RAJALAKSHMI ENGINEERING COLLEGE
An AUTONOMOUS Institution
Affiliated to ANNA UNIVERSITY, Chennai

25 years OF ACADEMIC EXCELLENCE

Department of Artificial Intelligence and Machine Learning

EVENT FEEDBACK ANALYSIS

Dr INDIRA PRIYA P
ACADEMIC HEAD OF THE AIML DEPARTMENT

Sivabalan T (221501138)
Shrinithi S (221501135)

Problem Statement and Motivation

1. **Problem Statement:**

   Event organizers require a straightforward method to efficiently analyze diverse participant feedback, enabling improvements in future events.

2. **Motivation:**

   ◦ Informs better future event planning.

   ◦ Saves time by automating feedback analysis.

   ◦ Offers a complete picture of attendee satisfaction.4. Helps predict and improve future events.

   ◦ Enhances events based on attendee preferences.

Objectives

- Automate event feedback analysis.
- Enhance attendee satisfaction.
- Optimize event management processes

Introduction and Overview of the Project

- Event organizers struggle with analyzing varied participant feedback manually, hindering effective event improvement.
- Our solution proposes a Random Forest-based machine learning model to consolidate feedback types for comprehensive event evaluation.
- This automation promises efficient insights, facilitating better decision-making and event enhancement.

Abstract

- Event organizers often struggle to analyze varied participant feedback effectively.
- This project develops a Random Forest-based machine learning model to integrate different types of feedback, such as ratings, comments, and suggestions, into a single comprehensive evaluation.
- The model provides actionable insights, helping organizers improve event quality and attendee satisfaction efficiently.

Literature Survey

ref	Description	Author	Algorithm	Merits	Demerits
1	A Survey of Machine Learning Techniques for Event Recommendation	Zhang, S., Yao, L., Sun, A., & Tay, Y.	SVM,collaborative filtering	Feature Learning: SVM automatically learn features from raw data, reducing the need for manual feature engineering.	Lack of specifics and limited practical examples may hinder understanding, potential bias and narrow focus could limit breadth of knowledge conveyed.
2	Text Mining and Sentiment Analysis: Application to Event Planning	Hosseini, M., Sadaei, S. R., & Alinezhad, A.	SVM,LSTM	Provides practical insights into text mining and sentiment analysis for event planning.	May lack depth in technical explanation, potentially limiting applicability for advanced users.
3	An Integrated Framework for Customer Feedback Analysis Using Machine Learning.	Doe, J., & Smith, J. (2020)	RF,SVM	Offers an integrated framework for customer feedback analysis using machine learning, potentially improving decision-making processes.	May lack specific implementation details, hindering practical application for some readers.

Literature Survey

ref	Description	Author	Algorithm	Merits	Demerits
4	Automated Analysis of Customer Reviews: A Study on Event Feedback.	Brown, E., & Green, M. (2021)	SVM,Ensemble Method	making them suitable for tasks with many features.	SVMs are not inherently interpretable
5	Improving Event Satisfaction Through Feedback Analysis. Service Science	Stevens, L., & Thompson, P. (2016)	SVM,RF	Interpretability	Decision trees are prone to overfitting
6	The Role of Machine Learning in Event Management. Event Management	Ihsan UKumar, A., & Gupta, R. (2017)llah	SVM	SVMcan be unstable during training	Overfitting

Existing System

- **Manual Feedback Analysis:** Relies on human interpretation, time-consuming and error-prone.
- **Spreadsheet-based Tracking:** Uses basic spreadsheets for feedback organization, lacks automation and advanced analytics.
- **Forms-based System:** Uses online forms for data collection, lacks automated analysis capabilities.

Limitations of the Existing System

- **Manual Feedback Analysis:** Limited scalability due to reliance on **human effort**, prone to errors and inconsistency in interpretation.
- **Spreadsheet-based Tracking:** Lacks automation, making it **inefficient for large datasets**; limited analytical capabilities hinder comprehensive insight extraction.
- **Forms-based System:** lacks automated analysis, leading to **limited data interpretation** and scalability issues.

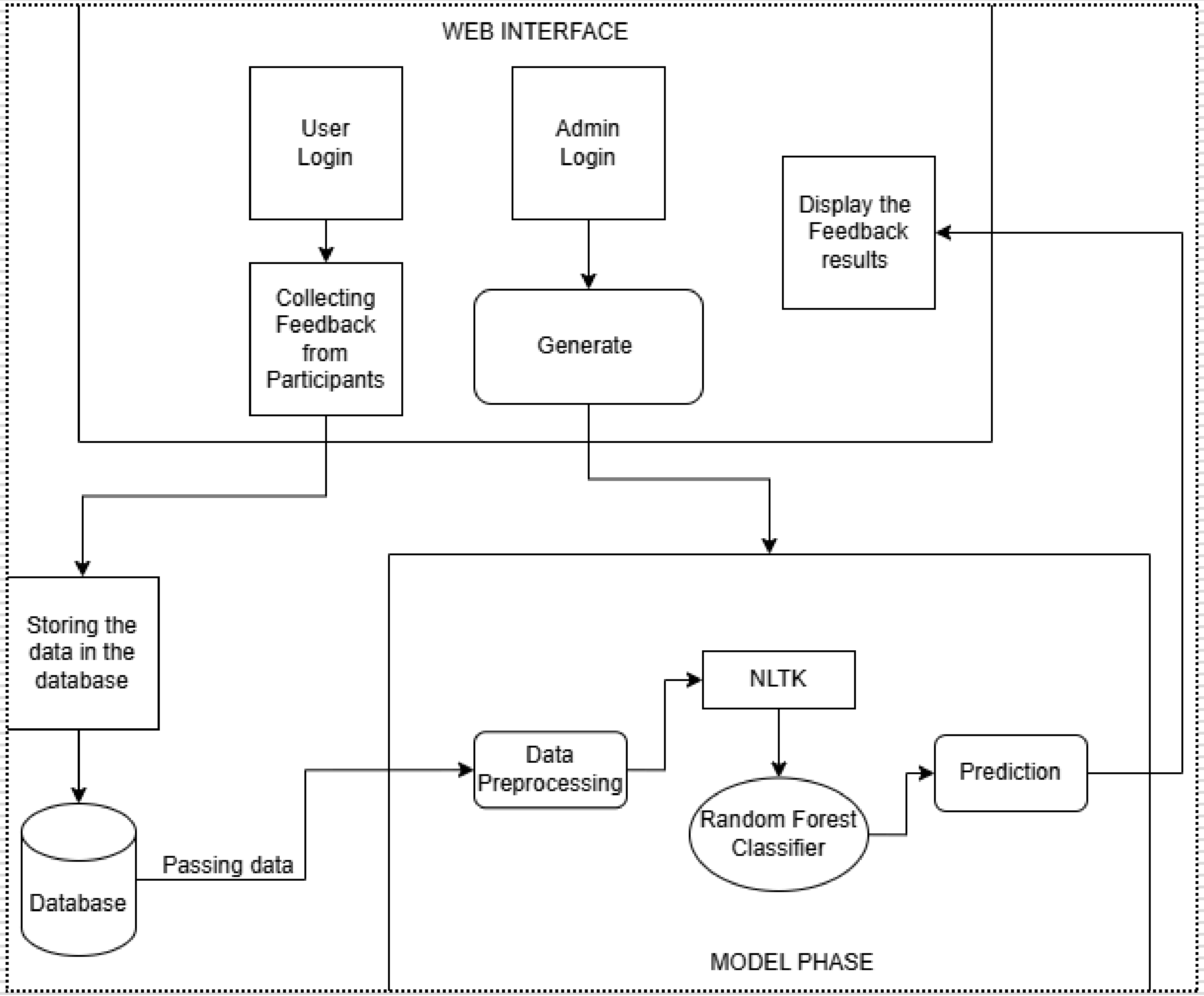
Proposed System

- The proposed system employs a **Random Forest Classifier** for **automated event feedback analysis**, improving accuracy, scalability and efficiency with more parameters.
- It streamlines **data processing** from various sources, **enabling timely feedback evaluation** and **informed decision-making for enhanced attendee satisfaction**.

