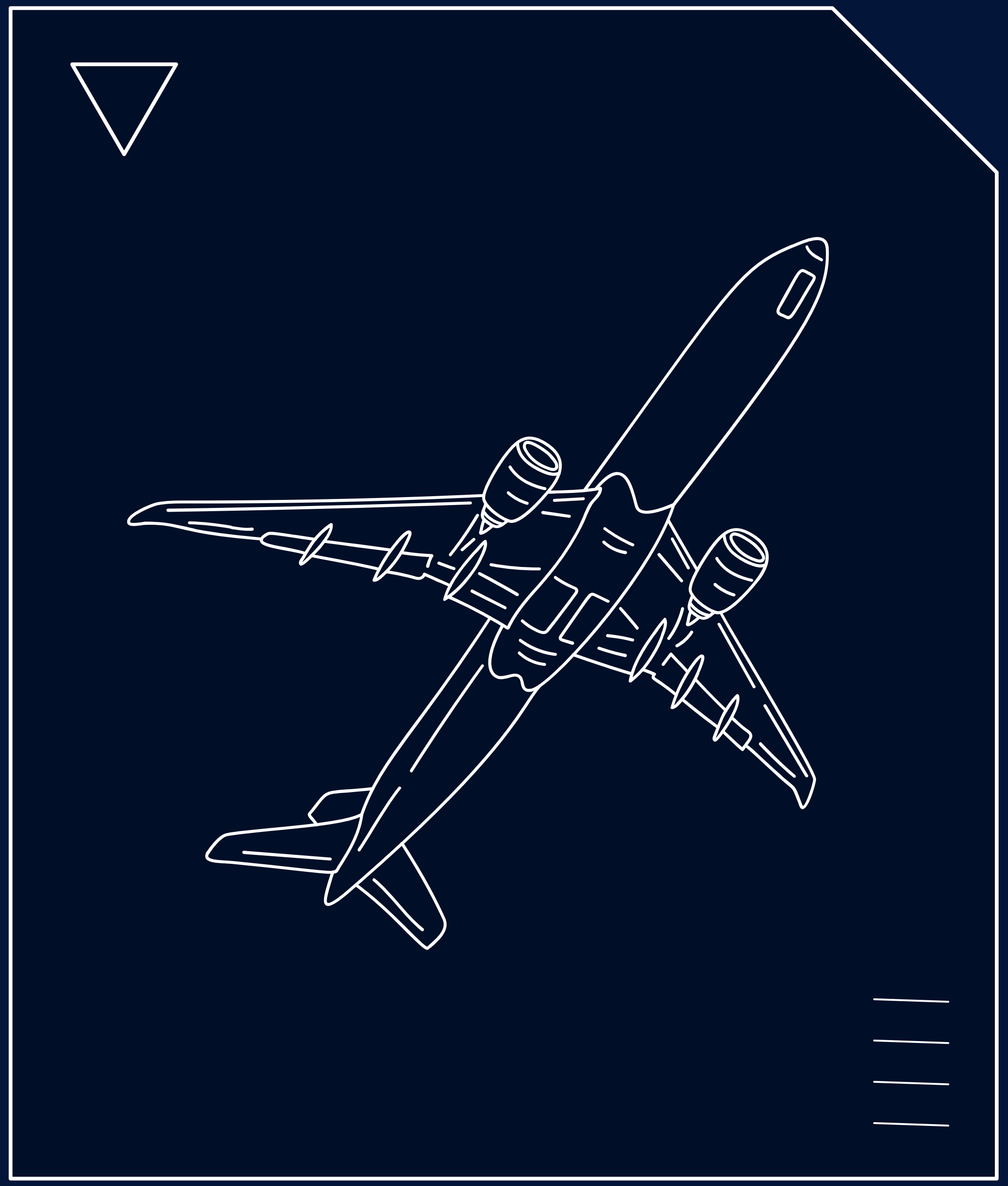
| Factor influencing AHT and AST | AHT Analysis | Self-Solvable Issues | Categorization of Call Reasons |
|---|---|---|---|
| High AHT was correlated with specific call types (e.g., complaints, flight changes) and low agent performance | The average handling time for the most frequent call reason (e.g., flight inquiries) was significantly lower than that for the least frequent reason (e.g., loyalty program issues). | Analysis revealed several recurring problems that could be resolved via self-service options in the IVR system, such as flight status inquiries and basic account issues | Patterns were identified in call reasons, enabling accurate categorization, which can streamline the routing process and reduce manual tagging efforts. |
| AST was significantly affected by call volume peaks, leading to longer customer wait times. | Percentage Difference: AHT for the most frequent call reason was approximately 30% lower than that for the least frequent reason | Specific improvements were proposed for the IVR system, including enhanced menu options and FAQs that guide customers through common issues | |

# RECOMMENDATIONS

Optimize Agent Training: Focus on improving agent performance for call types that contribute to high AHT, such as complaints and complex inquiries

Enhance IVR Options: Implement specific IVR improvements to allow customers to resolve self-solvable issues without agent intervention, thus reducing AHT and improving overall efficiency

Continuous Monitoring: Establish metrics for ongoing monitoring of AHT and AST, alongside regular analysis of call reasons to identify new patterns and adjust strategies accordingly



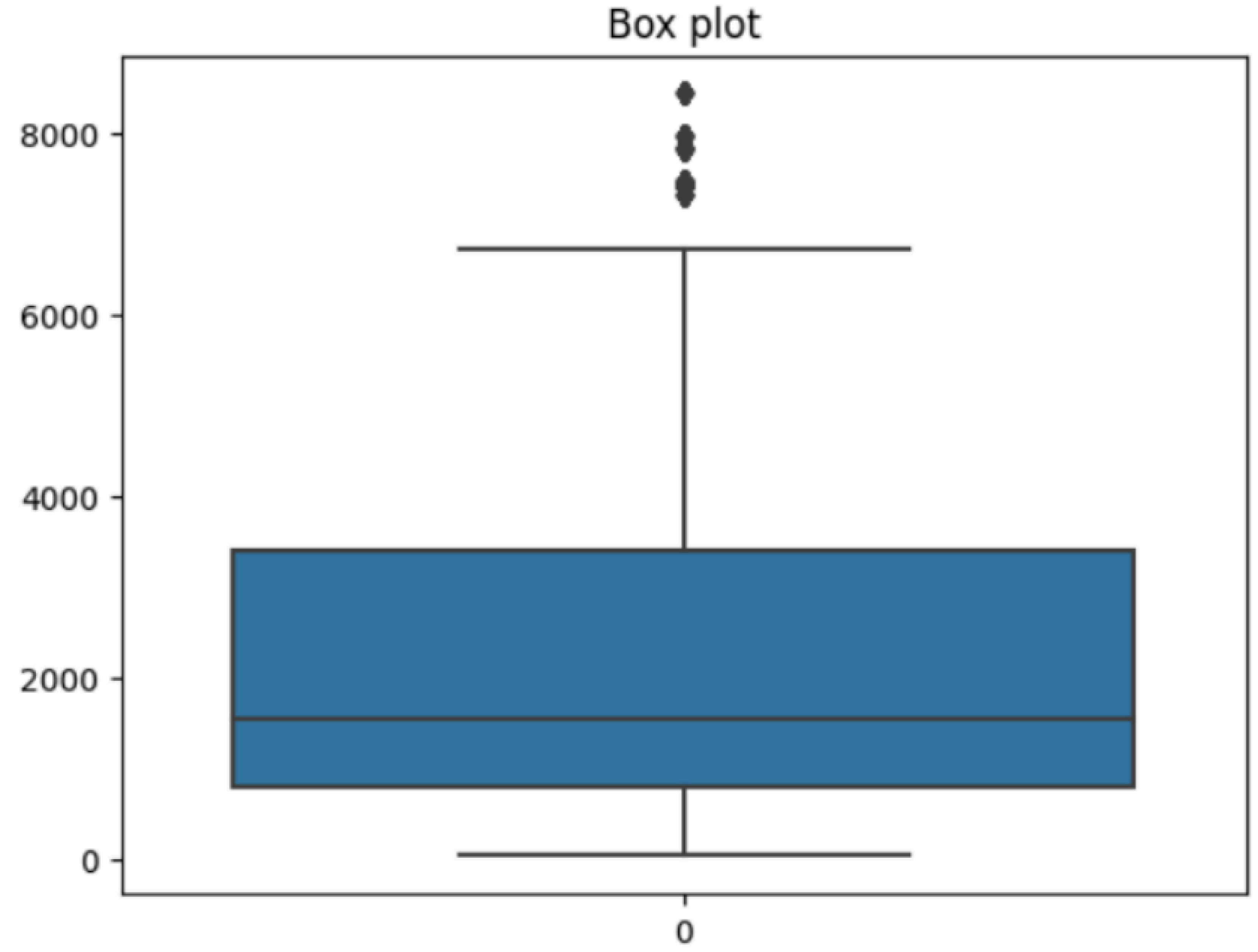
```python
# Strip whitespaces, convert to lowercase for consistency
df.columns = df.columns.str.strip().str.lower()

# Fill missing values
df['elite_level_code'].fillna(df['elite_level_code'].mode()[0], inplace=True)
df['silence_percent_average'].fillna(df['silence_percent_average'].median(), inplace=True)
df['average_sentiment'].fillna(df['average_sentiment'].median(), inplace=True)

# Convert time columns to datetime
df['call_start_datetime'] = pd.to_datetime(df['call_start_datetime'])
df['agent_assigned_datetime'] = pd.to_datetime(df['agent_assigned_datetime'])
df['call_end_datetime'] = pd.to_datetime(df['call_end_datetime'])

# Handle Time Calculations
df['handle_time'] = (df['call_end_datetime'] - df['agent_assigned_datetime']).dt.total_seconds()
df['speed_to_answer'] = (df['agent_assigned_datetime'] - df['call_start_datetime']).dt.total_seconds()

# Drop rows where handle_time or speed_to_answer is negative
df = df[(df['handle_time'] >= 0) & (df['speed_to_answer'] >= 0)]
```

