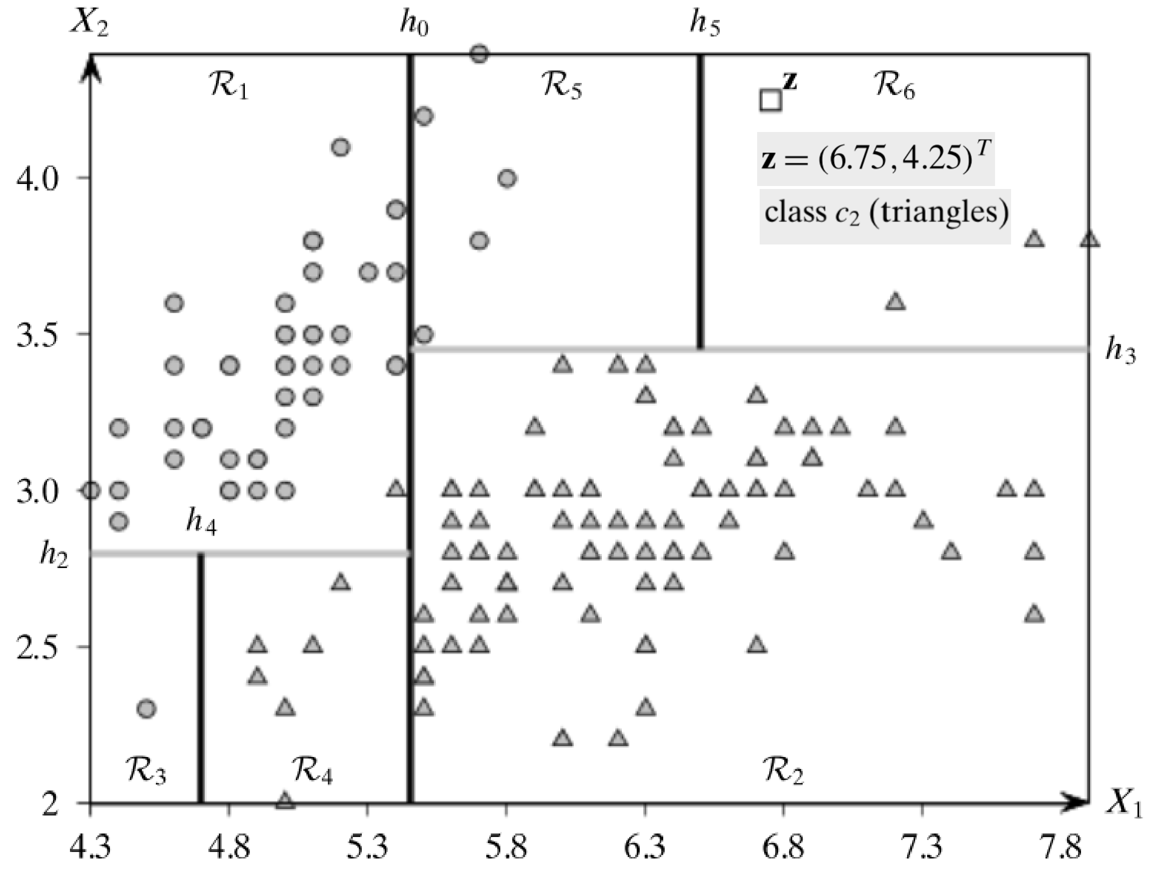
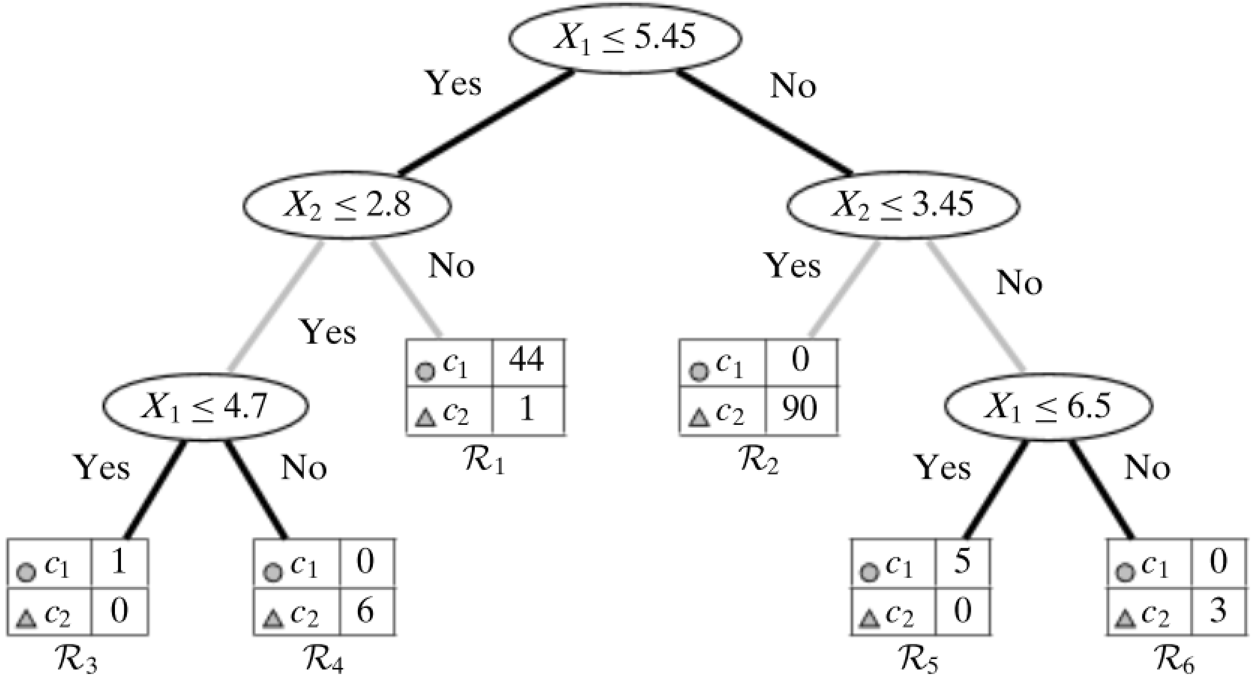
دسته‌بندی‌گر درخت تصمیم

