In [1]:
```
import pandas as pd
```

In [4]:
```
df = pd.read_csv('Social_Network_Ads.csv')
df
```

Out[4]:

	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 [5]:
```
#input data
x=df[['Age','EstimatedSalary']]

#output data
y=df['Purchased']
```

In [6]:
```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
x_scaled = scaler.fit_transform(x)
```

In [7]:
```
from sklearn.model_selection import train_test_split
```

In [13]:
```
x_train, x_test, y_train, y_test = train_test_split(x_scaled, y, random_state=0, test_size=0.25)
x_train
```

Out[13]:
```
array([[0.61904762, 0.17777778],
       [0.33333333, 0.77777778],
       [0.47619048, 0.25925926],
       [0.33333333, 0.88888889],
       [0.80952381, 0.04444444],
       [0.83333333, 0.65925926],
       [0.5       , 0.2       ],
       [0.47619048, 0.34074074],
       [0.42857143, 0.25925926],
       [0.42857143, 0.35555556],
       [0.4047619 , 0.07407407],
       [0.4047619 , 0.25925926],
       [0.57142857, 0.42962963],
       [0.69047619, 0.25185185],
       [0.97619048, 0.1037037 ],
       [0.73809524, 0.37037037],
       [0.64285714, 0.85925926],
       [0.30952381, 0.54814815],
       [0.66666667, 0.4962963 ],
       [0.69047619, 0.26666667],
       [0.19047619, 0.        ],
       [1.        , 0.64444444],
       [0.47619048, 0.71851852],
       [0.52380952, 0.68148148],
       [0.57142857, 0.28148148],
       [0.4047619 , 0.32592593],
       [0.71428571, 0.19259259],
       [0.71428571, 0.88148148],
       [0.47619048, 0.72592593],
       [0.26190476, 0.98518519],
       [0.19047619, 0.        ],
       [1.        , 0.2       ],
       [0.14285714, 0.02962963],
       [0.57142857, 0.99259259],
       [0.66666667, 0.6       ],
       [0.23809524, 0.32592593],
       [0.5       , 0.6       ],
```

```
[0.23809524, 0.54814815],
[0.54761905, 0.42222222],
[0.64285714, 0.08148148],
[0.35714286, 0.4       ],
[0.04761905, 0.4962963 ],
[0.30952381, 0.43703704],
[0.57142857, 0.48148148],
[0.4047619 , 0.42222222],
[0.35714286, 0.99259259],
[0.52380952, 0.41481481],
[0.78571429, 0.97037037],
[0.66666667, 0.47407407],
[0.4047619 , 0.44444444],
[0.47619048, 0.26666667],
[0.42857143, 0.44444444],
[0.45238095, 0.46666667],
[0.47619048, 0.34074074],
[1.        , 0.68888889],
[0.04761905, 0.4962963 ],
[0.92857143, 0.43703704],
[0.57142857, 0.37037037],
[0.19047619, 0.48148148],
[0.66666667, 0.75555556],
[0.4047619 , 0.34074074],
[0.07142857, 0.39259259],
[0.23809524, 0.21481481],
[0.54761905, 0.53333333],
[0.45238095, 0.13333333],
[0.21428571, 0.55555556],
[0.5       , 0.2       ],
[0.23809524, 0.8       ],
[0.30952381, 0.76296296],
[0.16666667, 0.53333333],
[0.4047619 , 0.41481481],
[0.45238095, 0.40740741],
[0.4047619 , 0.17777778],
[0.69047619, 0.05925926],
[0.4047619 , 0.97777778],
[0.71428571, 0.91111111],
[0.19047619, 0.52592593],
[0.16666667, 0.47407407],
[0.80952381, 0.91111111],
[0.78571429, 0.05925926],
[0.4047619 , 0.33333333],
[0.35714286, 0.72592593],
[0.28571429, 0.68148148],
[0.71428571, 0.13333333],
[0.54761905, 0.48148148],
[0.71428571, 0.6       ],
[0.30952381, 0.02222222],
[0.30952381, 0.41481481],
[0.5952381 , 0.84444444],
[0.97619048, 0.45185185],
[0.        , 0.21481481],
[0.42857143, 0.76296296],
[0.57142857, 0.55555556],
[0.69047619, 0.11111111],
[0.19047619, 0.20740741],
[0.52380952, 0.46666667],
[0.66666667, 0.32592593],
[0.97619048, 0.2       ],
[0.66666667, 0.43703704],
[0.4047619 , 0.56296296],
[0.23809524, 0.32592593],
[0.52380952, 0.31111111],
[0.97619048, 0.94814815],
[0.92857143, 0.08148148],
[0.80952381, 0.17037037],
[0.69047619, 0.72592593],
[0.83333333, 0.94814815],
[0.4047619 , 0.08888889],
[0.95238095, 0.63703704],
[0.64285714, 0.22222222],
[0.11904762, 0.4962963 ],
[0.66666667, 0.05925926],
[0.57142857, 0.37037037],
[0.23809524, 0.51111111],
[0.47619048, 0.32592593],
[0.19047619, 0.51111111],
[0.26190476, 0.0962963 ],
[0.45238095, 0.41481481],
[0.0952381 , 0.2962963 ],
[0.71428571, 0.14814815],
```

```
[0.73809524, 0.0962963 ],
[0.47619048, 0.37037037],
[0.21428571, 0.01481481],
[0.66666667, 0.0962963 ],
[0.71428571, 0.93333333],
[0.19047619, 0.01481481],
[0.4047619 , 0.60740741],
[0.5       , 0.32592593],
[0.14285714, 0.08888889],
[0.33333333, 0.02222222],
[0.66666667, 0.54074074],
[0.4047619 , 0.31851852],
[0.9047619 , 0.33333333],
[0.69047619, 0.14074074],
[0.52380952, 0.42222222],
[0.33333333, 0.62962963],
[0.02380952, 0.04444444],
[0.16666667, 0.55555556],
[0.4047619 , 0.54074074],
[0.23809524, 0.12592593],
[0.76190476, 0.03703704],
[0.52380952, 0.32592593],
[0.76190476, 0.21481481],
[0.4047619 , 0.42222222],
[0.52380952, 0.94074074],
[0.66666667, 0.12592593],
[0.5       , 0.41481481],
[0.04761905, 0.43703704],
[0.26190476, 0.44444444],
[0.30952381, 0.45185185],
[0.69047619, 0.07407407],
[0.52380952, 0.34074074],
[0.38095238, 0.71851852],
[0.47619048, 0.48148148],
[0.57142857, 0.44444444],
[0.69047619, 0.23703704],
[0.5       , 0.44444444],
[0.02380952, 0.07407407],
[0.45238095, 0.48148148],
[0.42857143, 0.33333333],
[0.54761905, 0.27407407],
[0.42857143, 0.81481481],
[0.71428571, 0.1037037 ],
[0.42857143, 0.82222222],
[0.78571429, 0.88148148],
[0.21428571, 0.31111111],
[0.47619048, 0.41481481],
[0.5       , 0.34074074],
[0.0952381 , 0.08888889],
[0.35714286, 0.33333333],
[0.71428571, 0.43703704],
[0.95238095, 0.05925926],
[0.83333333, 0.42222222],
[0.33333333, 0.75555556],
[0.85714286, 0.40740741],
[0.28571429, 0.48148148],
[0.95238095, 0.59259259],
[0.19047619, 0.27407407],
[0.64285714, 0.47407407],
[0.14285714, 0.2962963 ],
[0.52380952, 0.44444444],
[0.35714286, 0.0962963 ],
[0.61904762, 0.91851852],
[0.0952381 , 0.02222222],
[0.35714286, 0.26666667],
[0.5952381 , 0.87407407],
[0.14285714, 0.12592593],
[0.66666667, 0.05185185],
[0.4047619 , 0.2962963 ],
[0.85714286, 0.65925926],
[0.71428571, 0.77037037],
[0.4047619 , 0.28148148],
[0.45238095, 0.95555556],
