Logistic regression

predict students are pass or fail based on their study hours

In [57]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import math
import seaborn as sns

# function for finding linear combination
def linear_combination(i, theta_0, theta_1):
    return theta_0 + theta_1 * i


# fucntion finding sigmoid()
def sigmoid(z):
    return 1 / (1 + math.exp(-z))


# function finding the predicted class
def predict(i, theta_0, theta_1, threshold=0.5):
    z = linear_combination(i, theta_0, theta_1)
    probability = sigmoid(z)
    predicted_class = 1 if probability > threshold else 0
    return predicted_class, probability,z

#getting value of intercept and coefficient
theta_0 = -3
theta_1 = 1

# this is study Hours
x = [4, 5, 6, 7,3, 1,2,3.5,2.3,1.5]


for i in x:
    predicted_class, probability,z = predict(i, theta_0, theta_1)

    print("Hours Studied:",i)
    print("Linear Combination (z):",z)
    print("Predicted Probability:", probability)
    print("Predicted Class:",predicted_class)
    print()
```

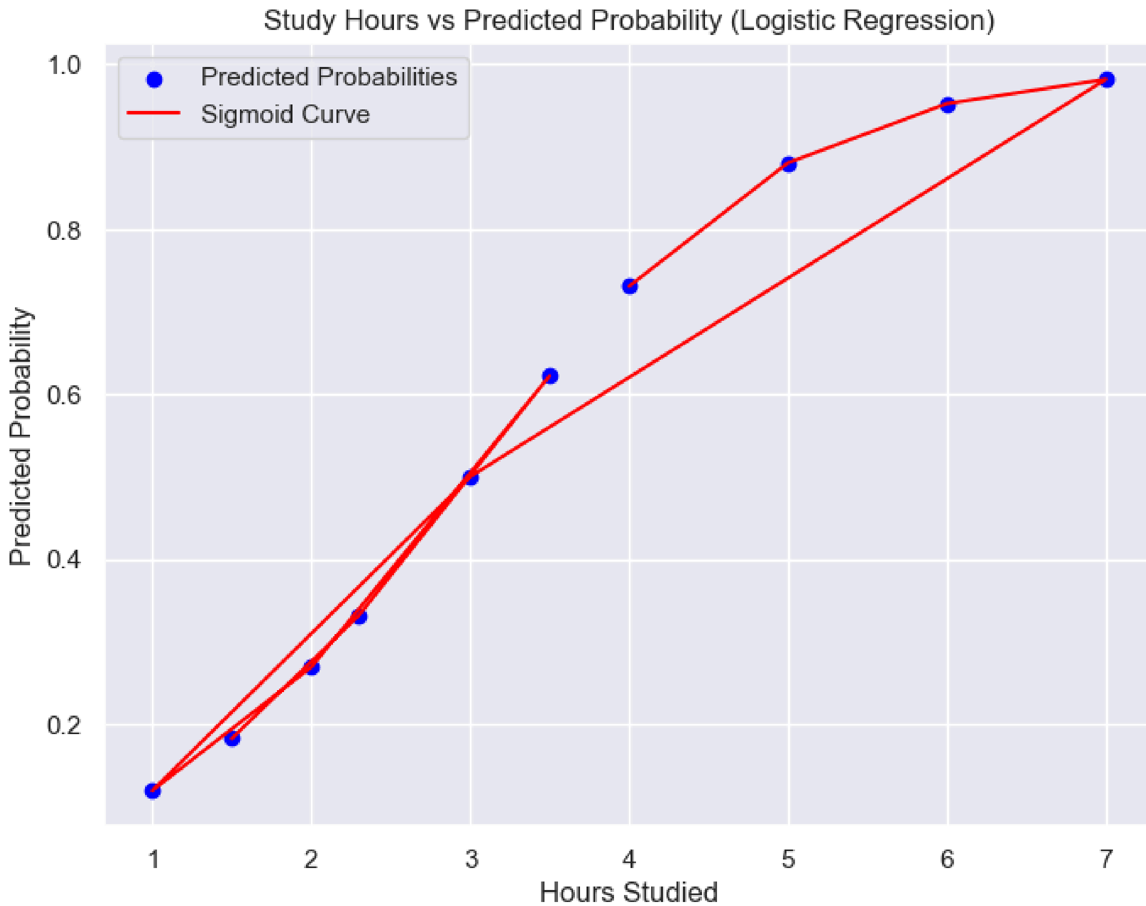
```
Hours Studied: 4
Linear Combination (z): 1
Predicted Probability: 0.7310585786300049
Predicted Class: 1

