Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Department of Computer Science and Engineering (Data Science)

S.Y.B.Tech. **Sem: IV**

Subject: Computational Methods and Pricing Models Laboratory

Experiment 3

Name: Smayan Kulkarni **SAP ID: 60009230142**

Date:	**Experiment Title:** Black-Scholes Option Pricing Model and Sensitivity Analysis
Aim	1. To understand the Black-Scholes model for pricing European call options. 2. To analyze the sensitivity of option price with respect to volatility, time to maturity, and interest rate. 3. To visualize the impact of key parameters on option pricing.
Software	Python on Google Colab
Theory	The Black-Scholes model is used to calculate the theoretical price of European call options using the following formula: $$C = SN(d_1) - Ke^{-rT}N(d_2)$$ Where: S = Current stock price K = Strike price T = Time to maturity (in years) r = Risk-free interest rate σ = Volatility of the stock N(d) = Cumulative distribution function (CDF) of the standard normal distribution $$d_1 = \frac{\ln(S/K)+(r+0.5\sigma^2)T}{\sigma\sqrt{T}}$$ $$d_2 = d_1 - \sigma\sqrt{T}$$ • **d1**: Shows the chance that the stock price will be above the option's strike price when it expires. It helps measure how sensitive the option's price is to changes in the stock price. • **d2**: Shows the chance that the option will have value (be profitable) at expiration, taking into account time and interest rates. Both d1 and d2 help calculate the option's price by considering these probabilities. **Application of Black-Scholes Model:** • Used by traders and investors to price European options on stocks, indices, and futures. • Helps in risk management by assessing the impact of different factors on option pricing. • Used for hedging strategies in financial markets. **Sensitivity Analysis** The option price is affected by several parameters: 1. Volatility (σ): Higher volatility generally increases the option price.

	2. Time to Maturity (T): Longer time to maturity often results in a higher option price. 3. Interest Rate (r): Changes in the risk-free rate can impact option pricing. **The Impact of Option Pricing Parameters** • Volatility Effect: Higher volatility increases the call option price, whereas lower volatility reduces it. • Time to Maturity Effect: A longer time to maturity generally increases the option price, and a shorter time to maturity reduces it. • Interest Rate Effect: A higher interest rate generally increases the call option price, whereas a lower interest rate reduces it. **Example:** **Given Data:** Enter Stock Price: 60 Enter Strike Price: 56 Enter Time to Maturity (in years): 0.5 Enter Risk-Free Rate (as decimal, e.g., 0.05 for 5%): 0.14 Enter Volatility (as decimal, e.g., 0.2 for 20%): 0.3 **Step 1** calculate d1,d2 after calculation d1= 0.76134, d2=0.54924 **Step 2** calculate normal distribution of d1 and d2 which is 0.77 and 0.70 respectively {use scientific calculator} **Step 3** calculate present value of exercise price using continuous discounting principle E/e^rt where e is approximately 2.71 ans = 52.21 **Step 4** c= 60*0.7768-52.21*0.7086 =46.61-37.00 =9.61
Implementation	**Step 1**: Compute Call Option Price Using Black-Scholes Model • Define the given stock parameters. • Use the Black-Scholes formula to compute the call option price. **Step 2**: Sensitivity Analysis • Vary volatility between 10% to 50% and observe the option price. • Vary time to maturity between 0.1 years to 2 years and observe the option price. • Vary interest rate between 1% to 10% and observe the option price. **Step 3**: Visual Analysis • Plot option price vs volatility. • Plot option price vs time to maturity. • Plot option price vs interest rate.
Conclusion	In this experiment, we explored the Black-Scholes Option Pricing Model for valuing European call options. Through sensitivity analysis, we observed that option prices increase with higher volatility and longer time to maturity, while the effect of interest rates varies based on market conditions. Visualizing these relationships provided insights into how

	key parameters influence option pricing, reinforcing the model's practical applications in financial markets.