[0.11904762, 0.37777778],
[0.45238095, 0.9037037 ],
[0.30952381, 0.31851852],
[0.35714286, 0.19259259],
[0.64285714, 0.05185185],
[0.28571429, 0.        ],
[0.02380952, 0.02962963],
[0.73809524, 0.43703704],
[0.5       , 0.79259259],
[0.4047619 , 0.42962963],
```

```
[0.5       , 0.41481481],
[0.14285714, 0.05925926],
[0.54761905, 0.42222222],
[0.26190476, 0.5037037 ],
[0.85714286, 0.08148148],
[0.4047619 , 0.21481481],
[0.45238095, 0.44444444],
[0.26190476, 0.23703704],
[0.30952381, 0.39259259],
[0.57142857, 0.28888889],
[0.28571429, 0.88888889],
[0.80952381, 0.73333333],
[0.76190476, 0.15555556],
[0.9047619 , 0.87407407],
[0.26190476, 0.34074074],
[0.28571429, 0.54814815],
[0.19047619, 0.00740741],
[0.35714286, 0.11851852],
[0.54761905, 0.42222222],
[0.42857143, 0.13333333],
[0.88095238, 0.81481481],
[0.71428571, 0.85925926],
[0.54761905, 0.41481481],
[0.28571429, 0.34814815],
[0.45238095, 0.42222222],
[0.54761905, 0.35555556],
[0.95238095, 0.23703704],
[0.28571429, 0.74814815],
[0.04761905, 0.25185185],
[0.45238095, 0.43703704],
[0.54761905, 0.32592593],
[0.73809524, 0.54814815],
[0.23809524, 0.47407407],
[0.83333333, 0.4962963 ],
[0.52380952, 0.31111111],
[1.        , 0.14074074],
[0.4047619 , 0.68888889],
[0.07142857, 0.42222222],
[0.47619048, 0.41481481],
[0.5       , 0.67407407],
[0.45238095, 0.31111111],
[0.19047619, 0.42222222],
[0.4047619 , 0.05925926],
[0.85714286, 0.68888889],
[0.28571429, 0.01481481],
[0.5       , 0.88148148],
[0.26190476, 0.20740741],
[0.35714286, 0.20740741],
[0.4047619 , 0.17037037],
[0.54761905, 0.22222222],
[0.54761905, 0.42222222],
[0.5       , 0.88148148],
[0.21428571, 0.9037037 ],
[0.07142857, 0.00740741],
[0.19047619, 0.12592593],
[0.30952381, 0.37777778],
[0.5       , 0.42962963],
[0.54761905, 0.47407407],
[0.69047619, 0.25925926],
[0.54761905, 0.11111111],
[0.45238095, 0.57777778],
[1.        , 0.22962963],
[0.16666667, 0.05185185],
[0.23809524, 0.16296296],
[0.47619048, 0.2962963 ],
[0.42857143, 0.28888889],
[0.04761905, 0.15555556],
[0.9047619 , 0.65925926],
[0.52380952, 0.31111111],
[0.57142857, 0.68888889],
[0.04761905, 0.05925926],
[0.52380952, 0.37037037],
[0.69047619, 0.03703704],
[0.        , 0.52592593],
[0.4047619 , 0.47407407],
[0.92857143, 0.13333333],
[0.38095238, 0.42222222],
[0.73809524, 0.17777778],
[0.21428571, 0.11851852],
[0.02380952, 0.40740741],
[0.5       , 0.47407407],
[0.19047619, 0.48888889],
[0.16666667, 0.48148148],
```

```
        [0.23809524, 0.51851852],
        [0.88095238, 0.17777778],
        [0.76190476, 0.54074074],
        [0.73809524, 0.54074074],
        [0.80952381, 1.        ],
        [0.4047619 , 0.37037037],
        [0.57142857, 0.28888889],
        [0.38095238, 0.20740741],
        [0.45238095, 0.27407407],
        [0.71428571, 0.11111111],
        [0.26190476, 0.20740741],
        [0.42857143, 0.27407407],
        [0.21428571, 0.28888889],
        [0.19047619, 0.76296296]])
```

