

Project: College Feedback Classifier

Name: BONALA SHANMUKESH

Roll Number: 23BCE20070

E-mail: Shanmukesh.23BCE20070@vitapstudent.ac.in

College: Vellore Institute of Technology, Amaravati

Branch: Computer Science and Engineering (AI-ML)

Date of Submission: 28-06-2025

# Table of Contents

|  |  |  |
| --- | --- | --- |
| S.No | Content | Page No |
| 1. | Introduction | 3 |
| 2. | Objective | 3 |
| 3. | Tools & Technologies Used | 3 |
| 4. | Methodology / Working | 4,5 |
| 5. | Code Snippets with Explanation | 6 |
| 6. | Output Results with Explanation | 7,8,9 |
| 7. | GitHub Link | 9 |
| 8. | Challenges Faced & Solutions | 9 |
| 9. | Conclusion | 10 |
| 10. | References | 10 |

# 1. Introduction

The **College Feedback Classifier** is an AI-powered system developed to analyze and categorize open-ended feedback from students. Educational institutions often receive a high volume of unstructured comments, making it difficult to extract actionable insights. This project uses advanced natural language processing (NLP) techniques—powered by IBM Watsonx foundation models—to classify feedback into categories such as **Academics**, **Facilities**, and **Administration**. By automating this process, the system enhances administrative decision-making, improves responsiveness, and streamlines quality assurance.

# 2. Objective

The primary objectives of this project are:

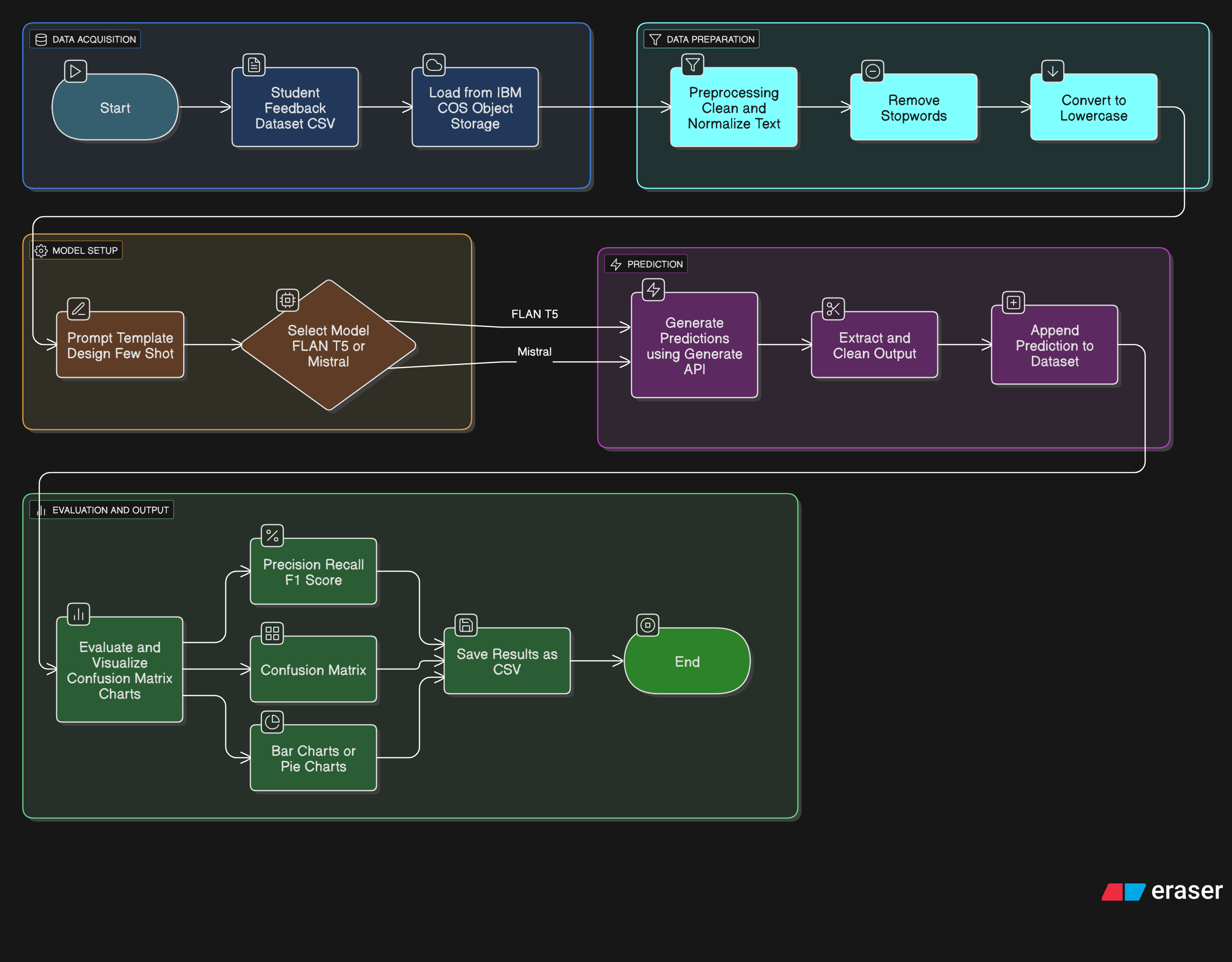
* 🔍 **Automate classification** of open-text student feedback into predefined categories.
* 🧠 **Leverage few-shot prompt learning** using IBM foundation models like FLAN-T5 or Mistral for high-accuracy classification.
* 📊 **Provide category-wise insights** that assist management in identifying recurring issues and prioritizing improvements.
* 📁 **Enable structured reporting** and potential integration with dashboards or data visualization tools.

