REAL TIME SOCIAL MEDIA ANALYTIC

Phase 3: Real time social media analytic Model Training and Evaluation

3.1 Overview of Model Training and Evaluation

Training and evaluating models for real-time social media analytics involves a few key steps. These steps typically focus on processing large amounts of social media data in real time, extracting meaningful insights, and using machine learning models to predict or classify information. Here's a general overview of how this can be done:

3.2 Choosing Suitable Algorithms

When choosing suitable algorithms for real-time social media analytics, several factors come into play, such as the type of data being analyzed, the required speed of processing, and the ultimate goals of the analysis. Below are some algorithms commonly used for real-time social media analytics and how they align with different types of data and objectives:

Sentiment Analysis Algorithms

- **Goal**: To gauge the sentiment (positive, negative, neutral) in social media posts,

- **Algorithms**:

- **Naive Bayes**: A probabilistic classifier that works well for text classification, including sentiment analysis.
- **Support Vector Machines (SVM)**: Effective for binary sentiment classification, especially in high-dimensional feature spaces.
- **LSTM (Long Short-Term Memory)**: A type of recurrent neural network that is good at analyzing sequences of words and capturing contextual sentiment over time.

- **Use Case**: Monitoring brand sentiment, detecting crises, or understanding public opinion.

Real-Time Trending Topic Detection

- **Goal**: To identify and track what topics are becoming popular in real-time.
- **Algorithms**:
    - **Sliding Window Approach**: Tracks the frequency of hashtags or keywords over a rolling window of time, identifying spikes in activity.

- **Hashing Techniques (e.g., Count-Min Sketch)**: Used for efficiently counting frequently occurring items (hashtags, keywords) in streams of social media data.
- **Stream-based Clustering (e.g., CluStream)**: Allows clustering of data streams to detect clusters of emerging topics as they evolve.

Anomaly Detection Algorithms

- **Goal**: To detect unusual or unexpected patterns in social media data, such as bot activity, misinformation, or unusual spikes in mentions.

Algorithms:

- **Isolation Forest**: A tree-based algorithm that isolates outliers by recursively partitioning the data.
- **Autoencoders**: Neural networks used to detect anomalies by learning a compressed representation of normal behavior.

3.3 Hyperparameter Tuning

Hyperparameter tuning for real-time social media analytics involves optimizing the hyperparameters of machine learning models and algorithms used to process and analyze social media data in real-time. Given that real-time social media data is dynamic, fast, and unstructured, hyperparameter tuning is crucial for ensuring efficient, accurate, and scalable performance.

Source code for grid search for K-Means to find the best number of clusters # Import necessary libraries

```
import tweepy

import numpy as np

import pandas as pd

from sklearn.model_selection import GridSearchCV

from sklearn.linear_model import LogisticRegression

from sklearn.feature_extraction.text import CountVectorizer

from sklearn.metrics import accuracy_score

from textblob import TextBlob

from sklearn.model_selection import train_test_split



# 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 Tweepy API connection

auth = tweepy.OAuth1UserHandler(consumer_key, consumer_secret, access_token, access_token_secret)

api = tweepy.API(auth)


# Function to fetch real-time tweets

def fetch_tweets(query, count=100):

    tweets = tweepy.Cursor(api.search, q=query, lang="en", tweet_mode="extended").items(count)

    tweet_data = []

    for tweet in tweets:

        tweet_data.append(tweet.full_text)

    return tweet_data


# Function to classify the sentiment of tweets (positive/negative)

def classify_sentiment(tweet):

    analysis = TextBlob(tweet)

    # A simple sentiment analysis: Positive if polarity > 0, Negative if polarity < 0, Neutral if polarity == 0

    if analysis.sentiment.polarity > 0:

        return 1  # Positive

    elif analysis.sentiment.polarity < 0:

        return 0  # Negative

    else:
```

```
        return 2  # Neutral


# Fetch some real-time tweets related to a topic (e.g., "football")
tweets = fetch_tweets("football", 200)


# Preprocess the tweets: Convert to sentiment labels (positive/negative)
labels = [classify_sentiment(tweet) for tweet in tweets]


# Vectorize the tweets (convert text into numerical data using CountVectorizer)
vectorizer = CountVectorizer(stop_words='english')
X = vectorizer.fit_transform(tweets)


# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, labels, test_size=0.2, random_state=42)


# Define the Logistic Regression model
lr_model = LogisticRegression(max_iter=1000)


# Define the hyperparameters to tune
param_grid = {
    'C': [0.1, 1, 10],  # Regularization strength
    'solver': ['liblinear', 'saga'],  # Solvers for optimization
    'penalty': ['l2', 'none']  # Regularization type
}
```

```
Fitting 5 folds for each of 12 candidates, totalling 60 fits
Best Hyperparameters: {'C': 1, 'penalty': 'l2', 'solver': 'liblinear'}
```

