REDDIT SENTIMENT TRACKER

MINI PROJECT REPORT

Submitted
by

Bharath Kumar S (9517202209008)
Jai Chandran S (9517202209021)
Nantha Gopal R (9517202209035)

in

19AD751 – BIG DATA ANALYTICS LABORATORY

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

MEPCO SCHLENK ENGINEERING COLLEGE

SIVAKASI

NOVEMBER 2025

MEPCO SCHLENK ENGINEERING COLLEGE, SIVAKASI

AUTONOMOUS

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

BONAFIDE CERTIFICATE

This is to certify that it is the bonafide work of **"BHARATH KUMAR S (202209008), JAICHANDRAN (202209021), NANTHA GOPAL R (202209035)"** for the mini project titled **"REDDIT SENTIMENT TRACKER"** in 19AD751 –Big Data Analytics Laboratory during the seventh semester July 2025 – November 2025 under my supervision.

Signature

Dr.S.Shiny,
M.E., Ph.D., Assistant Professor,
AI&DS Department,
Mepco Schlenk Engg., College,
Sivakasi - 626 005.

Signature

Dr. J. Angela Jennifa Sujana,
M.TECH., Ph.D., Professor & Head,
AI&DS Department,
Mepco Schlenk Engg., College,
Sivakasi - 626 005.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENT

First and foremost we **praise and thank "The Almighty",** the lord of all creations, who by his abundant grace has sustained us and helped us to work on this project successfully.

We really find unique pleasure and immense gratitude in thanking our respected management members**,** who is the backbone of our college.

A deep bouquet of thanks to respected Principal **Dr.S.Arivazhagan M.E.,Ph.D.,** for having provided the facilities required for our mini project.

We sincerely thank our Head of the Department **Dr. J. Angela Jennifa Sujana M.TECH., Ph.D.,** Professor & Head, Department of Artificial Intelligence and Data Science, for her guidance and support throughout the mini project.

We extremely thank our project coordinators **, Dr.S.Shiny M.E., Ph.D** Assistant Professor and **Dr.R.Nirmalan M.E., Ph.D** Assistant Professor**,** Department of Artificial Intelligence and Data Science, who inspired us and supported us throughout the mini project.

We extend our heartfelt thanks and profound gratitude to all the faculty members of Artificial Intelligence and Data Science department for their kind help during our mini project work.

We also thank our parents and our friends who had been providing us with constant support during the course of the mini project work.

SUSTAINABLITIY DEVELOPMENT GOALS

Goal Name	Logo	Justification
SDG 9: Industry, Innovation and Infrastructure		The project leverages AI and NLP technologies to analyze online discussions, demonstrating innovation in digital infrastructure and data-driven systems.
SDG 10: Reduced Inequalities		By tracking sentiments across diverse topics and communities, the system helps identify online biases, hate speech, or marginalization patterns, promoting inclusivity.
SDG 11: Sustainable Cities and Communities		Analyzing public opinions aids policymakers and organizations in understanding community sentiment, leading to more responsive and sustainable social platforms.

SDG 16: Peace, Justice and Strong Institutions		The system detects toxic or harmful content trends, supporting efforts toward peaceful online discourse and transparent digital governance.
SDG 17: Partnerships for the Goals		This project can collaborate with NGOs, researchers, and governments to monitor digital sentiment for social good and global awareness.
SDG 4: Quality Education		The sentiment tracker serves as a valuable educational tool for learning about data science, AI ethics, and responsible technology use.

ABSTRACT

In the era of digital communication, social media platforms like Reddit generate vast amounts of user-generated content that reflect public opinion and emerging trends. Analyzing this data in real time provides valuable insights into the collective sentiment of users across different topics. The *Reddit Sentiment Tracker* project aims to build an automated pipeline that continuously monitors and analyzes posts from multiple Reddit communities to determine their emotional tone using advanced data processing and machine learning techniques.

The system architecture integrates several key technologies to achieve scalable and efficient data handling. Reddit data is collected through the Reddit API using PRAW, streamed via Apache Kafka, and processed in real time with Apache Spark's MLlib. The sentiment analysis pipeline classifies posts into categories such as positive, negative, or neutral based on their textual content. The processed results are stored in a MySQL database, enabling further trend analysis, topic-based comparisons, and visualization of sentiment distribution across different subreddits.

This project demonstrates how distributed data streaming and machine learning can be combined to create an intelligent social media analytics system. Beyond Reddit, the framework can be extended to other platforms such as Twitter or YouTube for broader sentiment monitoring. By providing real-time insights into public opinion, the system can assist researchers, organizations, and policymakers in understanding online discourse dynamics and identifying emerging trends within social communities.

CHAPTER 1
INTRODUCTON

1.1 Project Description

The Reddit Sentiment Tracker is a real-time data analytics system designed to collect, process, and analyze public opinions expressed across multiple Reddit communities. The project focuses on understanding user sentiment by continuously fetching posts from subreddits such as technology, science, worldnews, sports, movies, gaming, and music. Using the Reddit API, the system extracts key post attributes including titles, scores, comment counts, and timestamps, which are then enriched through sentiment analysis to determine the emotional tone of each post.

The architecture of the system is built around a distributed and scalable data pipeline. Data from Reddit is first published to Apache Kafka, which serves as a reliable and fault-tolerant message broker. The consumer component retrieves these messages, applies a sentiment classification model built with PySpark MLlib, and stores the results in a MySQL database. This modular design ensures real-time processing, robustness against system failures, and flexibility for integrating additional analytics components such as topic modeling or trend forecasting in the future.

The Reddit Sentiment Tracker provides valuable insights into public opinion and emerging online trends by aggregating and analyzing user-generated content in real time. It can be extended for various applications, such as media monitoring, social research, or brand reputation analysis. By combining technologies like Kafka, Spark, and machine learning, the project highlights the power of big data ecosystems in transforming unstructured social media data into actionable intelligence.

1.2 Project Objectives

The main objective of the Reddit Sentiment Tracker project is to design and implement an automated system that continuously collects and analyzes Reddit posts to identify public sentiment and emerging trends in real time. The project aims to integrate data engineering, machine learning, and big data tools to process large-scale unstructured social media data efficiently.

Specific objectives include:

1. **Data Collection:** To automatically fetch the latest posts from multiple Reddit communities using the Reddit API and ensure seamless data flow through Apache Kafka.
2. **Real-time Data Streaming:** To implement a robust producer–consumer pipeline using Kafka for reliable message delivery and scalable data ingestion.
3. **Sentiment Analysis:** To classify the emotional tone of Reddit posts (positive, negative, or neutral) using machine learning models built with PySpark MLlib.
4. **Data Storage and Management:** To design a structured MySQL database for storing Reddit post information along with sentiment scores for easy retrieval and analysis.
5. **Visualization and Insights:** To enable interactive dashboards using Streamlit for visualizing sentiment trends, subreddit comparisons, and engagement metrics.
6. **Scalability and Extensibility:** To build an architecture that can be easily extended to include additional analytics such as topic modeling, trend prediction, or user engagement forecasting.

Through these objectives, the project demonstrates how real-time data engineering and sentiment analytics can be integrated to provide actionable insights from large-scale social media platforms like Reddit.

1.3 Scope Of The Project

The scope of the Reddit Sentiment Tracker project is to collect live Reddit posts from multiple subreddits, analyze their sentiments using PySpark MLlib, and store the processed data in a MySQL database for real-time visualization through a Streamlit dashboard. The project covers the entire pipeline — from data extraction and streaming using Kafka to machine learning–based sentiment analysis and interactive analytics — providing users with insights into trending topics, public opinions, and engagement patterns across Reddit communities.

CHAPTER 2
PROPOSED SYSTEM

2.1 Architecture Diagram

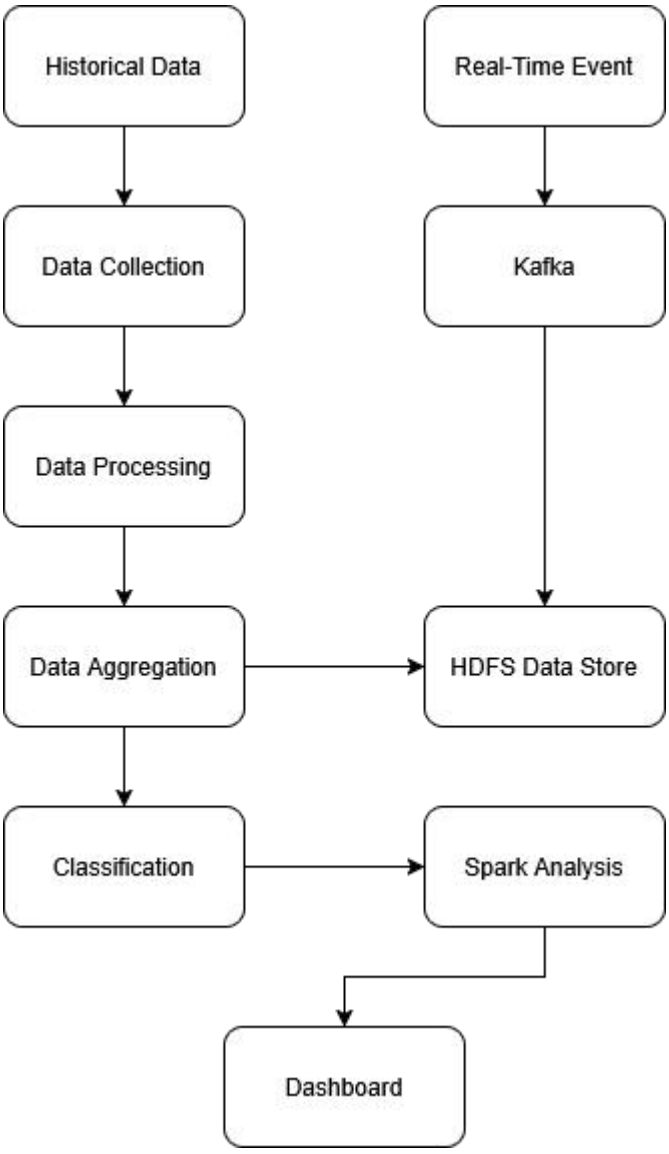

Fig 2.1 System Design

The architecture of the **Reddit Sentiment Tracker** project follows a **modular big data pipeline** that integrates data ingestion, processing, storage, and visualization layers:

1. **Data Ingestion Layer** – Reddit posts are fetched in real time using the Reddit API (via the producer.py script). The producer acts as a data source, continuously collecting posts from multiple subreddits such as *technology*, *science*, *worldnews*, *sports*, *movies*, *gaming*, and *music*. These posts are published to a **Kafka topic**, ensuring scalable and fault-tolerant message streaming.
2. **Data Processing Layer** – The **Kafka consumer** (consumer.py) listens to the topic and retrieves incoming posts. Each post is processed through the **PySpark MLlib sentiment analysis model** (sentiment_model.py in the analytics folder). This layer is responsible for performing machine learning–based sentiment classification (e.g., positive, negative, neutral) in real time before the data is stored.
3. **Data Storage & Visualization Layer** – The processed data (with sentiment, score, comments, and timestamps) is inserted into a **MySQL database**. A **Streamlit dashboard** then fetches the latest records periodically to display analytics such as sentiment trends, subreddit comparisons, and engagement metrics.

Overall, the architecture ensures an **end-to-end big data pipeline** that transforms unstructured Reddit data into **real-time, sentiment-driven insights** through automation, scalability, and interactive visualization.

2.2 Implementaton Steps

The implementation of the *Reddit Sentiment Tracker* project involves several well-defined stages, each ensuring smooth data flow, accurate processing, and meaningful sentiment analysis.

