1 Probability

a. Calculating Simple probabilities

#Import necessary libraries
import pandas as pd
 # Load your dataset
df = pd.read_csv('train.csv')

Calculate probability of an event
probability_event = df['Survived'].value_counts() / len(df['Survived'])
print(probability_event)

 # OUTPUT
Survived
0   0.616162
1   0.383838
Name: count, dtype: float64

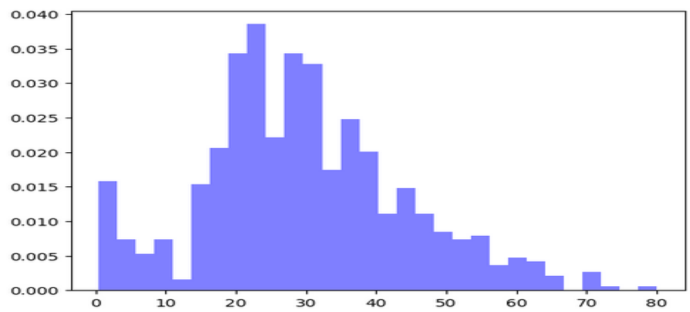
b.Applications of Probability Distributions

 # Import necessary libraries
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import norm

Drop missing values in the 'Age' column for simplicity
titanic_data = df.dropna(subset=['Age'])

 # Plot the histogram
plt.hist(df['Age'], bins=30, density=True, alpha=0.5, color='b',label='Age Distribution')

```
Out[8]: (array([0.01583946, 0.00739175, 0.00527982, 0.00739175, 0.00158395,
                0.01531148, 0.0205913 , 0.03431883, 0.03854269, 0.02217524,
                0.03431883, 0.03273488, 0.01742341, 0.02481515, 0.02006332,
                0.01108762, 0.0147835 , 0.01108762, 0.00844771, 0.00739175,
                0.00791973, 0.00369587, 0.00475184, 0.00422386, 0.00211193,
                0.        , 0.00263991, 0.00052798, 0.        , 0.00052798]),
         array([ 0.42      ,  3.07266667,  5.72533333,  8.378     , 11.03066667,
                13.68333333, 16.336     , 18.98866667, 21.64133333, 24.294     ,
                26.94666667, 29.59933333, 32.252     , 34.90466667, 37.55733333,
                40.21      , 42.86266667, 45.51533333, 48.168     , 50.82066667,
                53.47333333, 56.126     , 58.77866667, 61.43133333, 64.084     ,
                66.73666667, 69.38933333, 72.042     , 74.69466667, 77.34733333,
                80.        ]),
         <BarContainer object of 30 artists>)
```

