Logistic regression is a statistical method used for binary classification tasks, where the dependent variable is categorical and has two possible outcomes (e.g., yes/no, 0/1, true/false). It models the probability that an instance belongs to a particular class based on one or more independent variables. Despite its name, logistic regression is a classification algorithm rather than a regression algorithm.

python

```
# Importing necessary libraries
import numpy as np
import matplotlib.pyplot as plt
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, confusion_matrix

# Generating example data
X, y = make_classification(n_samples=1000, n_features=2, n_classes=2, n_clusters_per_class=1,
random_state=42)

# Splitting the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Creating and fitting the logistic regression model
model = LogisticRegression()
model.fit(X_train, y_train)

# Making predictions
y_pred = model.predict(X_test)

# Calculating accuracy
accuracy = accuracy_score(y_test, y_pred)

# Calculating confusion matrix
cm = confusion_matrix(y_test, y_pred)

# Plotting decision boundary
h = .02
x_min, x_max = X[:, 0].min() - .5, X[:, 0].max() + .5
y_min, y_max = X[:, 1].min() - .5, X[:, 1].max() + .5
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
Z = model.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)

plt.figure(1, figsize=(8, 6))
plt.contourf(xx, yy, Z, cmap=plt.cm.coolwarm, alpha=0.8)
plt.scatter(X[:, 0], X[:, 1], c=y, edgecolors='k', cmap=plt.cm.coolwarm)
```

```
plt.xlabel('Feature 1')
plt.ylabel('Feature 2')
plt.title('Logistic Regression Decision Boundary')
plt.show()

# Printing accuracy and confusion matrix
print("Accuracy:", accuracy)
print("Confusion Matrix:\n", cm)
```

In this example:

We generate synthetic data using make_classification from scikit-learn, with two features and two classes.

We split the data into training and testing sets using train_test_split.

We create an instance of the LogisticRegression class, fit the model to the training data, and make predictions on the testing data.

We calculate the accuracy of the model and the confusion matrix to evaluate its performance.

Finally, we plot the decision boundary to visualize how the logistic regression model separates the classes.