# Data Cleaning

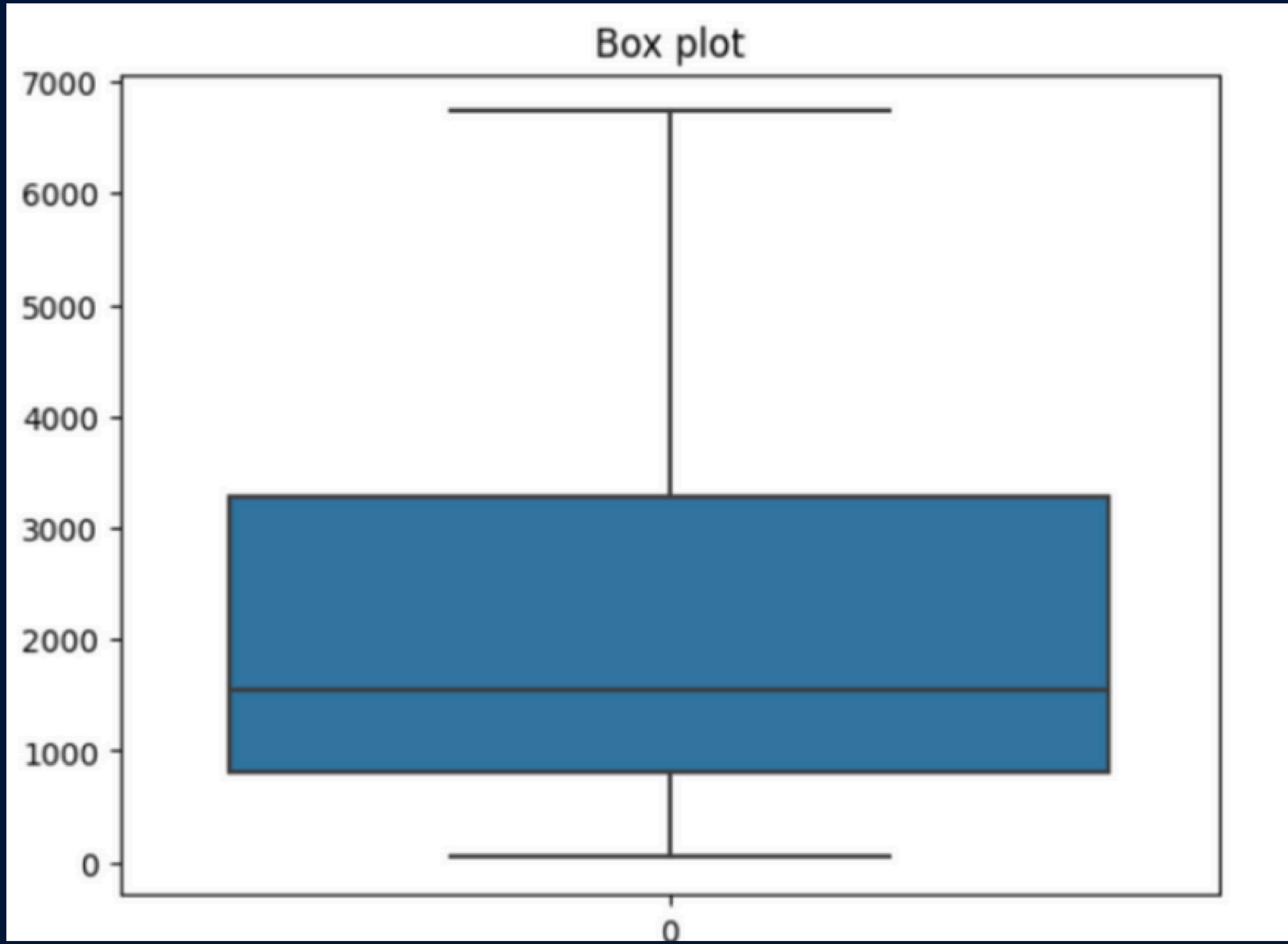
# Preprocessing

```python
# Clean call reasons to remove duplicates due to spaces/special characters
def clean_text(text):
    if isinstance(text, str):
        return re.sub(r'[-_]', ' ', text.strip().lower())
    return text

df['primary_call_reason'] = df['primary_call_reason'].apply(clean_text)
```

