SPEECH RECOGNITION & EMOTIONAL DETECTION

20 February 2025

ABOUT THE PROJECT

This project aims to analyze speech and text data to classify sentiments into positive, negative, or neutral emotions. It integrates Natural Language Processing (NLP) and Machine Learning (ML) techniques for text-based sentiment analysis.

OBJECTIVES

- Develop a sentiment analysis system using text data from YouTube comments.
- Explore possible integration with speech recognition .
- Compare multiple machine learning models for sentiment classification.
- Visualize sentiment trends and provide insights.

PROJECT TIMELINE & MILESTONES:

Phase	Start Date	End Date
Data Collection & Cleaning	10/11/2024	30/11/2024
Model Development	5/12/2024	28/2/2025
Testing & Evaluation	1/3/2025	10/3/2025
Documentation & Final Report	15/3/2025	20/3/2025

OUR BEST TEAM

AHMED WAHBA

MOHAMED EBRAHIM

MANAR EL-BELTAGY

NOHA SALAH

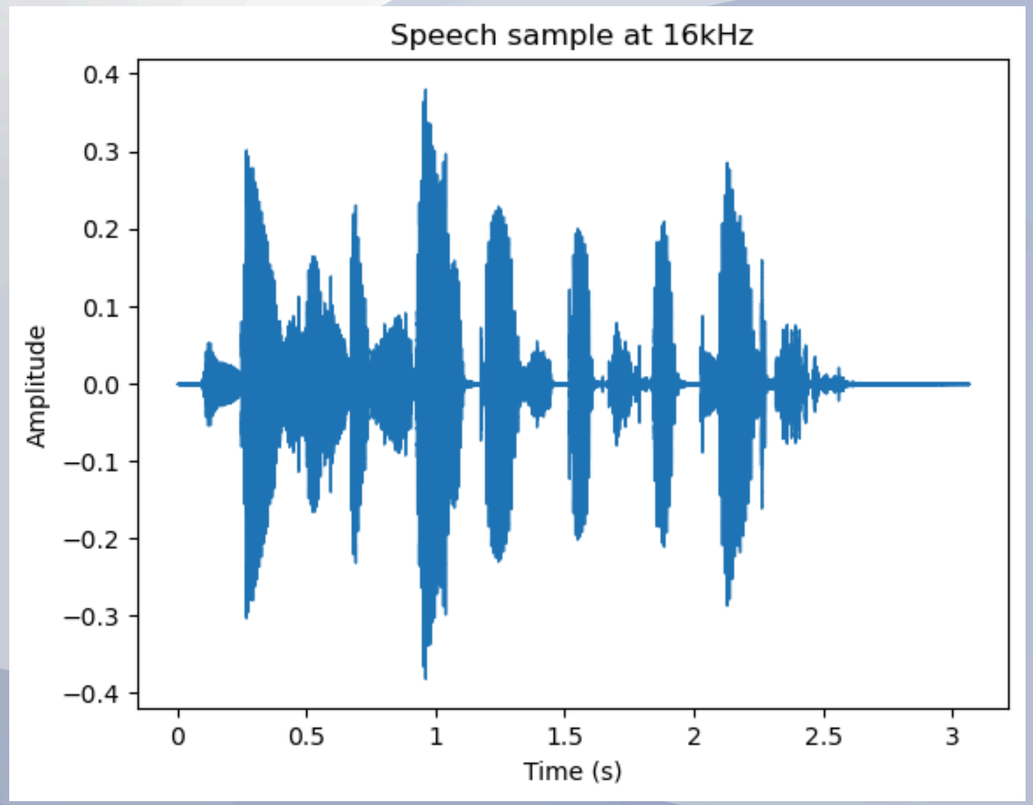
2. LITERATURE REVIEW

- Natural Language Processing (NLP): The role of NLP in sentiment analysis and emotion detection.
- Machine Learning for Sentiment Analysis: Comparison of classifiers.
- Speech Recognition: Methods used for speech-to-text conversion and sentiment extraction (if applicable).

3. REQUIREMENTS GATHERING

Functional Requirements:

