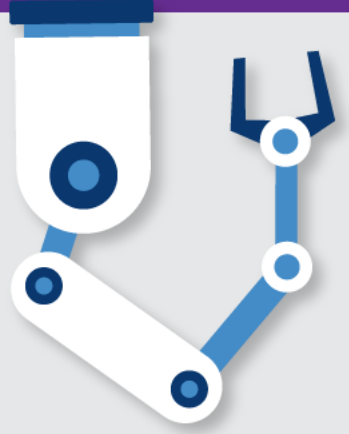
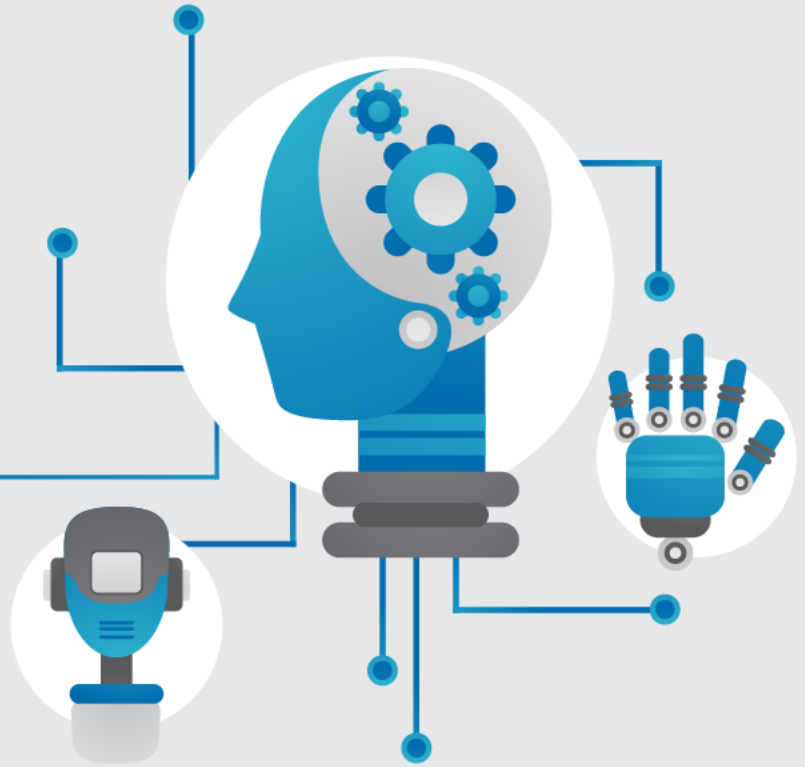


Scikit-learn Kneighbors Classifier



KNeighbors Classifier 介紹

- **K-近鄰演算法（k-Nearest Neighbor, KNN）**
  最早是由 N.S. Altman 於1992年所提出，是一種用於**分類和迴歸**的無母數統計方法。
- **sklearn.neighbors.KNeighborsClassifier**
  為K-近鄰演算法的分類實作

KNeighbors Classifier 參數說明

- class sklearn.neighbors.KNeighborsClassifier(n_neighbors=5, weights='uniform', algorithm='auto', leaf_size=30, p=2, metric='minkowski', metric_params=None, n_jobs=None, **kwargs)

- KNeighbors Classifier 類別常用參數
  - n_neighbors
  - weights
  - algorithm
  - metric
  - p
  - n_jobs

參數 n_neighbors

- n_neighbors : int, optional (default = 5)
  - ✓近鄰的數量（K）

參數 weights

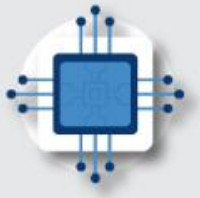
- weights : str or callable, optional (default = ‘uniform’)

