Decision Tree Classifier: CART with Gini and Entropy Criteria

1. Pendahuluan

Model *Decision Tree Classifier* yang diimplementasikan merupakan varian dari **CART (Classification and Regression Trees)** yang mengadopsi pemisahan biner terhadap fitur kontinu. Fungsi objektif utama model adalah meminimalkan impuritas di setiap node, dengan dua pilihan fungsi impuritas yang dapat digunakan: **Gini impurity** dan **entropy**.

2. Struktur Pohon dan Representasi Node

Struktur pohon terdiri dari simpul-simpul (*TreeNode*) yang merepresentasikan kondisi pemisahan atau nilai prediksi akhir. Setiap simpul memiliki atribut berikut:

- Fitur pemisah: x_j
- Ambang batas pemisahan: $t \in \mathrm{R}$
- Sub-pohon kiri dan kanan
- Nilai prediksi kelas $\hat{y}$ jika simpul adalah daun
- Impuritas simpul: I
- Jumlah sampel n, dan distribusi kelas lokal

3. Fungsi Impuritas

3.1. Gini Impurity

Diberikan $\mathrm{D} = \{y_1, \ldots, y_n\}$, Gini impurity didefinisikan sebagai:

$$G(\mathrm{D}) = 1 - \sum_{k=1}^{K} p_k^2$$

dengan $p_k = \frac{n_k}{n}$ adalah proporsi kelas ke-k dalam simpul.

3.2. Entropy

Entropy dihitung sebagai:

$$H(\mathrm{D}) = -\sum_{k=1}^{K} p_k \log_2 p_k$$

Kedua fungsi ini digunakan dalam seleksi split terbaik berdasarkan penurunan impuritas.

4. Kriteria Pemilihan Split

Split dipilih untuk memaksimalkan pengurangan impuritas:

$$\Delta I = I_{\text{parent}} - (\frac{n_L}{n} I_L + \frac{n_R}{n} I_R)$$

dengan n_L dan n_R adalah jumlah sampel di subtree kiri dan kanan.

Split hanya valid jika kedua subtree memiliki setidaknya min_samples_leaf sampel, dan penurunan impuritas $\Delta I \geq \text{min\_impurity\_decrease}$.

5. Algoritma Pelatihan

5.1. Pseudocode

```
function build_tree(X, y, depth):
    if stopping_condition_met:
        return leaf_node(most_common_class(y))

    best_feature, best_threshold = find_best_split(X, y)

    if no_valid_split:
        return leaf_node(most_common_class(y))

    split X, y into X_left, y_left and X_right, y_right
    left_subtree = build_tree(X_left, y_left, depth+1)
    right_subtree = build_tree(X_right, y_right, depth+1)

    return TreeNode(best_feature, best_threshold, left_subtree,
right_subtree)
```

5.2. Kondisi Berhenti

- Kedalaman maksimum tercapai: $\text{depth} = \text{max\_depth}$
- Jumlah sampel kurang dari min_samples_split
- Impuritas nol: $I = 0$
- Hanya satu kelas tersisa

6. Prediksi

6.1. Kelas

Untuk prediksi kelas, simpul dilewati secara rekursif berdasarkan:

$$\text{if } x_j \leq t \Rightarrow \text{left}, \quad \text{else} \Rightarrow \text{right}$$

hingga daun tercapai, dan nilai $\hat{y} = \arg\max_k n_k$ dikembalikan.

6.2. Probabilitas

Distribusi probabilitas dikembalikan dari daun:

$$P(y = k \mid x) = \frac{n_k}{\sum_j n_j}$$

dengan n_k jumlah sampel kelas k pada simpul daun tempat x jatuh.