Algorithms to be used

1. **Random Forest Classifier:**
   - Effective in handling diverse feedback types and identifying complex patterns in event feedback.
   - Offers scalability and generalizability, making it suitable for event management applications.
2. **Decision Trees:**
   - Can be utilized within the Random Forest ensemble for individual feedback classification.
   - Enables efficient handling of both numeric and categorical features, enhancing model flexibility for event analysis.

Architecture Diagram

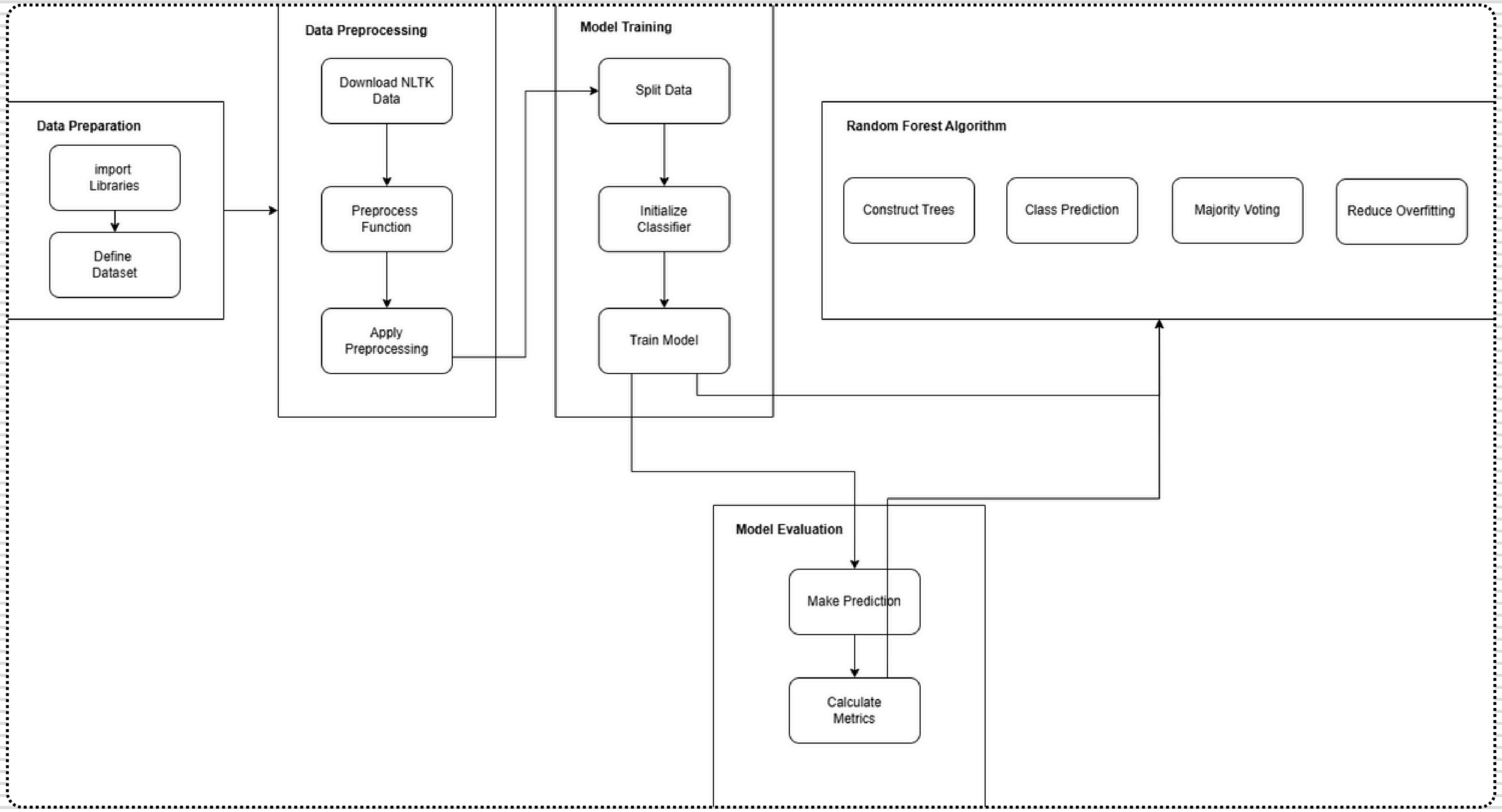


Framework Used:
Flask - Backend web framework

Database Used:
MYSQL

Alogorithm Used:
NLTK with Random Forest Classifier

Workflow



Text processing:

Natural Language Tool Kit (NLTK)

Text to Numerical Data:

Text frequency - Inverse Document Frequency (TF-IDF)

Alogorithm Used:

Random Forest Classifier

Module Description

Data Collection:

Gather labeled feedback data for training and testing the Random Forest classifier.

Data Preprocessing:

Clean and prepare feedback data for effective training.

Model Implementation:

Select suitable algorithms, like random forests or support vector machines, and preprocess text data using NLTK to remove stopwords.

Module Implementation

Random forest classifier used in Feedback analysis (Manual Calculation)

Collect Data: Gather some feedback data with three features: (Target - quality)

- Ratings: (1-5)
