Profundiza más

Recurso de Profundización

Ejemplo de implementación de k–vecinos más cercanos en Python:

```python
# Importamos las librerías necesarias
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.datasets import load_iris
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score

# Cargar un conjunto de datos (en este caso, el conjunto de datos Iris)
data = load_iris()
X = data.data  # Características (features)
y = data.target  # Etiquetas (labels)

# Dividir el conjunto de datos en entrenamiento y prueba
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Crear el modelo de KNN con k=3
knn = KNeighborsClassifier(n_neighbors=3)

# Entrenar el modelo con los datos de entrenamiento
knn.fit(X_train, y_train)

# Hacer predicciones sobre los datos de prueba
y_pred = knn.predict(X_test)

# Evaluar el modelo con la precisión (accuracy)
accuracy = accuracy_score(y_test, y_pred)
print(f'Precisión del modelo KNN: {accuracy * 100:.2f}%')

# Imprimir algunas predicciones junto con las verdaderas etiquetas
print("Predicciones:", y_pred[:10])
print("Etiquetas reales:", y_test[:10])
```

Explicación:

1. **Carga de datos**: usamos el conjunto de datos Iris de *Scikit-learn*, que es un conjunto de datos común para problemas de clasificación. Contiene medidas de flores (características) y sus respectivas especies (etiquetas).
2. **División de datos**: se divide el conjunto de datos en un 70 % para entrenamiento y un 30 % para prueba utilizando train_test_split.
3. **Creación del modelo**: creamos el clasificador KNN con n_neighbors=3, lo que significa que el modelo usará los 3 vecinos más cercanos para hacer predicciones.

4. **Entrenamiento del modelo**: usamos el método .fit() para entrenar el modelo con los datos de entrenamiento.
5. **Predicción y evaluación**: después de entrenar, se hace la predicción con los datos de prueba usando .predict(). La precisión del modelo se evalúa con accuracy_score, que compara las predicciones con las etiquetas reales.

Este es un ejemplo básico para entender cómo funciona el algoritmo KNN en la clasificación. Puedes ajustarlo según tus necesidades, cambiando el valor de n_neighbors o utilizando diferentes conjuntos de datos.