Hours Studied: 5
Linear Combination (z): 2
Predicted Probability: 0.8807970779778823
Predicted Class: 1

Hours Studied: 6
Linear Combination (z): 3
Predicted Probability: 0.9525741268224334
Predicted Class: 1

Hours Studied: 7
Linear Combination (z): 4
Predicted Probability: 0.9820137900379085
Predicted Class: 1

Hours Studied: 3
Linear Combination (z): 0
Predicted Probability: 0.5
Predicted Class: 0

Hours Studied: 1
Linear Combination (z): -2
Predicted Probability: 0.11920292202211755
Predicted Class: 0

Hours Studied: 2
Linear Combination (z): -1
Predicted Probability: 0.2689414213699951
Predicted Class: 0

Hours Studied: 3.5
Linear Combination (z): 0.5
Predicted Probability: 0.6224593312018546
Predicted Class: 1

Hours Studied: 2.3
Linear Combination (z): -0.7000000000000002
Predicted Probability: 0.33181222783183384
Predicted Class: 0

Hours Studied: 1.5
Linear Combination (z): -1.5
Predicted Probability: 0.18242552380635635
Predicted Class: 0
```

In [35]:

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import math
import seaborn as sns

# Seaborn style for the plots
sns.set()

# Linear combination function
def linear_combination(i, theta_0, theta_1):
    return theta_0 + theta_1 * i

# Sigmoid function
def sigmoid(z):
    return 1 / (1 + math.exp(-z))

# Predict class and probability function
def predict(i, theta_0, theta_1, threshold=0.5):
    z = linear_combination(i, theta_0, theta_1)
    probability = sigmoid(z)
    predicted_class = 1 if probability > threshold else 0
    return predicted_class, probability, z

theta_0 = -3
theta_1 = 1

# Study hours
x = [4, 5, 6, 7, 3, 1, 2, 3.5, 2.3, 1.5]

# To store probabilities for graphing
probabilities = []

# Predict and print results
for i in x:
    predicted_class, probability, z = predict(i, theta_0, theta_1)
    probabilities.append(probability)

    print("Hours Studied:", i)
    print("Linear Combination (z):", z)
    print("Predicted Probability:", probability)
    print("Predicted Class:", predicted_class)
    print()

# Plot the graph of study hours vs. predicted probabilities
plt.figure(figsize=(8, 6))
plt.scatter(x, probabilities, color='blue', label='Predicted Probabilitie
plt.plot(x, probabilities, color='red', label='Sigmoid Curve')
plt.xlabel('Hours Studied')
plt.ylabel('Predicted Probability')
plt.title('Study Hours vs Predicted Probability (Logistic Regression)')
plt.legend()
plt.show()
```

```
54
```

```
Hours Studied: 4
Linear Combination (z): 1
Predicted Probability: 0.7310585786300049
Predicted Class: 1

Hours Studied: 5
Linear Combination (z): 2
Predicted Probability: 0.8807970779778823
Predicted Class: 1

Hours Studied: 6
Linear Combination (z): 3
Predicted Probability: 0.9525741268224334
Predicted Class: 1

Hours Studied: 7
Linear Combination (z): 4
Predicted Probability: 0.9820137900379085
Predicted Class: 1

Hours Studied: 3
Linear Combination (z): 0
Predicted Probability: 0.5
Predicted Class: 0

Hours Studied: 1
Linear Combination (z): -2
Predicted Probability: 0.11920292202211755
Predicted Class: 0

Hours Studied: 2
Linear Combination (z): -1
Predicted Probability: 0.2689414213699951
Predicted Class: 0

Hours Studied: 3.5
Linear Combination (z): 0.5
Predicted Probability: 0.6224593312018546
Predicted Class: 1

Hours Studied: 2.3
Linear Combination (z): -0.7000000000000002
Predicted Probability: 0.33181222783183384
Predicted Class: 0

Hours Studied: 1.5
Linear Combination (z): -1.5
Predicted Probability: 0.18242552380635635
Predicted Class: 0
```



