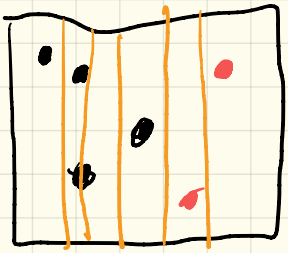
Split Points

10/12

Let $U = \{v_1, \dots, v_m\}$ be our set of midpoints

$\exists\ v_1 < v_2 < \dots < v_m$



we need to estimate

$\hat{P}(c_i \mid D_v) = \hat{P}(c_i \mid X \le v)$

$\hat{P}(c_i \mid D_w) = \hat{P}(c_i \mid X > v)$

Let I indicator random var

1 when arg is true

0 when arg is false

E.g Let B be a boolean Statement

$I(B) = \begin{cases} 0 & \text{if } B \text{ is false} \\ 1 & \text{if } B \text{ is true} \end{cases}$

Let $\vec{x}_i$ be labeled $y_i = c_3$

$y_i = c_1$

$I(y_i = c_1) = 0$
$I(y_i = c_2) = 0$
$I(y_i = c_3) = 1$
$I(y_i = c_4) = 0$
$\vdots$

using Bayes thm

$$\hat{P}(c_i \mid X \le v) = \frac{\hat{P}(X \le v \mid c_i)\hat{P}(c_i)}{\hat{P}(X \le v)} = \frac{\hat{P}(X \le v \mid c_i)\hat{P}(c_i)}{\sum_{j=1}^{k} \hat{P}(X \le v \mid c_j)\hat{P}(c_j)}$$

note 1 $\sum_{v=1}^{k} \hat{P}(X \le v \mid c_j)$

note 2 $P(X \le v \mid c_j) = \frac{P(X \le v \text{ and } c_j)}{P(c_j)} \Leftrightarrow P(X \le v \text{ and } c_j) = P(X \le v \mid c_j) \cdot P(c_j)$

$$\hat{P}(c_i | x \le v) = \frac{\hat{P}(x \le v | c_i)\hat{P}(c_i)}{\sum_{j=1}^{k} \hat{P}(x \le v | c_j)\hat{P}(c_j)}$$

$$\hat{P}(c_i) = \frac{1}{n}\sum_{j=1}^{n} I(y_i = c_i) = \frac{n_i}{n}$$

n_i: # of pts w/ label i; n: # of pts

y_i is the label of pt $\hat{x}_i$

$$\hat{P}(x \le v | c_i)\hat{P}(c_i) \Big/ \sum_{j=1}^{k} \hat{P}(x \le v | c_j)\hat{P}(c_j)$$

Next $\hat{P}(x \le v | c_i) = \frac{N_{vi}}{n_i}$

Let N_{vi} be the number of pts w/ class c_i and val less than our split

$$N_{vi} = \sum_{j=1}^{n} I(x_j \le v \text{ and } y_j = c_i)$$

$$\hat{P}(x \le v | c_i) = \frac{\hat{P}(x \le v \text{ and } c_i)}{\hat{P}(c_i)} = \frac{\frac{1}{n} N_{vi}}{\left(\frac{n_i}{n}\right)} = \frac{N_{vi}}{n_i}$$

$P(c_i | D_y)$

$$\hat{P}(c_i | x \le v) = \frac{\hat{P}(x \le v | c_i)\hat{P}(c_i)}{\sum_{j=1}^{k} \hat{P}(x \le v | c_j)\hat{P}(c_j)} = \frac{\frac{N_{vi}}{n}}{\sum_{j=1}^{k} \frac{N_{vj}}{n}} = \frac{\frac{N_{vi}}{n}}{\frac{1}{n}\sum_{j=1}^{k} N_{vj}}$$