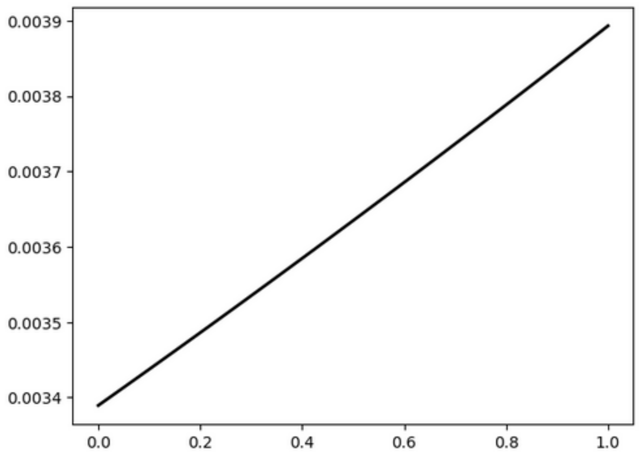


SHIVASWAMY D S
ASSISTANT PROFESSOR
DEPARTMENT OF COMPUETR SCIENCE
SHESHADRIPURAM COLLEGE B-20

 # Fit a normal distribution to the data

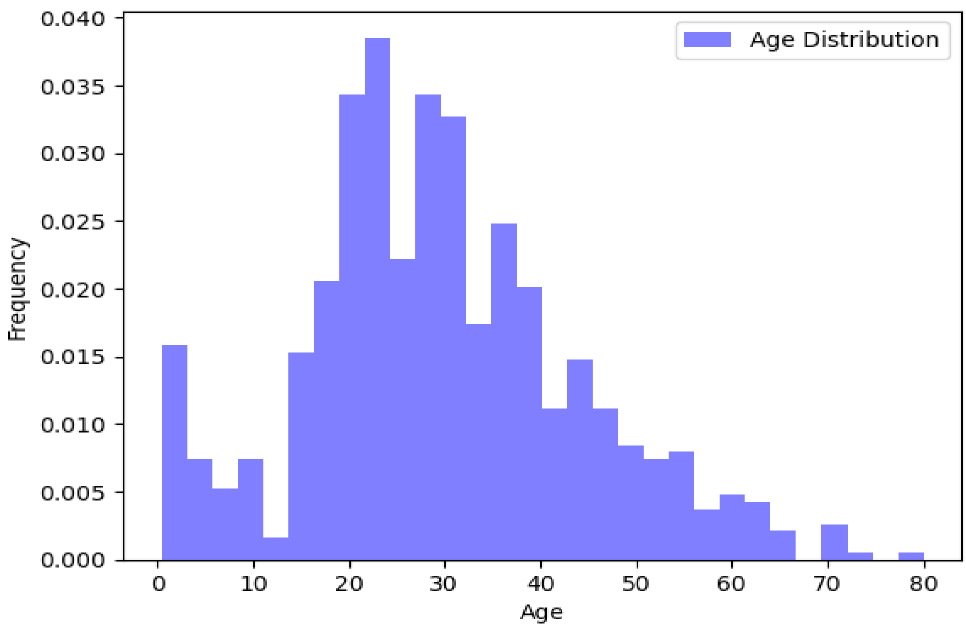
```
mu, std = norm.fit(titanic_data['Age'])
xmin, xmax = plt.xlim()
x = np.linspace(xmin, xmax, 100)
p = norm.pdf(x, mu, std)
plt.plot(x, p, 'k', linewidth=2)
```

```
Out[10]: [<matplotlib.lines.Line2D at 0x26f8f55a310>]
```



 # Display the plot

```
plt.hist(df['Age'], bins=30, density=True, alpha=0.5, color='b',label='Age Distribution')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.legend()
plt.show()
```



SHIVASWAMY D S
ASSISTANT PROFESSOR
DEPARTMENT OF COMPUETR SCIENCE
SHESHADRIPURAM COLLEGE B-20

2 Test of Significance

a. t-Test: one sample, two independent samples and Paired

from scipy.stats import ttest_ind

 # Load the dataset
df = pd.read_csv('StudentsPerformance.csv')

 # Separate data for male and female students
from scipy.stats import ttest_ind
import pandas as pd

 # Separate data for male students
male_scores = df[df['gender'] == 'male']['math score']
male_scores

OUTPUT

```
3      47
4      76
7      40
8      64
10     58
       ..
985    57
987    81
990    86
994    63
996    62
Name: math score, Length: 482, dtype: int64
```

 # Separate data for female students
female_scores = df[df['gender'] == 'female']['math score']
female_scores

```
0      72
1      69
2      90
5      71
6      88
       ..
993    62
995    88
997    59
998    68
999    77
Name: math score, Length: 518, dtype: int64
```


SHIVASWAMY D S
ASSISTANT PROFESSOR
DEPARTMENT OF COMPUETR SCIENCE
SHESHADRIPURAM COLLEGE B-20

Perform independent two-sample t-test
t_statistic, p_value = ttest_ind(male_scores, female_scores)

# Print the results
print(f'T-Statistic: {t_statistic}')
print(f'P-Value: {p_value}')

OUTPUT

```
T-Statistic: 5.383245869828983
P-Value: 9.120185549328822e-08
```

# Interpret the results
alpha = 0.05
if p_value < alpha:
    print("There is a significant difference in math scores between male and female students.")
else:
    print("There is no significant difference in math scores between male and female students.")