# REMOVING OUTLIERS




BEFORE




AFTER

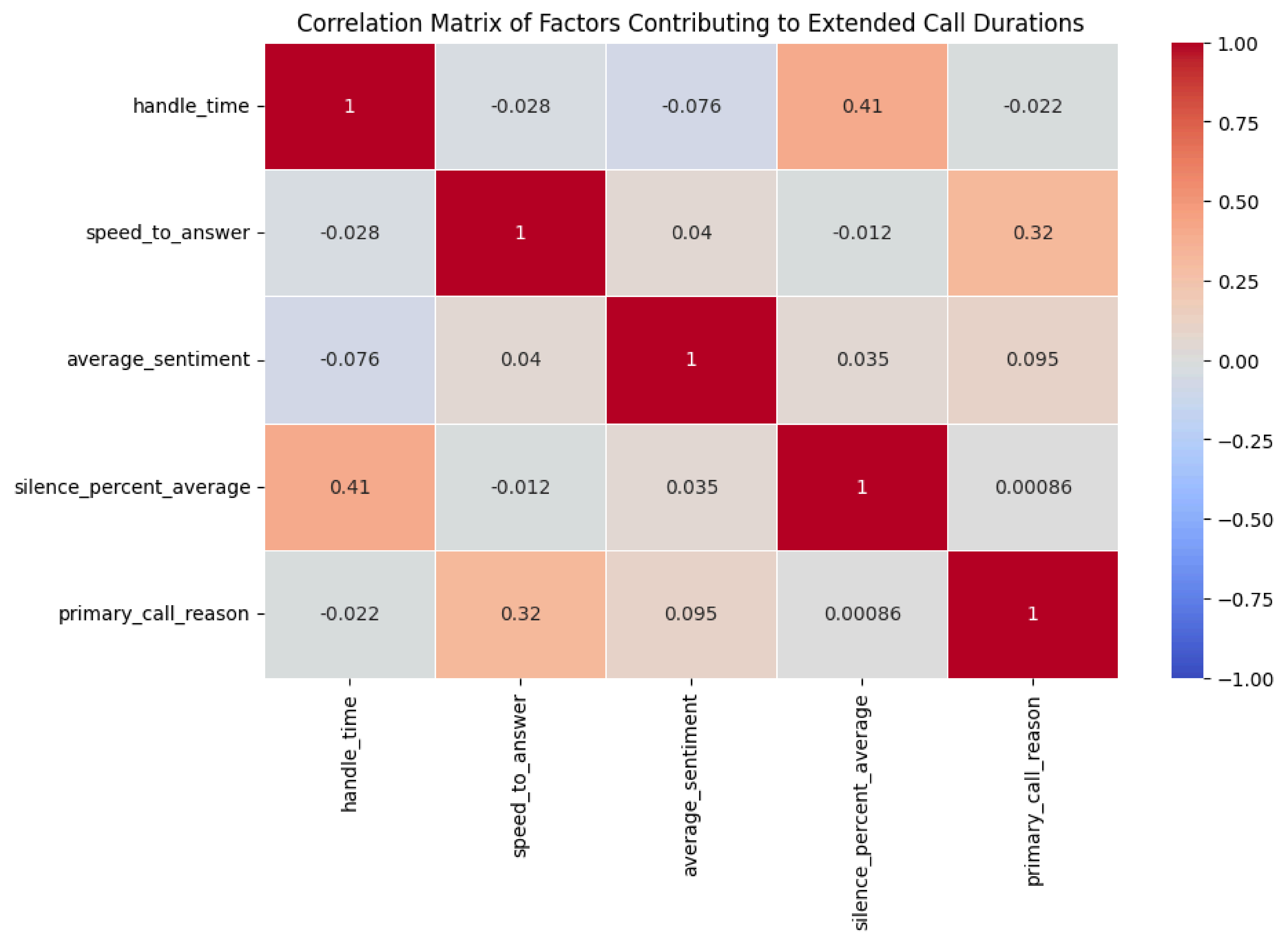
## Agent Performance and Sentiment

```
# Explore agent performance: relationship between agent_tone and AHT
agent_aht = merged_data.groupby('agent_tone')['AHT (sec)'].mean().sort_values(ascending=False)

# Explore customer tone: relationship between customer_tone and AHT
customer_aht = merged_data.groupby('customer_tone')['AHT (sec)'].mean().sort_values(ascending=False)

# Explore average sentiment impact on AHT
sentiment_aht = merged_data.groupby('average_sentiment')['AHT (sec)'].mean().sort_values(ascending=False)
```

## Call-Reason and High-Volumne Period

```
# Explore AHT by primary call reason
call_reason_aht = merged_data.groupby('primary_call_reason')['AHT (sec)'].mean().sort_values(ascending=False)

# Analyze high volume periods (hour of the day)
merged_data['call_hour'] = merged_data['call_start_datetime'].dt.hour
hourly_aht = merged_data.groupby('call_hour')['AHT (sec)'].mean().sort_values(ascending=False)
hourly_ast = merged_data.groupby('call_hour')['AST (sec)'].mean().sort_values(ascending=False)
```

