```
[113]
print("Class probabilities for the first 5 samples:", clf.predict_proba(X_
```

```
Class probabilities for the first 5 samples: [[0. 1. 0.]
 [1. 0. 0.]
 [0. 0. 1.]
 [0. 1. 0.]
 [0. 1. 0.]]
```

```
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy of the model:", accuracy)
```

```
Accuracy of the model: 1.0
```

```
[115] accuracy_alternative = clf.score(X_test, y_test)
print("Accuracy of the model (using score method):", accuracy_alternative)
```

```
Accuracy of the model (using score method): 1.0
```

```
df = pd.read_csv('/content/drive/MyDrive/Ai/heart.csv')
df.head()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
[120] model = LinearRegression()
      model.fit(X_train, y_train)
```

```
LinearRegression
LinearRegression()
```

```
[124] df = pd.read_csv('/content/drive/MyDrive/Ai/gender_submission.csv')
      df.head()
```

	PassengerId	Survived
0	892	0
1	893	1
2	894	0
3	895	0
4	896	1

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
LogisticRegression
LogisticRegression(max_iter=1000)
```

```
[ ]
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning:
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning:
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning:
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

```
print(f"Accuracy: {accuracy}")
print(f"Confusion Matrix:\n{conf_matrix}")
print(f"Classification Report:\n{class_report}")
```

```
Accuracy: 0.5952380952380952
Confusion Matrix:
[[50  0]
 [34  0]]
Classification Report:
              precision    recall  f1-score   support

           0       0.60      1.00      0.75        50
           1       0.00      0.00      0.00        34

    accuracy                           0.60        84
   macro avg       0.30      0.50      0.37        84
weighted avg       0.35      0.60      0.44        84
```