  - ✓預測中使用的權重函數
  - ✓‘uniform’ : 統一權重。每個鄰域中的所有點均被加權。
  - ✓‘distance’ : 權重點與其距離的倒數。在這種情況下，查詢點的近鄰比遠處的近鄰具有更大的影響力。
  - ✓[callable] : 用戶定義的函數，輸入距離數組參數，並返回包含權重的相同型態的數組。

參數 algorithm

- algorithm : {‘auto’, ‘ball_tree’ , ‘kd_tree’ , ‘brute’}, optional

  - ✓用於計算最近鄰居的算法
  - ✓ “ ball_tree” : 將使用 BallTree
  - ✓ “ kd_tree” : 將使用 KDTree
  - ✓ “brute” : 將使用暴力搜索
  - ✓ “auto” : 將嘗試根據傳遞給fit方法的值，來決定最合適的算法



參數 metric

- metric : string or callable, default ‘minkowski’

  - ✓使用的距離度量
  - ✓預設是 minkowski
  - ✓有關可用度量的列表，請參見 DistanceMetric 類的文檔

參數 p

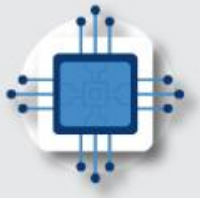
- p : integer, optional (default = 2)

  - ✓Minkowski 指標的功率參數
  - ✓當p = 1時，這等效於使用曼哈頓距離（l1）
  - ✓當p = 2時，這等效於使用歐氏距離（l2）

KNeighbors Classifier 函式說明

› KNeighbors Classifier 常用函式

- fit
- predict
- score



訓練（fit）

- 指令：fit(self, X, y)
- 參數
  - x : 訓練數據
  - y : 目標值（分類中的類標籤）
- 說明：使用X作為訓練數據和y作為目標值擬合模型
- 範例程式

```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=3)
knn.fit(X_train, y_train)
```

預測（predict）

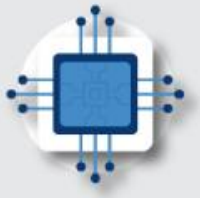
- 指令：predict(self, X)
- 參數
  - x : 測試數據
- 回傳：測試數據X的預測結果
- 範例程式

```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=3)
knn.fit(X_train, y_train)
predictions = knn.predict(x_test)
```

評分（score）

- 指令：score(self, X, y[, sample_weight])
- 參數
  - x : 測試樣本
  - y : 測試樣本的正確答案
- 回傳：測試樣本的平均準確度
- 範例程式

```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=3)
knn.fit(X_train, y_train)
accuracy = knn.score(x_test, y_test)
```



程式範例（IRIS）

程式碼

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.neighbors import KNeighborsClassifier
from sklearn import datasets

# 載入資料
iris = datasets.load_iris()
X = iris.data[:, :2] # 只取前兩種特徵
Y = iris.target

# 建立 k-nearest neighbors
knn = KNeighborsClassifier(n_neighbors=3)

# 進行訓練
knn.fit(X, Y)

# 繪製座標軸
x_min, x_max = X[:, 0].min() - .5, X[:, 0].max() + .5
y_min, y_max = X[:, 1].min() - .5, X[:, 1].max() + .5
h = .02  # 單位間隔
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
```

程式範例（IRIS）

程式碼

```
# 進行預測
Z = knn.predict(np.c_[xx.ravel(), yy.ravel()])

# 繪製預測結果
Z = Z.reshape(xx.shape)
plt.figure(1, figsize=(4, 3))
plt.pcolormesh(xx, yy, Z, cmap=plt.cm.Paired)

plt.scatter(X[:, 0], X[:, 1], c=Y, edgecolors='k',
cmap=plt.cm.Paired)
plt.xlabel('Sepal length')
plt.ylabel('Sepal width')

plt.xlim(xx.min(), xx.max())
plt.ylim(yy.min(), yy.max())
plt.xticks(())
plt.yticks(())

plt.show()
```

程式範例（IRIS）

- 輸出結果