1. **Data Collection:**
   The system connects to the Reddit API to extract posts and comments based on selected topics or subreddits. This raw text data serves as the foundation for sentiment analysis.
2. **Data Preprocessing:**
   Collected data undergoes cleaning and transformation using PySpark. Steps include removing special characters, stop words, and irrelevant content to prepare the text for analysis.
3. **Sentiment Model Integration:**
   The preprocessed text is passed to a PySpark MLlib-based sentiment analysis model. This

model classifies sentiments as *positive*, *negative*, or *neutral* based on the linguistic patterns and tone of the content.

4. **Kafka Stream Processing:**

   The producer sends cleaned Reddit data to a Kafka topic, enabling real-time streaming. Consumers (like analytics modules) can subscribe to this topic to process and visualize data dynamically.

5. **Data Storage and Analytics:**

   Processed sentiment results are stored in a MySQL database for future insights. The analytics component generates summaries and trends for visualization or reporting.

6. **Result Visualization and Reporting:**

   Finally, the analyzed data is used to generate sentiment distribution reports and dashboards. These visual insights help in understanding public opinion trends on various topics.

2.3 ML Workflow Diagram

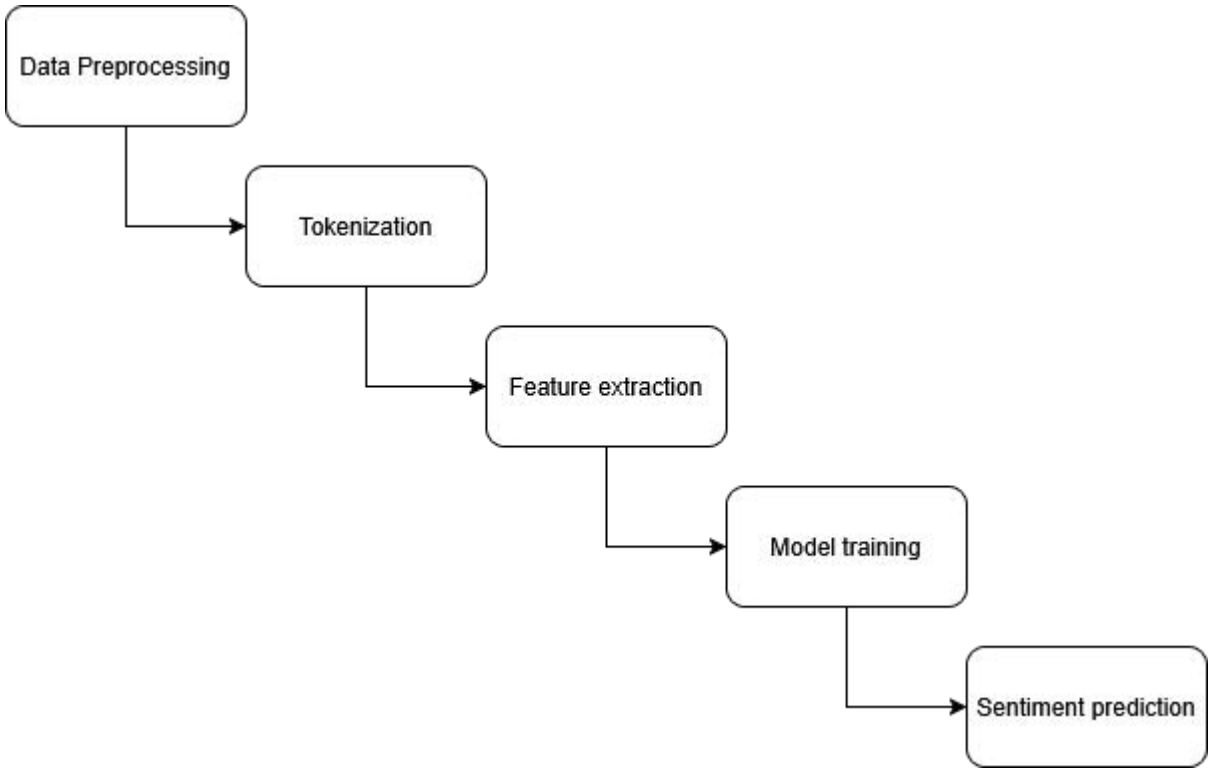

Fig 2.2 : ML Workflow Diagram

The **Machine Learning Workflow** in the Reddit Sentiment Tracker project focuses on transforming raw textual Reddit data into meaningful sentiment insights using **PySpark MLlib**. It follows a structured sequence of stages, ensuring that data is efficiently processed, modeled, and evaluated:

1. **Data Collection and Preprocessing** – Reddit post titles and metadata are first collected through the API and stored in the MySQL database. The text data (titles) is then extracted and preprocessed to remove noise — including punctuation, stopwords, and irrelevant characters. Tokenization and normalization (lowercasing, stemming/lemmatization) are applied to prepare the text for analysis.
2. **Feature Extraction** – The cleaned text is transformed into numerical vectors using **TF-IDF (Term Frequency–Inverse Document Frequency)** or **Word2Vec** representations available in PySpark MLlib. This step converts unstructured text into structured input features that machine learning algorithms can interpret.
3. **Model Training and Evaluation** – The processed dataset is split into training and testing subsets. Various MLlib models like **Logistic Regression**, **Naive Bayes**, or **Random Forest** are trained to predict the sentiment of Reddit posts (Positive, Negative, Neutral). The trained model is then evaluated using metrics such as **accuracy, precision, recall, and F1-score** to ensure reliability and robustness.
4. **Prediction and Continuous Learning** – Once deployed, new incoming Reddit posts are passed through the trained model in real time via the Kafka consumer. The model predicts their sentiment before storing the results in the database. Over time, the model can be retrained with fresh data to improve accuracy and adapt to changing language trends.

This workflow enables a **fully automated sentiment analysis pipeline**, converting continuous Reddit data streams into **actionable emotional intelligence** for visualization and further analytics.