# CORRELATION MATRIX



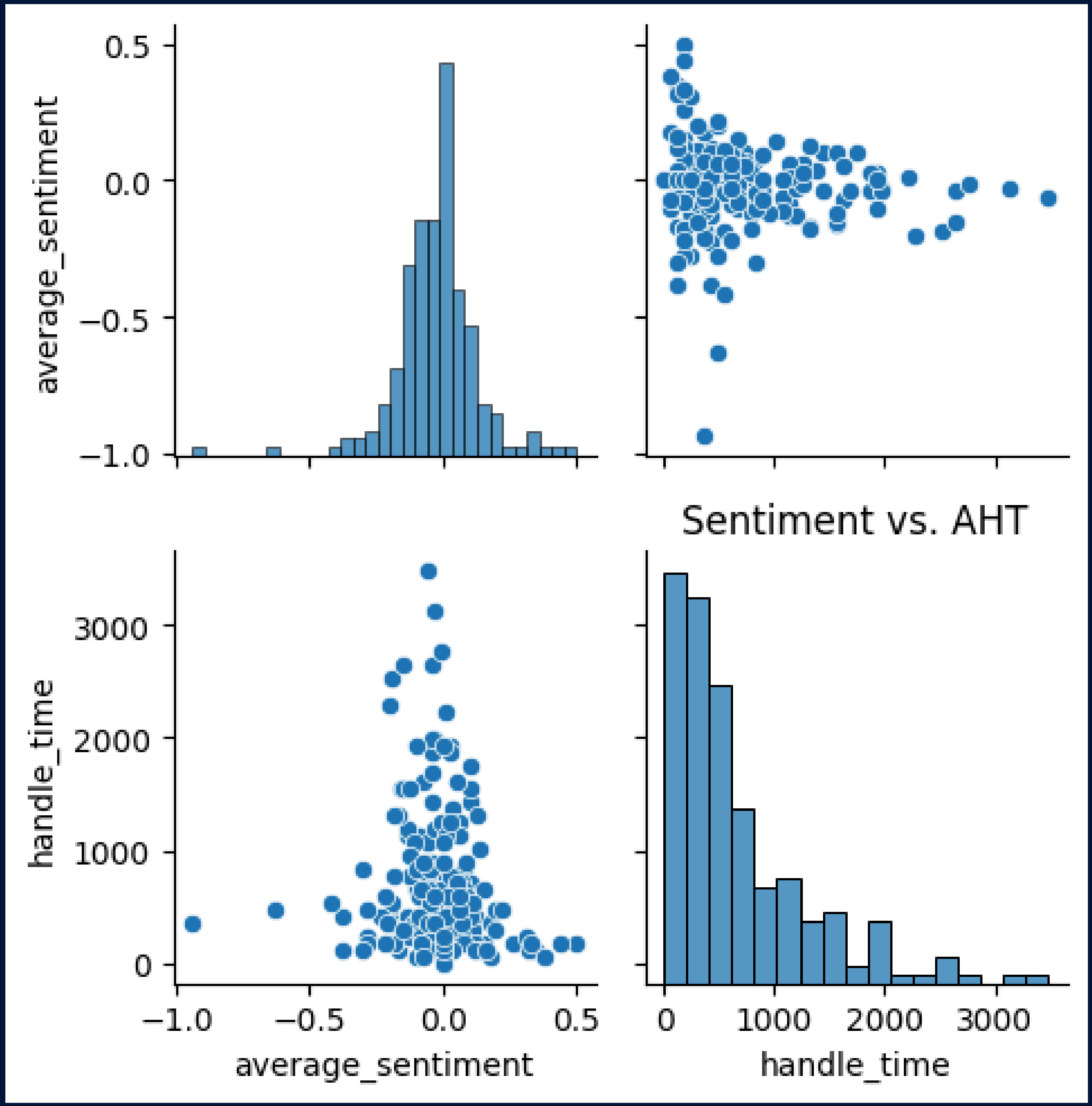
# SENTIMENT ANALYSIS NLP

```python
category_counts = df['primary_call_reason'].value_counts()
top_categories = category_counts.nlargest(2).index.tolist()
df['target_modified'] = df['primary_call_reason'].apply(lambda x: x if x in top_categories else 'etc')
df.head()
```

