import yfinance as yf

import pandas as pd

from sklearn.model\_selection import train\_test\_split

# Fetch Coca-Cola stock data

ticker = "AAPL" # Coca-Cola stock ticker

data = yf.download(ticker, *start*='2015-01-01',

*end*='2023-12-31')

data = data.ffill()

# Reset index for easier handling

data.reset\_index(*inplace*=True)

# Display data structure

#print(data.isnull().sum())

# Confirm no missing values remain

#print(data.isnull().sum())

#print(data.isnull().sum())

data['MA\_20'] = data['Close'].rolling(*window*=20).mean()

data['MA\_50'] = data['Close'].rolling(*window*=50).mean()

data['Daily\_Return'] = data['Close'].pct\_change()

data['Volatility'] = data['Daily\_Return'].rolling(*window*=20).std()

# Drop rows with NA due to rolling calculations

data.dropna(*inplace*=True)

#print(data.head())

#print(data.describe())

import matplotlib.pyplot as plt

import seaborn as sns

# Line plot for stock prices

#plt.figure(figsize=(12, 6))

#plt.plot(data['Date'], data['Close'], label='Close Price')

#plt.plot(data['Date'], data['MA\_20'], label='MA 20',

#linestyle='--')

#plt.plot(data['Date'], data['MA\_50'], label='MA 50',

#linestyle='--')

#plt.title('Coca-Cola Stock Prices with Moving Averages')

#plt.xlabel('Date')

#plt.ylabel('Price')

#plt.legend()

#plt.show()

#plt.figure(figsize=(10, 8))

#sns.heatmap(data.corr(), annot=True, cmap='coolwarm')

#plt.title('Correlation Heatmap')

#plt.show()

features = ["Open", "High", "Low", "Volume", "MA\_20", "MA\_50", "Daily\_Return", "Volatility"]

target = "Close"

X = data[features]

y = data[target]

X\_train, X\_test, y\_train, y\_test = train\_test\_split(

    X, y, *test\_size*=0.2, *random\_state*=42, *shuffle*=False

)

#print(X\_train.head())

#print(y\_train.head())

from sklearn.ensemble import RandomForestRegressor

from sklearn.metrics import mean\_absolute\_error, mean\_squared\_error

model = RandomForestRegressor(*n\_estimators*=100, *random\_state*=42)

model.fit(X\_train, y\_train)

# Predict

y\_pred = model.predict(X\_test)

# Evaluate

mae = mean\_absolute\_error(y\_test, y\_pred)

mse = mean\_squared\_error(y\_test, y\_pred)

#print("MAE:", mae)

#print("MSE:", mse)

import streamlit as st

import yfinance as yf

st.title("Coca-Cola Stock Price Prediction")

data = yf.download("KO", *start*="2015-01-01")

data["MA\_20"] = data["Close"].rolling(20).mean()

data["MA\_50"] = data["Close"].rolling(50).mean()

data = data.dropna()

st.line\_chart(data[["Close", "MA\_20", "MA\_50"]])

st.write(f"Predicted Closing Price: {live\_prediction[0]}")

live\_data = yf.download(ticker, *period*='1d', *interval*='1m')

# Prepare live data for prediction

live\_data['MA\_20'] = live\_data['Close'].rolling(*window*=20).mean()

live\_data['MA\_50'] = live\_data['Close'].rolling(*window*=50).mean()

live\_data['Daily\_Return'] = live\_data['Close'].pct\_change()

live\_data['Volatility'] = live\_data['Daily\_Return'].rolling(*window*=20).std()

# Ensure no missing values

live\_data.fillna(0, *inplace*=True)

# Use the latest data point for prediction

latest\_features = live\_data[features].iloc[-1:].dropna()

live\_prediction = model.predict(latest\_features)

print(f"Predicted Closing Price: {live\_prediction[0]}")