# Final Project Report: Smart Customer Support System

# Natural Language Processing Laboratory (Assignment 5)

## 1. Abstract

In the modern digital service landscape, customer support teams are often overwhelmed by the sheer volume of incoming queries. This project presents an **End-to-End Natural Language Processing (NLP) Pipeline** designed to automate the initial triage of support tickets. By implementing a dual-model architecture, the system simultaneously categorizes the *topic* of a ticket (Billing vs. Technical) and analyzes the *sentiment* (Positive vs. Negative). The result is an intelligent routing system that can flag urgent issues from angry customers while automatically organizing standard queries, thereby reducing response times and human error.

## 2. Problem Statement & Motivation

### The Challenge

Manual ticket sorting is inefficient. A human agent spends valuable time reading an email just to decide which department should handle it. Furthermore, amidst thousands of "How do I reset my password?" emails, a critical complaint from a furious customer might go unnoticed for hours, leading to churn.

### The Solution

We propose an automated "Smart Triage" system. The goal is not to replace human agents, but to empower them by:

1. **Filtering:** Instantly routing tickets to the correct department (Finance, IT, or General).
2. **Prioritizing:** acting as a "traffic light" system where angry customers get a "Red Light" (Escalation) and standard queries get a "Green Light" (Auto-reply).

## 3. System Design & Methodology

### 3.1 Dataset Generation

Due to privacy concerns with real-world support emails, a **synthetic dataset** was programmatically generated for this project. The dataset consists of 300 labeled examples across three categories:

- **Billing:** Keywords like *invoice, charge, refund, credit card*.
- **Technical Support:** Keywords like *crash, error, login, install, server*.
- **General Inquiry:** Keywords like *hours, location, partnership, policy*.

### 3.2 Data Preprocessing

Raw text is noisy and unstructured. To prepare the data for modeling, a preprocessing pipeline was implemented using the NLTK library:

- **Noise Removal:** Regular expressions (re) were used to strip punctuation and special characters, leaving only alphabetic text.
- **Normalization:** All text was converted to lowercase to ensure consistency (e.g., treating "Invoice" and "invoice" as identical).
- **Stop-word Removal:** Common English words (e.g., "the", "is", "and") were removed to reduce feature space dimensionality.
- **Lemmatization:** Words were reduced to their root forms (e.g., "crashing" $\rightarrow$ "crash") using the WordNet Lemmatizer.

### 3.3 Feature Engineering (TF-IDF)

Instead of simple word counts (Bag of Words), we utilized **Term Frequency-Inverse Document Frequency (TF-IDF)**. This method was chosen because support tickets often contain generic polite words ("please", "help"). TF-IDF down-weights these common terms and highlights domain-specific keywords like "invoice" or "bug," which are crucial for classification.

### 3.4 Model Architecture

The core innovation of this project is the use of **two parallel Logistic Regression classifiers**:

1. **The Topic Model:** Predicts the Department (Billing / Tech / General).
2. **The Sentiment Model:** Predicts the Urgency (Positive / Neutral / Negative).

Logistic Regression was selected over deeper neural networks because, for text classification with limited data, it offers an excellent balance of speed, interpretability, and performance.

## 4. Implementation Details

The system functions as a live command-line interface (CLI). When a user inputs a ticket, the system processes it in real-time using the following logic flow:

1. **Input:** User types "I want my money back!"
2. **Vectorization:** The text is cleaned and converted to a numeric vector.
3. **Inference:** Both models make a prediction.
4. **Routing Logic (The "Traffic Light" System):**
    - **Red Flag:** If Sentiment is *Negative* Escalate immediately (overrides topic).
    - **Yellow Flag:** If Topic is *Billing* Route to Finance.
    - **Blue Flag:** If Topic is *Tech Support* Route to IT.
    - **Green Flag:** If Topic is *General* Send automated FAQ.

## 5. Results & Evaluation

### 5.1 Performance Metrics

The models were evaluated on a 20% test split. The system achieved high accuracy (>90%) across both tasks.

- **Topic Classification:** The model showed near-perfect separation between categories. This is expected, as the vocabulary difference between "money/invoice" and "crash/login" is distinct.
- **Sentiment Analysis:** Performance was strong for clearly negative words ("hate", "terrible"), though the model showed slightly lower recall for "passive-aggressive" complaints that lacked strong negative keywords.

### 5.2 Visualization

The confusion matrix reveals that the primary source of error was between "General Inquiries" and "Tech Support" when the user used vague language (e.g., "How does this work?").

## 6. Challenges & Limitations

While the system performs well in a controlled environment, several challenges were identified during development:

1. **The Sarcasm Blind Spot:** The current TF-IDF approach struggles with sarcasm. A review stating *"Great job deleting my data!"* is classified as **Positive** because of the words "Great job," whereas a human would instantly recognize it as **Negative**.
2. **Context Limitations:** The model treats sentences as a "bag of words." It does not understand sequence. "Not good, but bad" and "Not bad, but good" look very similar to the model mathematically.
3. **Data Imbalance:** In a real-world scenario, 80% of tickets might be simple password resets. The model might become biased toward the majority class without careful resampling techniques.

## 7. Future Scope

To evolve this from a mini-project to a production-ready system, the following improvements are proposed:

- **Deep Learning (BERT):** Replacing Logistic Regression with a Transformer model (like BERT) would solve the sarcasm and context issues, as BERT understands the *relationship* between words.
- **Named Entity Recognition (NER):** Integrating an NER module to automatically extract Order IDs, Dates, and Email Addresses from the ticket body.
- **Web Deployment:** Wrapping the Python script in a **Streamlit** or **Flask** web interface to allow non-technical support staff to use the tool via a browser.

## 8. Conclusion

This project successfully demonstrated the power of NLP in automating business workflows. By combining text preprocessing, TF-IDF vectorization, and logical routing, we built a system that not only saves time but also prioritizes customer satisfaction. It serves as a strong

foundation for understanding how AI can be applied to solve real-world communication problems.