CHAPTER 3
SYSTEM REQUIREMENTS

3.1 HARDWARE REQUIREMENTS:

To effectively process large-scale climate datasets and perform complex analytics, the following hardware configurations are recommended to ensure optimal performance and smooth execution throughout the project:

1. **Processor**: Intel Core i5 or higher, with a **Quad-core or higher** configuration recommended. A multi-core processor is essential for parallel data processing, especially when working with big data tools like Apache Spark.

2. **RAM**: A minimum of **8 GB** is required, but **16 GB or higher** is recommended for handling large datasets efficiently. Ample memory is necessary for in-memory data processing, particularly in distributed computing environments.

3. **Storage**: **500 GB HDD** or **SSD**. While HDD is sufficient, an SSD is recommended for its faster read/write speeds, which will improve data loading, saving, and real-time processing tasks.

4. **Graphics Card**: Basic **integrated graphics** are sufficient for this project if advanced rendering is not required. However, if high-end data visualizations or machine learning models involving GPUs are required, a discrete GPU could enhance performance.

5. **Network**: A **high-speed internet connection** is crucial for API interactions (e.g., retrieving real-time weather data from AbusiveAPI) and cloud-based operations, such as data storage or distributed processing in a cloud environment.

3.2 SOFTWARE REQUIREMENTS

The development and deployment of the Reddit Sentiment Tracker system require a combination of programming languages, frameworks, libraries, and supporting tools to enable smooth data extraction, processing, sentiment classification, and visualization. This section outlines all the essential software components used in the project.

1. Operating System

The system is platform-independent and can run on multiple operating systems with minimal configuration.

However, for development and deployment purposes, the following operating systems are recommended:

1. **Windows 10 / 11 (64-bit):** Provides a stable environment with compatibility for Python, MySQL, and Kafka.
2. **Linux (Ubuntu 20.04 or later):** Preferred for server-side deployment due to its performance, scalability, and command-line flexibility.
3. **macOS (Monterey or later):** Can be used for development with minimal adjustments to dependencies.

2. Programming Language

1. **Python 3.8 or higher:**

The entire system is built using Python due to its simplicity, readability, and vast ecosystem of libraries for data analytics and machine learning. Python also integrates seamlessly with APIs and external data sources like Reddit, making it ideal for real-time data collection and analysis.

2. **PySpark:**

A Python API for Apache Spark used for distributed data processing and large-scale sentiment classification. PySpark allows handling large volumes of streaming data efficiently using its MLlib machine learning library.

3. Frameworks and Libraries

The project leverages several open-source frameworks and libraries to ensure scalability, efficiency, and reliability.

1. **PRAW (Python Reddit API Wrapper):**
   PRAW is used to connect with the Reddit API, allowing easy access to posts, comments, and metadata. It simplifies the process of authentication, subreddit exploration, and post retrieval without dealing with complex REST calls.
2. **Apache Kafka:**
   Kafka acts as the backbone for real-time data streaming in this project. It allows asynchronous communication between the producer (data collection module) and consumer (sentiment analytics module). Kafka ensures fault-tolerant, high-throughput, and low-latency message transfer.
3. **PySpark MLlib:**
   Used for performing large-scale machine learning tasks such as sentiment classification. It provides built-in algorithms for text preprocessing, feature extraction (TF-IDF, Word2Vec), and classification (Logistic Regression, Naive Bayes). MLlib ensures that the model can handle massive data volumes efficiently.
4. **dotenv:**
   Manages environment variables securely by storing credentials such as API keys, database connections, and Kafka configurations in a .env file. This ensures code security and flexibility across environments.
5. **pandas and numpy:**
   Essential for structured data handling and numerical computations. Pandas provides DataFrame support for tabular data manipulation, while NumPy supports efficient mathematical and statistical operations on large arrays.
6. **mysql-connector-python:**
   Used to connect Python applications with MySQL databases. It enables smooth read/write operations, transaction handling, and supports efficient storage of large datasets collected from Reddit.

7. **streamlit:**

   Powers the frontend visualization dashboard. It allows developers to build interactive and data-driven web interfaces directly in Python without HTML or JavaScript.

4. Database

MySQL Database:
MySQL serves as the centralized data storage component in the project. It stores metadata about Reddit posts, sentiment results, subreddit categories, and time-based statistics. MySQL is chosen for its robustness, reliability, and structured query support.

The schema is designed to handle continuous streaming data with efficient indexing and querying. Additionally, MySQL supports integration with visualization and reporting tools for analytics purposes.

5. Development Environment

1. **Apache Spark:**

   The Spark engine is used for distributed computation and machine learning via PySpark. It manages large-scale sentiment processing using resilient distributed datasets (RDDs) and supports real-time streaming analytics.

2. **Apache Kafka Server & Zookeeper:**

   Kafka and Zookeeper form the communication bridge between data producers and consumers. Kafka handles high-throughput data ingestion, while Zookeeper coordinates distributed Kafka clusters for synchronization and failover handling.

3. **Visual Studio Code :**

   Used as the primary development IDEs for writing, debugging, and testing Python scripts. These environments provide plugin support for virtual environments, linting, and code formatting.

4. **Git and GitHub:**

   Used for version control and collaborative development. All project code, documentation, and updates are maintained in a Git repository.