OUTPUT

There is a significant difference in math scores between male and female students.

# ANOVA: Comparing Multiple Groups (e.g., Ethnicity)
from scipy.stats import f_oneway

# Load the dataset
df = pd.read_csv('StudentsPerformance.csv')

# Extract math scores for each ethnicity_groups
ethnicity_groups = df['ethnicity'].unique()
ethnicity_groups

OUTPUT

```
array(['group B', 'group C', 'group A', 'group D', 'group E'],
      dtype=object)
```


SHIVASWAMY D S
ASSISTANT PROFESSOR
DEPARTMENT OF COMPUETR SCIENCE
SHESHADRIPURAM COLLEGE B-20

 # Extract math scores for each ethnicity_data
ethnicity_data = {ethnicity: df[df['ethnicity'] == ethnicity]['math score'] for ethnicity in ethnicity_groups}
ethnicity_data

OUTPUT

```
{'group B': 0      72
 2      90
 5      71
 6      88
 7      40
        ..
 969    75
 976    60
 980     8
 982    79
 991    65
 Name: math score, Length: 190, dtype: int64,
 'group C': 1      69
 4      76
 10     58
 15     69
 16     88
        ..
 979    91
 984    74
 986    40
 996    62
 997    59
 Name: math score, Length: 319, dtype: int64,
 'group A': 3      47
 13     78
 14     50
 25     73
 46     55
        ..
 974    54
 983    78
 985    57
 988    44
 994    63
 Name: math score, Length: 89, dtype: int64,
 'group D': 8      64
 11     40
 20     66
 22     44
 24     74
        ..
 989    67
 992    55
 993    62
 998    68
 999    77
 Name: math score, Length: 262, dtype: int64,
 'group E': 32      56
 34      97
 35      81
 44      50
 50      53
```


SHIVASWAMY D S
ASSISTANT PROFESSOR
DEPARTMENT OF COMPUETR SCIENCE
SHESHADRIPURAM COLLEGE B-20

```
    ...
962    100
968     68
987     81
990     86
995     88
Name: math score, Length: 140, dtype: int64}
```

 # Perform one-way ANOVA

```
f_statistic, p_value_anova = f_oneway(*ethnicity_data.values())
```

 # Print the results

```
print(f'F-Statistic: {f_statistic}')
print(f'P-Value (ANOVA): {p_value_anova}')
```

OUTPUT

```
F-Statistic: 14.593885166332637
P-Value (ANOVA): 1.3732194030370688e-11
```

 # Interpret the results

```
if p_value_anova < alpha:
   print("There is a significant difference in math scores among different ethnicities.")
else:
   print("There is no significant difference in math scores among different ethnicities.")
```

OUTPUT

There is a significant difference in math scores among different ethnicities.

SHIVASWAMY D S
ASSISTANT PROFESSOR
DEPARTMENT OF COMPUETR SCIENCE
SHESHADRIPURAM COLLEGE B-20

3. Correlation and Regression Analysis

a. Scatter Diagram, Calculating of Crrelation coefficient

#Importing necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import statsmodels.api as sm
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression, LogisticRegression
from sklearn.metrics import mean_squared_error, r2_score, accuracy_score,
confusion_matrix

