Classification and Regression Tree (CART)

Evelyn Yosiana / 13522083

1. Konsep CART yaitu pembentukan tree dimana di tiap cabangnya akan ada “pertanyaan” yang nantinya akan di-split menjadi “no” dan “yes”. Cara memprediksi suatu data yaitu dengan mengikuti alur “yes” dan “no” tadi sampai ke daun.

   Proses train: untuk proses train, intinya yaitu pembentukan pohon secara rekursif. Untuk tiap percabangan, dicari fitur dengan threshold terbaik (gain paling tinggi), kemudian kumpulan data akan dipecah berdasarkan threshold tadi. Kemudian dari sana akan dibentuk cabang kiri dan cabang kanan berdasarkan hasil splitting data tadi. Proses grow tree akan terus dilakukan sampai mencapai syarat tertentu, yaitu kedalaman pohon sudah mencapai max depth, atau total data yang akan di-split kurang dari min samples.

   Proses predict: untuk proses ini, intinya tiap data akan di-traverse dari satu cabang ke cabang lainnya yang lebih di bawahnya sampai mencapai leaf. Dari leaf tersebut, data yang akan di-predict mendapatkan labelnya. Hal tersebut dilakukan untuk tiap data yang ingin di-predict.

4. Pada model ini, hasil perbandingan model yang saya buat dan model dari library kurang sama. Hal ini dapat disebabkan oleh beberapa faktor, antara lain:
   - **Pemilihan threshold**: dalam kode yang saya buat, pemilihan threshold dilakukan secara manual, sedangkan kode dalam library menggunakan pendekatan yang lebih kompleks, misalnya binning, untuk mendapatkan hasil yang optimal.
   - **Perbedaan pembulatan angka**: perbedaan presisi mungkin akan berpengaruh pada tree yang terbentuk.
   - **Perbedaan penanganan fitur**: kode dari library dapat menangani data kategorikal, null values, dan dapat secara otomatis menstandarisasi data, namun kode saya belum dapat menangani hal ini.

4. Improvement yang dapat saya lakukan antara lain:
   - **Optimasi threshold selection** agar komputasi menjadi lebih ringan dengan waktu kompilasi yang lebih cepat. Hal ini dapat dilakukan dengan metode median split, dibandingkan dengan melakukan brute force.
   - **Melakukan post-pruning dan pre-pruning** untuk menghindari model yang overfitting. Untuk post-pruning, setelah model di-train secara

menyeluruh, beberapa cabang dapat di-pruning. Sedangkan untuk pre-pruning, sebelum pembentukan cabang baru, dapat diberikan syarat tertentu.

- **Optimasi dengan vectorization** (numpy) untuk mempercepat komputasi.
- **Optimasi dengan multi-threading dan parallelization** untuk mempercepat komputasi.
- **Hyperparameter tuning** untuk mendapatkan kombinasi parameter yang optimal (salah satunya dengan menggunakan metode grid search atau library optuna).
- Menggunakan **confusion metrics** untuk model evaluation.

Lampiran

Contoh perhitungan manual:

Evelyn Yosiana / 13522083

CART

$$x_{train} = \begin{bmatrix} 1 & 3 \\ 4 & 2 \\ 5 & 2 \end{bmatrix} \quad y_{train} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

classes = [0, 1]

counts = [2, 1]

probabilities = $\left[\frac{2}{3}, \frac{1}{3}\right]$

gini impurity = $1 - \left(\frac{2}{3}\right)^2 - \left(\frac{1}{3}\right)^2 = \frac{4}{9}$

feature 1

$x_{left} = [1, 3]$ $y_{left} = [0]$ gini left $= 1 - 1^2 - 0^2 = 0$

$x_{right} = [[4, 2], [5, 2]]$ $y_{right} = [1, 0]$ gini right $= 1 - \left(\frac{1}{2}\right)^2 - \left(\frac{1}{2}\right)^2 = \frac{1}{2}$

gini split $= \frac{1}{3}(0) + \frac{2}{3} \cdot \left(\frac{1}{2}\right) = \frac{1}{3}$

gain $= \frac{4}{9} - \frac{1}{3} = \frac{1}{9}$

feature 2

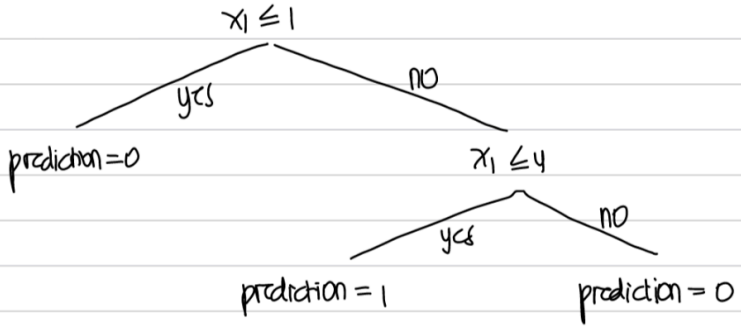
$x_{left} = [[1, 3], [4, 2]]$ $y_{left} = [0, 1]$ gini left $= 1 - \left(\frac{1}{2}\right) - \left(\frac{1}{2}\right)^2 = \frac{1}{2}$

$x_{right} = [5, 2]$ $y_{right} = [0]$ gini right $= 1 - 1^2 - 0^2 = 0$

gini split $= \frac{2}{3}\left(\frac{1}{2}\right) + \frac{1}{3}(0) = \frac{1}{3}$

gain $= \frac{4}{9} - \frac{1}{3} = \frac{1}{9}$

Pilih gain terkecil, karena sama jadi pilih split paling awal aja



predict [3, 7] :

$3 \leq 1$ NO

$3 \leq 4$ YES

prediction = 1