Decision Tree Classifier

هر یک از مناطق $\mathcal{R}_i$ با تقسیم مکرر ابرصفحه‌ها موازی محورها تا زمانی که نقاط درون یک پارتیشن از نظر برچسب دسته خود نسبتاً خالص باشند، یعنی بیشتر نقاط متعلق به یک دسته هستند.



(a) Recursive Splits



(b) Decision Tree

Figure 19.1. Decision trees: recursive partitioning via axis-parallel hyperplanes.

برای نسبت دادن یک نقطه‌ی آزمون $\boldsymbol{x}$ به یک دسته، فقط باید مشخص کنیم به کدام منطقه تعلق دارد و در آن منطقه، دسته‌ی غالب کدام است.

ابرصفحه‌های موازی محور (Axis-Parallel Hyperplanes)

$$\boxed{h(\mathbf{x}): \mathbf{w}^T\mathbf{x} + b = 0} \qquad h(\boldsymbol{x}): \boldsymbol{e}_j^T \boldsymbol{x} + b = x_j + b = 0$$

ابرصفحه‌ی $h(\boldsymbol{x}) = 0$ که عمود بر مشخصه‌ی X_j می‌باشد و دامنه‌ی $\mathcal{R}$ برای مشخصه‌ی X_j را به دو ناحیه $\mathcal{R}_Y$ و $\mathcal{R}_N$ تقسیم می‌کند.
ناحیه‌ی $\mathcal{R}_Y$ نقاطی هستند که در شرط $h(\boldsymbol{x}) \leq 0$ صدق می‌کنند و ناحیه‌ی $\mathcal{R}_N$ نقاط در ناحیه‌ی مقابل هستند.
نقطه‌ی تقسیم (Split Point) ناحیه‌ی $\mathcal{R}$ با انتخاب $v = -b$ مشخص می‌شود. با این تقسیم مجموعه‌داده‌های $\boldsymbol{D}$ نیز به دو زیر مجموعه‌ی $\boldsymbol{D}_Y$ و $\boldsymbol{D}_N$ بنابر این که شرط $X_j \leq v$ را ارضا نمایند و یا خیر، تقسیم می‌شوند.

