**به نام خدا**

**Decision Tree**

خب! حالا که با **مدل‌های خطی** کار کردی، وقتشه سراغ **روش‌های غیرخطی** بریم. امروز قراره با **درخت تصمیم** (Decision Tree) آشنا بشی، یکی از کاربردی‌ترین مدل‌های یادگیری ماشین!

**درخت تصمیم (Decision Tree) چیست؟**

درخت تصمیم یک مدل یادگیری ماشین **غیرپارامتری** و **تابع-محور** است که از ساختار درختی برای تصمیم‌گیری استفاده می‌کند. این الگوریتم هم برای **طبقه‌بندی (Classification)** و هم برای **رگرسیون (Regression)** قابل استفاده است.

* **کاربردها:**
  + تشخیص بیماری‌ها (طبقه‌بندی)
  + پیش‌بینی قیمت خانه (رگرسیون)

**۱. ساختار درخت تصمیم:**

* **ریشه (Root Node)**: نقطه شروع تصمیم‌گیری (بهترین ویژگی برای تقسیم)
* **شاخه‌ها (Branches)**: مسیرهای تصمیم بر اساس مقادیر ویژگی‌ها
* **گره‌های داخلی (Internal Nodes)**: سوالات/شرط‌های میانی
* **برگ‌ها (Leaf Nodes)**: نتایج نهایی (پیش‌بینی)

### ****۲. نحوه کار الگوریتم:****

۱. **انتخاب بهترین ویژگی** برای تقسیم داده‌ها با معیارهای:

* **آنتروپی (Entropy)**: میزان بی‌نظمی داده‌ها  
  Entropy=−∑pilog⁡2(pi)Entropy=−∑*pi*​log2​(*pi*​)
* **شاخص جینی (Gini Index)**: احتمال خطا در طبقه‌بندی  
  Gini=1−∑pi2Gini=1−∑*pi*2​

۲. **تقسیم بازگشتی** داده‌ها تا رسیدن به شرایط توقف:

* تمام نمونه‌ها در یک کلاس باشند
* عمق حداکثر (max\_depth) رسیده باشد
* تعداد نمونه‌ها در گره کمتر از حد آستانه باشد

### ****3. مزایا و معایب:****

| **مزایا** | **معایب** |
| --- | --- |
| تفسیرپذیری بالا (قابل نمایش گرافیکی) | مستعد بیش‌برازش (Overfitting) |
| نیاز به پیش‌پردازش کم (حساسیت کم به مقیاس داده) | حساس به تغییرات کوچک در داده |
| کار با داده‌های عددی و دسته‌ای | عملکرد ضعیف در روابط غیرخطی پیچیده |
| سرعت بالا در پیش‌بینی | نیاز به تنظیم دقیق هیپرپارامترها |

### ****4. هیپرپارامترهای کلیدی:****

* **max\_depth** :حداکثر عمق درخت
* **min\_samples\_split** : حداقل نمونه مورد نیاز برای تقسیم گره
* **min\_samples\_leaf** :حداقل نمونه در گره‌های برگ
* **max\_features** :تعداد ویژگی‌های مورد بررسی در هر تقسیم
* **criterion** :معیار تقسیم (gini/entropy برای طبقه‌بندی، squared\_error برای رگرسیون)

### ****5. کاربردهای واقعی:****

۱. **تشخیص بیماری‌ها** بر اساس علائم بیمار  
۲. **تصمیم‌گیری اعتباری** در بانک‌ها  
۳. **پیش‌بینی قیمت مسکن** بر اساس ویژگی‌های ملک  
۴. **سیستم‌های توصیه‌گر** ساده

### ****6. مقایسه با سایر مدل‌ها:****

| **معیار** | **درخت تصمیم** | **رگرسیون لجستیک** | **SVM** |
| --- | --- | --- | --- |
| تفسیرپذیری | ✅ عالی | ✅ خوب | ❌ ضعیف |
| نیاز به پیش‌پردازش | ❌ کم | ✅ زیاد | ✅ زیاد |
| عملکرد روی داده‌های پیچیده | ❌ متوسط | ✅ خوب | ✅ عالی |

**توضیحات مربوط کد :**

هر خانه با ۸ ویژگی توصیف می‌شود:

1. MedInc: درآمد متوسط ساکنان
2. HouseAge: میانگین سن خانه‌ها
3. AveRooms: میانگین تعداد اتاق‌ها
4. AveBedrms: میانگین تعداد خواب‌ها
5. Population: جمعیت منطقه
6. AveOccup: میانگین تعداد ساکنان هر خانه
7. Latitude: مختصات جغرافیایی
8. Longitude: مختصات جغرافیایی

**تذکرات :**

* اگر R² بین **۰.۶ تا ۰.۸** باشد، مدل قابل قبول است.
* اگر زیر **۰.۵** باشد، نیاز به بازنگری اساسی دارد.

**توضیح کد :**

percentiles = np.percentile(X, [25, 50, 75])

### نحوه محاسبه درصدیل‌ها یا چارک ها ****outlier****

### داده‌های ما:

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

(10 عدد به ترتیب صعودی)

### محاسبه درصدیل‌ها:

تابع  np.percentile  مقدار متناظر با هر درصد را محاسبه می‌کند.

#### **الف) درصدیل 25% (Q1):**

* محل قرارگیری: (25/100) \* (n + 1) = 0.25 \* 11 = 2.75
* مقدار: بین عنصر 2ام و 3ام
* محاسبه: 2 + 0.75\*(3-2) = 2.75

#### **ب) درصدیل 50% (میانه - Q2):**

* محل قرارگیری: 0.5 \* 11 = 5.5
* مقدار: بین عنصر 5ام و 6ام
* محاسبه: 5 + 0.5\*(6-5) = 5.5

#### **ج) درصدیل 75% (Q3):**

* محل قرارگیری: 0.75 \* 11 = 8.25
* مقدار: بین عنصر 8ام و 9ام
* محاسبه: 8 + 0.25\*(9-8) = 8.25

### خروجی نهایی:

print(percentiles) *# [2.75 5.5 8.25]*

### توضیح کد :

   for percentile in percentiles:

        left\_idx = X <= percentile

        right\_idx = X > percentile

### این دستور یک **ماسک بولی (Boolean mask)** ایجاد می‌کند که مشخص می‌کند کدام عناصر آرایه X کوچکتر یا مساوی مقدار درصدیل (percentile) هستند

### .۱. نحوه عملکرد:

* X <= percentile: یک مقایسه عنصر به عنصر انجام می‌دهد
* نتیجه: یک آرایه بولی با همان طول X که:
  + True: اگر عنصر ≤ درصدیل باشد
  + False: اگر عنصر > درصدیل باشد

### ۲. مثال عملی:

فرض کنید:

import numpy as np

X = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10])

percentile = 5.5 *# درصدیل 50% (میانه)*

left\_idx = X <= percentile

print(left\_idx)

### ۳. خروجی:

[ True True True True True False False False False False]

تفسیر:

* 5 عنصر اول (1 تا 5) ≤ 5.5 هستند → True
* 5 عنصر بعدی (6 تا 10) > 5.5 هستند → False

### ۴. کاربرد در درخت تصمیم:

*# تقسیم داده‌ها به دو زیرمجموعه*

left\_data = X[left\_idx] *# مقادیر ≤ درصدیل*

right\_data = X[~left\_idx] *# مقادیر > درصدیل*

print("زیرمجموعه چپ:", left\_data) *# [1 2 3 4 5]*

print("زیرمجموعه راست:", right\_data) *# [6 7 8 9 10]*