In [5]:

```
!pip install scikit-learn
```

```
Requirement already satisfied: scikit-learn in c:\users\mr_pr\appdata\local\progr
ams\python\python313\lib\site-packages (1.7.2)
Requirement already satisfied: numpy>=1.22.0 in c:\users\mr_pr\appdata\local\prog
rams\python\python313\lib\site-packages (from scikit-learn) (2.3.4)
Requirement already satisfied: scipy>=1.8.0 in c:\users\mr_pr\appdata\local\progr
ams\python\python313\lib\site-packages (from scikit-learn) (1.16.3)
Requirement already satisfied: joblib>=1.2.0 in c:\users\mr_pr\appdata\local\prog
rams\python\python313\lib\site-packages (from scikit-learn) (1.5.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in c:\users\mr_pr\appdata\loc
al\programs\python\python313\lib\site-packages (from scikit-learn) (3.6.0)
```

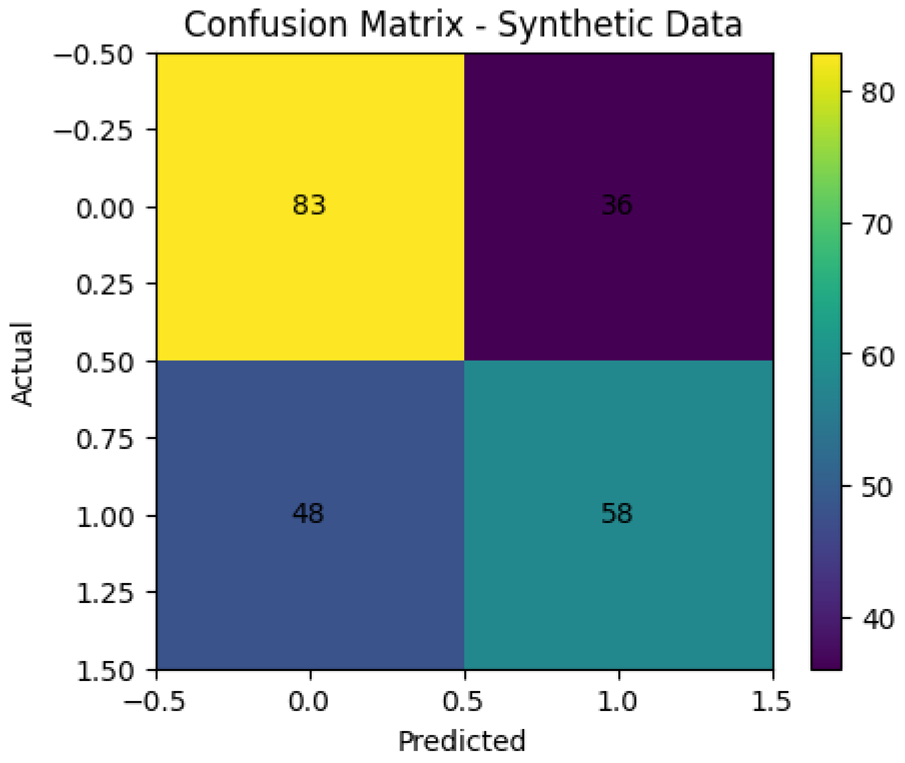
```
[notice] A new release of pip is available: 25.2 -> 25.3
[notice] To update, run: python.exe -m pip install --upgrade pip
```

In [4]:

```
# Logistic Regression Classification on Synthetic Dataset
# Fully unique code - no CSV required

import numpy as np
import pandas as pd
from sklearn.datasets import make_classification
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix, classification_rep
import matplotlib.pyplot as plt

# 1) Generate Synthetic Classification Data
X, y = make_classification(
    n_samples=900,
    n_features=12,
    n_informative=6,
    n_redundant=2,
    random_state=50
)

print("Synthetic Dataset Created!")
print("Shape =", X.shape)

# Convert to DataFrame for clarity
feature_names = [f"Feature_{i}" for i in range(1, 13)]
df = pd.DataFrame(X, columns=feature_names)
df["Target"] = y

print(df.head())

# 2) Split Data
X_train, X_test, y_train, y_test = train_test_split(
    df[feature_names], df["Target"], test_size=0.25, random_state=20
)

# 3) Scaling
sc = StandardScaler()
X_train_s = sc.fit_transform(X_train)
X_test_s = sc.transform(X_test)

# 4) Logistic Regression Model
log_model = LogisticRegression(max_iter=1000)
```

```
log_model.fit(X_train_s, y_train)

# 5) Predictions
pred = log_model.predict(X_test_s)

# 6) Evaluation
acc = accuracy_score(y_test, pred)
print("\nModel Accuracy :", acc)

print("\nClassification Report:\n")
print(classification_report(y_test, pred))

# 7) Confusion Matrix Plot
cm = confusion_matrix(y_test, pred)
plt.figure(figsize=(5,4))
plt.imshow(cm)
plt.title("Confusion Matrix - Synthetic Data")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.colorbar()

for i in range(2):
    for j in range(2):
        plt.text(j, i, cm[i,j], ha="center", va="center")

plt.show()

# 8) Feature Importance
coef_df = pd.DataFrame({
    "Feature": feature_names,
    "Weight": log_model.coef_[0]
})

top_features = coef_df.reindex(coef_df["Weight"].abs().sort_values(ascending=Fal
print("\nTop Important Features:\n")
print(top_features.head(10))
```

```
Synthetic Dataset Created!
Shape = (900, 12)
   Feature_1  Feature_2  Feature_3  Feature_4  Feature_5  Feature_6  \
0  -0.840133   0.102959  -2.143297   0.443025  -3.109692   0.625330   
1   1.082529   0.366530   1.978024   1.576367   0.492825  -0.614997   
2   0.313079   0.406360  -1.493028  -1.290461  -0.545914   1.874460   
3   0.676281  -2.399057  -1.868399  -1.242226  -0.596219   3.946337   
4   0.273239   1.005360  -0.185943  -0.312617   2.907923   1.696554   

   Feature_7  Feature_8  Feature_9  Feature_10  Feature_11  Feature_12  Target  
0  -0.870656  -1.824718  -0.655700    1.298951    0.076265    1.936307       1  
1   0.476142  -1.007300  -1.369090    0.499349   -0.891352   -0.522335       1  
2   0.351657  -0.660898  -0.652426   -1.175338    1.797599    1.123237       1  
3  -1.062198   0.477131  -0.795997   -0.918098    1.934648    1.609816       1  
4   1.024243  -1.908218  -0.369516    0.356602    0.359840   -3.392515       0  

Model Accuracy : 0.6266666666666667

Classification Report:

              precision    recall  f1-score   support

           0       0.63      0.70      0.66       119
           1       0.62      0.55      0.58       106

    accuracy                           0.63       225
   macro avg       0.63      0.62      0.62       225
weighted avg       0.63      0.63      0.62       225
```



```
Top Important Features:

       Feature    Weight
10  Feature_11  0.436506
2    Feature_3 -0.406361
8    Feature_9  0.247584
6    Feature_7  0.204808
3    Feature_4  0.198851
0    Feature_1 -0.145400
5    Feature_6  0.142823
9   Feature_10 -0.123465
4    Feature_5 -0.090665
1    Feature_2  0.062456
```

In [ ]: