YouTube Sentiment Insights

An End-to-End NLP & MLOps-Based Sentiment Analysis System

---

1. Introduction

With the rapid growth of user-generated content on platforms like YouTube, understanding public opinion at scale has become increasingly important for creators, brands, and organizations. Millions of comments are posted daily, making manual analysis infeasible.

This project, **YouTube Sentiment Insights**, aims to build an **end-to-end sentiment analysis system** that automatically classifies YouTube comments into sentiment categories using Natural Language Processing (NLP) and Machine Learning techniques. Beyond model building, the project emphasizes **production readiness**, incorporating MLOps tools, a REST API, and a Chrome extension for real-world usability.

---

2. Problem Statement

Given a large volume of YouTube comments in textual form:

- Automatically determine the **sentiment polarity** of each comment
- Ensure the solution is **scalable, reproducible, and deployable**
- Provide predictions through a **user-facing interface**

---

3. Objectives

The key objectives of the project are:

1. Perform **data preprocessing and exploratory analysis** on YouTube comments
2. Experiment with multiple NLP feature extraction techniques
3. Train and evaluate different machine learning models
4. Track experiments and models using **MLflow**
5. Build a reproducible pipeline using **DVC**
6. Deploy the best-performing model using a **Flask REST API**
7. Integrate the model with a **Chrome Extension frontend**

---

4. Dataset Description

- **Source**: Publicly available YouTube comment sentiment dataset
- **Features**:
    - `comment_text`: Raw YouTube comment
    - `sentiment`: Target label (Positive / Negative / Neutral)

Data Challenges Identified

- Noisy text (URLs, emojis, special characters)
- Informal language and slang
- Class imbalance in sentiment labels

---

5. Exploratory Data Analysis (EDA)

EDA was conducted using Jupyter notebooks to understand:

- Sentiment class distribution
- Comment length distribution
- Frequently occurring words
- Presence of noise such as emojis and symbols

Key Insight:
The dataset showed a noticeable **class imbalance**, which influenced model selection and evaluation metrics.

---

6. Data Preprocessing Pipeline

A dedicated preprocessing module was implemented to ensure consistency across experiments.

Steps Included:

- Lowercasing text
- Removal of URLs, HTML tags, and special characters
- Stopword removal
- Whitespace normalization

Preprocessing logic was centralized in `data_preprocessing.py` to allow reuse across notebooks, training scripts, and deployment.

---

7. Feature Engineering

Multiple text vectorization techniques were explored:

1. Bag of Words (BoW)

- Simple frequency-based representation
- Served as a baseline

2. TF-IDF (Term Frequency–Inverse Document Frequency)

- Captures word importance across documents
- Performed significantly better than BoW

Enhancements:

- N-gram tuning
- Vocabulary size optimization
- Sparsity control

---

8. Model Development

Several machine learning models were trained and evaluated:

Models Explored:

- Logistic Regression (baseline)
- Tree-based models
- XGBoost
- LightGBM
- Stacking Ensemble (final experiment)

Model Selection Criteria:

- F1-score (primary metric due to class imbalance)
- Precision–Recall tradeoff
- Generalization on unseen data

Final Model Chosen:
TF-IDF + LightGBM, based on consistent performance and inference efficiency.

---

9. Model Evaluation

The evaluation module computed:

- Accuracy
- Precision
- Recall
- F1-score
- Confusion Matrix

The final model demonstrated strong generalization performance and handled class imbalance better than baseline models.

---

10. Experiment Tracking with MLflow

To ensure reproducibility and systematic experimentation:

- All parameters were logged
- Metrics were tracked per experiment
- Models were versioned using **MLflow Model Registry**

This allowed easy comparison between models and safe promotion of the best-performing model to production.

---

11. Data & Pipeline Versioning with DVC

DVC (Data Version Control) was used to:

- Track datasets
- Define pipeline stages (ingestion → preprocessing → training)
- Ensure reproducible experiments across environments

Pipeline configurations were managed using `dvc.yaml` and `params.yaml`.

---

12. Deployment Architecture

Backend: Flask API

- Loads trained TF-IDF vectorizer and model artifacts

- Exposes a `/predict` endpoint
- Accepts text input and returns sentiment predictions in JSON format

Frontend: Chrome Extension

- User-friendly popup interface
- Sends comment text to the Flask API
- Displays predicted sentiment instantly

This transforms the project from a research prototype into a **usable product**.

---

13. Testing & Validation

Basic testing scripts were included to:

- Validate model loading
- Ensure API stability
- Check inference consistency

---

14. Tools & Technologies Used

- **Programming Language**: Python
- **NLP**: Scikit-learn, TF-IDF
- **ML Models**: LightGBM, XGBoost
- **MLOps**: MLflow, DVC
- **Backend**: Flask
- **Frontend**: JavaScript (Chrome Extension)
- **Version Control**: Git

---

15. Results & Key Outcomes

- Built a complete NLP pipeline from raw data to deployment
- Achieved strong sentiment classification performance
- Implemented experiment tracking and model versioning
- Delivered a real-world usable sentiment analysis tool

---

16. Limitations

- Model is trained on English comments only

- Sarcasm and context-dependent sentiment remain challenging
- Real-time large-scale deployment would require additional infrastructure

---

17. Future Enhancements

- Transformer-based models (BERT / DistilBERT)
- Dockerization for easier deployment
- Batch inference support
- Dashboard for sentiment analytics
- Multilingual sentiment analysis

---

18. Conclusion

This project demonstrates a strong understanding of **NLP, machine learning, and MLOps principles**. By combining rigorous experimentation with real-world deployment, **YouTube Sentiment Insights** goes beyond a traditional academic project and showcases industry-relevant skills suitable for Machine Learning, Data Science, and Applied AI roles.