Simplify with:

$$\hat{P}(x \le v | c_i)\, P(c_i) = \left(\frac{N_{vi}}{n_i}\right)\left(\frac{n_i}{n}\right) = \frac{N_{vi}}{n}$$

$$\frac{N_{vi}}{\sum_{j=1}^{k} N_{vj}}$$

$$\hat{P}(c_i | D_N) = \hat{P}(c_i | x > v)$$

using same idea as before:

$$\hat{P}(c_i|\mathbf{D}_N) = \hat{P}(c_i|X > v) = \frac{\hat{P}(X > v|c_i)\hat{P}(c_i)}{\sum_{j=1}^{k} \hat{P}(X > v|c_j)\hat{P}(c_j)}$$

$$\hat{P}(x > v | c_i) = 1 - \hat{P}(x \le v | c_i) = 1 - \frac{N_{vi}}{n_i} = \frac{n_i - N_{vi}}{n_i}$$

and plug each in and simplify

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

Now we can evaluate split points

compute midpts (sort midpts) $O(n \lg n)$

Algorithm 19.2: Evaluate Numeric Attribute (Using Gain)

```
EVALUATE-NUMERIC-ATTRIBUTE (D, X):
1  sort D on attribute X, so that x_j ≤ x_{j+1}, ∀j = 1,...,n−1
2  M ← ∅ // set of midpoints
3  for i = 1,...,k do n_i ← 0
4  for j = 1,...,n−1 do
5      if y_j = c_i then n_i ← n_i + 1
       // running count for class c_i
6      if x_{j+1} ≠ x_j then
7          v ← (x_{j+1} + x_j)/2 ; M ← M ∪ {v} // midpoints
8          for i = 1,...,k do
9              N_vi ← n_i // Number of points such that x_j ≤ v and
                   y_j = c_i
10 if y_n = c_i then n_i ← n_i + 1
   // evaluate split points of the form X ≤ v
11 v* ← ∅; score* ← 0 // initialize best split point
12 forall v ∈ M do
13     for i = 1,...,k do
14         P̂(c_i|D_Y) ← N_vi / Σ_{j=1}^{k} N_vj
15         P̂(c_i|D_N) ← (n_i − N_vi) / Σ_{j=1}^{k} (n_j − N_vj)
16     score(X ≤ v) ← Gain(D, D_Y, D_N) // use Eq. (19.5)
17     if score(X ≤ v) > score* then
18         v* ← v; score* ← score(X ≤ v)
19 return (v*, score*)
```

Note: DO NOT COMPUTE MIDPOINTS LIKE THIS CAN CAUSE OVERFLOW $x_j + \frac{x_{j+1} - x_j}{2}$

Compute N_{vi}'s $O(nk)$

$O(nk)$

$O(k)$

$O(k)$

total time

$O(n \lg n + nk)$ as k is usually small (e.g. # labels) we treat k as a const

$O(n \lg n)$ work to find best split in each dim

$n = |D|$

assume # of labels k is const

$\Rightarrow$ complexity of computing split point in a dim is $O(n \lg n)$

Algorithm 19.1: Decision Tree Algorithm

```
DECISIONTREE (D, η, π):
1  n ← |D| // partition size
2  n_i ← |{x_j | x_j ∈ D, y_j = c_i}| // size of class c_i
3  purity(D) ← max_i {n_i / n}
4  if n ≤ η or purity(D) ≥ π then // stopping condition
5      c* ← argmax_{c_i} {n_i / n} // majority class
6      create leaf node, and label it with class c*
7      return
8  (split point*, score*) ← (∅, 0) // initialize best split point
9  foreach (attribute X_j) do
10     if (X_j is numeric) then
11         (v, score) ← EVALUATE-NUMERIC-ATTRIBUTE(D, X_j)
12         if score > score* then (split point*, score*) ← (X_j ≤ v, score)
13     else if (X_j is categorical) then
14         (V, score) ← EVALUATE-CATEGORICAL-ATTRIBUTE(D, X_j)
15         if score > score* then (split point*, score*) ← (X_j ∈ V, score)
   // partition D into D_Y and D_N using split point*, and call
      recursively
16 D_Y ← {x^T | x ∈ D satisfies split point*}
17 D_N ← {x^T | x ∈ D does not satisfy split point*}
18 create internal node split point*, with two child nodes, D_Y and D_N
19 DECISIONTREE(D_Y); DECISIONTREE(D_N)
```

find good splitpoint for each dim d

run $O(n \lg n)$ time alg to find a split point in dim

$\Downarrow$

$O(dn \log n)$ for each recursive call

How many times do we recurse?

$\Rightarrow$ How deep can our tree be at worst $O(n)$

$\Rightarrow$ routine that is $O(dn \log n)$ $O(n)$ times

$\Rightarrow O(n \cdot d \cdot n \log n)$ time $= O(dn^2 \log n)$