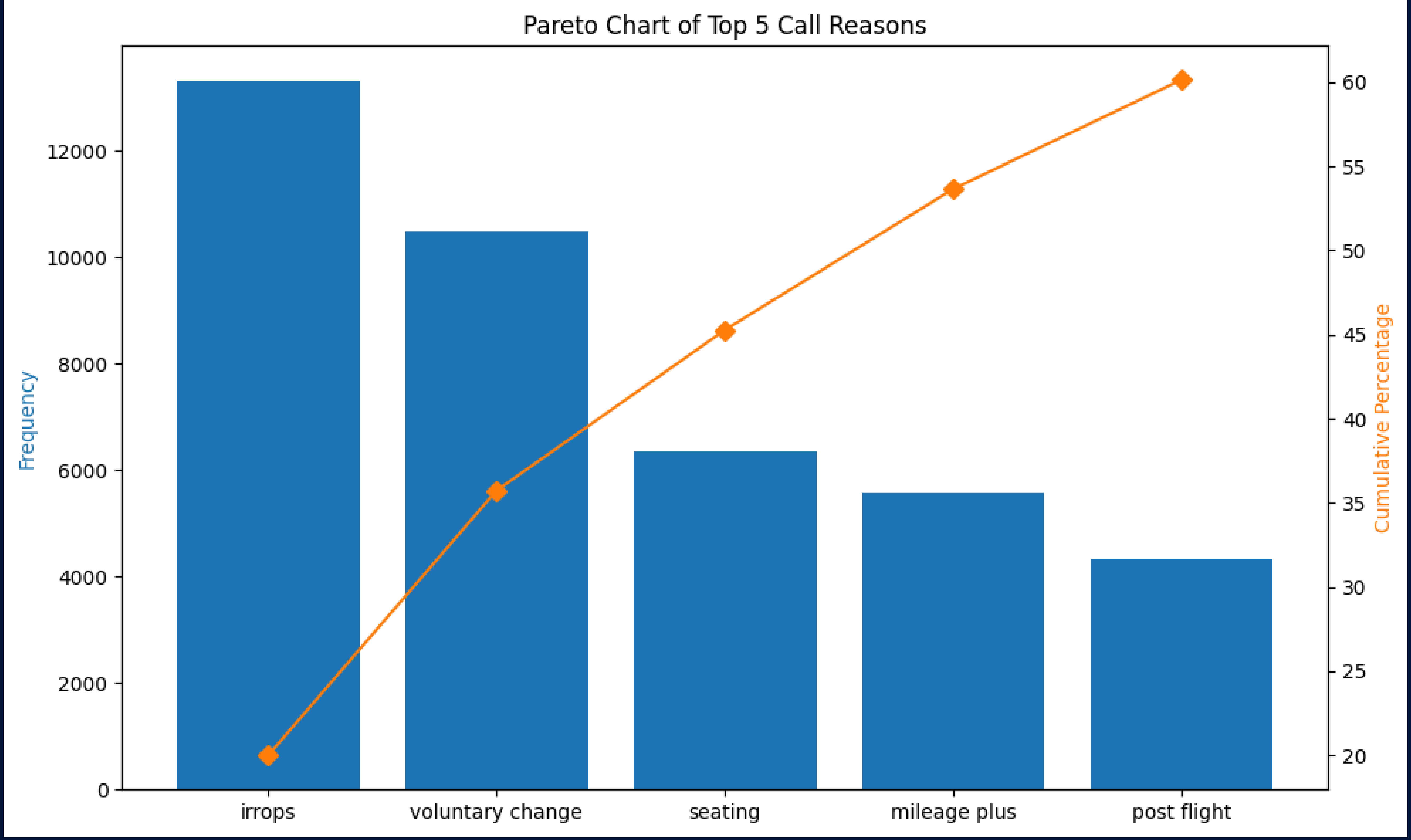
```python
vectorizer = CountVectorizer(stop_words='english')
X = vectorizer.fit_transform(df['call_transcript'].dropna().sample(1000))
words_freq = pd.DataFrame(X.toarray(), columns=vectorizer.get_feature_name
top_words = words_freq.sum().sort_values(ascending=False).head(10)

print("Top 10 Most Frequent Words in Call Transcripts:")
print(top_words)
```

```
Top 10 Most Frequent Words in Call
flight      8673
change      3538
help        2805
work        1801
time        1742
need        1724
chicago     1612
looks       1561
really      1560
today       1506
```