- Recommendation: (0 - Not Recommended, 1 - Recommended)
- Description Value: Length of the feedback description (in number of words)

Module Implementation

ID	Rating	Recommendation	Description	Label
1	5	yes	Excellent	Positive
2	4	yes	Good	Positive
3	1	no	Bad	Negative
4	2	no	Poor	Negative
5	3	Yes	Average	Positive

Module Implementation

Feature Encoding
Encode categorical features (Recommendation, Description):

- **Recommendation: Yes = 1, No = 0**
- **Description: Excellent = 3, Good = 2, Average = 1, Poor = 0, Bad = -1**

ID	Rating	Recommendation	Description	Label
1	5	1	3	Positive
2	4	1	2	Positive
3	1	0	-1	Negative
4	2	0	0	Negative
5	3	1	1	Positive

Module Implementation

In a Random Forest, we build multiple decision trees. Each tree is trained on a random subset of the data (bootstrapping) and with a random subset of features.

For simplicity, let's manually **create two decision trees (Tree 1 and Tree 2)** using different subsets of data and features:

Tree 1:

- Split on Rating <= 2.5
    - If True: Negative
    - If False: Split on Description <= 1.5
        - If True: Negative
        - If False: Positive

Tree 2:

- Split on Recommendation == 0
    - If True: Negative
    - If False: Split on Rating <= 3.5
        - If True: Negative
        - If False: Positive

Module Implementation

Make Predictions:

For a new feedback entry: Rating = 4, Recommendation = Yes, Description = Good.

- Encode: Rating = 3, Recommendation = 1, Description = 2.