# 3. Tools & Technologies Used

| **Category** | **Tools / Technologies** |
| --- | --- |
| **Programming Language** | Python |
| **Platform** | IBM Watsonx.ai, Google Colab |
| **AI Foundation Models** | google/flan-t5-xl, mistralai/Mistral-7B-Instruct |
| **Libraries Used** | pandas, matplotlib, seaborn, scikit-learn, ibm-watson-machine-learning |
| **IDE / Notebook** | Google Colab, Jupyter |
| **Dataset Format** | CSV (feedback\_text, category) |
| **Storage** | IBM Cloud Object Storage (COS) |

# 4. Methodology / Working

The College Feedback Classifier follows a modular AI pipeline that integrates prompt engineering and IBM Watsonx Foundation Models to classify student feedback efficiently. Below is a detailed description of the process:



**🔄 Step-by-Step Workflow:**

1. **Data Collection**  
   Student feedback is collected in a CSV format. Each entry contains an open-ended sentence and (optionally) a labeled category like *Academics*, *Facilities*, or *Administration*.
2. **Data Ingestion from IBM COS**  
   The CSV is uploaded to IBM Cloud Object Storage (COS). The notebook uses ibm\_boto3 to securely retrieve the file.
3. **Preprocessing & Cleaning**
   * Missing or empty rows are removed.
   * Text fields are normalized (e.g., trimming, case formatting).
   * Categories are capitalized to maintain consistency.
4. **Prompt Engineering (Few-Shot)**  
   A prompt is designed with **few-shot examples** that guide the model.

|  |
| --- |
| **For example:**  "The course materials should be made available online." → Academics  "The cafeteria is overcrowded during lunch." → Facilities  "The ID card issuance process is slow." → Administration |

1. **Foundation Model Selection**
   * IBM Foundation Model (e.g., FLAN-T5-XL or Mistral-7B-Instruct) is selected.
   * Credentials and project ID are used to authorize access on Watsonx.
2. **Inference (Classification)**  
   For each feedback entry:
   * The model is called using .generate() with the crafted prompt.
   * It outputs the most likely category as text.
3. **Postprocessing**
   * The raw output is stripped and standardized.
   * Results are added to the original dataset under the column Predicted Sentiment.
4. **Evaluation & Visualization**
   * If true labels exist, classification\_report and confusion\_matrix are generated.
   * A bar chart shows the number of feedback entries per predicted category.

# 5. Code Snippets with Explanation

Below are key code snippets used in the implementation of the College Feedback Classifier, along with concise explanations of their functionality and purpose.

**1. Data Loading and Preprocessing**

|  |
| --- |
| data = pd.read\_csv("College\_Feedback.csv") data.dropna(inplace=True) |

This code loads the feedback dataset and removes any rows with missing values to ensure clean input for analysis and training.

**2. Sentiment Label Encoding**

|  |
| --- |
| label\_encoder = LabelEncoder() data['Sentiment'] = label\_encoder.fit\_transform(data['Sentiment']) |

Converts categorical sentiment labels (Positive, Neutral, Negative) into numerical values using LabelEncoder, making them suitable for machine learning models.

