1205ED_2_

Predicción de Enfermedades Cardíacas con KNN y Árboles de Decisión

Dataset base

Kaggle: Heart Disease UCI Dataset

Archivo recomendado: `heart.csv`

Github:
`https://github.com/mihifidem/1205ED_2_enfermedades-corazon-knn-arboles-de-decision.git`

Objetivo del ejercicio

Entrenar y comparar modelos de **KNN** y **DecisionTreeClassifier** para **predecir si un paciente tiene enfermedad cardíaca (target = 1)** a partir de factores clínicos.

Parte 1: Carga y exploración de datos

```
import pandas as pd

# 1️⃣ Cargar el dataset
data = pd.read_csv("heart.csv")

# 2️⃣ Visualizar primeras filas
print(data.head())
print("\nInformación del dataset:")
print(data.info())
print("\nResumen estadístico:")
print(data.describe())
```

Tareas:

1. Identifica cuántas variables hay y de qué tipo (numéricas/categóricas).
2. Determina qué variable es el **objetivo de predicción**.
3. Observa si hay valores atípicos o nulos.

1205ED_2_

Predicción de Enfermedades Cardíacas con KNN y Árboles de Decisión

Parte 2: Preprocesamiento y división de datos

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

# Variables predictoras y objetivo
X = data.drop('target', axis=1)
y = data['target']

# División en entrenamiento y test
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

# Escalado (recomendado para KNN)
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

Tareas:

- Explica por qué se aplica `StandardScaler()` solo a las variables numéricas.
- Comprueba la forma de los conjuntos `X_train` y `X_test`.

1205ED_2_

Predicción de Enfermedades Cardíacas con KNN y Árboles de Decisión

Parte 3: Entrenamiento de modelos

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix

# 1️⃣ Modelo KNN
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train_scaled, y_train)
y_pred_knn = knn.predict(X_test_scaled)

# 2️⃣ Árbol de decisión
tree = DecisionTreeClassifier(max_depth=4, random_state=42)
tree.fit(X_train, y_train)  # No requiere escalado
y_pred_tree = tree.predict(X_test)

# Evaluación
print("=== KNN ===")
print(classification_report(y_test, y_pred_knn))
print("Matriz de confusión:\n", confusion_matrix(y_test,
y_pred_knn))

print("\n=== Árbol de Decisión ===")
print(classification_report(y_test, y_pred_tree))
print("Matriz de confusión:\n", confusion_matrix(y_test,
y_pred_tree))
```

Tareas:

- Compara la precisión (`accuracy`) y el `recall` de ambos modelos.
- ¿Qué modelo parece más estable entre entrenamiento y test?

1205ED_2_

Predicción de Enfermedades Cardíacas con KNN y Árboles de Decisión

Parte 4: Optimización con GridSearchCV

```
from sklearn.model_selection import GridSearchCV

# KNN
param_knn = {'n_neighbors': range(1, 31), 'weights': ['uniform',
'distance']}
grid_knn = GridSearchCV(KNeighborsClassifier(), param_knn, cv=5,
scoring='accuracy')
grid_knn.fit(X_train_scaled, y_train)

print("🔹 Mejor configuración KNN:", grid_knn.best_params_)

# Árbol
param_tree = {
    'criterion': ['gini', 'entropy'],
    'max_depth': [3, 4, 5, 6, 8, None],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}
grid_tree = GridSearchCV(DecisionTreeClassifier(random_state=42),
param_tree, cv=5, scoring='accuracy')
grid_tree.fit(X_train, y_train)

print("🔹 Mejor configuración Árbol:", grid_tree.best_params_)
```

Tareas:

- Compara los parámetros óptimos de ambos modelos.
- ¿Qué hiperparámetro tiene más impacto en cada modelo (`n_neighbors` vs `max_depth`)?

1205ED_2_

Predicción de Enfermedades Cardíacas con KNN y Árboles de Decisión

Parte 5: Visualización e interpretación

```
import matplotlib.pyplot as plt
from sklearn.tree import plot_tree

# Visualizar árbol final
best_tree = grid_tree.best_estimator_

plt.figure(figsize=(16,8))
plot_tree(best_tree, feature_names=X.columns, class_names=['Sano',
'Enfermo'], filled=True)
plt.title("Árbol de Decisión Óptimo - Dataset Heart")
plt.show()
```

Tareas:

1. Identifica qué variables tienen más peso en las decisiones del árbol.
2. Explica cómo interpretas una rama del árbol.

1205ED_2_

Predicción de Enfermedades Cardíacas con KNN y Árboles de Decisión

Parte 6: Comparación final y conclusiones

```
# Precisión final
acc_knn = accuracy_score(y_test,
grid_knn.best_estimator_.predict(X_test_scaled))
acc_tree = accuracy_score(y_test, best_tree.predict(X_test))

print(f"🔹 Precisión KNN: {acc_knn:.4f}")
print(f"🔹 Precisión Árbol: {acc_tree:.4f}")
```

Conclusiones esperadas:

- El modelo **Árbol de Decisión** suele ser más **interpretativo**, mostrando reglas clínicas (edad, presión, colesterol…).
- El **KNN** puede alcanzar precisión similar, pero es **más sensible al escalado y al ruido**.
- Ambos modelos complementan distintas perspectivas:
    - Árbol = **transparencia y reglas claras**
    - KNN = **similitud basada en los pacientes más parecidos**

1205ED_2_

Predicción de Enfermedades Cardíacas con KNN y Árboles de Decisión

Competencias que se practican

Área	Habilidad
ML clásico	KNN y Árbol de Decisión
Preprocesamiento	Escalado y codificación
Evaluación	Métricas, matriz de confusión
Optimización	GridSearchCV
Interpretabilidad	Visualización de árboles y reglas