DON BOSCO INSTITUTE OF TECHNOLOGY

Bengaluru, Karnataka – 74

TEAM COUNT: FOUR

PROJECT TITLE: REAL TIME SOCIAL MEDIA ANALYTICS PIPELINE : BUILDING A ROBUST DATA PROCESSING FRAMEWORK

TEAM LEAD NAME: RAHEESHA S

TEAM LEAD CAN ID: CAN_33695975

1) NAME: RAHEESHA S
CAN ID : CAN_33695975
ROLE : PROJECT MANAGER & RESEARCHER

2) NAME: SANJANA D
CAN ID : CAN_33698964
ROLE: MACHINE LEARNING ENGINEER

3) NAME: MAHESH N
CAN ID : CAN_33760190
ROLE: BACKEND DEVELOPER

4) NAME: SUNIL KUMAR T R
CAN ID: CAN_33706158
ROLE: FORTEND DEVELOPER

Real-Time Social Media Analytics

Pipeline: Building a Robust Data Processing Framework

Real time social media Using Deep Clustering

Phase 4: Real time social media analytic Model Deployment and Interface

Development

4.1 Overview of Model Deployment and Interface Development

Once the model is trained and validated, it's time to deploy it in a production-ready environment. The deployment process involves several steps, including:

4.2 Deploying the Model

Once the model is trained and validated, it's time to deploy it in a production-ready environment. The deployment process involves several steps, including: The trained model is deployed on a model serving platform, such as TensorFlow Serving, AWS SageMaker, or Azure Machine Learning, which provides a scalable and secure way to serve the model The process involves the following steps:

1. **Model Serving: The trained model is deployed on a model serving platform, such as TensorFlow Serving, AWS SageMaker, or Azure Machine Learning, which provides a scalable and secure way to serve the model.**
2. **API Development: An API is developed to interact with the model, allowing users to send requests and receive responses. The API can be built using frameworks such as Flask or Django.**
3. **Data Ingestion: A data ingestion pipeline is set up to collect and process social media data in real-time. This can be done using tools such as Apache Kafka, Apache Storm, or AWS Kinesis.**

```
import pandas as pd import
numpy as np import
tensorflow as tf from
tensorflow import keras
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score from
sklearn.preprocessing import StandardScaler
import tweepy
import json

# Twitter API credentials consumer_key =
"your_consumer_key" consumer_secret =
"your_consumer_secret" access_token =
"your_access_token"
access_token_secret = "your_access_token_secret"

# Set up Twitter API
auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_token_secret) api =
tweepy.API(auth)

# Define a function to collect tweets def
collect_tweets(query, count):     tweets
= []     for tweet in
tweepy.Cursor(api.search, q=query,
count=count).items():
        tweets.append(tweet.text)
return tweets

# Define a function to preprocess tweets def
preprocess_tweets(tweets):
    tweets = [tweet.lower() for tweet in tweets]     tweets
= [tweet.replace("\n", " ") for tweet in tweets]     tweets
= [tweet.replace("\t", " ") for tweet in tweets]     return
tweets
```

```
# Define a function to split data into training and testing sets def split_data(tweets,
labels):     train_tweets, test_tweets, train_labels, test_labels =
train_test_split(tweets, labels, test_size=0.2, random_state=42)
   return train_tweets, test_tweets, train_labels, test_labels

# Define a function to create a TensorFlow model
def create_model():     model = keras.Sequential([
       keras.layers.Embedding(input_dim=10000, output_dim=128, input_length=100),
keras.layers.LSTM(128, dropout=0.2),       keras.layers.Dense(64, activation="relu"),
       keras.layers.Dense(1, activation="sigmoid")
   ])
   model.compile(loss="binary_crossentropy", optimizer="adam", metrics=["accuracy"])
return model

# Define a function to train the model def
train_model(model, train_tweets, train_labels):
   model.fit(train_tweets, train_labels, epochs=10, batch_size=32, validation_split=0.2)

# Define a function to evaluate the model def
evaluate_model(model, test_tweets, test_labels):
   loss, accuracy = model.evaluate(test_tweets, test_labels)
return accuracy

# Collect tweets
tweets = collect_tweets("#machinelearning", 1000)

# Preprocess tweets
tweets = preprocess_tweets(tweets)

# Split data into training and testing sets
train_tweets, test_tweets, train_labels, test_labels = split_data(tweets, [1 if
tweet.contains("positive") else 0 for tweet in tweets])

# Create a TensorFlow model
model = create_model()

# Train the model
train_model(model, train_tweets, train_labels)

# Evaluate the model
accuracy = evaluate_model(model, test_tweets, test_labels) print("Accuracy:",
accuracy)
```