CHAPTER 4
IMPLEMENTATION

3.3 SOURCE CODE

producer.py

```
import praw
import mysql.connector
import time
from config import MYSQL_CONFIG, REDDIT_CONFIG
from analytics.sentiment_model import get_sentiment


SUBREDDITS = [
    "technology",
    "science",
    "worldnews",
    "sports",
    "movies",
    "gaming",
    "music"
]


def create_db_connection():
    while True:
        try:
            conn = mysql.connector.connect(**MYSQL_CONFIG)
            return conn
        except mysql.connector.Error as e:
```

```
            print(f"✗ Database connection failed, retrying in 5s: {e}")
            time.sleep(5)

db = create_db_connection()
cursor = db.cursor()

cursor.execute("""
CREATE TABLE IF NOT EXISTS reddit_posts (
    id VARCHAR(20) PRIMARY KEY,
    title TEXT,
    subreddit VARCHAR(50),
    score INT,
    num_comments INT,
    sentiment VARCHAR(20),
    created_utc DATETIME
)
""")
db.commit()

reddit = praw.Reddit(**REDDIT_CONFIG)

def insert_post(post):
    try:
        sentiment = get_sentiment(post.title)
    except Exception as e:
        print(f"   Sentiment analysis failed for post '{post.id}': {e}")
        sentiment = "Unknown"
```

```
    sql = """INSERT INTO reddit_posts (id, title, subreddit, score, num_comments, sentiment,
created_utc)
            VALUES (%s, %s, %s, %s, %s, %s, FROM_UNIXTIME(%s))
            ON DUPLICATE KEY UPDATE score=%s, num_comments=%s, sentiment=%s"""
    values = (
        post.id, post.title, post.subreddit.display_name, post.score,
        post.num_comments, sentiment, post.created_utc,
        post.score, post.num_comments, sentiment
    )

    try:
        cursor.execute(sql, values)
        db.commit()
        print(f"    Stored: {post.title[:60]} | Sentiment: {sentiment}")
    except mysql.connector.Error as e:
        print(f"❌ Error storing {post.id}: {e}")
        db.rollback()

for subreddit_name in SUBREDDITS:
    subreddit = reddit.subreddit(subreddit_name)
    print(f"    Fetching posts from {subreddit_name} ...")
    for post in subreddit.hot(limit=100):
        insert_post(post)
        time.sleep(2)

print("✅ Data fetch and store completed!")
cursor.close()
db.close()
```

consumer.py

```
from kafka import KafkaConsumer
import json
import mysql.connector
from config import KAFKA_TOPIC, KAFKA_SERVER, MYSQL_CONFIG
from analytics.sentiment_model import get_sentiment
import time
from reddit_to_mysql import add_to_mysql

def create_db_connection():
    """Create a persistent DB connection."""
    while True:
        try:
            conn = mysql.connector.connect(**MYSQL_CONFIG)
            return conn
        except mysql.connector.Error as e:
            print(f"❌ DB connection failed, retrying in 5s: {e}")
            time.sleep(5)

# Connect to MySQL
db = create_db_connection()
cursor = db.cursor()

# Ensure table exists
cursor.execute("""
```

```
CREATE TABLE IF NOT EXISTS reddit_posts (
    id VARCHAR(20) PRIMARY KEY,
    title TEXT,
    subreddit VARCHAR(50),
    score INT,
    num_comments INT,
    sentiment VARCHAR(20),
    created_utc DATETIME
)
""")
db.commit()

# Create KafkaConsumer once
consumer = KafkaConsumer(
    KAFKA_TOPIC,
    bootstrap_servers=[KAFKA_SERVER],
    value_deserializer=lambda m: json.loads(m.decode('utf-8')),
    auto_offset_reset='earliest',
    enable_auto_commit=True
)

print("✅ Listening to Kafka topic...")

# Continuous processing
while True:
    try:
        for message in consumer:
            post = message.value
```

```
            sentiment = get_sentiment(post["title"])

            sql = """INSERT INTO reddit_posts (id, title, subreddit, score, num_comments, sentiment,
created_utc)
                    VALUES (%s,%s,%s,%s,%s,%s,FROM_UNIXTIME(%s))
                    ON DUPLICATE KEY UPDATE score=%s, num_comments=%s, sentiment=%s"""
            values = (
                post["id"], post["title"], post["subreddit"], post["score"],
                post["num_comments"], sentiment, post["created_utc"],
                post["score"], post["num_comments"], sentiment
            )
            try:
                cursor.execute(sql, values)
                db.commit()
                print(f"    Stored {post['title'][:30]} | {sentiment}")
            except mysql.connector.Error as e:
                print(f"✗ Error storing {post['id']}: {e}")
                db.rollback()
        except Exception as e:
            print(f"   Consumer error: {e}, reconnecting in 5s...")
            time.sleep(5)
            db = create_db_connection()
            cursor = db.cursor()
```