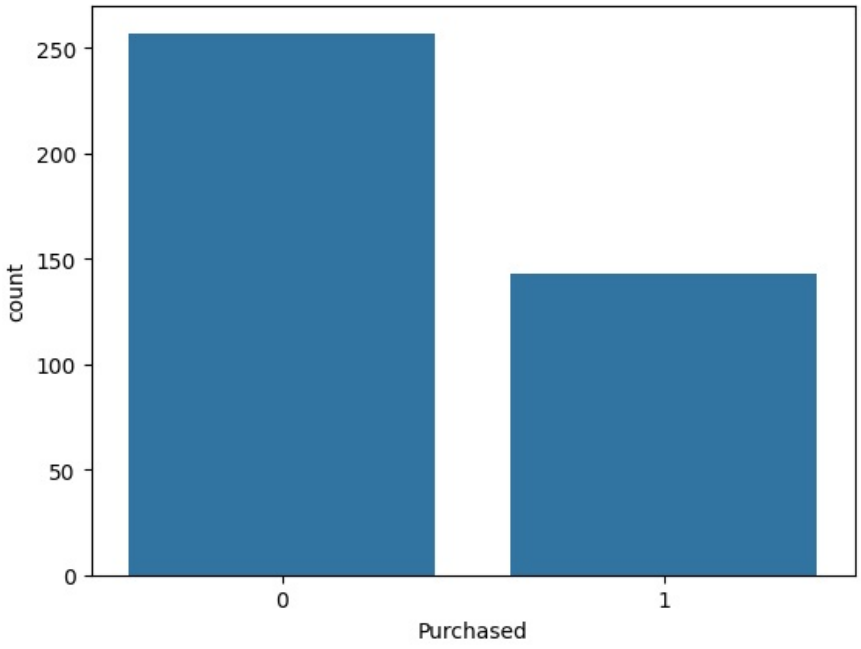
In [14]:
```
y_train
```

Out[14]:
```
250    0
63     1
312    0
159    1
283    1
      ..
323    1
192    0
117    0
47     0
172    0
Name: Purchased, Length: 300, dtype: int64
```

In [15]:
```
from sklearn.linear_model import LogisticRegression
```

In [16]:
```
import seaborn as sns
sns.countplot(x=y)
```

Out[16]:
```
<Axes: xlabel='Purchased', ylabel='count'>
```



In [17]:
```
y.value_counts()
```

Out[17]:
```
Purchased
0    257
1    143
Name: count, dtype: int64
```

In [18]:
```
classifier = LogisticRegression()
```

In [19]:
```
 classifier.fit(x_train,y_train)
```

Out[19]:
```
▾ LogisticRegression
LogisticRegression()
```

In [20]:
```
y_pred = classifier.predict(x_test)
y_train.shape
```

Out[20]:
```
(300,)
```

In [21]:
```
x_train.shape
```

Out[21]:
```
(300, 2)
```

In [22]:
```
y_pred
```

Out[22]:
```
array([0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1,
       0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1,
       0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1,
       0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1], dtype=int64)
```