7e6n7wlb6

February 27, 2025

Google Colab Link : https://colab.research.google.com/github/SmayanKulkarni/AI-and-ML-Course/blob/master/D100%20CMPM/exp-3.ipynb

[1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

[2]:
```
def cal_d1(S,K,sigma,T,r):
    d1 = (np.log(S/K) + (r+ (0.5 * (sigma)**2) )*T) / (sigma * np.sqrt(T))
    return d1
```

[3]:
```
def cal_d2(d1, sigma,T):
    d2 = d1 - (sigma * (np.sqrt(T)))
    return d2
```

[4]:
```
from scipy.stats import norm
import math
def cal_C(S,K, d1,d2,r,T):
    """
    S = Current stock price
    K = Strike price
    T = Time to maturity (in years)
    r = Risk-free interest rate
      = Volatility of the stock
    N(d) = Cumulative distribution function (CDF) of the standard normal␣
  ↪distribution
    """
    c = S * norm.cdf(d1) - K * math.exp(- r  * T) * norm.cdf(d2)

    return c
```

[5]:
```
here_d1= cal_d1(60,56,0.3,0.5,0.14)
print(here_d1)
```

```
0.7612846969447399
```

[6]:
```
here_d2 = cal_d2(here_d1, 0.3,0.5)
here_d2
```

[6]: 0.5491526625887756

[7]:
```
c_val = cal_C(60,56,here_d1,here_d2,0.14,0.5)
```

[8]:
```
c_val
```

[8]: 9.60913887372783

[9]:
```
def compute_Black_Scholes_Option_Pricing_Model(S,K,r,T,sigma):
    here_d1 = cal_d1(S,K,sigma,T,r)
    here_d2 = cal_d2(here_d1, sigma,T)
    c_val = cal_C(S,K, here_d1,here_d2,r,T)

    return c_val
```

[10]:
```
c_val = compute_Black_Scholes_Option_Pricing_Model(60,56,0.14,0.5,0.3)
```

[11]:
```
c_val
```

[11]: 9.60913887372783

[12]:
```
variation_vol = np.arange(0.1, 0.6, 0.1)
```

[13]:
```
variation_vol
```

[13]: array([0.1, 0.2, 0.3, 0.4, 0.5])

[14]:
```
val_variations = []

for var in variation_vol:
    temp_c_val = compute_Black_Scholes_Option_Pricing_Model(60,56,0.14,0.5,var)
    val_variations.append(temp_c_val)
```

[15]:
```
val_variations
```

[15]:
```
[7.822753913421124,
 8.46621414492224,
 9.60913887372783,
 10.94339920921135,
 12.35893254464969]
```

[16]:
```
variation_T = np.arange(0.1,2.1, 0.1)
```

[17]:
```
T_variations = []

for T in variation_T:
    temp_c_val = compute_Black_Scholes_Option_Pricing_Model(60,56,0.14,T,0.3)
```

```
    T_variations.append(temp_c_val)

T_variations
```

[17]:
```
[5.3523436059603,
 6.598664019842751,
 7.689492527404234,
 8.6833079963654,
 9.60913887372783,
 10.48349259225845,
 11.316762465048804,
 12.115981207773139,
 12.886167219754533,
 13.631050468250208,
 14.353494249111762,
 15.05575504225795,
 15.73965034287766,
 16.406671335504086,
 17.058061024736624,
 17.694869919417933,
 18.317996664967378,
 18.928218305033543,
 19.526213226910166,
 20.11257883718394]
```

[18]:
```
int_variations = np.arange(0.01, 0.11, 0.01)
```

[19]:
```
int_variations
```

[19]:
```
array([0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 0.1 ])
```

[20]:
```
interest_variations = []

for inte in int_variations:
    temp_c_val = compute_Black_Scholes_Option_Pricing_Model(60,56,inte,0.5,0.3)
    interest_variations.append(temp_c_val)

interest_variations
```

[20]:
```
[7.3201864357995845,
 7.487056067932336,
 7.655602926067537,
 7.825789817382386,
 7.99757870389854,
 8.170930752066702,
 8.345806383176885,
 8.522165324473981,
```

```
 8.699966660858252,
 8.879168887049723]
```