sentiment_model.py

```
from pyspark.sql import SparkSession
from pyspark.ml.feature import Tokenizer, StopWordsRemover, HashingTF, IDF, StringIndexer
```

```
from pyspark.ml.classification import LogisticRegression
from pyspark.ml import Pipeline
from pyspark.sql.functions import col
import time

spark = SparkSession.builder \
    .appName("RedditSentimentAnalysis") \
    .config("spark.jars", "mysql-connector-j-8.0.33.jar") \
    .config("spark.sql.streaming.checkpointLocation", "./checkpoint") \
    .getOrCreate()

MYSQL_CONFIG = {
    "url": "jdbc:mysql://localhost:3306/redditdb",
    "properties": {
        "user": "root",
        "password": "root",
        "driver": "com.mysql.cj.jdbc.Driver"
    },
    "table": "reddit_posts"
}

def load_data():
    df = spark.read.jdbc(
        url=MYSQL_CONFIG["url"],
        table=MYSQL_CONFIG["table"],
        properties=MYSQL_CONFIG["properties"]
    )
    return df.select("id", "title", "sentiment")
```

```
def preprocess_data(df):
    df = df.na.drop(subset=["title", "sentiment"])
    label_indexer = StringIndexer(inputCol="sentiment", outputCol="label")
    tokenizer = Tokenizer(inputCol="title", outputCol="words")
    remover = StopWordsRemover(inputCol="words", outputCol="filtered")
    hashingTF = HashingTF(inputCol="filtered", outputCol="rawFeatures", numFeatures=10000)
    idf = IDF(inputCol="rawFeatures", outputCol="features")
    pipeline = Pipeline(stages=[label_indexer, tokenizer, remover, hashingTF, idf])
    processed_df = pipeline.fit(df).transform(df)
    return processed_df

def train_model(df):
    train_data, test_data = df.randomSplit([0.8, 0.2], seed=42)
    lr = LogisticRegression(maxIter=20, regParam=0.3, elasticNetParam=0.8)
    model = lr.fit(train_data)
    predictions = model.transform(test_data)
    correct = predictions.filter(predictions.label == predictions.prediction).count()
    total = predictions.count()
    accuracy = (correct / total) * 100 if total > 0 else 0
    print(f"Model Trained Successfully | Accuracy: {accuracy:.2f}%")
    return model

def stream_training(interval=60):
    latest_count = 0
    model = None
    while True:
        df = load_data()
```

```
        current_count = df.count()
        if current_count > latest_count:
            print(f"New data detected ({current_count - latest_count} new posts). Updating model...")
            processed_df = preprocess_data(df)
            model = train_model(processed_df)
            latest_count = current_count
        else:
            print("No new posts found. Waiting for next cycle...")
        time.sleep(interval)

def get_sentiment(text):
    global model
    if model is None:
        print("Model not trained yet. Returning 'Neutral'.")
        return "Neutral"
    new_df = spark.createDataFrame([(text,)], ["title"])
    processed_df = preprocess_data(new_df)
    prediction = model.transform(processed_df).collect()[0].prediction
    if prediction == 1.0:
        return "Positive"
    elif prediction == 0.0:
        return "Neutral"
    else:
        return "Negative"

if __name__ == "__main__":
    print("Starting Real-Time Sentiment Training using PySpark MLlib ...")
    stream_training(interval=120)
```