 # Load the Housedata dataset
df = pd.read_csv('data.csv')

#To display the Columns in dataset
df.columns

 # Display the first five rows of the dataset
print(df.head())

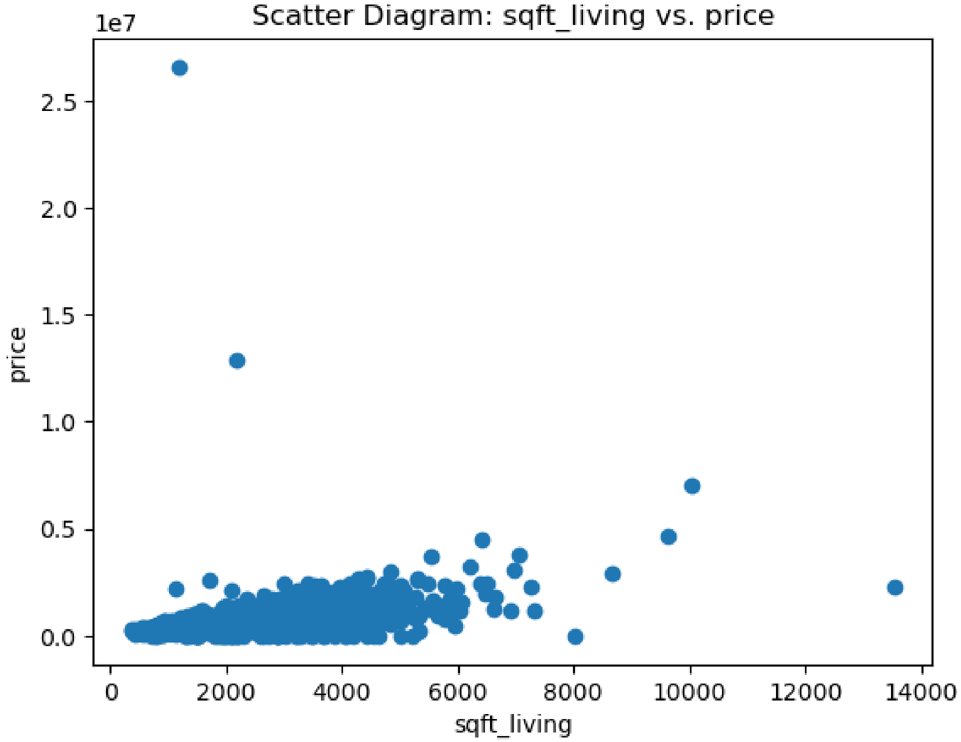
```
date      price  bedrooms  bathrooms  sqft_living  sqft_lot  \
0  2014-05-02 00:00:00   313000.0       3.0       1.50         1340      7912
1  2014-05-02 00:00:00  2384000.0       5.0       2.50         3650      9050
2  2014-05-02 00:00:00   342000.0       3.0       2.00         1930     11947
3  2014-05-02 00:00:00   420000.0       3.0       2.25         2000      8030
4  2014-05-02 00:00:00   550000.0       4.0       2.50         1940     10500

   floors  waterfront  view  condition  sqft_above  sqft_basement  yr_built  \
0     1.5           0     0          3        1340              0      1955
1     2.0           0     4          5        3370            280      1921
2     1.0           0     0          4        1930              0      1966
3     1.0           0     0          4        1000           1000      1963
4     1.0           0     0          4        1140            800      1976

   yr_renovated                    street       city  statezip country
0          2005      18810 Densmore Ave N  Shoreline  WA 98133     USA
1             0           709 W Blaine St    Seattle  WA 98119     USA
2             0  26206-26214 143rd Ave SE       Kent  WA 98042     USA
3             0           857 170th Pl NE   Bellevue  WA 98008     USA
4          1992         9105 170th Ave NE    Redmond  WA 98052     USA
```


SHIVASWAMY D S
ASSISTANT PROFESSOR
DEPARTMENT OF COMPUETR SCIENCE
SHESHADRIPURAM COLLEGE B-20

a. Scatter Diagram

 # Scatter diagram for two variables (e.g., sqft_living vs. price)

```
plt.scatter(df['sqft_living'], df['price'])
plt.title('Scatter Diagram: sqft_living vs. price')
plt.xlabel('sqft_living')
plt.ylabel('price')
plt.show()
```