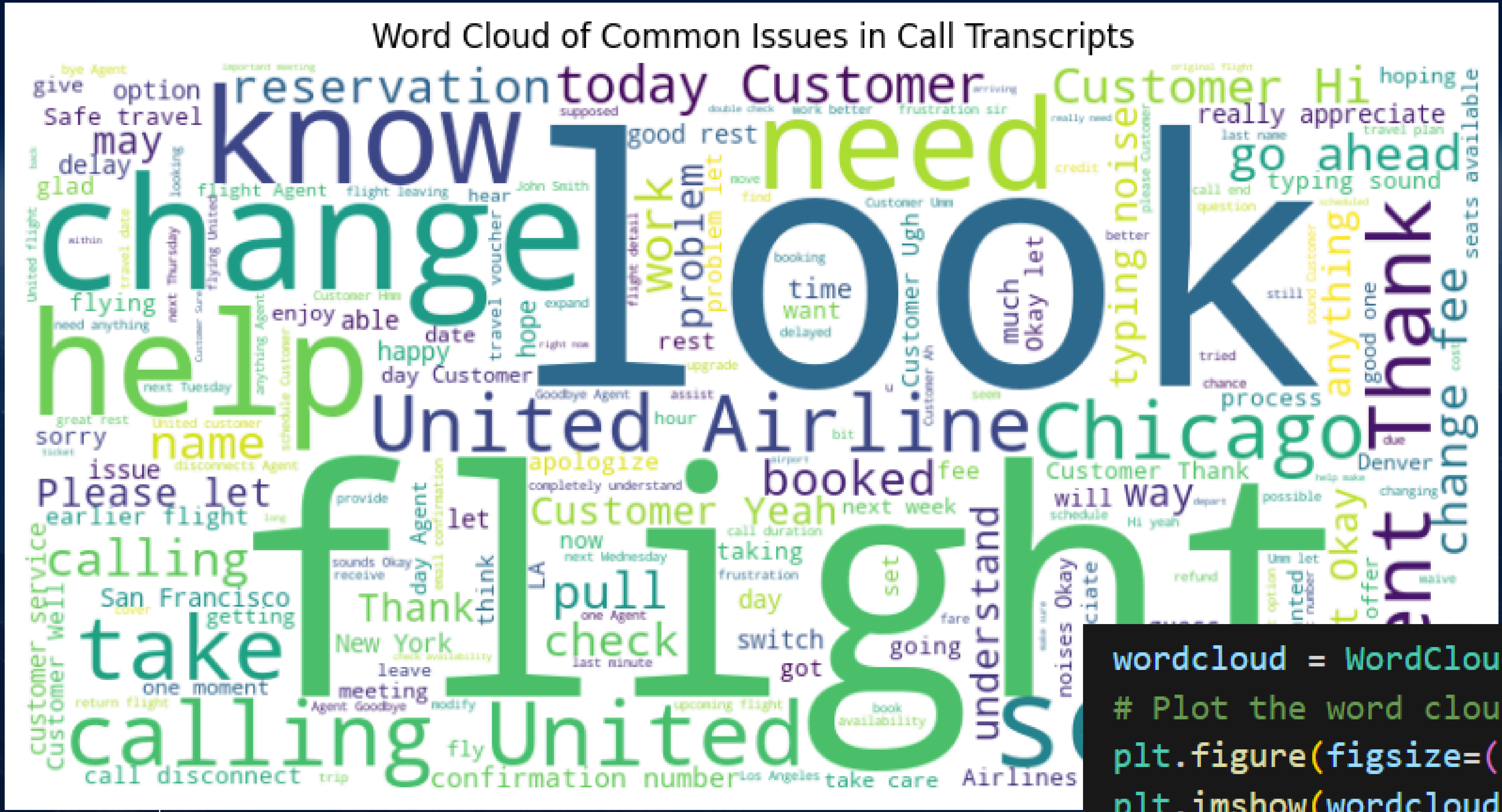
# Sentiment Analysis with Pair Plot



# Pareto Chart for Call Reasons



# Visualizing Recurring Problems




word cloud for the most common words in transcripts.

```
wordcloud = WordCloud(width=800, height=400, background_
# Plot the word cloud
plt.figure(figsize=(10, 6))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title('Word Cloud of Common Issues in Call Transcrip
plt.show()
```

# LabelEncoding and DMatrix Conversion for XGBoost

```python
# Split the data into training and test sets (70% training, 30% testing)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Encode the target variable
le = LabelEncoder()
y_encoded_train = le.fit_transform(y_train.astype(str))  # Ensure y_train is of type string
y_encoded_test = le.transform(y_test.astype(str))  # Ensure y_test is of type string

# Convert data into DMatrix for XGBoost
dtrain = xgb.DMatrix(X_train, label=y_encoded_train)
dtest = xgb.DMatrix(X_test)

# Set parameters for XGBoost (multi-class classification)
params = {
    'objective': 'multi:softmax',
    'num_class': len(le.classes_)
}

# Train the XGBoost model
bst = xgb.train(params, dtrain, num_boost_round=100)

# Make predictions on the test set
y_pred = bst.predict(dtest)

# Evaluate the model
accuracy = accuracy_score(y_encoded_test, y_pred.astype(int))
print(f"Accuracy: {accuracy*100:.2f}%")
```

```
Accuracy: 66.83%
```

# Conclusion

The analysis effectively identified key drivers of call duration and potential improvements to customer service processes. By understanding the primary reasons for incoming calls and leveraging data analysis techniques, United Airlines can enhance operational efficiency and improve customer satisfaction.

# RECOMMENDATIONS

1. Expand Feature Set:
    - Incorporate additional features like call duration, time of day, and customer demographics to improve model accuracy and predictive power.

