Sentiment Analysis Using NLP Techniques

1. Business Understanding

The goal of this project is to perform sentiment and emotion analysis on text data using Natural Language Processing (NLP) and machine learning.
We aim to automatically identify emotions like 'happy' or 'frustrated' from text such as reviews or feedback.

This task is widely applicable in:
- Understanding customer satisfaction
- Monitoring brand sentiment
- Automating emotion recognition in conversational AI systems

Project Objective:
- Perform text preprocessing and feature extraction
- Build and train machine learning models for sentiment classification
- Evaluate model performance and identify improvement areas

2. Data Understanding

Dataset size: ~25 samples (≈30KB)
Each record contains:
- Review text
- Emotion label (happy or frustrated)

Findings:
- Vocabulary size after preprocessing ≈ 1,404 unique words
- Slight imbalance in class distribution (more 'frustrated' than 'happy')
- No missing values detected

Observation: The dataset is small and meant for prototype demonstration rather than production-scale analysis.

3. Data Preparation

Preprocessing Steps:
- Converted all text to lowercase
- Removed punctuation, digits, and stopwords
- Tokenized and lemmatized text
- Added a 'cleaned_text' column to preserve original reviews

Feature Extraction:
- Used TF-IDF Vectorization → (25 samples, 1404 features)
- Captures frequency and importance of words across reviews

4. Modeling

Models Trained:
- Naive Bayes
- K-Nearest Neighbors (KNN)
- Decision Tree
- Random Forest
- Logistic Regression

Hyperparameter Tuning:
- Logistic Regression → C
- KNN → n_neighbors
- Random Forest → n_estimators, max_depth

Best Parameters and Scores:
Model	Best Params	Train Acc	Test Acc
Logistic Regression	C=0.01	100%	40%
KNN	n_neighbors=1	100%	40%
Random Forest	n_estimators=50, max_depth=None	100%	40%

Observation: Test accuracy remains 40% across models → data too small for stable learning.

5. Evaluation

Model Accuracy: 0.40 → 2 of 5 test samples correctly classified.

Confusion Matrix:
Actual ↓ vs Predicted →	Frustrated	Happy
Frustrated	0	3
Happy	0	2

Classification Report:
Emotion	Precision	Recall	F1-Score	Support
Frustrated	0.00	0.00	0.00	3
Happy	0.40	1.00	0.57	2
Accuracy			0.40	5

Interpretation:
- The model performs poorly due to class imbalance.
- Predicts 'happy' more often, ignoring 'frustrated'.
- Precision and Recall indicate overfitting on the majority class.

6. Observations

Training vs Test Comparison:

Metric	Training	Testing
Accuracy	0.80	0.40
Precision	0.84	0.16
Recall	0.80	0.40
F1	0.80	0.23

Interpretation:
- Overfitting evident: perfect training performance, poor test generalization.
- Causes: very small dataset, imbalance, sparse TF-IDF features, limited semantics from embeddings.

7. Deployment

While not production-ready, the project includes all steps for deployment.
Future deployment plan:
1. Save the model (.pkl or .joblib)
2. Create a REST API for predictions
3. Build a web interface for user input

8. Results & Conclusion

Summary:
- Dataset: 25 samples, 2 emotion classes
- Features: 1,404 TF-IDF terms
- Best Model: Logistic Regression
- Accuracy: 40%
- Overfitting due to data size and imbalance

Conclusion:
Project demonstrates an end-to-end sentiment analysis workflow using CRISP–DM methodology.
Despite low accuracy, it validates the pipeline structure, preprocessing, and evaluation process.

9. Recommendations

- Collect more data (≥ 1000 samples)
- Apply class balancing (SMOTE, oversampling)
- Use cross-validation for stable metrics
- Try BERT, LSTM, or Word2Vec for better semantic understanding
- Perform hyperparameter tuning with more folds