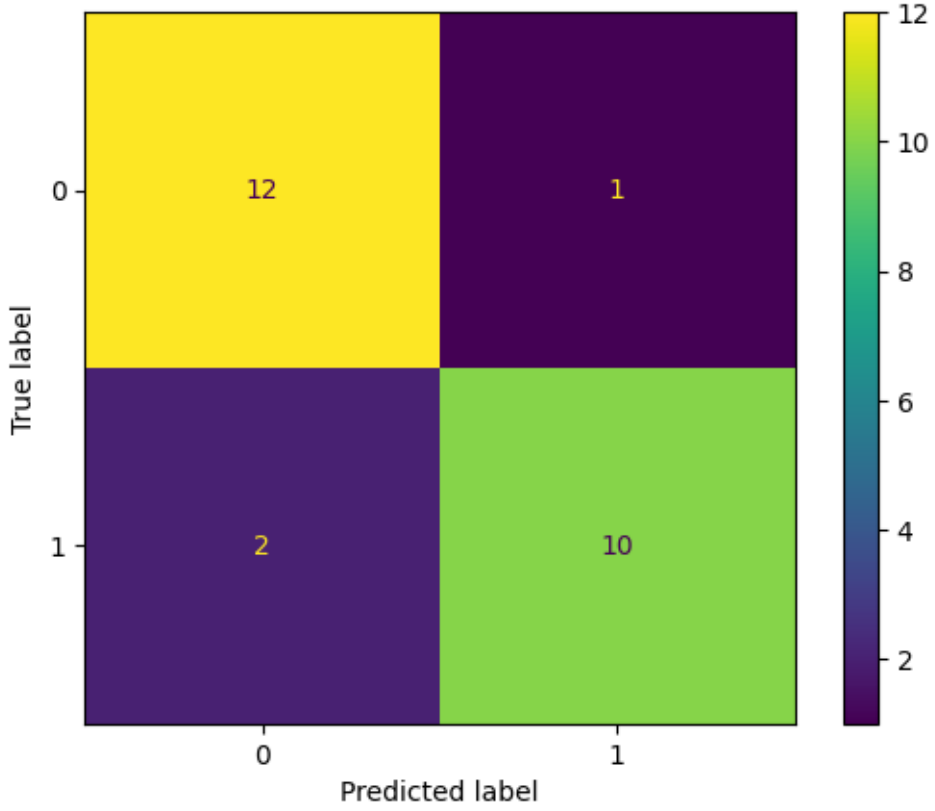
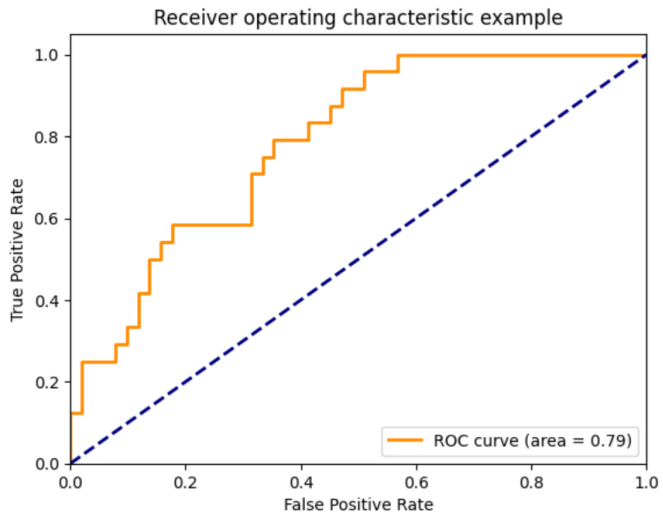
Activity 6

In this activity, we explored different performance evaluation techniques to assess model effectiveness. Metrics such as the F1-score and R-squared were used to measure how well the model balanced precision and recall or explained variance, depending on the task. Additionally, we constructed a confusion matrix to visualise true positives, false positives, true negatives, and false negatives. This helped us understand the practical impact of model errors, especially in sensitive applications where misclassification can have real-world consequences.



Commands + Code + Text Run all



Variables Terminal

Commands + Code + Text Run all

RMSE

```
[14] from sklearn.metrics import mean_squared_error
     y_true = [3, -0.5, 2, 7]
     y_pred = [2.5, 0.0, 2, 8]
     mean_squared_error(y_true, y_pred)
```

```
0.375
```

MAE

```
[15] from sklearn.metrics import mean_absolute_error
     y_true = [3, -0.5, 2, 7]
     y_pred = [2.5, 0.0, 2, 8]
     mean_absolute_error(y_true, y_pred)
```

```
0.5
```

r squared

Variables Terminal

r squared

```
[16] from sklearn.metrics import r2_score

     r2_score(y_true, y_pred)
```

```
0.9486081370449679
```

+ Code + Text

[8]
```
from sklearn.metrics import classification_report
y_true = [0, 1, 2, 2, 2]
y_pred = [0, 0, 2, 2, 1]
target_names = ['class 0', 'class 1', 'class 2']
print(classification_report(y_true, y_pred, target_names=target_names))
```

```
              precision    recall  f1-score   support

     class 0       0.50      1.00      0.67         1
     class 1       0.00      0.00      0.00         1
     class 2       1.00      0.67      0.80         3

    accuracy                           0.60         5
   macro avg       0.50      0.56      0.49         5
weighted avg       0.70      0.60      0.61         5
```

[9]
```
from sklearn.datasets import load_breast_cancer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import roc_auc_score
X, y = load_breast_cancer(return_X_y=True)
clf = LogisticRegression(solver="liblinear", random_state=0).fit(X, y)
roc_auc_score(y, clf.predict_proba(X)[:, 1])
```

Variables Terminal