

```
## Imports and Data Loading
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix, roc_auc_score, roc_curve
import matplotlib.pyplot as plt
import seaborn as sns
```

```
# Load dataset
df = pd.read_csv("data.csv")
```

```
## Data Preprocessing
```

```
# Drop unnecessary columns
df = df.drop(columns=["id", "Unnamed: 32"])
```

```
# Encode diagnosis column
df["diagnosis"] = df["diagnosis"].map({"M": 1, "B": 0})
```




```
# Split into features and target
X = df.drop(columns=["diagnosis"])
y = df["diagnosis"]
```

```
# Train/test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
# Feature scaling
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

```
## Logistic Regression Model
```

```
# Fit model
model = LogisticRegression(random_state=42)
model.fit(X_train_scaled, y_train)
```

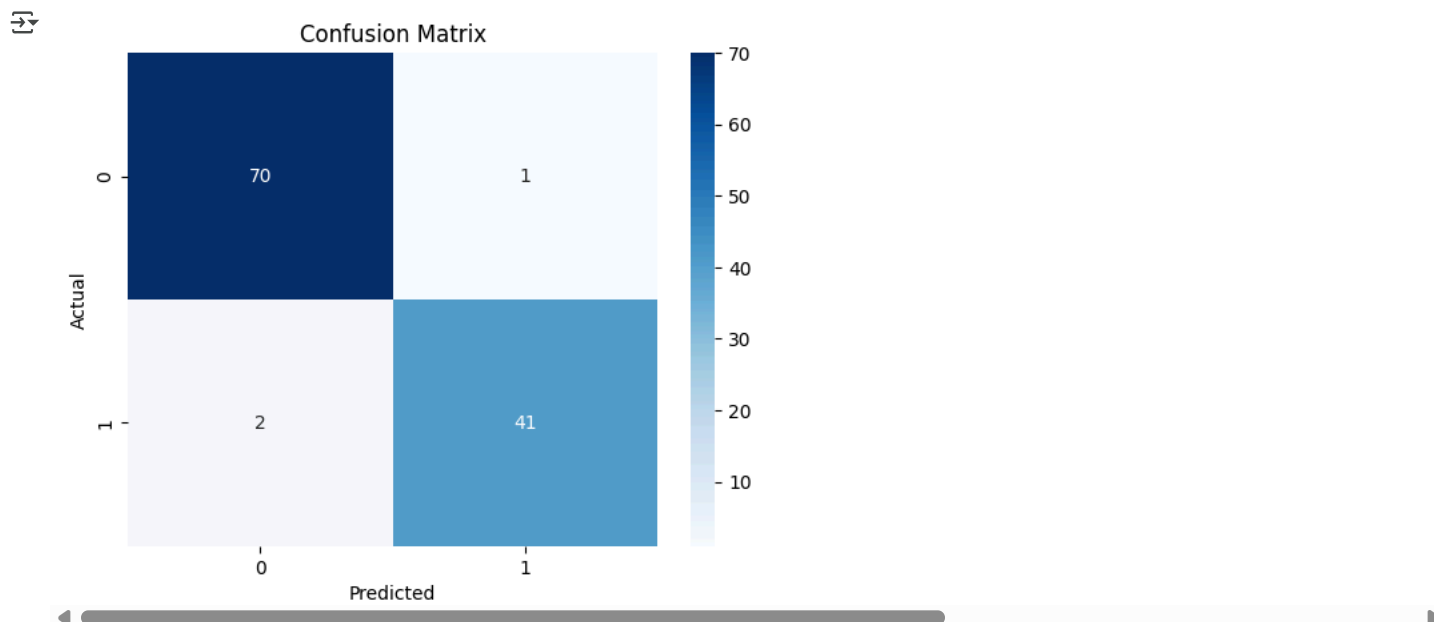
 LogisticRegression    
LogisticRegression(random\_state=42)

```
# Predictions
y_pred = model.predict(X_test_scaled)
y_proba = model.predict_proba(X_test_scaled)[: , 1]
```

```
## Evaluation
```

```
# Confusion Matrix
conf_matrix = confusion_matrix(y_test, y_pred)
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues')
plt.title("Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()
```





```
# Classification Report
print(classification_report(y_test, y_pred))
```

```
# ROC-AUC Score
roc_auc = roc_auc_score(y_test, y_proba)
print("ROC-AUC Score:", roc_auc)
```

```
precision    recall  f1-score   support

0           0.97      0.99      0.98         71
1           0.98      0.95      0.96         43

accuracy          0.97          0.97          0.97        114
macro avg          0.97          0.97          0.97        114
weighted avg          0.97          0.97          0.97        114
```

ROC-AUC Score: 0.99737962659679

```
## ROC Curve and Sigmoid Explanation
```

```
# ROC Curve
fpr, tpr, thresholds = roc_curve(y_test, y_proba)
plt.plot(fpr, tpr, label=f"ROC Curve (AUC = {roc_auc:.2f})")
plt.plot([0, 1], [0, 1], 'k--')
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title("ROC Curve")
plt.legend()
plt.show()
```





ROC Curve



```
# Sigmoid Function
def sigmoid(z):
    return 1 / (1 + np.exp(-z))

z = np.linspace(-10, 10, 100)
plt.plot(z, sigmoid(z))
plt.title("Sigmoid Function")
plt.xlabel("z")
plt.ylabel("sigmoid(z)")
plt.grid(True)
plt.show()
```



Sigmoid Function

