Report on Crude Oil Price Prediction using Black-Scholes Concepts and Brownian Motion

1 Model Explanation: Black-Scholes Concepts and Brownian Motion

1.1 Brownian Motion

Brownian motion is a mathematical model used to describe the random movement of particles suspended in a fluid. In finance, it is adapted to model the random fluctuations of asset prices over time. The key idea is that price movements are unpredictable in the short term and can be represented as a random walk.

A common model for asset prices is **Geometric Brownian Motion (GBM)**, where the logarithm of the asset price follows a random walk with a drift. The stochastic process underlying the Black-Scholes model is:

$$dS = \mu S\, dt + \sigma S\, dW_t$$

Where: - S: Asset price - μ: Drift (expected rate of return) - σ: Volatility (standard deviation of returns) - dt: Small time interval - dW_t: Wiener process, representing random shock

1.2 Black-Scholes Model Concepts

The Black-Scholes model is a famous mathematical model for pricing European-style options. Its underlying assumption is that asset prices follow a random walk with constant drift and volatility.

In this task, we borrow the idea from Black-Scholes that volatility (σ) is a key parameter in determining the magnitude of random price movements. Historical data is used to estimate volatility, which is then applied to simulate future price movement using Brownian motion.

—

2 Explanation of Code Steps

The Python code predicts crude oil spot price one day ahead by following these steps:

2.1 1. Data Loading and Preparation

- Load historical crude oil price data into a pandas DataFrame.
- Convert the `observation_date` column to datetime and set as index.
- Ensure the `DCOILWTICO` column is numeric.
- Filter data to last 10 years.

2.2 2. Volatility Calculation

- Calculate daily returns using `pct_change()`.
- Compute daily volatility (std. dev. of returns).
- Annualize volatility:

$$\sigma_{annual} = \sigma_{daily} \times \sqrt{252}$$

2.3 3. Price Prediction (One Day Ahead)

- Retrieve the last available price.
- Define time step: $dt = \frac{1}{252}$.
- Generate a random shock $Z \sim N(0, 1)$.
- Predict price using Geometric Brownian Motion:

$$S_{t+dt} = S_t \times \exp\left((\mu - 0.5\sigma^2)dt + \sigma\sqrt{dt}Z\right)$$

- Here, $\mu = 0$ for simplicity in a one-day forecast.

—

3 Python Code

```
import pandas as pd
import numpy as np
from datetime import datetime, timedelta

# Load the data from the CSV file
df_oil_prices = pd.read_csv("/content/wti crude oil price.csv")

# Convert to numeric and clean data
df_oil_prices['DCOILWTICO'] = pd.to_numeric(df_oil_prices['DCOILWTICO'],
    errors='coerce')
df_oil_prices.dropna(subset=['DCOILWTICO'], inplace=True)

```

```
# Ensure datetime index
df_oil_prices['observation_date'] = pd.to_datetime(df_oil_prices['
    observation_date'])
df_oil_prices.set_index('observation_date', inplace=True)

# Filter for last 10 years
current_date = datetime.now()
start_date_10_years_ago = current_date - timedelta(days=10*365)
df_recent_prices = df_oil_prices[df_oil_prices.index >=
    start_date_10_years_ago].copy()

# Calculate returns and volatility
df_recent_prices['Daily_Return'] = df_recent_prices['DCOILWTICO'].
    pct_change()
daily_volatility = df_recent_prices['Daily_Return'].std()
annualized_volatility = daily_volatility * np.sqrt(252)

# Last price
last_price = df_recent_prices['DCOILWTICO'].iloc[-1]

# Time step
time_step_days = 1
time_step_years = time_step_days / 252.0

# Brownian motion simulation
drift = 0
np.random.seed(42)
z = np.random.normal(0, 1)

# Predicted price
predicted_price = last_price * np.exp(
    (drift - 0.5 * annualized_volatility**2) * time_step_years
    + annualized_volatility * np.sqrt(time_step_years) * z
)
```

—

4 Historical and Predicted Prices Chart

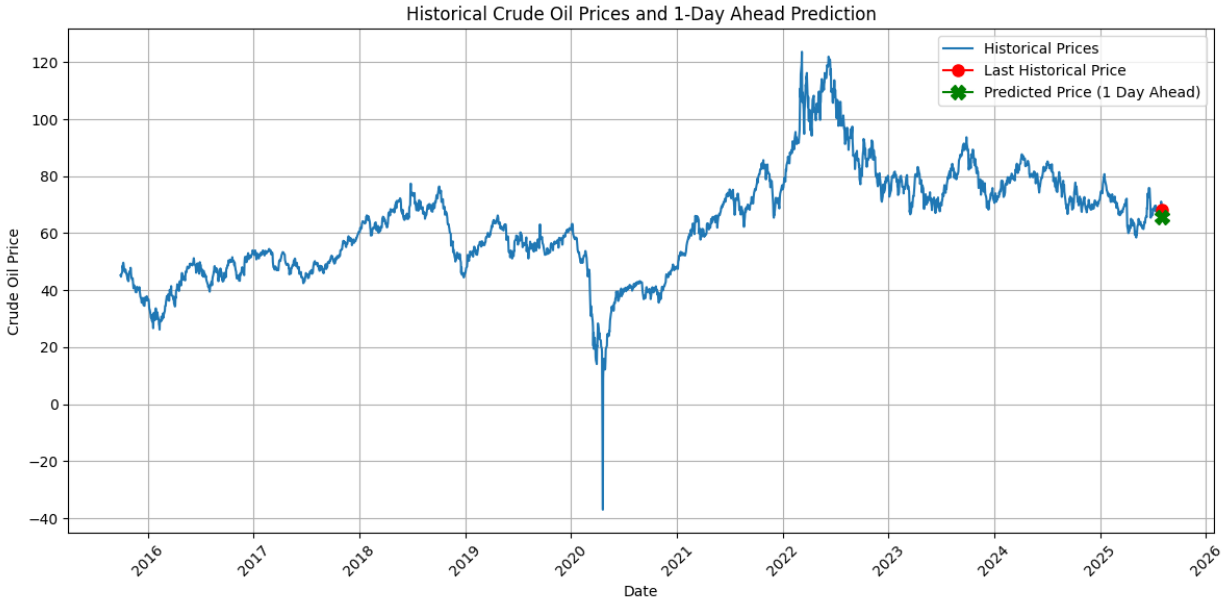

Figure 1: Historical Crude Oil Prices and 1-Day Ahead Prediction