In [1]:
```
import numpy as np
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn import model_selection
from sklearn.datasets import make_classification

class AdaBoostClassifier:
    def __init__(self, n_estimators=50):
        self.n_estimators = n_estimators
        self.alphas = []
        self.models = []

    def fit(self, x, y):
        weights = np.ones(len(x)) / len(x)
        for _ in range(self.n_estimators):
            model = DecisionTreeClassifier(max_depth=1)
            model.fit(x, y, sample_weight=weights)

            predictions = model.predict(x)

            error = np.sum(weights[predictions != y])
            alpha = 0.5 * np.log((1.0 - error) / error)

            self.alphas.append(alpha)
            self.models.append(model)

            # 更新权重
            weighted_error = np.exp(-alpha * y * predictions)
            weights *= weighted_error
            weights /= np.sum(weights)

    def predict(self, x):
        predictions = np.array([(model.predict(x) * alpha) for model, alpha in z
        weights_sum = np.sum(predictions, axis=0)
        return np.sign(weights_sum).astype(int)

if __name__ == "__main__":
    x, y = make_classification(n_samples=100, n_features=2, n_informative=2, n_r

    x_train, x_test, y_train, y_test = model_selection.train_test_split(x, y, ra

    adaboost = AdaBoostClassifier(n_estimators=50)
    adaboost.fit(x_train, y_train)

    y_pred = adaboost.predict(x_test)
    accuracy = accuracy_score(y_test, y_pred)
    print(accuracy)
```

```
0.95
```