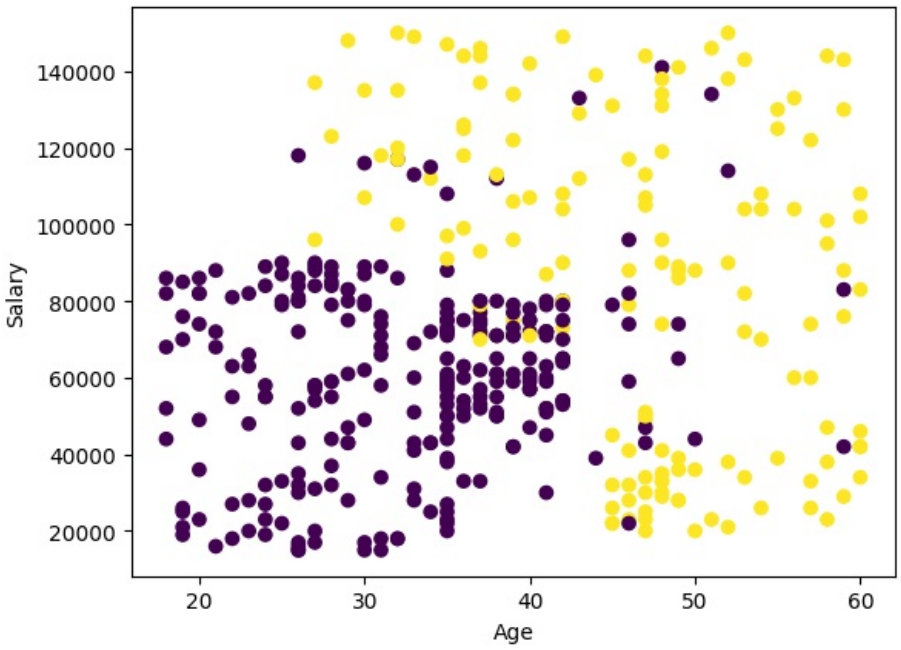
In [23]:
```
y_test
```

Out[23]:
```
132    0
309    0
341    0
196    0
246    0
      ..
146    1
135    0
390    1
264    1
364    1
Name: Purchased, Length: 100, dtype: int64
```

In [24]:
```
import matplotlib.pyplot as plt
```

In [25]:
```
plt.xlabel('Age')
plt.ylabel('Salary')
plt.scatter(x['Age'],x['EstimatedSalary'],c=y)
```

Out[25]:
```
<matplotlib.collections.PathCollection at 0x27778a82630>
```



In [26]:
```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
x_scaled = scaler.fit_transform(x)
```

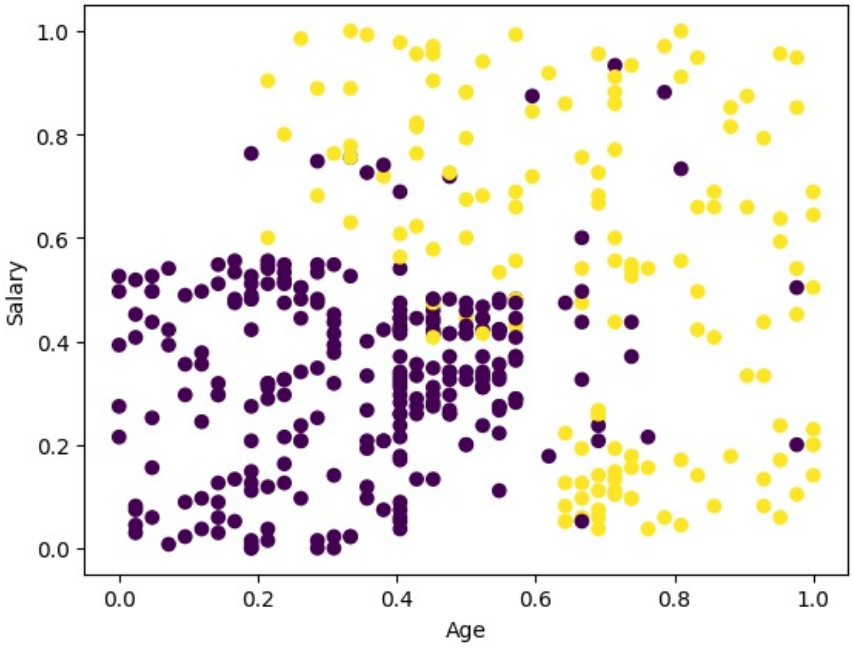
In [27]:
```
pd.DataFrame(x_scaled).describe()
```

Out[27]:

	0	1
count	400.000000	400.000000
mean	0.467976	0.405500
std	0.249592	0.252570
min	0.000000	0.000000
25%	0.279762	0.207407
50%	0.452381	0.407407
75%	0.666667	0.540741
max	1.000000	1.000000

In [28]:
```
plt.xlabel('Age')
plt.ylabel('Salary')
plt.scatter(x_scaled[:,0],x_scaled[:,1],c=y)
```

Out[28]:
```
<matplotlib.collections.PathCollection at 0x2777ab6dbe0>
```



In [29]:
```
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,y_pred)
```

Out[29]:
```
array([[67,  1],
       [10, 22]], dtype=int64)
```

In [30]:
```
y_test.value_counts()
```

Out[30]:
```
Purchased
0    68
1    32
Name: count, dtype: int64
```

In [31]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import ConfusionMatrixDisplay, confusion_matrix
```

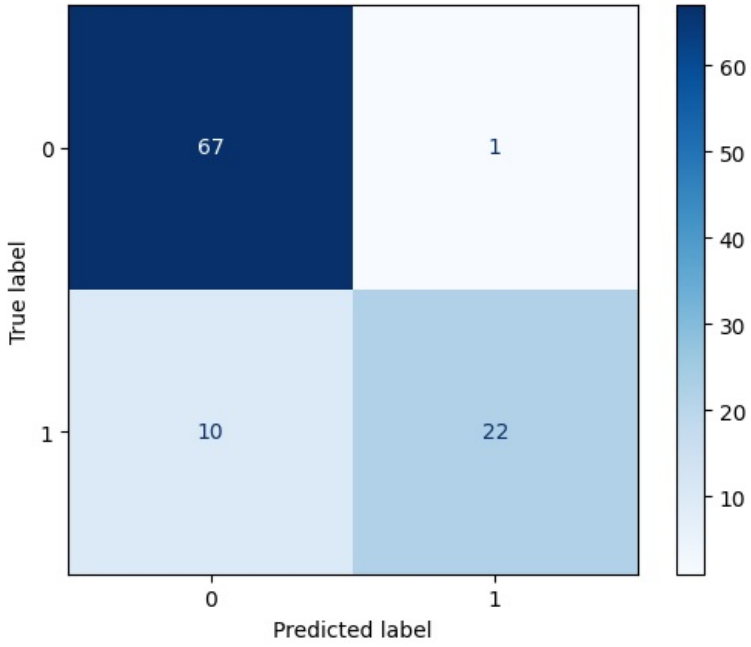
In [32]:
```
model = LogisticRegression()
model.fit(x_train, y_train)
```

Out[32]:
```
LogisticRegression()
```

In [33]:
```
y_pred = model.predict(x_test)
cm = confusion_matrix(y_test, y_pred)
```

In [34]:
```
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=model.classes_)
disp.plot(cmap='Blues')
```

Out[34]:
```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x2777ab1dc10>
```



In [35]:
```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

Out[35]: 0.89

In [36]:
```
from sklearn.metrics import classification_report
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.87      0.99      0.92        68
           1       0.96      0.69      0.80        32

    accuracy                           0.89       100
   macro avg       0.91      0.84      0.86       100
weighted avg       0.90      0.89      0.88       100
```

In [37]:
```
new1=[[26,34000]]
new2=[[57,138000]]
```

In [38]:
```
classifier.predict(scaler.transform(new1))
```

```
C:\Users\Apurva\AppData\Local\Programs\Python\Python312\Lib\site-packages\sklearn\utils\validation.py:2739: User
Warning: X does not have valid feature names, but MinMaxScaler was fitted with feature names
  warnings.warn(
```

Out[38]: array([0], dtype=int64)

In [39]:
```
classifier.predict(scaler.transform(new2))
```

```
C:\Users\Apurva\AppData\Local\Programs\Python\Python312\Lib\site-packages\sklearn\utils\validation.py:2739: User
Warning: X does not have valid feature names, but MinMaxScaler was fitted with feature names
  warnings.warn(
```

Out[39]: array([1], dtype=int64)

In [ ]: