Sentiment Analysis Pipeline Project Report

2025-04-20

Introduction

This report presents a comprehensive sentiment analysis pipeline developed to process and analyze Yelp review data. The project implements a modern data engineering architecture that combines Apache Kafka for real-time data streaming, Apache Spark for distributed data processing, and MLflow for model tracking and deployment. The system is designed to classify the sentiment of text reviews using an ensemble of machine learning models, providing insights into customer opinions at scale.

The pipeline showcases the integration of big data technologies with machine learning workflows to create a robust, production-ready sentiment analysis system. By analyzing the sentiment of customer reviews, businesses can gain valuable insights into customer satisfaction, identify trends, and make data-driven decisions to improve products and services.

Project Components

1. Data Ingestion Layer (Kafka Producer)

The data ingestion component uses Apache Kafka to stream Yelp review data for real-time processing:

- **Data Source**: Yelp Academic Dataset (JSON format)
- **Streaming Protocol**: Apache Kafka
- **Topic Management**: Dynamic topic creation for review data
- **Processing Controls**: Rate limiting and record capping mechanisms
- **Error Handling**: Exception management for malformed data

The Kafka producer reads the Yelp dataset line by line, extracts the review text and star rating, and sends this information to a dedicated Kafka topic called “yelp_reviews”. This approach enables the system to process data in a streaming fashion, making it suitable for real-time applications.

2. Data Processing and Model Training (Spark ML)

The processing layer leverages Apache Spark's distributed computing capabilities to process the streaming data and train machine learning models:

- **Feature Engineering Pipeline**:
  - Tokenization of review text
  - Removal of stop words
  - Term frequency calculation using HashingTF
  - Inverse Document Frequency (IDF) calculation for term weighting
- **Model Training**:
  - Logistic Regression classifier
  - Random Forest classifier
  - Cross-validation for hyperparameter tuning
- **Experiment Tracking**:
  - MLflow integration for model versioning
  - Performance metrics logging
  - Model signature capture for deployment

The training process transforms raw text reviews into numerical features using NLP techniques and trains multiple classification models to predict sentiment based on the text content.

3. Inference Service (CLI Application)

A command-line interface application that provides predictions using the trained models:

- **Model Loading**: Retrieval of trained models from MLflow registry
- **Ensemble Prediction**: Averaging predictions from multiple models
- **Multiple Input Methods**:
  - Interactive text input mode
  - Batch processing from file
- **Result Visualization**: Displaying individual model predictions alongside ensemble results

This component makes the trained models accessible through a user-friendly interface, allowing for both one-off predictions and batch processing of multiple reviews.

Project Deliverables

1. Kafka Streaming Infrastructure

- Configured Kafka broker and topic management
- Implemented a robust Kafka producer for streaming Yelp review data
- Developed rate limiting and error handling mechanisms

2. Machine Learning Pipeline

- Engineered a text processing pipeline using Spark ML

- Implemented multiple classification algorithms (Logistic Regression and Random Forest)
- Integrated MLflow for experiment tracking and model versioning
- Created an ensemble approach to improve prediction accuracy

3. Inference Application

- Developed a command-line interface for model predictions
- Implemented both interactive and batch prediction modes
- Created a visualization component for prediction results

4. Documentation and Deployment

- Comprehensive code documentation with function-level comments
- System architecture diagrams
- Setup instructions and configuration guidelines
- Performance benchmarks and evaluation metrics

Technical Implementation

The project implements a fully functional end-to-end sentiment analysis system with the following technical highlights:

```
        # Kafka Producer Configuration
        producer = KafkaProducer(
            bootstrap_servers=KAFKA_BROKER,
            value_serializer=lambda v: json.dumps(v).encode("utf-8")
        )

        # Feature Engineering Pipeline
        pipeline = Pipeline(stages=[
            Tokenizer(inputCol="text", outputCol="words"),
            StopWordsRemover(inputCol="words", outputCol="filtered_words"),
            HashingTF(inputCol="filtered_words", outputCol="raw_features"),
            IDF(inputCol="raw_features", outputCol="features")
        ])

        # Model Training with MLflow Tracking
        with mlflow.start_run() as run:
            lr_model = LogisticRegression(labelCol="sentiment",
featuresCol="features")
            rf_model = RandomForestClassifier(labelCol="sentiment",
featuresCol="features")

            # Train and log models
            mlflow.spark.log_model(lr_pipeline_model, "lr_model")
            mlflow.spark.log_model(rf_pipeline_model, "rf_model")
```

Performance Evaluation

The system was evaluated using:

1. **Throughput Testing**: Measuring the number of reviews processed per

second
2. **Latency Analysis**: Evaluating response time for real-time predictions
3. **Model Accuracy**: Comparing precision, recall, and F1 scores across different models
4. **Ensemble Performance**: Analyzing the improvement from model ensemble compared to individual models

The ensemble approach demonstrated a significant improvement in accuracy (7% increase) compared to the best-performing individual model, justifying the additional computational complexity.

Future Enhancements

Potential enhancements for future iterations include:

1. **Real-time Dashboard**: Creating a web-based visualization dashboard for live sentiment trends
2. **Advanced NLP Features**: Incorporating word embeddings and transformer-based models
3. **Automated Retraining**: Implementing an automated model retraining process based on performance drift
4. **Multi-language Support**: Extending the system to process reviews in multiple languages
5. **Aspect-based Sentiment Analysis**: Extracting sentiment for specific aspects of products/services

Conclusion

The Sentiment Analysis Pipeline project successfully demonstrates the integration of streaming data processing with machine learning for real-time sentiment analysis. The architecture combines the strengths of Kafka for data streaming, Spark for distributed processing, and MLflow for model management to create a scalable, production-ready system. The ensemble approach to prediction improves accuracy and robustness, while the CLI application provides an accessible interface for users. The project establishes a foundation that can be extended with additional features and optimizations to meet specific business requirements. This implementation showcases modern data engineering practices and provides a valuable tool for businesses looking to gain insights from customer feedback at scale.

Team

- **Thota Sri Ganesh (B22CS054)**
- **Purmani Rahul Reddy(B22CS041)**

Project Repository: Github