[21]:
```python
df_interest = pd.DataFrame()
```

[22]:
```python
df_interest['Interest Variation'] = interest_variations
```

[23]:
```python
df_interest
```

[23]:

	Interest Variation
0	7.320186
1	7.487056
2	7.655603
3	7.825790
4	7.997579
5	8.170931
6	8.345806
7	8.522165
8	8.699967
9	8.879169

[24]:
```python
df_interest['Interest rate'] = int_variations
```

[25]:
```python
df_interest
```

[25]:

	Interest Variation	Interest rate
0	7.320186	0.01
1	7.487056	0.02
2	7.655603	0.03
3	7.825790	0.04
4	7.997579	0.05
5	8.170931	0.06
6	8.345806	0.07
7	8.522165	0.08
8	8.699967	0.09
9	8.879169	0.10

[26]:
```python
df_volatality = pd.DataFrame()
```

[27]:
```python
df_volatality['Volatility Variation'] = variation_vol
```

[28]:
```python
df_volatality['Variation with volatility'] = val_variations
```

[29]:
```python
df_volatality
```

[29]:

	Volatility Variation	Variation with volatility
0	0.1	7.822754

```
1                    0.2                     8.466214
2                    0.3                     9.609139
3                    0.4                    10.943399
4                    0.5                    12.358933
```

[30]:
```python
df_time = pd.DataFrame()
```

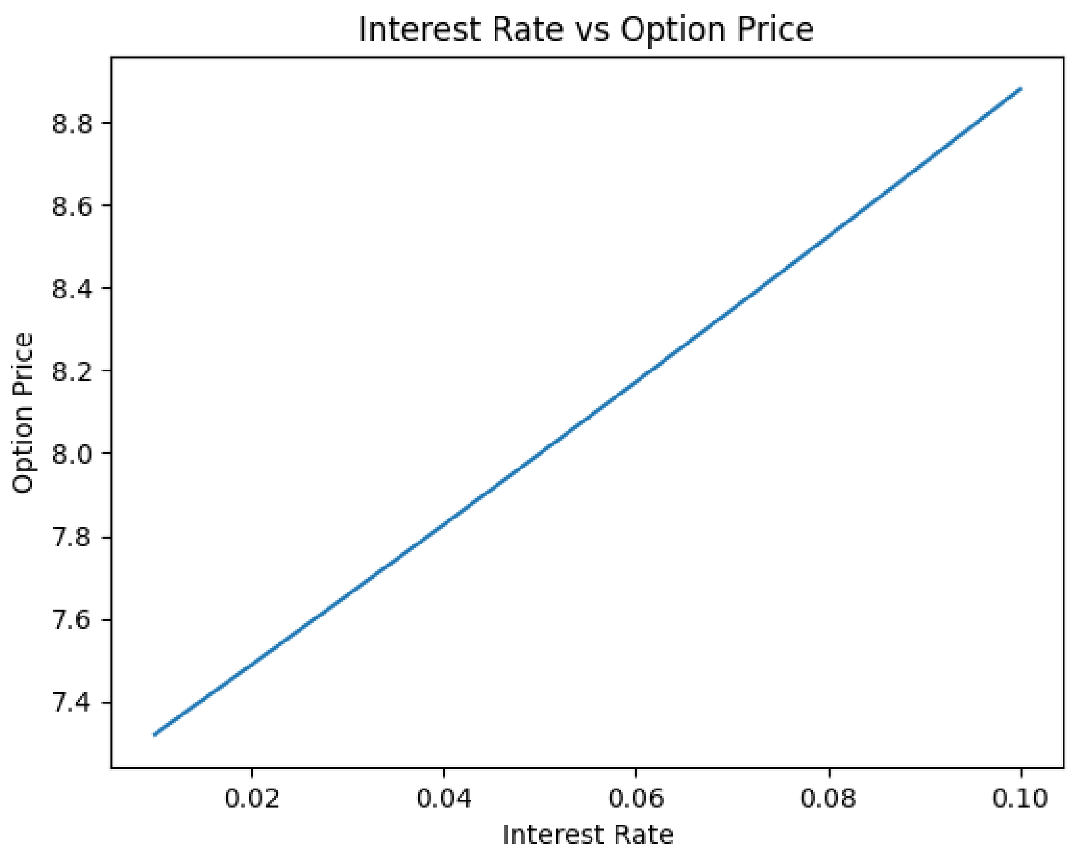
[31]:
```python
df_time['Year Variation'] = variation_T
df_time['Value Variations'] = T_variations
```