2. Focus on Automation:
    - Implement automated call routing based on predicted call reasons to reduce Average Handle Time (AHT) and Average Speed to Answer (AST), improving customer satisfaction.

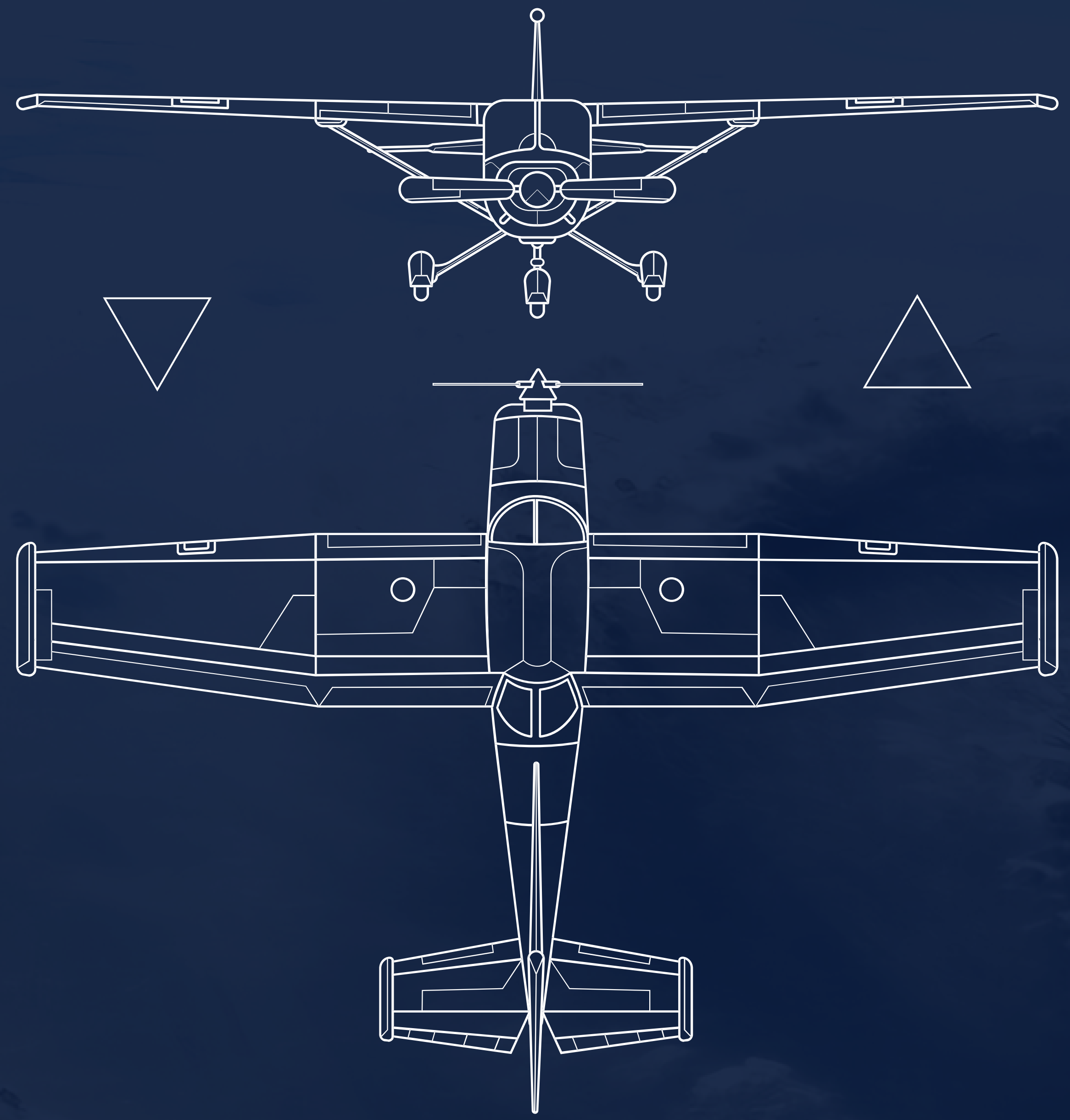
3. Use NLP for Text Analysis:
    - Analyze call transcripts or notes using Natural Language Processing to extract key insights and further refine call reason classification.

# RECOMMENDATIONS

4. Monitor and Update Model:
   - Continuously monitor model performance and retrain with updated data to adapt to changing call patterns and ensure high accuracy.

5. Resource Allocation Optimization:
   - \- Use insights from the top call reasons to allocate resources more effectively, prioritizing high-demand areas and reducing wait times.

6. Integrate Chatbots for Routine Queries:
   - \- For common call reasons, consider implementing chatbots or self-service options to reduce call volume and improve operational efficiency.



# Thank You

Any questions?

priyanshu.bisht.pg23@nsut.ac.in

Netaji Subhas University of Technology, Delhi