**Задачи за Decision Trees, Random Forests и k-NN**

⸻

**Задача 1: Класификация на цветя с Decision Tree**

**Имате данни за различни цветя (примерно от набора данни Iris), съдържащи информация за дължина и ширина на листенцата. Целта ви е да използвате Decision Tree Classifier, за да класифицирате вида на цветето.**

**Данни (примерни):**

from sklearn.datasets import load\_iris

from sklearn.tree import DecisionTreeClassifier

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import accuracy\_score

# Зареждаме данните

iris = load\_iris()

X = iris.data # Характеристики на цветята

y = iris.target # Вид на цветето

# Разделяне на тренировъчни и тестови данни

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Създаване и обучение на Decision Tree модел

dt\_model = DecisionTreeClassifier()

dt\_model.fit(X\_train, y\_train)

# Тестване на модела

y\_pred = dt\_model.predict(X\_test)

accuracy = accuracy\_score(y\_test, y\_pred)

print(f"Точност на модела: {accuracy:.2f}")

**Вашите задачи:**

1. Използвайте различни стойности за max\_depth, за да видите как се променя точността на модела.

2. Визуализирайте дървото на решенията с plot\_tree() от sklearn.tree.

3. Опитайте се да предскажете вида на цвете с характеристики [5.0, 3.5, 1.3, 0.3].

**Задача 2: Прогнозиране на цената на къща с Random Forest**

**Имате набор от данни с различни характеристики на къщи, като брой стаи, квадратура и възраст на имота. Целта ви е да използвате Random Forest Regressor, за да предскажете цената на къща.**

**Данни (примерни):**

import numpy as np

import matplotlib.pyplot as plt

from sklearn.ensemble import RandomForestRegressor

from sklearn.model\_selection import train\_test\_split

# Симулирани данни: квадратура (X) и цена на къщата (y)

X = np.array([50, 60, 75, 80, 100, 120, 150, 200, 250, 300]).reshape(-1, 1)

y = np.array([100000, 120000, 140000, 160000, 180000, 210000, 250000, 310000, 370000, 450000])

# Разделяне на данните

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Създаване и обучение на Random Forest модел

rf\_model = RandomForestRegressor(n\_estimators=100, random\_state=42)

rf\_model.fit(X\_train, y\_train)

# Прогнозиране на цената за къща с 180 кв.м

predicted\_price = rf\_model.predict([[180]])

print(f"Прогнозна цена за 180 кв.м: {predicted\_price[0]:.2f} BGN")

**Вашите задачи:**

1. Опитайте различни стойности за n\_estimators (брой дървета) и анализирайте как влияе на точността.

2. Визуализирайте прогнозите на Random Forest спрямо реалните данни.

3. Предскажете цената на къща с 220 кв.м и сравнете с прогнозата за 180 кв.м.

**Задача 3: Класификация на ръчно написани цифри с k-NN**

**Ще използваме известния набор от данни MNIST, който съдържа изображения на ръчно написани цифри (0-9). Целта е да изградите k-NN класификатор, който разпознава цифрите.**

**Данни и код:**

from sklearn.datasets import load\_digits

from sklearn.neighbors import KNeighborsClassifier

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import accuracy\_score

import matplotlib.pyplot as plt

# Зареждане на данните

digits = load\_digits()

X = digits.data # Векторизирани изображения (8x8 пиксела)

y = digits.target # Цифри (0-9)

# Разделяне на данните

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Създаване и обучение на k-NN модел

knn\_model = KNeighborsClassifier(n\_neighbors=3)

knn\_model.fit(X\_train, y\_train)

# Тестване на модела

y\_pred = knn\_model.predict(X\_test)

accuracy = accuracy\_score(y\_test, y\_pred)

print(f"Точност на k-NN модела: {accuracy:.2f}")

# Визуализиране на някои от изображенията

fig, axes = plt.subplots(1, 5, figsize=(10, 3))

for i, ax in enumerate(axes):

ax.imshow(X\_test[i].reshape(8, 8), cmap='gray')

ax.set\_title(f"Прогноза: {y\_pred[i]}")

ax.axis('off')

plt.show()

**Вашите задачи:**

1. Променете броя на съседите (n\_neighbors) и вижте как се променя точността на модела.

2. Визуализирайте още изображения и проверете дали моделът греши.