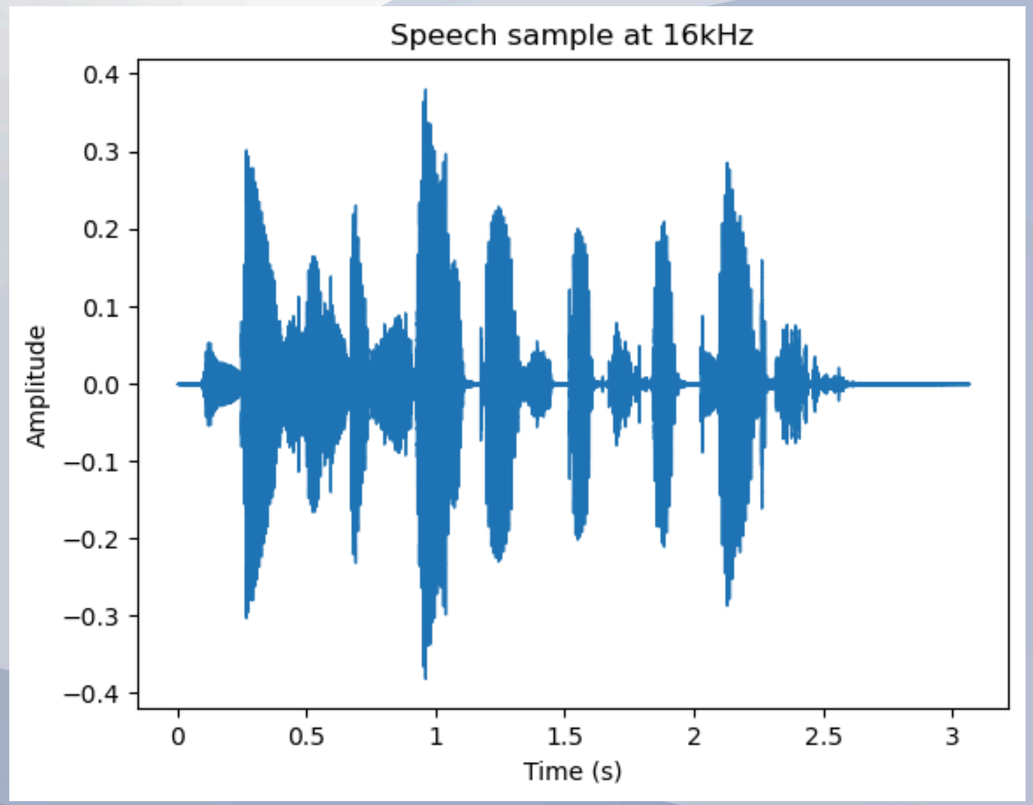
- System should classify text into positive, negative, or neutral categories.
- Support dataset processing (YouTube comments dataset used).
- Train and evaluate ML models on labeled sentiment data.
- Provide visual insights (word clouds, sentiment trends).



3. REQUIREMENTS GATHERING

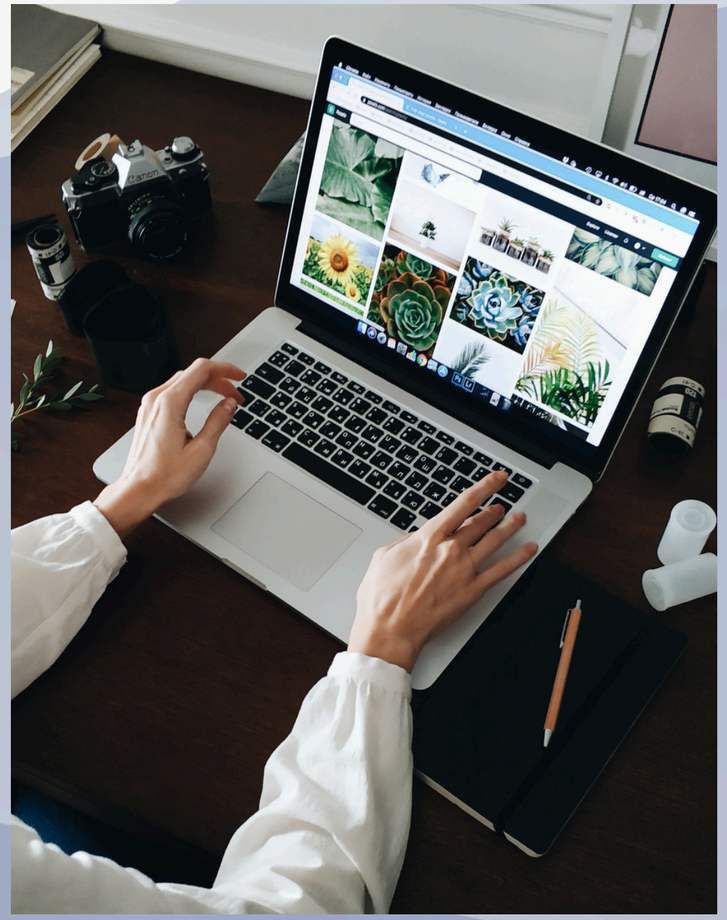
Non- Requirements:

- Efficient execution and scalability.
- High accuracy in sentiment prediction.



SYSTEM ANALYSIS & DESIGN DATA FLOW & PROCESSING STEPS:

1. Load YouTube comments dataset.
2. Apply preprocessing (tokenization, stopword removal, lemmatization).
3. Extract features using TF-IDF vectorization.
4. Train different ML models (Naïve Bayes, SVM, Random Forest).
5. Evaluate model performance using accuracy, F1-score, precision, recall.
6. Visualize results (word clouds, sentiment distributions).



5. Implementation (Source Code & Execution)

Technologies Used:

- Programming Language: Python
- Libraries: NLTK, Scikit-learn, Matplotlib, Seaborn, Gensim, WordCloud
- Data Processing: Pandas, NumPy

Execution Steps:

1. Run the Jupyter Notebook (text-sentiment-analysis-with-speech-recognition.ipynb).
2. Load and preprocess the dataset (comments.csv).
3. Train multiple ML models and compare results.
4. Evaluate and visualize sentiment predictions.

6. Testing & Quality Assurance

- Train/Test Split: 80% training, 20% testing.
- Evaluation Metrics: Accuracy, Precision, Recall, F1-score.
- Cross-validation applied to improve generalization.

7. Final Presentation & Reports

- Presentation of Results: Graphs, word clouds, and sentiment trends.
- Project Report: Summarizing implementation, results, and future improvements.
- Future Work: Possible integration of speech-to-text for voice-based sentiment analysis.

THANK YOU!