 # Calculate the correlation coefficient

```
correlation_coefficient = df['sqft_living'].corr(df['price'])
print(f'Correlation Coefficient (sqft_living vs. price): {correlation_coefficient}')
```

OUTPUT

```
Correlation Coefficient (sqft_living vs. price): 0.43041002543262824
```

b. Linear Regression: Fitting, Testing Model Adequacy, and Prediction (Simple and Multiple)

 # Simple Linear Regression

```
X_simple = sm.add_constant(df[['sqft_living']])
y_simple = df['price']
model_simple = sm.OLS(y_simple, X_simple).fit()
```


SHIVASWAMY D S
ASSISTANT PROFESSOR
DEPARTMENT OF COMPUETR SCIENCE
SHESHADRIPURAM COLLEGE B-20

# Summary of the simple linear regression
print(model_simple.summary())

OUTPUT

```
              OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.185
Model:                            OLS   Adj. R-squared:                  0.185
Method:                 Least Squares   F-statistic:                     1045.
Date:                Fri, 24 Nov 2023   Prob (F-statistic):          7.55e-207
Time:                        17:00:16   Log-Likelihood:                 -66971.
No. Observations:                4600   AIC:                         1.339e+05
Df Residuals:                    4598   BIC:                         1.340e+05
Df Model:                           1
Covariance Type:            nonrobust
===============================================================================
                  coef    std err          t      P>|t|      [0.025      0.975]
-------------------------------------------------------------------------------
const        1.295e+04   1.83e+04      0.709      0.479   -2.29e+04    4.88e+04
sqft_living   251.9501      7.792     32.334      0.000     236.674     267.227
==============================================================================
Omnibus:                    12550.690   Durbin-Watson:                   1.980
Prob(Omnibus):                  0.000   Jarque-Bera (JB):        504349454.972
Skew:                          33.420   Prob(JB):                         0.00
Kurtosis:                    1623.778   Cond. No.                     5.72e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 5.72e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

# Multiple Linear Regression
X_multi = sm.add_constant(df[['sqft_living', 'bedrooms', 'bathrooms']])
y_multi = df['price']
model_multi = sm.OLS(y_multi, X_multi).fit()
# Summary of the multiple linear regression
print(model_multi.summary())

OUTPUT

```
OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.190
Model:                            OLS   Adj. R-squared:                  0.190
Method:                 Least Squares   F-statistic:                     359.8
Date:                Fri, 24 Nov 2023   Prob (F-statistic):          6.78e-210
Time:                        17:00:49   Log-Likelihood:                 -66957.
No. Observations:                4600   AIC:                         1.339e+05
Df Residuals:                    4596   BIC:                         1.339e+05
Df Model:                           3
Covariance Type:            nonrobust
===============================================================================
                  coef    std err          t      P>|t|      [0.025      0.975]
```


SHIVASWAMY D S
ASSISTANT PROFESSOR
DEPARTMENT OF COMPUETR SCIENCE
SHESHADRIPURAM COLLEGE B-20

```
---------------------------------------------------------------------------------
const          1.232e+05   3.02e+04      4.080      0.000      6.4e+04    1.82e+05
sqft_living   274.6629     12.692      21.641      0.000     249.781     299.545
bedrooms     -5.514e+04   1.04e+04     -5.296      0.000    -7.56e+04   -3.47e+04
bathrooms     1.33e+04    1.5e+04      0.889      0.374     -1.6e+04    4.26e+04
==============================================================================
Omnibus:                    12588.478   Durbin-Watson:                   1.978
Prob(Omnibus):                  0.000   Jarque-Bera (JB):        516518988.559
Skew:                          33.683   Prob(JB):                         0.00
Kurtosis:                    1643.227   Cond. No.                     9.83e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 9.83e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