- This code collects tweets using the Twitter API, preprocesses the tweets, splits the data into training and testing sets, creates a TensorFlow model, trains the model, evaluates the model, and uses the model to make predictions on new tweets.
- - scikit-learn for machine learning tasks
- - pandas for data manipulation and analysis

- **- numpy for numerical computations**
- **- matplotlib and seaborn for data visualization**
- **- flask or django for building web applications**
- **- aws or google cloud for deploying the model on cloud platforms.**.

4.3 Developing the Web Interface

The web interface is a crucial component of real-time social media analytics, as it provides a user-friendly platform for users to interact with the analytics system. The web interface should be designed to be intuitive, responsive, and scalable, with a focus on providing real-time insights and analytics. 1. Dashboard: A customizable dashboard that provides an overview of key metrics and analytics, such as sentiment analysis, topic modeling, and influencer identification.

2. Real-time Data Visualization: Real-time data visualization tools, such as charts, graphs, and maps, to provide users with a dynamic and interactive view of social media data. 3. Search and Filter: Search and filter functionality to enable users to quickly and easily find specific data and analytics.

4. Alerts and Notifications: Alerts and notifications to inform users of important events, such as changes in sentiment or the emergence of new trends.

5. User Management: User management features, such as user authentication and authorization, to ensure that only authorized users can access the analytics system.

6. Reporting and Exporting: Reporting and exporting features to enable users to generate reports and export data for further analysis.

Technologies Used

1. Front-end Frameworks: Front-end frameworks, such as React, Angular, or Vue.js, to build the user interface and provide a responsive and interactive experience.

```
import React, { useState, useEffect } from 'react'; import
axios from 'axios';

function App() {   const [data,
setData] = useState([]);
  const [search, setSearch] = useState('');

  useEffect(() => {
axios.get('/api/data')
.then(response => {
       setData(response.data);
      })
      .catch(error => {
        console.error(error);
```

```
    });
  }, []);

  const handleSearch = (event) => {
    setSearch(event.target.value);
  };

  const filteredData = data.filter((item) => {
    return item.name.toLowerCase().includes(search.toLowerCase());
  });

  return (
    <div>
      <h1>Real-Time Social Media Analytics</h1>
      <input type="search" value={search} onChange={handleSearch} />
      <ul>
        {filteredData.map((item) => {
          return <li key={item.id}>{item.name}</li>;
        })}
      </ul>
    </div>
  );
}
```

This code creates a simple web interface with a search bar and a list of data. The data is fetched from a Node.js API using Axios.

4.4 Cloud Platform Considerations

When deploying a real-time social media analytics system on a cloud platform, there are several considerations to keep in mind. Here are some key factors to consider:

1. Scalability: The cloud platform should be able to scale to handle large volumes of social media data and user traffic.
2. Performance: The cloud platform should provide high-performance computing resources to handle the processing and analysis of social media data in real-time.
3. Security: The cloud platform should provide robust security measures to protect user data and prevent unauthorized access.
4. Cost: The cloud platform should provide a cost-effective solution for deploying and maintaining the real-time social media analytics system.
5. Integration: The cloud platform should provide easy integration with social media APIs and other data sources.
6. Data Storage: The cloud platform should provide scalable and secure data storage solutions for storing social media data.

4.5 Conclusion of Phase 4

In Phase 4, we discussed the development of a real-time social media analytics system. We covered the following topics:

1. Introduction to Real-Time Social Media Analytics: We introduced the concept of real-time social media analytics and its importance in today's digital world.

2. Data Collection: We discussed the various methods of collecting social media data, including APIs, web scraping, and social media listening tools.

3. Data Processing: We covered the processing of social media data, including data cleaning, data transformation, and data storage.

4. Data Analysis: We discussed the analysis of social media data, including sentiment analysis, topic modeling, and influencer identification.

5. Data Visualization: We covered the visualization of social media data, including the use of charts, graphs, and maps to represent data insights.

6. Cloud Platform Considerations: We discussed the considerations for deploying a real-time social media analytics system on a cloud platform, including scalability, performance, security, and cost.