CHAPTER 5
RESULT

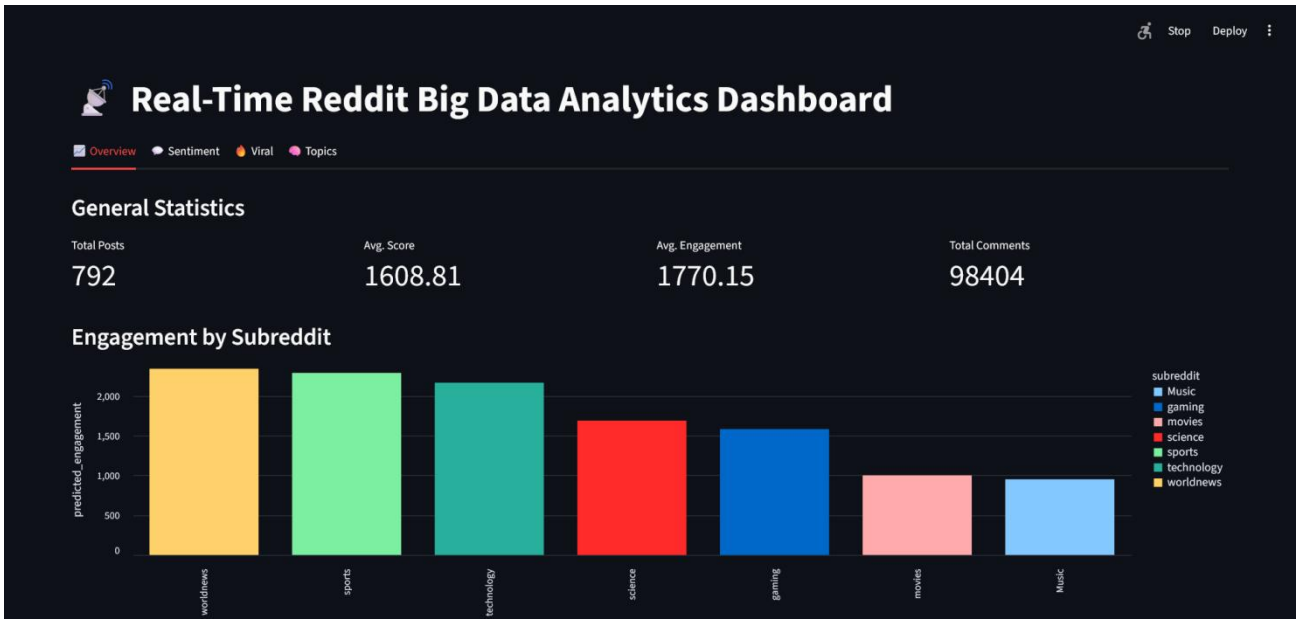

Fig 4.1 General Statistics

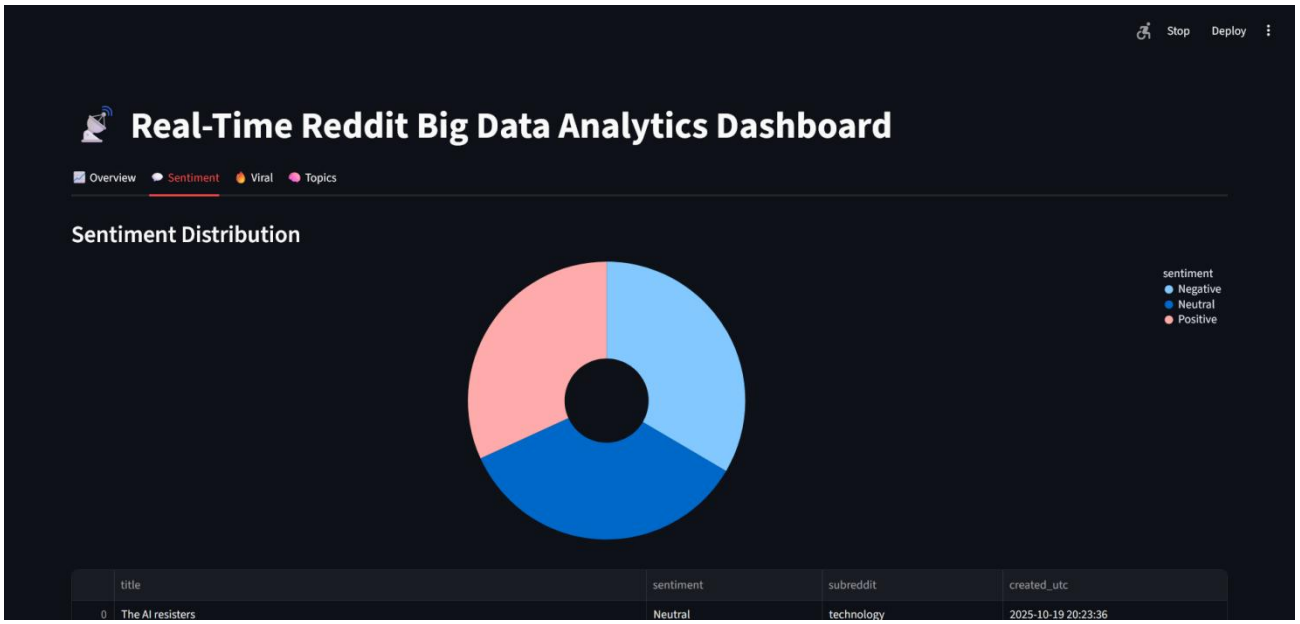

Fig 4.2 Sentiment Distribution

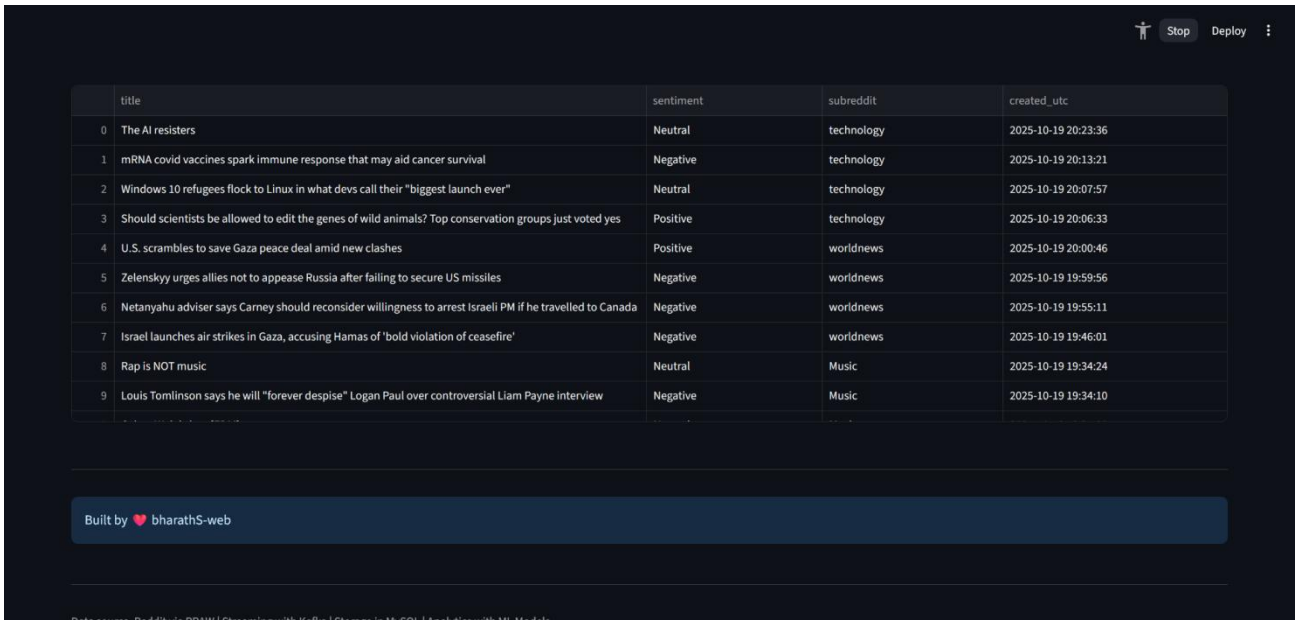

Fig 4.3 Real Time Data Summary

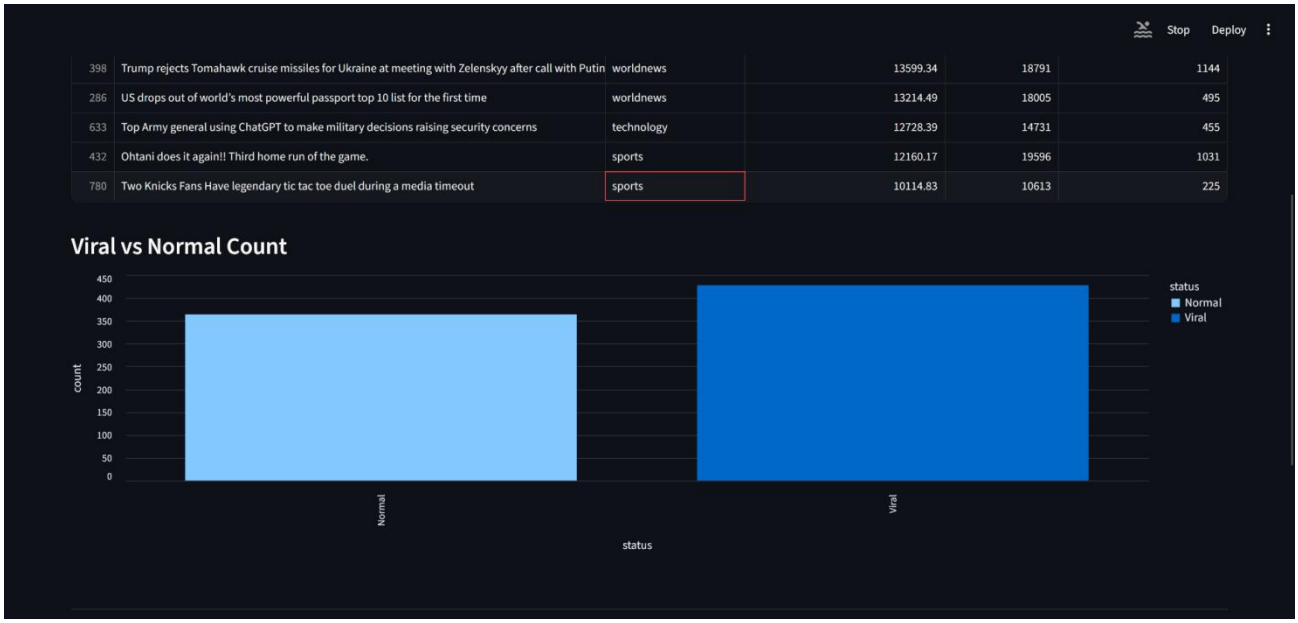

Fig 4.4 Subreddit Viral Statistics

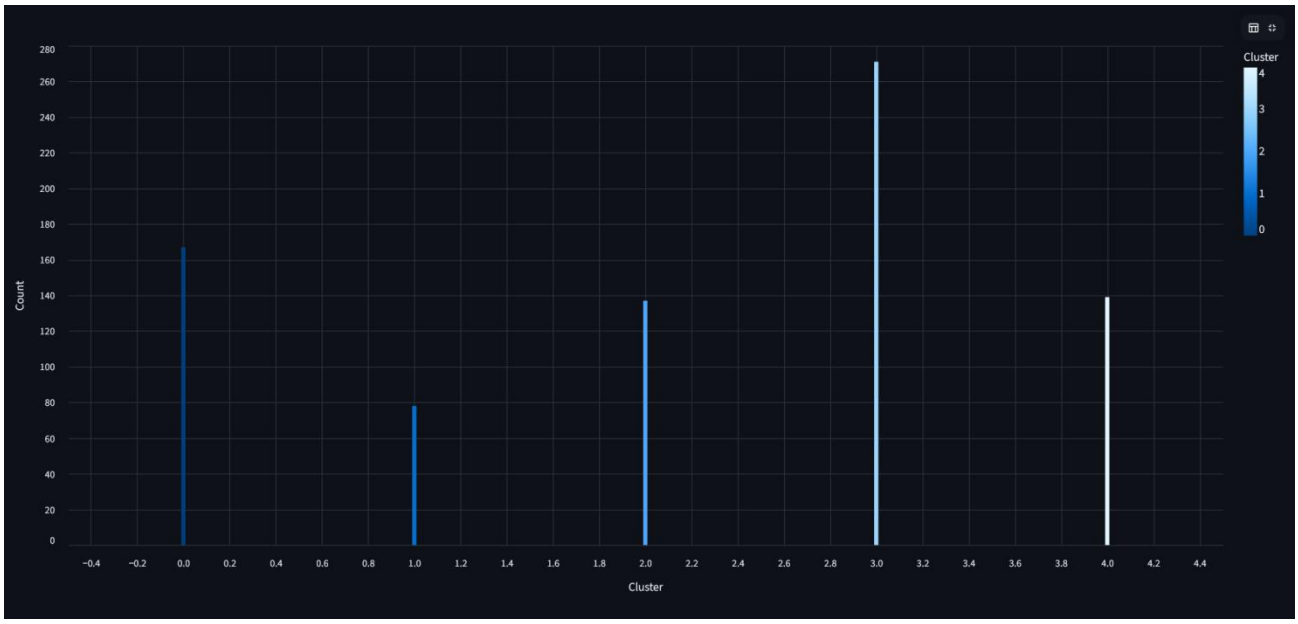

Fig 4.5 Trending Topic Statistics

CHAPTER 6
CONCLUSION

The Reddit Sentiment Tracker project successfully demonstrates the integration of big data streaming, natural language processing, and machine learning to perform real-time sentiment analysis on Reddit posts. By continuously fetching data from multiple Reddit communities, the system captures a wide range of public opinions across different topics such as technology, science, movies, and world news. This live data pipeline enables continuous insights into trending discussions and sentiment variations in a dynamic online environment.

The project leverages **Apache Kafka** for data streaming, **MySQL** for structured data storage, and **PySpark MLlib** for scalable sentiment model training. Through automated workflows, the system processes incoming Reddit posts, performs sentiment classification, and updates predictive models using streaming data. This combination of real-time analytics and continuous model refinement ensures that the analysis remains accurate and adaptive to new information and evolving trends.