 # Prediction

predictions_simple = model_simple.predict(X_simple)
predictions_simple

OUTPUT

```
0       350567.418016
1       932572.220763
2       499217.995341
3       516854.504515
4       501737.496651
            ...
4595    393398.940296
4596    380801.433743
4597    771324.136885
4598    539530.016310
4599    388359.937675
Length: 4600, dtype: float64
```

 # Prediction

X_multi = sm.add_constant(df[['sqft_living', 'bedrooms', 'bathrooms']])
y_multi = df['price']
model_multi = sm.OLS(y_multi, X_multi).fit()
predictions_multi = model_multi.predict(X_multi)
predictions_multipredictions_multi = model_multi.predict(X_multi)
predictions_multi

OUTPUT

```
0       345726.155445
1       883214.890294
2       514428.895515
3       536981.112223
4       468684.238234
            ...
4595    395744.662521
4596    391988.957691
4597    817716.458384
4598    503232.046882
4599    400228.844801
Length: 4600, dtype: float64
```


SHIVASWAMY D S
ASSISTANT PROFESSOR
DEPARTMENT OF COMPUETR SCIENCE
SHESHADRIPURAM COLLEGE B-20

c. Fitting of Linear regression

 # Assuming 'waterfront' is a binary variable indicating waterfront or not

```
X_logistic = df[['sqft_living', 'waterfront']]
y_logistic = (df['price'] > df['price'].median()).astype(int)
# Binary target variable
X_train_log, X_test_log, y_train_log, y_test_log = train_test_split(X_logistic, y_logistic, test_size=0.2, random_state=42)
print(X_logistic)
print(y_logistic)
```

OUTPUT

X_logistic:

```
      sqft_living  waterfront
0            1340           0
1            3650           0
2            1930           0
3            2000           0
4            1940           0
...           ...         ...
4595         1510           0
4596         1460           0
4597         3010           0
4598         2090           0
4599         1490           0

[4600 rows x 2 columns]
```

y_logistic:

```
0       0
1       1
2       0
3       0
4       1
       ..
4595    0
4596    1
4597    0
4598    0
4599    0
Name: price, Length: 4600, dtype: int32
```

 # Logistic Regression

```
logreg = LogisticRegression()
logreg.fit(X_train_log, y_train_log)
```

OUTPUT

```
▾ LogisticRegression
LogisticRegression()
```

`

SHIVASWAMY D S
ASSISTANT PROFESSOR
DEPARTMENT OF COMPUETR SCIENCE
SHESHADRIPURAM COLLEGE B-20

 # Predictions

```
y_pred_log = logreg.predict(X_test_log)
y_pred_log
```

OUTPUT

```
array([0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1,
       1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0,
       0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1,
       1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1,
       1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1,
       1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0,
       1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0,
       1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1,
       0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1,
       0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0,
       1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1,
       1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0,
       1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0,
       0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0,
       0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0,
       1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0,
       0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1,
       0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1,
       0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1,
       1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 0,
       0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0,
       0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1,
       0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0,
       0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1,
       0, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0,
       1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0,
       1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1,
       0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0,
       0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0,
       1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0,
       0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1,
       1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0,
       0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1,
       0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0,
       0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0,
       1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1,
       0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1,
       1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1,
       1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 0,
       0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1])
```

 # Model Evaluation

```
accuracy_log = accuracy_score(y_test_log, y_pred_log)
conf_matrix_log = confusion_matrix(y_test_log, y_pred_log)
print(f'Accuracy (Logistic Regression): {accuracy_log}')
print(f'Confusion Matrix (Logistic Regression): \n{conf_matrix_log}')
```

OUTPUT

```
Accuracy (Logistic Regression): 0.7184782608695652
Confusion Matrix (Logistic Regression):
[[351 119]
 [140 310]]
```


SHIVASWAMY D S
ASSISTANT PROFESSOR
DEPARTMENT OF COMPUETR SCIENCE
SHESHADRIPURAM COLLEGE B-20