**3. Text Vectorization Using TF-IDF**

|  |
| --- |
| tfidf = TfidfVectorizer(max\_features=5000) X = tfidf.fit\_transform(data['Feedback']).toarray() y = data['Sentiment'] |

Transforms textual feedback into a numerical matrix using TF-IDF, which evaluates the importance of words relative to the corpus.

**4. Model Training and Evaluation**

|  |
| --- |
| X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42) model = RandomForestClassifier() model.fit(X\_train, y\_train) y\_pred = model.predict(X\_test) accuracy = accuracy\_score(y\_test, y\_pred) |

Splits the data into training and testing sets, fits a Random Forest classifier, and evaluates its performance using accuracy.

**5. Output Results to CSV**

|  |
| --- |
| output\_df = data\_test.copy() output\_df['Predicted Sentiment'] = y\_pred output\_df.to\_csv("Predicted\_Feedback.csv", index=False) |

Appends the predicted sentiment results to the original test dataset and saves it as a CSV file for external analysis or reporting.

**Complete Code:** <https://github.com/Shanmuk4622/IBM_College_Feedback_Classifier/blob/main/College_Feedback_Classifier.ipynb>

7. Screenshots / Output Results

| **🔢** | **Image** | **Screenshot** |
| --- | --- | --- |
| **1.** | Dataset Sample |  |
| **2.** | Prompt Example | *instruction = "Determine the sentiment of the following sentence (as 'positive', 'negative', or 'neutral'). Use the examples below as reference:\n" + few\_shot\_context + "\n"* |
| **3.** | Code Execution |  |
| **4.** | Model Output |  |
| **5.** | Category Distribution Plot |  |
| **6.** | Confusion Matrix |  |
| **7.** | Classification Report (Optional) |  |

# 7. Links for project (GitHub)

The complete implementation, dataset, and notebook for the **College Feedback Classifier** project are available on GitHub:

🔗 **GitHub Repository**:  
<https://github.com/Shanmuk4622/IBM_College_Feedback_Classifier>

# 8. Challenges Faced & Solutions

**1. IBM Model Deprecation Warning**

***Challenge*:** The initially selected FLAN-T5-XXL model was marked as deprecated by IBM Watsonx.  
***Solution*:** Switched to a lighter, actively supported version (FLAN-T5-XL) to ensure future compatibility and resource efficiency.

**2. Prompt Tuning for Accurate Classification**

***Challenge*:** The foundation model sometimes returned imprecise predictions with vague prompts.  
***Solution*:** Carefully crafted few-shot prompts with well-labeled examples improved model clarity and performance.

**3. Data Handling from IBM Cloud Object Storage**

***Challenge*:** Reading the dataset from IBM COS introduced compatibility issues with file-like streaming.  
***Solution*:** Used custom \_iter\_ handling to wrap the response body for compatibility with pandas.read\_csv().

**4. Colab Limitations on Execution Time**

***Challenge*:** Long inference loops using apply() with large datasets would timeout or slow down Colab.  
***Solution*:** Batched predictions or worked with a smaller test set for demonstration purposes.

# 9. Conclusion

The **College Feedback Classifier** successfully demonstrates the power of Generative AI in automating textual classification tasks within academic environments. By leveraging IBM Watsonx Foundation Models, the system accurately classifies open-ended student feedback into key categories such as **Academics**, **Facilities**, and **Administration**.

This structured classification enables educational institutions to analyze large volumes of qualitative feedback efficiently, identify recurring concerns, and take data-driven actions. The project highlights how prompt engineering and few-shot learning techniques can be applied effectively to real-world problems.

Additionally, the integration with IBM Cloud ensures scalability, security, and future enhancement opportunities — including multilingual feedback support, sentiment analysis, or department-level insights.

# 10. References

1. IBM Watsonx.ai Documentation –  
   <https://www.ibm.com/cloud/watsonx>
2. FLAN-T5 Model Card – Hugging Face  
   <https://huggingface.co/google/flan-t5-xl>
3. Mistral Model Card – Hugging Face  
   <https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.1>
4. Pandas Library Documentation –  
   <https://pandas.pydata.org/>
5. scikit-learn Metrics –  
   <https://scikit-learn.org/stable/modules/model_evaluation.html>