$$\mathbf{D}_Y = \left\{\mathbf{x}^T \mid \mathbf{x} \in \mathbf{D}, x_j \leq v\right\}$$

$$\mathbf{D}_N = \left\{\mathbf{x}^T \mid \mathbf{x} \in \mathbf{D}, x_j > v\right\}$$

برای **مشخصه‌های دسته‌ای (Categorical Attributes)**، زیردامنه‌ای از X_j به شکل $V \subset dom(X_j)$ برای تقسیم ناحیه $\mathcal{R}$ انتخاب می‌شود.
ناحیه‌ی $\mathcal{R}_Y$ نقاطی از مجموعه داده‌های $\boldsymbol{D}$ هستند که $x_i \in V$

خلوص (Purity) برای ناحیه‌ی $\mathcal{R}_j$ نسبت داده‌های دسته‌ای غالب در ناحیه به کل داده‌های موجود در آن ناحیه است.

$$purity(\boldsymbol{D}_j) = \max_i \left\{\frac{n_{ji}}{n_j}\right\}$$

که در آن $n_j = |\boldsymbol{D}_j|$ تعداد داده‌ها در ناحیه‌ی $\mathcal{R}_j$ و n_{ji} تعداد نقاط در این ناحیه با برچسب دسته‌ی c_i می‌باشد.

الگوریتم درخت تصمیم (Decision Tree Algorithm)

Algorithm 19.1: Decision Tree Algorithm

DecisionTree $(\mathbf{D}, \eta, \pi)$:

$n \leftarrow |\mathbf{D}|$ `// partition size`

$n_i \leftarrow |\{\mathbf{x}_j | \mathbf{x}_j \in \mathbf{D}, y_j = c_i\}|$ `// size of class` c_i

$purity(\mathbf{D}) \leftarrow \max_i \left\{\frac{n_i}{n}\right\}$

if $n \leq \eta$ *or* $purity(\mathbf{D}) \geq \pi$ **then** `// stopping condition`

- $c^* \leftarrow \arg\max_{c_i} \left\{\frac{n_i}{n}\right\}$ `// majority class`
- create leaf node, and label it with class c^*
- **return**

$(split\ point^*, score^*) \leftarrow (\emptyset, 0)$ `// initialize best split point`

foreach *(attribute* X_j*)* **do**

- **if** *(*X_j *is numeric)* **then**
  - $(v, score) \leftarrow$ Evaluate-Numeric-Attribute$(\mathbf{D}, X_j)$
  - **if** $score > score^*$ **then** $(split\ point^*, score^*) \leftarrow (X_j \leq v, score)$
- **else if** *(*X_j *is categorical)* **then**
  - $(V, score) \leftarrow$ Evaluate-Categorical-Attribute$(\mathbf{D}, X_j)$
  - **if** $score > score^*$ **then** $(split\ point^*, score^*) \leftarrow (X_j \in V, score)$

`// partition` $\mathbf{D}$ `into` $\mathbf{D}_Y$ `and` $\mathbf{D}_N$ `using` *split point**`, and call recursively`

$\mathbf{D}_Y \leftarrow \{\mathbf{x}^T \mid \mathbf{x} \in \mathbf{D}$ satisfies *split point*$^*\}$

$\mathbf{D}_N \leftarrow \{\mathbf{x}^T \mid \mathbf{x} \in \mathbf{D}$ does not satisfy *split point*$^*\}$

create internal node *split point**, with two child nodes, $\mathbf{D}_Y$ and $\mathbf{D}_N$

DecisionTree$(\mathbf{D}_Y)$; DecisionTree$(\mathbf{D}_N)$

شرط توقف رسیدن به حداکثر خلوص π و یا حداقل تعداد نقاط η می‌باشد.
مقادیر π و η از پیش مشخص می‌شوند.

