**AI/ML Role – Scraping for Stock Movement Predictions**

**Introduction:**

The stock market is heavily influenced by public sentiment, news, and discussions, particularly those shared on social media platforms such as Twitter, Reddit, and Telegram. With the vast amount of data generated daily on these platforms, there is an opportunity to harness this information to predict stock movements. This project aims to scrape social media data, analyze sentiment and trends, and build a machine learning model to forecast stock price changes.

The project focuses on data scraping from platforms that discuss stock market activities, using Natural Language Processing (NLP) techniques to extract meaningful insights from the text data. By identifying sentiment, frequency of mentions, and trends in discussions, the project seeks to develop a predictive model that can assist in making informed stock market decisions. The combination of data scraping, sentiment analysis, and machine learning provides a comprehensive approach to utilizing alternative data sources for financial forecasting.

**Objective:**

Developed a machine learning model that predicts stock movements by scraping data from social media platforms like Twitter. The model should extract insights from user-generated content, such as stock discussions, predictions, or sentiment analysis, and accurately forecast stock price trends.

**Task Requirements for the Project:**

**1.Data Collection:**

For Twitter, I used the Tweepy API to scrape relevant tweets with stock market-related hashtags or from specific users discussing stocks.

**2.Data Preprocessing:**

Cleaned the scraped data by removing unnecessary noise (e.g., special characters, URLs).

Handled missing or incomplete data entries.

Convert text to lowercase and normalize data for consistency.

1. **Prediction Model:**

* **Model Development:**

Build a machine learning model to predict stock movements based on the processed data (e.g., sentiment scores, keyword frequencies).

Trained the model using historical stock data and sentiment analysis results.

* **Model Testing:**

Tested the model on historical stock data to compare predicted movements with actual stock performance.

Evaluated the model's performance using metrics such as accuracy, precision, recall, and F1-score.

* **Model Evaluation:**

Provided a detailed analysis of the model's performance, including suggestions for possible improvements (e.g., tuning hyper parameters, using more data, or incorporating additional features).

**4. Technical Skills Used:**

Python Programming: Proficiency in Python for web scraping, data analysis, and model building.

**Web Scraping Tools:** Experience with tools like Beautiful Soup, Scrapy, Tweepy (for Twitter), PRAW (for Reddit), or Selenium (for Telegram).

**Natural Language Processing (NLP):** Skills in text pre processing, sentiment analysis, and text mining.

**Machine Learning:** Knowledge of machine learning frameworks such as scikit-learn, TensorFlow, or PyTorch for building and evaluating prediction models.

**5. Usage:**

**For Stock Traders:** The model can be used by stock traders or financial analysts to get real-time insights into market sentiment. By identifying public sentiment trends, traders can make informed decisions about buying or selling stocks.

**For Data Scientists:** The project demonstrates how to integrate social media data into predictive models. It showcases how alternative data sources like tweets can enhance traditional financial analysis by providing real-time public sentiment.

**For Financial Analysts:** The geographical data visualization helps analysts understand which regions are generating the most discussion around certain stocks, potentially indicating regional interest or events that could impact stock prices.

**Additional Deliverables:**

A report detailing the data collection process, preprocessing steps, model architecture, and performance evaluation.

Visualization of key insights from the data (e.g., sentiment distribution, stock mentions frequency).

Suggestions for future improvements based on model evaluation.

**CODE :**

import tweepy as tw

import pandas as pd

import matplotlib.pyplot as plt

from collections import Counter

from opencage.geocoder import OpenCageGeocode

from mpl\_toolkits.basemap import Basemap

# Twitter API credentials (replace with your own)

consumer\_key = 'JuUg51XoUXALyPICw5DrO1VpP'

consumer\_secret = ‘Bpme70iq9vf4slWGPRJHNYJi5ZZVdNKySxvfY7ElNnHurMYTBL’

access\_token = '1835895473061683200-jNpuZqbqb54JWfgMCc3aLClmXSmVSr'

access\_token\_secret = 'hbs3Lz9dy3Xm1xO1ztwU2hwXOFw36re3Z23a2MsEvNdMd'

# Authenticate to the Twitter API

auth = tw.OAuthHandler(consumer\_key, consumer\_secret)

auth.set\_access\_token(access\_token, access\_token\_secret)

api = tw.API(auth, wait\_on\_rate\_limit=True)

# Define search parameters

search\_words = "#locusts"

date\_since = "2020-1-1"

# Collect tweets

tweets = tw.Cursor(api.search\_tweets, q=search\_words, lang="en", since=date\_since, include\_entities=True).items(200)

# Process tweet data into a DataFrame

users\_locs = [[tweet.user.screen\_name, tweet.user.location, tweet.coordinates] for tweet in tweets]

tweet\_data = pd.DataFrame(users\_locs, columns=['User', 'Location', 'Coordinates'])

# Preprocess the 'Location' column (convert to lowercase)

test\_list = tweet\_data['Location'].fillna('').apply(lambda x: x.lower())

# Count occurrences of each location

location\_counts = dict(Counter(test\_list))

# Filter locations with more than 2 occurrences

filtered\_locations = {k: v for k, v in location\_counts.items() if v > 2}

# Plot the location counts

plt.figure(figsize=(18, 5))

plt.bar(filtered\_locations.keys(), filtered\_locations.values(), color='g')

plt.xticks(rotation=90)

plt.xlabel("Location")

plt.ylabel("Frequency")

plt.title("Locations mentioned in tweets")

plt.show()

# Geocoding with OpenCage (replace with your OpenCage API key)

geocoder\_key = '0cd0a0a8da9243cd8303707c3271d79d'

geocoder = OpenCageGeocode(geocoder\_key)

latitudes, longitudes = [], []

# Geocode the locations

for location in test\_list:

results = geocoder.geocode(location)

if results and len(results):

latitudes.append(results[0]['geometry']['lat'])

longitudes.append(results[0]['geometry']['lng'])

# Plot the locations on a world map using Basemap

fig = plt.figure(figsize=(15, 15))

m = Basemap(projection='merc', lat\_0=0, lon\_0=0, llcrnrlat=-80, urcrnrlat=80, llcrnrlon=-180, urcrnrlon=180)

m.drawmapboundary(fill\_color='aqua')

m.fillcontinents(color='coral', lake\_color='aqua')

m.drawcoastlines()

# Plot points on the map

for lat, lon in zip(latitudes, longitudes):

x, y = m(lon, lat)

m.scatter(x, y, marker='D', color='white')

plt.show()

**Conclusion:**

This project leverages data scraping, natural language processing (NLP), and machine learning to predict stock movements based on sentiment and discussion trends from social media platforms. By collecting and analyzing data from sources like Twitter, Reddit, or Telegram, we gain insights into market sentiment and trends that can influence stock prices. The machine learning model built on this data helps predict stock movements with measurable accuracy, providing a potential edge for traders and investors in understanding market sentiment.

While the model shows promising results, there are areas for improvement, such as incorporating more features (e.g., economic indicators), using larger datasets, or applying advanced models (e.g., deep learning). Furthermore, this approach highlights the growing role of alternative data sources like social media in financial decision-making. Future work could involve real-time analysis and predictions, integrating external data (e.g., news articles or financial reports), and improving the model's robustness for better forecasting accuracy in the dynamic stock market environment.