[32]:
```python
df_time
```

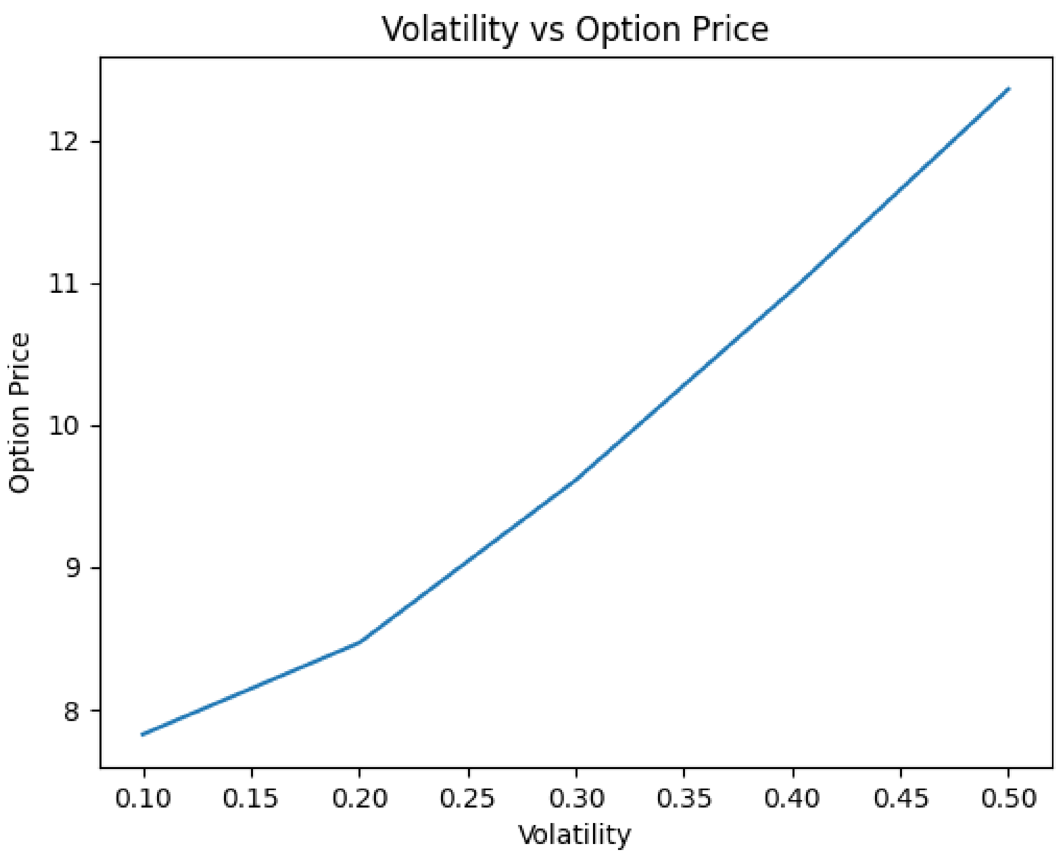
[32]:

	Year Variation	Value Variations
0	0.1	5.352344
1	0.2	6.598664
2	0.3	7.689493
3	0.4	8.683308
4	0.5	9.609139
5	0.6	10.483493
6	0.7	11.316762
7	0.8	12.115981
8	0.9	12.886167
9	1.0	13.631050
10	1.1	14.353494
11	1.2	15.055755
12	1.3	15.739650
13	1.4	16.406671
14	1.5	17.058061
15	1.6	17.694870
16	1.7	18.317997
17	1.8	18.928218
18	1.9	19.526213
19	2.0	20.112579