معیار انتخاب نقطه‌ی تقسیم (Split Point)
$X_j \in V$ یا $X_j \leq v$

آنتروپی (Entropy) از رابطه زیر تعریف می‌شود که در آن $P(c_i|\boldsymbol{D})$ احتمال دسته‌ی c_i در مجموعه داده‌ی $\boldsymbol{D}$ و k تعداد کل دسته‌ها است. اگر ناحیه‌ای کاملا خالص از دسته‌ی c_i باشد آنگاه $P(c_i|\boldsymbol{D}) = 1$ و $P(c_j|\boldsymbol{D})_{j\neq i} = 0$ و در نتیجه $H(\boldsymbol{D}) = 0$ می‌شود و اگر ناحیه کاملا مخلوط از هر k دسته باشد $P(c_i|\boldsymbol{D}) = \frac{1}{k}$ و آنتروپی می‌شود $H(\boldsymbol{D}) = \log_2 k$ پس هر چه ناحیه خالص‌تر باشد آنتروپی کمتر خواهد شد.

$$H(\mathbf{D}) = -\sum_{i=1}^{k} P(c_i|\mathbf{D}) \log_2 P(c_i|\mathbf{D})$$

$$H(\mathbf{D}_Y, \mathbf{D}_N) = \frac{n_Y}{n} H(\mathbf{D}_Y) + \frac{n_N}{n} H(\mathbf{D}_N)$$

نقطه‌ی تقسیمی که ناحیه را به نواحی خالص‌تر تقسیم کند، بهره‌ی اطلاعاتی (Information Gain) را افزایش می‌دهد.

$$Gain(\mathbf{D}, \mathbf{D}_Y, \mathbf{D}_N) = H(\mathbf{D}) - H(\mathbf{D}_Y, \mathbf{D}_N)$$

الگوریتم درخت تصمیم (Decision Tree Algorithm)

از هر طریق آنتروپی، بهره‌ی اطلاعاتی و یا عدد جینی بخواهیم نقطه‌ی تقسیم را محاسبه کنیم باید تابع جرمی احتمال (احتمال پسین) $P(c_i|\boldsymbol{D})$ را برآورد نماییم.

Algorithm 19.2: Evaluate Numeric Attribute (Using Gain)

EVALUATE-NUMERIC-ATTRIBUTE ($\mathbf{D}, X$):

sort $\mathbf{D}$ on attribute X, so that $x_j \le x_{j+1}, \forall j = 1, \ldots, n-1$
$\mathcal{M} \leftarrow \emptyset$ // set of midpoints
for $i = 1, \ldots, k$ **do** $n_i \leftarrow 0$
for $j = 1, \ldots, n-1$ **do**
    if $y_j = c_i$ **then** $n_i \leftarrow n_i + 1$
    // running count for class c_i
    if $x_{j+1} \ne x_j$ **then**
        $v \leftarrow \frac{x_{j+1} + x_j}{2}$; $\mathcal{M} \leftarrow \mathcal{M} \cup \{v\}$ // midpoints
        for $i = 1, \ldots, k$ **do**
            $N_{vi} \leftarrow n_i$ // Number of points such that $x_j \le v$ and $y_j = c_i$
if $y_n = c_i$ **then** $n_i \leftarrow n_i + 1$
// evaluate split points of the form $X \le v$
$v^* \leftarrow \emptyset$; $score^* \leftarrow 0$ // initialize best split point
forall $v \in \mathcal{M}$ **do**
    for $i = 1, \ldots, k$ **do**
        $\hat{P}(c_i|\mathbf{D}_Y) \leftarrow \frac{N_{vi}}{\sum_{j=1}^{k} N_{vj}}$
        $\hat{P}(c_i|\mathbf{D}_N) \leftarrow \frac{n_i - N_{vi}}{\sum_{j=1}^{k} n_j - N_{vj}}$
    $score(X \le v) \leftarrow Gain(\mathbf{D}, \mathbf{D}_Y, \mathbf{D}_N)$ // use Eq. (19.5)
    if $score(X \le v) > score^*$ **then**
        $v^* \leftarrow v$; $score^* \leftarrow score(X \le v)$
return $(v^*, score^*)$