predict product will be purchased or not based on Age and estimatedSalary

In [58]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import math
```

In [59]:
```
data= pd.read_csv('logistic regression dataset-Social_Network_Ads.csv')
data
```

Out[59]:

	User ID	Gender	Age	EstimatedSalary	Purchased
0	15624510	Male	19	19000	0
1	15810944	Male	35	20000	0
2	15668575	Female	26	43000	0
3	15603246	Female	27	57000	0
4	15804002	Male	19	76000	0
...	...	...	...	...	...
395	15691863	Female	46	41000	1
396	15706071	Male	51	23000	1
397	15654296	Female	50	20000	1
398	15755018	Male	36	33000	0
399	15594041	Female	49	36000	1

400 rows × 5 columns

In [39]:
```
data.shape
```

Out[39]:
```
(400, 5)
```

In [40]:
```
data.count()
```

Out[40]:
```
User ID            400
Gender             400
Age                400
EstimatedSalary    400
Purchased          400
dtype: int64
```

In [41]:
```
data.isnull().sum()
```

Out[41]:
```
User ID            0
Gender             0
Age                0
EstimatedSalary    0
Purchased          0
dtype: int64
```

In [42]:
```
data.duplicated().sum()
```

Out[42]:
```
0
```

In [43]:
```
data.describe()
```

Out[43]:

	User ID	Age	EstimatedSalary	Purchased
count	4.000000e+02	400.000000	400.000000	400.000000
mean	1.569154e+07	37.655000	69742.500000	0.357500
std	7.165832e+04	10.482877	34096.960282	0.479864
min	1.556669e+07	18.000000	15000.000000	0.000000
25%	1.562676e+07	29.750000	43000.000000	0.000000
50%	1.569434e+07	37.000000	70000.000000	0.000000
75%	1.575036e+07	46.000000	88000.000000	1.000000
max	1.581524e+07	60.000000	150000.000000	1.000000

In [44]:
```
data.nunique()
```

Out[44]:
```
User ID            400
Gender               2
Age                 43
EstimatedSalary    117
Purchased            2
dtype: int64
```

In [45]:
```
# Age and Salary take as a input feature..
X=data[["Age", 'EstimatedSalary']].values
pd.DataFrame(X,columns=['Age','Salary'])
```

Out[45]:

	Age	Salary
0	19	19000
1	35	20000
2	26	43000
3	27	57000
4	19	76000
...	...	...
395	46	41000
396	51	23000
397	50	20000
398	36	33000
399	49	36000

400 rows × 2 columns

In [46]:
```
y=data["Purchased"].values
pd.DataFrame(y,columns=['Purchased'])
```

Out[46]:

	Purchased
0	0
1	0
2	0
3	0
4	0
...	...
395	1
396	1
397	1
398	0
399	1

400 rows × 1 columns

In [47]:
```python

theta_0 = -3
theta_1 = 0.8
theta_2 = 3.5

def linear_combination(age, salary, theta_0, theta_1, theta_2):
    return theta_0 + theta_1 * age + theta_2 * salary

z_values = []

for index, row in data.iterrows():
    z = linear_combination(row['Age'], row['EstimatedSalary'], theta_0, t
    z_values.append(z)

data['Linear_Combination'] = z_values

def sigmoid(z):
    return 1 / (1 + np.exp(-z))

data['Sigmoid'] = data['Linear_Combination'].apply(sigmoid)


threshold = 0.5


data['Product'] = data['Sigmoid'].apply(lambda x: 'Purchased' if x >= thr

# Display the results
print(data[['Age', 'EstimatedSalary', 'Linear_Combination', 'Sigmoid', 'P

```

```
     Age  EstimatedSalary  Linear_Combination  Sigmoid    Product
0     19            19000             66512.2      1.0  Purchased
1     35            20000             70025.0      1.0  Purchased
2     26            43000            150517.8      1.0  Purchased
3     27            57000            199518.6      1.0  Purchased
4     19            76000            266012.2      1.0  Purchased
..   ...              ...                 ...      ...        ...
395   46            41000            143533.8      1.0  Purchased
396   51            23000             80537.8      1.0  Purchased
397   50            20000             70037.0      1.0  Purchased
398   36            33000            115525.8      1.0  Purchased
399   49            36000            126036.2      1.0  Purchased

[400 rows x 5 columns]
```

In [51]:

```
data['Product'].nunique
```

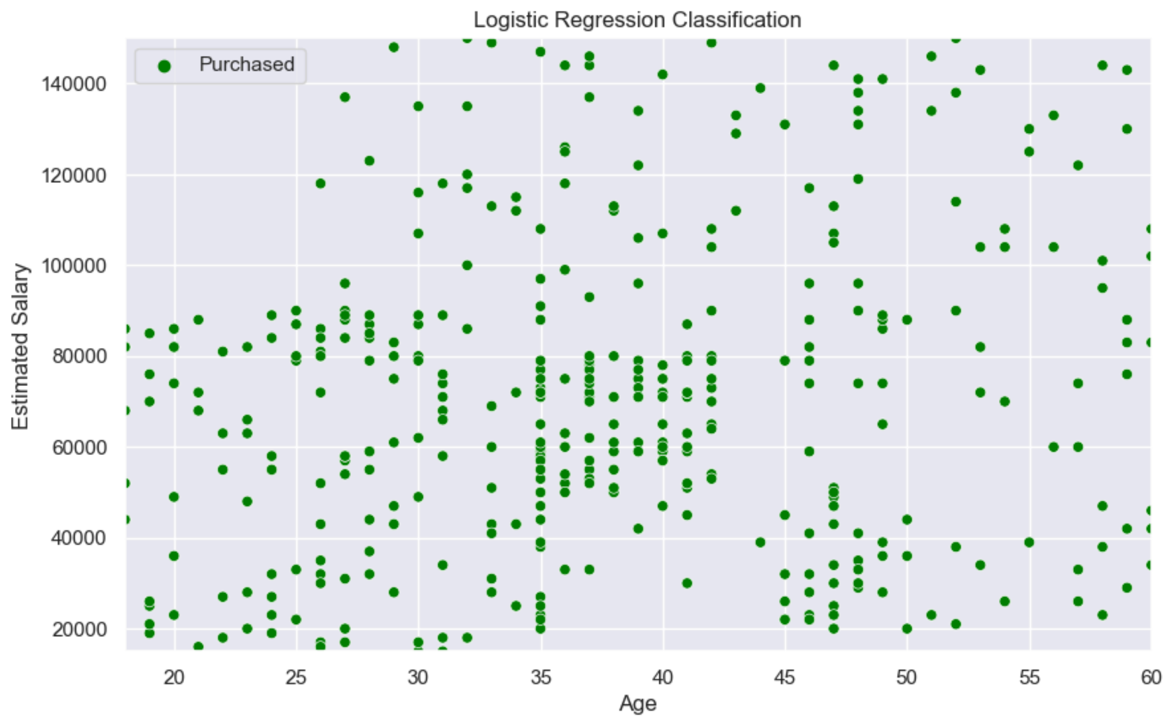
Out[51]:

```
<bound method IndexOpsMixin.nunique of 0      Purchased
1      Purchased
2      Purchased
3      Purchased
4      Purchased
         ...
395    Purchased
396    Purchased
397    Purchased
398    Purchased
399    Purchased
Name: Product, Length: 400, dtype: object>
```

In [61]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

data = pd.read_csv('logistic regression dataset-Social_Network_Ads.csv')

X = data[["Age", 'EstimatedSalary']].values
y = data["Purchased"].values

theta_0 = -3
theta_1 = 0.8
theta_2 = 3.5

def linear_combination(age, salary, theta_0, theta_1, theta_2):
    return theta_0 + theta_1 * age + theta_2 * salary

def sigmoid(z):

    return 1 / (1 + np.exp(-np.clip(z, -500, 500)))


data['Linear_Combination'] = data.apply(lambda row: linear_combination(rc
data['Sigmoid'] = data['Linear_Combination'].apply(sigmoid)


threshold = 0.5
data['Product'] = data['Sigmoid'].apply(lambda x: 'Purchased' if x >= thr

print(data[['Age', 'EstimatedSalary', 'Linear_Combination', 'Sigmoid', 'P

plt.figure(figsize=(10, 6))

sns.scatterplot(data=data, x='Age', y='EstimatedSalary', hue='Product', s

age_range = np.linspace(data['Age'].min(), data['Age'].max(), 100)
salary_range = np.linspace(data['EstimatedSalary'].min(), data['Estimated
age_grid, salary_grid = np.meshgrid(age_range, salary_range)
z_grid = linear_combination(age_grid.ravel(), salary_grid.ravel(), theta_
z_grid = sigmoid(z_grid).reshape(age_grid.shape)

# Plot decision boundary
plt.contour(age_grid, salary_grid, z_grid, levels=[0.5], colors='blue', l

plt.title('Logistic Regression Classification')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```

```
     Age  EstimatedSalary  Linear_Combination  Sigmoid    Product
0     19            19000             66512.2      1.0  Purchased
1     35            20000             70025.0      1.0  Purchased
2     26            43000            150517.8      1.0  Purchased
3     27            57000            199518.6      1.0  Purchased
4     19            76000            266012.2      1.0  Purchased
..   ...              ...                 ...      ...        ...
395   46            41000            143533.8      1.0  Purchased
396   51            23000             80537.8      1.0  Purchased
397   50            20000             70037.0      1.0  Purchased
398   36            33000            115525.8      1.0  Purchased
399   49            36000            126036.2      1.0  Purchased

[400 rows x 5 columns]
```



In [ ]:
```
1
```