**تمرین 1: دسته‌بندی دو بعدی با KNN**

فرض کنید داده‌های زیر را دارید:

X = [[2, 4], [5, 6], [3, 3], [8, 9], [7, 8]]

y = ['A', 'B', 'A', 'B', 'B']

* هدف: با استفاده از الگوریتم KNN و k=3، نقطه‌ی جدید [6, 5] را دسته‌بندی کنید.
* توضیح: فاصله‌ی اقلیدسی را برای محاسبه‌ی فاصله‌ها استفاده کنید.

**تمرین 2: پیاده‌سازی دستی KNN**

با استفاده از زبان برنامه‌نویسی Python، الگوریتم KNN را به صورت دستی پیاده‌سازی کنید. برای این کار:

1. یک تابع بنویسید که فاصله‌ی اقلیدسی بین دو نقطه را محاسبه کند.
2. یک تابع بنویسید که k نزدیک‌ترین همسایه را پیدا کند.
3. یک تابع بنویسید که بر اساس دسته‌های k همسایه‌ی نزدیک، دسته‌بندی نهایی را انجام دهد.

* تست: داده‌های زیر را به عنوان ورودی استفاده کنید:

X\_train = [[1, 2], [2, 3], [3, 1], [6, 5], [7, 7], [8, 6]]

y\_train = ['red', 'red', 'red', 'blue', 'blue', 'blue']

X\_test = [5, 4]

k = 3

**تمرین 3: تنظیم مقدار k**

داده‌های زیر را در نظر بگیرید:

1

2

X = [[1, 1], [2, 2], [3, 3], [6, 6], [7, 7], [8, 8]]

y = ['A', 'A', 'A', 'B', 'B', 'B']

* هدف: نقطه‌ی [5, 5] را با مقادیر مختلف k (مانند k=1, k=3, k=5) دسته‌بندی کنید.
* سوال: آیا تغییر مقدار k روی نتیجه تأثیر می‌گذارد؟ چرا؟

تمرین 4: استفاده از KNN در مجموعه داده Iris

مجموعه داده‌ی معروف Iris را از کتابخانه‌ی sklearn.datasets بارگذاری کنید. سپس:

داده‌ها را به دو بخش آموزش (train) و آزمون (test) تقسیم کنید.

الگوریتم KNN را با استفاده از کتابخانه‌ی sklearn.neighbors.KNeighborsClassifier پیاده‌سازی کنید.

دقت (Accuracy) مدل را برای مقادیر مختلف k (مانند k=1, k=3, k=5, k=7) محاسبه کنید.

بهترین مقدار k را پیدا کنید.

**تمرین 5: KNN برای رگرسیون**

فرض کنید داده‌های زیر را دارید:

X = [[1], [2], [3], [4], [5], [6], [7], [8], [9], [10]]

y = [1.1, 1.9, 3.1, 4.2, 5.1, 5.8, 7.2, 7.9, 9.1, 10.2]

* هدف: با استفاده از KNN برای رگرسیون و k=3، مقدار خروجی را برای ورودی X\_test = [6.5] پیش‌بینی کنید.
* توضیح: برای رگرسیون، میانگین مقادیر k همسایه‌ی نزدیک را به عنوان پیش‌بینی استفاده کنید