$$\hat{P}(X \le v|c_i) = \frac{\hat{P}(X \le v \text{ and } c_i)}{\hat{P}(c_i)} = \left(\frac{1}{n}\sum_{j=1}^{n} I(x_j \le v \text{ and } y_j = c_i)\right) \Big/ (n_i/n) = \frac{N_{vi}}{n_i}$$

که در آن N_{vi} تعداد نقاطی با برچسب c_i هستند
که شرط $x_j \le v$ را نیز ارضاع می‌کنند.

$$\hat{P}(c_i|\mathbf{D}_Y) = \hat{P}(c_i|X \le v) = \frac{N_{vi}}{\sum_{j=1}^{k} N_{vj}}$$

$$\boxed{\hat{P}(c_i|\mathbf{D}_N) = \hat{P}(c_i|X > v) = \frac{\hat{P}(X > v|c_i)\hat{P}(c_i)}{\sum_{j=1}^{k} \hat{P}(X > v|c_j)\hat{P}(c_j)} = \frac{n_i - N_{vi}}{\sum_{j=1}^{k}(n_j - N_{vj})}}$$

مشخصه‌های دسته‌ای (Categorical Attributes)

$$\hat{P}(X=v|c_i) = \frac{\hat{P}\big(X=v \text{ and } c_i\big)}{\hat{P}(c_i)}$$

$$= \left(\frac{1}{n}\sum_{j=1}^{n} I(x_j = v \text{ and } y_j = c_i)\right)\Big/ (n_i/n)$$

$$= \frac{n_{vi}}{n_i}$$

$$\boxed{\hat{P}(c_i|\mathbf{D}_Y) = \frac{\sum_{v\in V}\hat{P}(X=v|c_i)\hat{P}(c_i)}{\sum_{j=1}^{k}\sum_{v\in V}\hat{P}(X=v|c_j)\hat{P}(c_j)} = \frac{\sum_{v\in V} n_{vi}}{\sum_{j=1}^{k}\sum_{v\in V} n_{vj}}}$$

$$\boxed{\hat{P}(c_i|\mathbf{D}_N) = \hat{P}(c_i|X\notin V) = \frac{\sum_{v\notin V} n_{vi}}{\sum_{j=1}^{k}\sum_{v\notin V} n_{vj}}}$$

که در آن n_{vi} تعداد نقاطی هستند که برچسب c_i دارند و مشخصه‌ی X آن‌ها در دسته‌ی V قرار می‌گیرد.

Algorithm 19.3: Evaluate Categorical Attribute (Using Gain)

EVALUATE-CATEGORICAL-ATTRIBUTE ($\mathbf{D}, X, l$):

```
for i = 1, ..., k do
    n_i ← 0
    forall v ∈ dom(X) do n_vi ← 0
for j = 1, ..., n do
    if x_j = v and y_j = c_i then  n_vi ← n_vi + 1 // frequency statistics
// evaluate split points of the form   X ∈ V
V* ← ∅; score* ← 0 // initialize best split point
forall V ⊂ dom(X), such that 1 ≤ |V| ≤ l do
    for i = 1, ..., k do
        P̂(c_i|D_Y) ← Σ_{v∈V} n_vi / Σ_{j=1}^{k} Σ_{v∈V} n_vj
        P̂(c_i|D_N) ← Σ_{v∉V} n_vi / Σ_{j=1}^{k} Σ_{v∉V} n_vj
    score(X ∈ V) ← Gain(D, D_Y, D_N) // use Eq. (19.5)
    if score(X ∈ V) > score* then
        V* ← V; score* ← score(X ∈ V)
return (V*, score*)
```