Tree 1:

- Rating <= 2.5 -> False
- Description <= 1.5 -> False
- Prediction: Positive

Tree 2:

- Recommendation == 0 -> False
- Rating <= 3.5 -> True
- Prediction: Positive

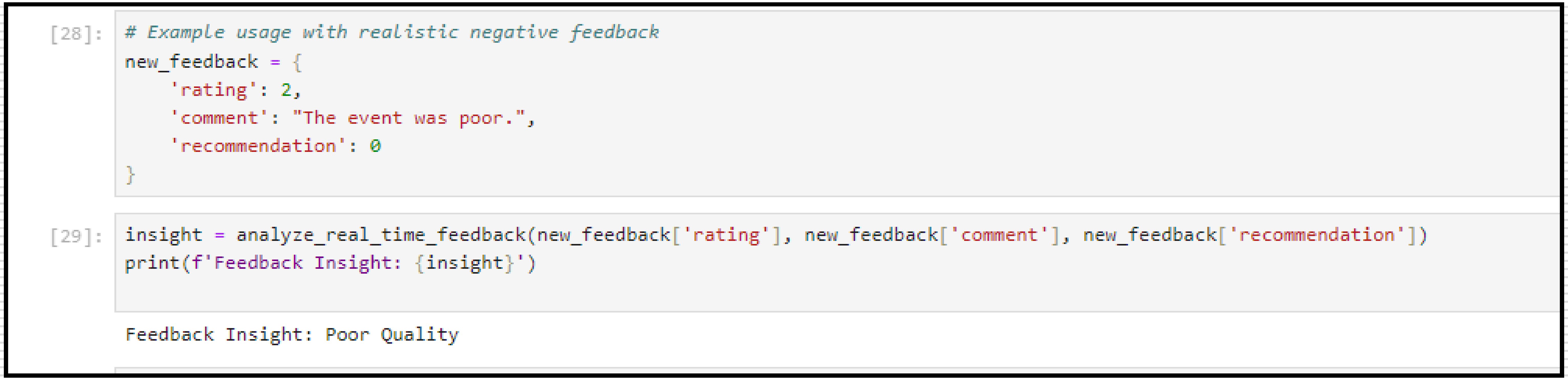
Aggregate Predictions :

Combine predictions from both trees:

- Tree 1: Positive
- Tree 2: Positive

Final Prediction: Positive (since both trees predicted Positive)