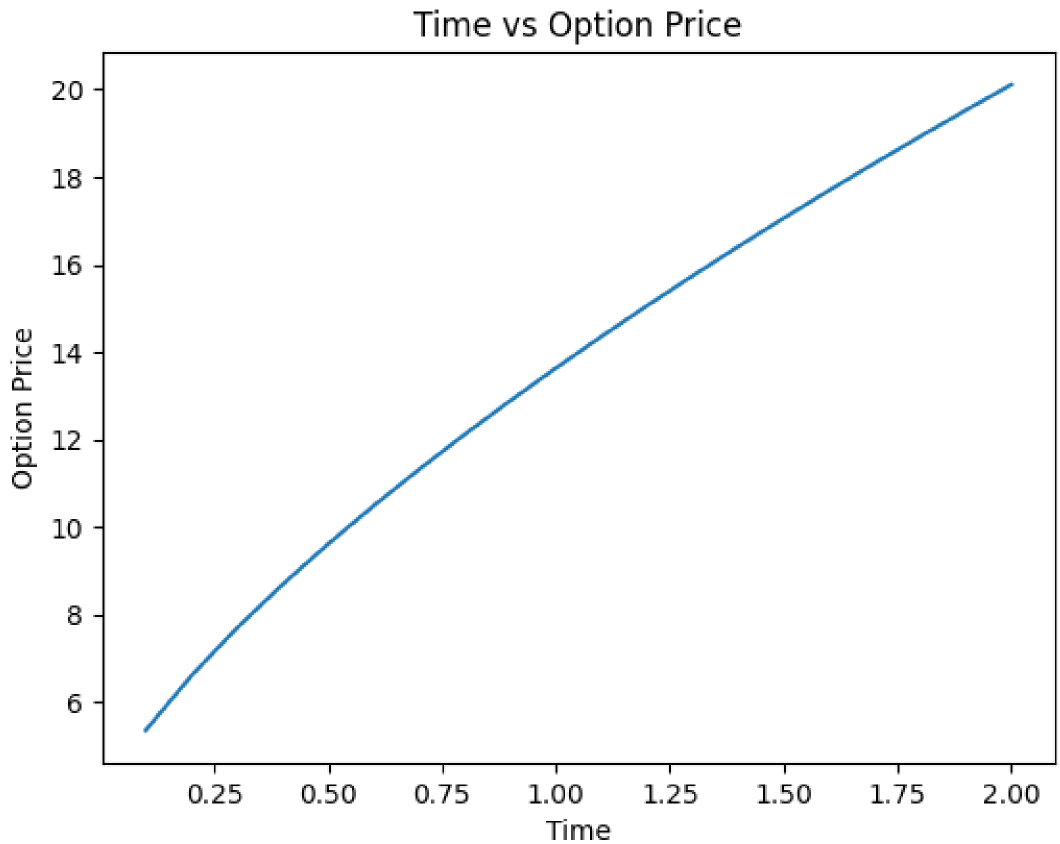
[33]:
```python
#Plotting the graph for interest rate
plt.plot(df_interest['Interest rate'], df_interest['Interest Variation'])
plt.xlabel('Interest Rate')
plt.ylabel('Option Price')
plt.title('Interest Rate vs Option Price')
plt.show()
```



[34]:
```
#Plotting the graph for volatility
plt.plot(df_volatality['Volatility Variation'], df_volatality['Variation with␣
↪volatility'])
plt.xlabel('Volatility')
plt.ylabel('Option Price')
plt.title('Volatility vs Option Price')
plt.show()
```



[35]:
```
#plotting the graph for time
plt.plot(df_time['Year Variation'], df_time['Value Variations'])
plt.xlabel('Time')
plt.ylabel('Option Price')
plt.title('Time vs Option Price')
plt.show()
```


Time vs Option Price
Option Price
20
18
16
14
12
10
8
6
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Time