```python
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, roc_auc_score, matthews_corrcoef, roc_curve

# Create dataset
np.random.seed(42)
X = np.vstack((
    np.random.normal(0.5, 1.0, (500, 2)),
    np.random.normal(0, 1.0, (500, 2))
))
y = np.array([1] * 500 + [0] * 500)

# Split data and train model
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
clf = LogisticRegression(C=0.1).fit(X_train, y_train)

# Make predictions
y_pred = clf.predict(X_test)
y_proba = clf.predict_proba(X_test)[:, 1]  # Probability for positive class

# Confusion matrix
cm = confusion_matrix(y_test, y_pred)
# Reorder confusion matrix to match: [[TP, FN], [FP, TN]]
cm_adjusted = np.array([
    [cm[1, 1], cm[1, 0]],  # TP, FN
    [cm[0, 1], cm[0, 0]]   # FP, TN
])
TP, FN, FP, TN = cm_adjusted.ravel()

# Calculate metrics
metrics = {
    'Accuracy': (TP + TN) / (TP + TN + FP + FN),
    'Precision': TP / (TP + FP) if (TP + FP) else 0,
    'Recall': TP / (TP + FN) if (TP + FN) else 0,
    'Specificity': TN / (TN + FP) if (TN + FP) else 0,
    'F1 Score': 2 * TP / (2 * TP + FP + FN) if (2 * TP + FP + FN) else 0,
    'MCC': matthews_corrcoef(y_test, y_pred),
    'AUC': roc_auc_score(y_test, y_proba)
}

# Print results
print("Confusion Matrix:\n", cm_adjusted)
print("\nMetrics:")
for metric, value in metrics.items():
    print(f"{metric}: {value:.4f}")

# Create figure with two subplots side by side
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(10, 4))

# Plot 1: Confusion Matrix
cax = ax1.matshow(cm_adjusted, cmap="Pastel2")
ax1.set_title("Confusion Matrix")
ax1.set_xlabel("Predicted")
ax1.set_ylabel("Actual")
for i in range(2):
    for j in range(2):
        ax1.text(j, i, str(cm_adjusted[i, j]), ha="center", va="center")
fig.colorbar(cax, ax=ax1)

# Plot 2: ROC Curve
fpr, tpr, _ = roc_curve(y_test, y_proba)
ax2.plot(fpr, tpr, color='blue', label=f'AUC = {metrics["AUC"]:.4f}')
ax2.plot([0, 1], [0, 1], color='gray', linestyle='--')
ax2.set_title("ROC Curve")
ax2.set_xlabel("False Positive Rate")
ax2.set_ylabel("True Positive Rate")
ax2.legend()
ax2.grid(True, linestyle='--', alpha=0.7)

plt.tight_layout()
plt.show()
```