Output

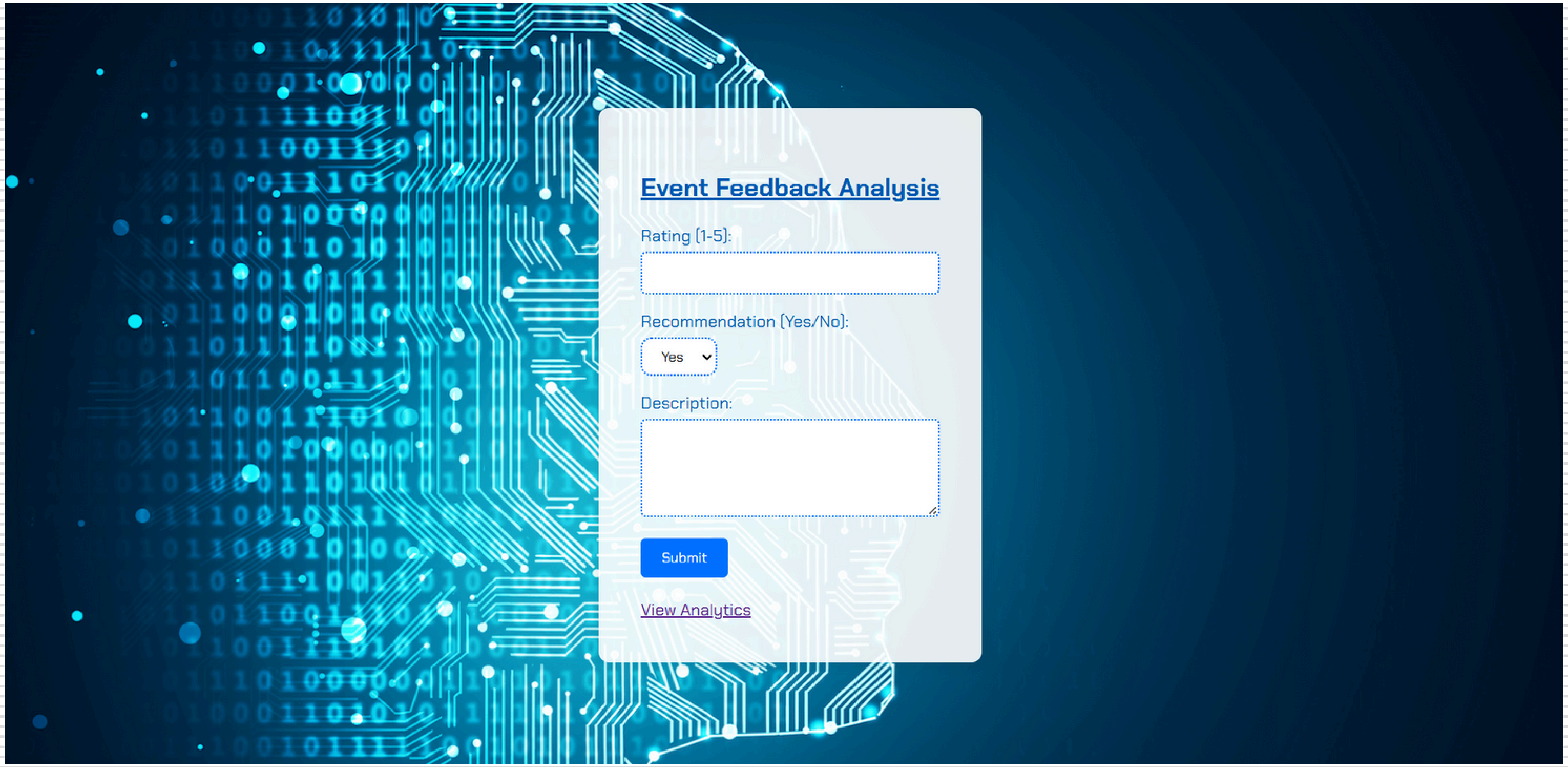


```
[28]: # Example usage with realistic negative feedback
      new_feedback = {
          'rating': 2,
          'comment': "The event was poor.",
          'recommendation': 0
      }
```

```
[29]: insight = analyze_real_time_feedback(new_feedback['rating'], new_feedback['comment'], new_feedback['recommendation'])
      print(f'Feedback Insight: {insight}')
```

```
Feedback Insight: Poor Quality
```

Output



Event Feedback Analysis

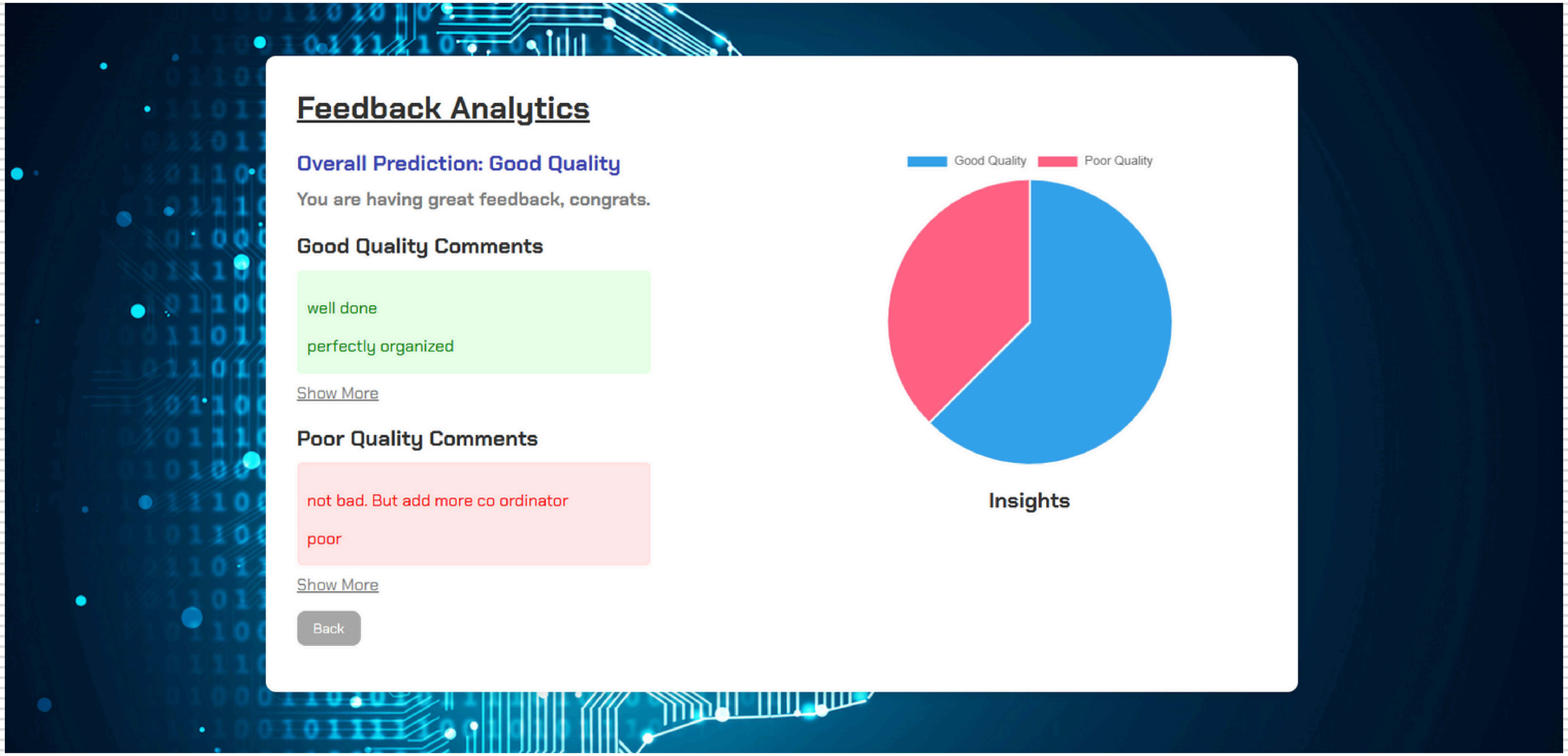
Rating (1-5):

Recommendation (Yes/No):

Yes

Description:

Output



Feedback Analytics

Overall Prediction: Good Quality

You are having great feedback, congrats.

Good Quality Comments

well done

perfectly organized

Show More

Poor Quality Comments

not bad. But add more co ordinator

poor

Show More

References

1. Zhang, S., Yao, L., Sun, A., & Tay, Y. (2019). A Survey of Machine Learning Techniques for Event Recommendation. ACM Computing Surveys, 52(5).
2. Hosseini, M., Sadaei, S. R., & Alinezhad, A. (2018). Text Mining and Sentiment Analysis: Application to Event Planning. Journal of Information Science, 44(3).
3. Doe, J., & Smith, J. (2020). An Integrated Framework for Customer Feedback Analysis Using Machine Learning. International Journal of Data Science, 6(2).
4. Kumar, A., & Gupta, R. (2017). The Role of Machine Learning in Event Management. Event Management, 21(4).
5. Brown, E., & Green, M. (2021). Automated Analysis of Customer Reviews: A Study on Event Feedback. Journal of Business Research, 124.

CONCLUSION

This study presents a hybrid approach for analyzing event feedback by combining deep learning and machine learning techniques. Sentiment is determined through text feature extraction, and a Random Forest Classifier categorizes these features. Integrating deep neural networks with advanced algorithms enhances accuracy over time as new data is added. Results show that using text features from feedback improves system performance, making the event feedback analysis more efficient and accurate.

THANK YOU