Project: College Feedback Classifier

Name: Ashrith Sai C

Roll Number: 23BCE8770

E-mail: ashrith.23bce8770@vitapstudent.ac.in

College: Vellore Institute of Technology

Branch: Computer Science and Engineering

Date of Submission: 02-06-2025

IBM

Table of Contents

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:

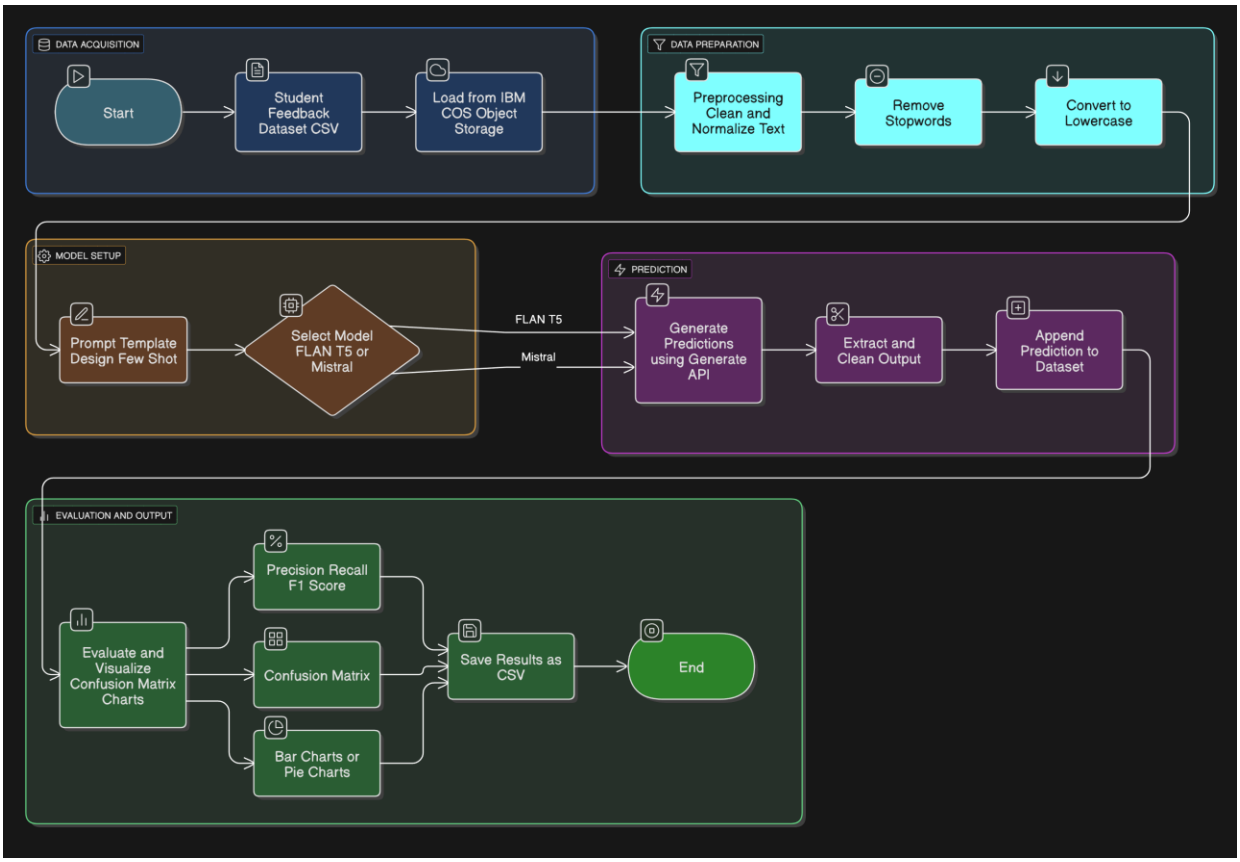
- 🔍 **Automate classification** of open-text student feedback into predefined categories.
- 🧠 **Leverage few-shot prompt learning** using IBM foundation models like FLAN-T5 or Mistral for high-accuracy classification.
- 📊 **Provide category-wise insights** that assist management in identifying recurring 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

5. **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.
6. **Inference (Classification)**
   For each feedback entry:
    - The model is called using .generate() with the crafted prompt.
    - It outputs the most likely category as text.
7. **Postprocessing**
    - The raw output is stripped and standardized.
    - Results are added to the original dataset under the column Predicted Sentiment.
8. **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/ashrithsai0214/IBM-Project

7. Screenshots / Output Results

Image **Screenshot**

1. Dataset Sample

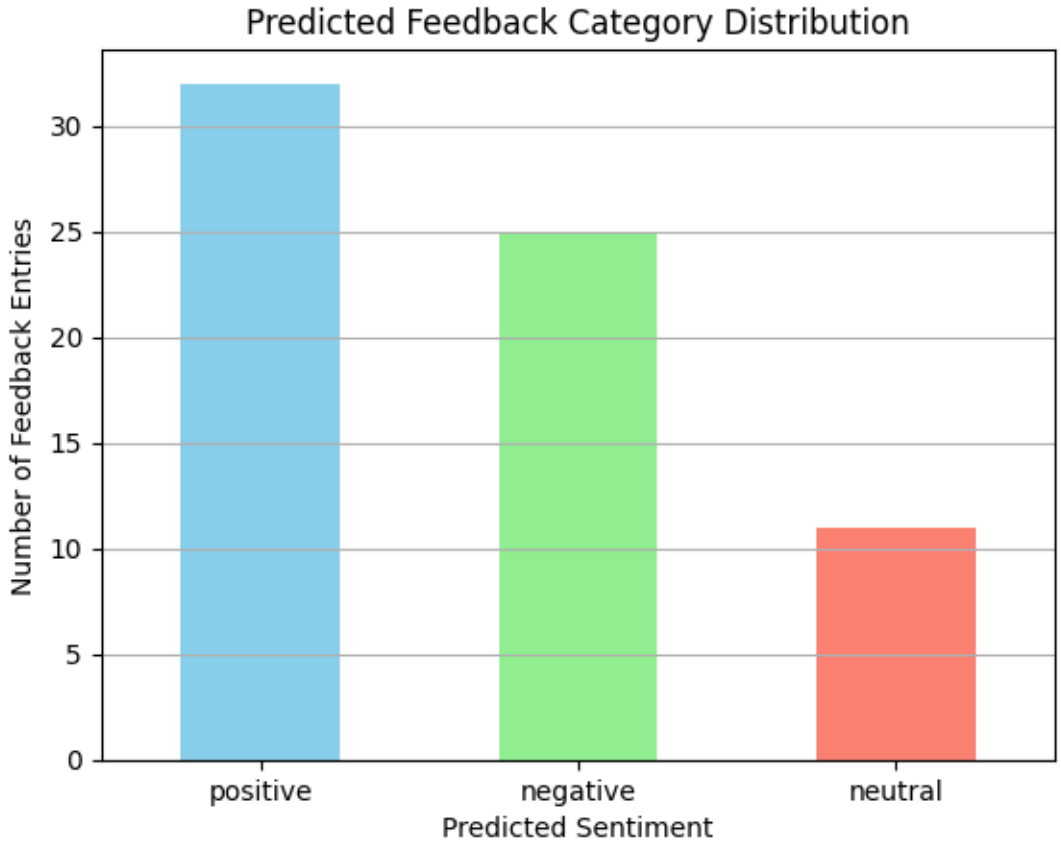
	feedback_text	category
1	feedback_text	category
2	The sports ground needs better lighting for evening practice.	Facilities
3	The course materials should be made available online.	Academics
4	The admissions helpline is difficult to reach during peak times.	Administration
5	More dustbins should be placed around the campus.	Facilities
6	The syllabus includes relevant case studies and real-world examples.	Academics
7	The process for applying for scholarships is confusing.	Administration
8	There are not enough benches in the common areas.	Facilities
9	Some subjects have too much theoretical content and less practical work.	Academics
10	The student ID card issuance takes longer than expected.	Administration
11	The gym equipment is well-maintained and modern.	Facilities
12	The exam results are declared on time.	Administration
13	The library staff are always willing to help students.	Administration
14	The classrooms are spacious and comfortable.	Facilities
15	The laboratory sessions are well-organized and supervised.	Academics
16	The online portal sometimes crashes during registration.	Administration
17	The cafeteria needs more healthy food options.	Facilities
18	The faculty encourages open discussions in class.	Academics
19	The fee payment process should be more streamlined.	Administration
20	The hostel rooms are regularly cleaned.	Facilities
21	The academic calendar is shared well in advance.	Administration

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

```
sentence: The course includes internships.
category: Academics
sentence: The teachers are inspiring.
category: Academics
sentence: The gym equipment is well-maintained and modern.
category: Facilities
sentence: The campus is clean and well-maintained.
category: Facilities
sentence: The administrative office should be more organized.
category: Administration
sentence: The process for getting character certificate is slow.
category: Administration
```

Image	Screenshot
4. Model Output	Enter a legal sentence to analyze sentiment (or type 'exit'): i always skip my hostel mess, as i Predicted Sentiment: neutral Enter a legal sentence to analyze sentiment (or type 'exit'): i skip my lunch in hostel mess , i Predicted Sentiment: negative Enter a legal sentence to analyze sentiment (or type 'exit'): my room is looking good compared to Predicted Sentiment: positive
5. Category Distribution Plot	
6. Confusion Matrix	Confusion Matrix: [[0 0 0 1 8 13] [0 0 0 19 1 3] [0 0 0 5 2 16] [0 0 0 0 0 0] [0 0 0 0 0 0] [0 0 0 0 0 0]]

🔢 **Image Screenshot**

7. Classification Report (Optional)

```
Classification Report:
                precision    recall  f1-score   support

     Academics       0.00      0.00      0.00      22.0
Administration       0.00      0.00      0.00      23.0
    Facilities       0.00      0.00      0.00      23.0
      negative       0.00      0.00      0.00       0.0
       neutral       0.00      0.00      0.00       0.0
      positive       0.00      0.00      0.00       0.0

      accuracy                           0.00      68.0
     macro avg       0.00      0.00      0.00      68.0
  weighted avg       0.00      0.00      0.00      68.0
```

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/ashrithsai0214/IBM-Project

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