logisticregressionnnn-1

November 18, 2025

[1]:
```
import numpy as np
```

[2]:
```
import pandas as pd
```

[34]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

[35]:
```
from sklearn.datasets import make_classification
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report,␣
 ↪confusion_matrix
```

[58]:
```
data_X, data_y =␣
 ↪make_classification(n_samples=1000,n_features=5,n_informative=3,
                                        n_redundant=0,random_state=42)
```

[37]:
```
cols = ["feature_1", "feature_2", "feature_3", "feature_4", "feature_5"]
df = pd.DataFrame(data_X, columns=cols)
df["target"] = data_y
```

[38]:
```
df.head()
```

[38]:

	feature_1	feature_2	feature_3	feature_4	feature_5	target
0	-0.529332	-0.093387	-1.526572	0.406847	-0.619699	0
1	-0.978500	-1.690672	1.229308	-0.703071	0.202055	1
2	-2.171571	0.545787	1.253433	1.527726	1.780785	1
3	-0.151299	-0.365506	1.335714	0.038355	-0.005317	1
4	-0.777371	1.146030	-2.479343	0.297014	1.518522	0

[39]:
```
inputs = df.iloc[:, :-1]
labels = df.iloc[:, -1]
```

[40]:
```
X_train, X_test, y_train, y_test = train_test_split(inputs, labels, test_size=0.
 ↪2, random_state=42)
```

[41]:
```
log_reg = LogisticRegression(max_iter=1000)
log_reg.fit(X_train, y_train)
```

[41]:
```
LogisticRegression(max_iter=1000)
```

[42]:
```
test_pred = log_reg.predict(X_test)
```

[43]:
```
print("\nClassification Report:\n", classification_report(y_test, test_pred))
```

```
Classification Report:
              precision    recall  f1-score   support

           0       0.91      0.93      0.92        92
           1       0.94      0.93      0.93       108

    accuracy                           0.93       200
   macro avg       0.93      0.93      0.93       200
weighted avg       0.93      0.93      0.93       200
```

[44]:
```
print("\nConfusion Matrix:\n", confusion_matrix(y_test, test_pred))
```

```
Confusion Matrix:
 [[ 86   6]
 [  8 100]]
```

[51]:
```
cm = confusion_matrix(y_test, test_pred)
```

[52]:
```
print("Accuracy:", accuracy_score(y_test, test_pred))
```

```
Accuracy: 0.93
```

[53]:
```
new_sample_df = pd.DataFrame([[0.2, -1.1, 0.5, 1.3, -0.7]],columns=cols)
```

[54]:
```
pred = log_reg.predict(new_sample_df)
prob = log_reg.predict_proba(new_sample_df)
print("Predicted class:", pred[0])
```

```
Predicted class: 1
```

[55]:
```
print("Probabilities:",prob)
```

```
Probabilities: [[0.18510287 0.81489713]]
```

[57]:
```
plt.figure(figsize=(6,4))
sns.heatmap(cm, annot=True, fmt="d")
```

```
plt.title("Confusion Matrix Heatmap")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()
```