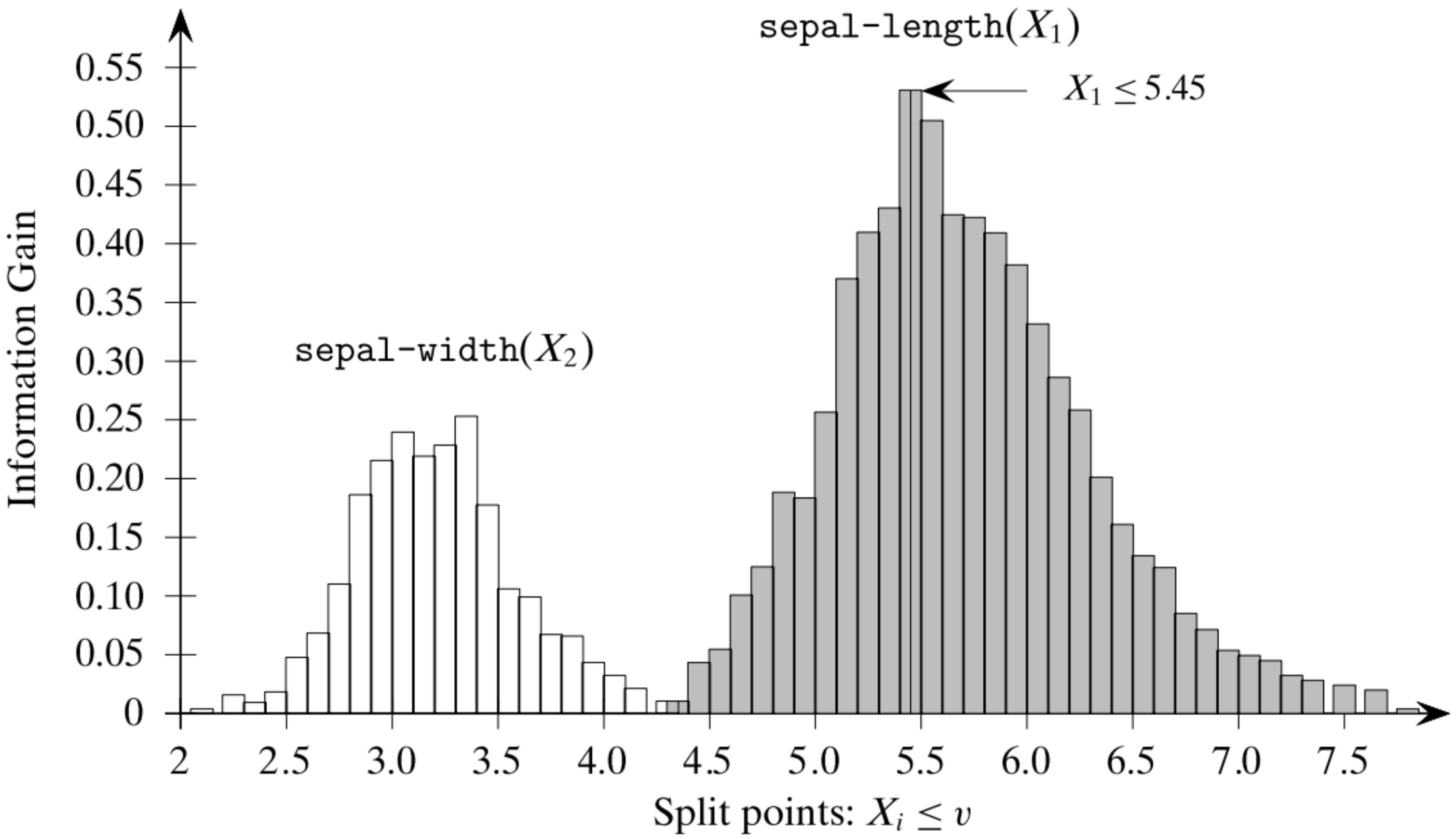


Figure 19.3. Iris: gain for different split points, for `sepal length` and `sepal width` .