Overall, the Reddit Sentiment Tracker provides a robust foundation for building intelligent social media monitoring tools. The project showcases the potential of integrating distributed systems with machine learning for real-world applications such as opinion mining, market research, and trend prediction. In future enhancements, the system could incorporate advanced deep learning models like BERT or LSTM for improved sentiment accuracy, and visualization dashboards for deeper analytical insights.

CHAPTER 7
REFRENCES

- Apache Software Foundation. (2024). Apache Kafka: A Distributed Streaming Platform.
- PRAW Documentation. (2024). Python Reddit API Wrapper (PRAW).
- The Apache Software Foundation. (2024). Apache Spark – Unified Analytics Engine for Big Data.
- MySQL Documentation. (2024). MySQL Reference Manual. Oracle Corporation.
- Python Software Foundation. (2024). Streamlit and PySpark MLlib Libraries for Machine Learning.
- Kumar, A., & Garg, N. (2022).
  "Real-Time Sentiment Analysis of Social Media Streams Using Apache Kafka and Spark Streaming."
  — Proposes a real-time sentiment analysis system using Kafka and Spark Streaming, similar to this project's data pipeline.
- Zhao, Y., Chen, L., & Xu, Y. (2023).
  "Mining Emotions from Reddit Data Using Deep Learning for Social Insights."
  — Focuses on extracting emotions and sentiments from Reddit posts using NLP and machine learning techniques.