

# **Emerald International College School of Computing and Analytics**

**Project title:** Real-Time Twitter Sentiment Analysis Using Kafka, Pyspark, MongoDB and Django.

|  |  |  |
| --- | --- | --- |
| **Name of Student** | **ID** | **Role** |
| Asmare Yenealem | DS0054/15 | Team leader |
| Amanuel Gebru | DS0050/15 | Supervisor |
| Wengel Zerihun | DS0082/15 | Member |
| Solomon Guesh | DS0075/15 | Member |
| Esubalew Dejene | DS0089/15 | Member |

***Submitted to Dr. Genet Shanko***

Contents

[**Emerald International College School of Computing and Analytics** 0](#_Toc175567469)

[1. Introduction 2](#_Toc175567470)

[2. Project Description 2](#_Toc175567471)

[3. Technologies Used 2](#_Toc175567472)

[3.1. PySpark 2](#_Toc175567473)

[3.2. Django 2](#_Toc175567474)

[3.3. Kafka 2](#_Toc175567475)

[3.4. Mongo Db 2](#_Toc175567476)

[4. Implementation 3](#_Toc175567477)

[4.1. Data Preprocessing with PySpark 3](#_Toc175567478)

[4.2. Model Training 3](#_Toc175567479)

[4.3. Interface Web Django 3](#_Toc175567480)

[4.4. Real-Time Streaming with Kafka 3](#_Toc175567481)

[4.5. Storage with MongoDB 4](#_Toc175567482)

[5. Analysis result 4](#_Toc175567483)

[6. Conclusion 4](#_Toc175567484)

# Introduction

Sentiment analysis is an important field of machine learning that aims to determine the sentiments expressed in a text, often opinions or reactions on social media. In this report, we present a project that combines machine learning, web technologies, and streaming systems to perform real-time sentiment analysis

# Project Description

The project consists of a web application that allows users to enter a tweet in a dedicated field, which is then processed by a machine-learning model to predict the sentiment associated with the tweet. The predicted results are instantly displayed on the web interface, accompanied by statistics on the distribution of sentiments in the submitted tweets. Additionally, the results are stored in a MongoDB database for later analysis.

# Technologies Used

## PySpark

We used PySpark for data preprocessing and training machine learning models.

## Django

Django was used to create the web interface that allows users to upload their validation data. Django offers a robust MVC architecture and built-in features to handle HTTP requests and generate dynamic responses, making it an ideal choice for web development.

## Kafka

Kafka was used for real-time streaming of predicted results from the machine-learning model to the web interface. Kafka is a distributed messaging system that enables real-time data streaming with low latency and high reliability

## Mongo Db

MongoDB was used as the database to store the predicted results. MongoDB is a flexible and scalable NoSQL database, well suited for store semi-structured data such as analysis results feelings.

# Implementation

## Data Preprocessing with PySpark

Data preprocessing with PySpark played a crucial role in preparing of our data for training sentiment analysis models. We used PySpark to perform a series of cleanup and transformation on our raw data.

What is essential for training machine-learning models.

This included:

* Deleting empty lines
* Removal of special characters
* Lowercase
* Tokenization
* Removal of stop-words

## Model Training

A removal of special characters a lowercase a tokenization deletion of this included in particular: After comparing the performance of each model, we selected the logistic regression model as the best model for our dataset. After preprocessing our data, we proceeded to train several sentiment analysis models using PySpark MLlib. We experimented with different classification algorithms. To evaluate the perform

## Interface Web Django

We chose the Django framework to create the web interface for our project. We structured our Django application using templates to represent data, views to manage business logic and HTML templates to generate dynamic web pages. We also used the Django form features for allow users to enter tweets directly on the interface of each model, we used techniques such as cross-validation and Accuracy.

## Real-Time Streaming with Kafka

The integration of Kafka into our project made it possible to manage streaming in real time tweets, from their processing to their storage. Here is how we implemented this integration.

***Producer Kafka***

We used a Kafka producer to read tweets from a CSV file and send to the Kafka topic named 'numtest'. The Kafka producer, written in Python, reads each line of the CSV file and sends the data to the topic every three seconds. This mechanism ensures that tweets are streamed for processing in real time.

***Kafka Consumer***

On the consumer side, we have implemented a Kafka consumer, which retrieves tweets in real time from the 'numtest' topic. Consumer uses PySpark to load a previously trained and stored logistic regression model. Each tweet received is cleaned and pre-processed before being transformed by the pipeline PySpark. The result of the prediction is then displayed and stored in a database. MongoDB data for further analysis.

The integration with PySpark allows us to harness the power of Spark for bulk data processing, while Kafka ensures efficient transmission and real-time data.

## Storage with MongoDB

To store the predicted results along with other relevant data, we used MongoDB as our NoSQL database.

# Analysis result

The analysis of the results showed that the logistic regression model trained on the validation data achieved satisfactory performance in predicting tweet sentiments. The statistics displayed on the web interface provided useful information about the distribution of sentiments in tweets, allowing users to better understand trends in the data

# Conclusion

This project successfully demonstrates the integration of different technologies to achieve real-time sentiment analysis. Using PySpark for preprocessing data and model training, combined with Django for the web interface and to Kafka for streaming results, made it possible to create a robust and efficient. The integration of MongoDB as a database also made it easier to management and analysis of results.