Accuracy of the best model: 0.72
In this case, after performing grid search, the best hyperparameters were found to be `C=1`, `penalty='l2'`, and `solver='liblinear'`, and the accuracy of the best model on the test set was

3.4 Model Evaluation Metrics

In real-time social media analytics, evaluating the performance of models is essential for ensuring that the insights generated are accurate, relevant, and actionable. Given the dynamic nature of social media data, choosing the right evaluation metrics is critical to making informed decisions. Below, we'll discuss common evaluation metrics used for models in real-time social media analytics.

For tasks like **sentiment analysis**, **topic classification**, **spam detection**, and **social media post categorization**, classification metrics are commonly used.

a. Accuracy

- **Definition**: Accuracy is the ratio of the number of correct predictions to the total number of predictions.
- **Formula**: Accuracy=Correct PredictionsTotal Predictions\text{Accuracy} = \frac{\text{Correct Predictions}}{\text{Total Predictions}}Accuracy=Total PredictionsCorrect Predictions
- **Use Case**: For tasks like sentiment analysis where you want to classify posts as positive, negative, or neutral, accuracy provides an overall measure of how well the model performs.
- **Limitation**: Accuracy can be misleading, especially in imbalanced datasets. For example, if the majority of social media posts are neutral, a model that always predicts neutral will have high accuracy but poor performance.

b. Precision

- **Definition**: Precision measures how many of the positively predicted instances were actually positive. It's the ratio of correctly predicted positive observations to the total predicted positives.
- **Formula**: Precision=TPTP+FP\text{Precision} = \frac{TP}{TP + FP}Precision=TP+FPTP where:
    - TPTPTP = True Positives
    - FPFPFP = False Positives
- **Use Case**: Precision is critical when the cost of false positives is high. For example, when detecting **spam posts** or **fake news**, a high precision means fewer irrelevant posts are labeled as spam or fake.
- **Limitation**: Precision alone doesn't provide a complete picture of model performance, especially when the class distribution is imbalanced.

Recall (Sensitivity)

- **Definition**: Recall measures how many of the actual positive instances were correctly predicted by the model. It is the ratio of correctly predicted positive observations to all observations in the actual class.
- **Formula**: Recall=TPTP+FN\text{Recall} = \frac{TP}{TP + FN}Recall=TP+FNTP where:
    - FNFNFN = False Negatives

Use Case: Recall is essential in scenarios where missing a positive prediction is costly. For instance, in **trend detection** on social media, if the model fails to identify a trending topic, it might result in missing a valuable marketing or PR opportunity._

3.5 Cross-Validation

Cross-validation is a crucial technique in machine learning to assess how a model will generalize to an independent dataset, especially in dynamic and rapidly changing environments like **real-time social media analytics**. In real-time social media analytics, data is often collected continuously, and the underlying distribution can change over time due to trends, events, or shifts in user behavior. Cross-validation helps ensure that a model is robust, unbiased, and performs well on unseen data, which is essential when deploying models in production systems where the data continuously evolves. the data into multiple subsets, training the model on some subsets, and testing it on others.

Source code:

The example will include:

- Collecting real-time data from Twitter using Tweepy
- Performing basic preprocessing on the data (e.g., text cleaning)
- Performing sentiment analysis using a pre-trained machine learning model
- Using **TimeSeriesSplit** cross-validation for model evaluation

, pip install tweepy pandas scikit-learn nltk textblob

```
import tweepy

import nltk

from nltk.sentiment import SentimentIntensityAnalyzer

# 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)
```

3.6 Conclusion of Phase 3

In conclusion, real-time social media analytics is a powerful tool that can help businesses and organizations gain valuable insights into their customers' behavior, preferences, and opinions. By leveraging various algorithms and techniques, such as sentiment analysis, topic modeling, and influencer identification, businesses can make data-driven decisions to improve their marketing strategies, customer engagement, and overall business performance.