Decision Tree Classifier

CSCI 347

Adiesha Liyana Ralalage

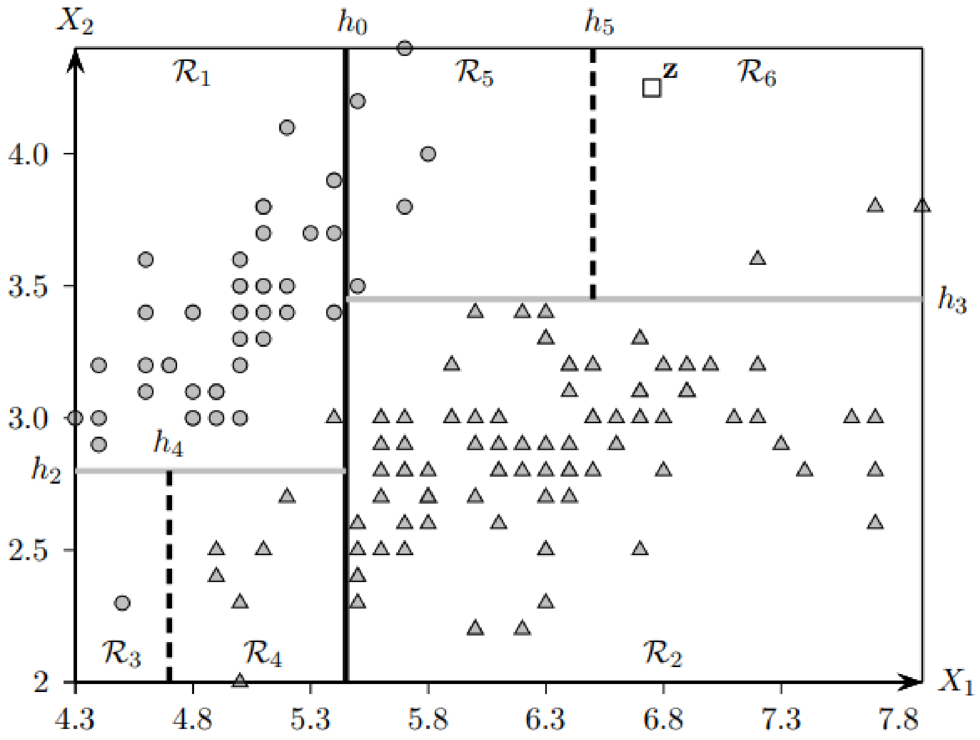
Decision Tree classifier

- What is the basic Idea?
- Works by splitting the datasets into subsets based on the feature values.
- This classification technique forms a tree-like structure where each internal node represents a decision on a feature.
- Recursive algorithm.
- A decision tree uses an axis-parallel hyperplane to split the data space.

Preliminaries

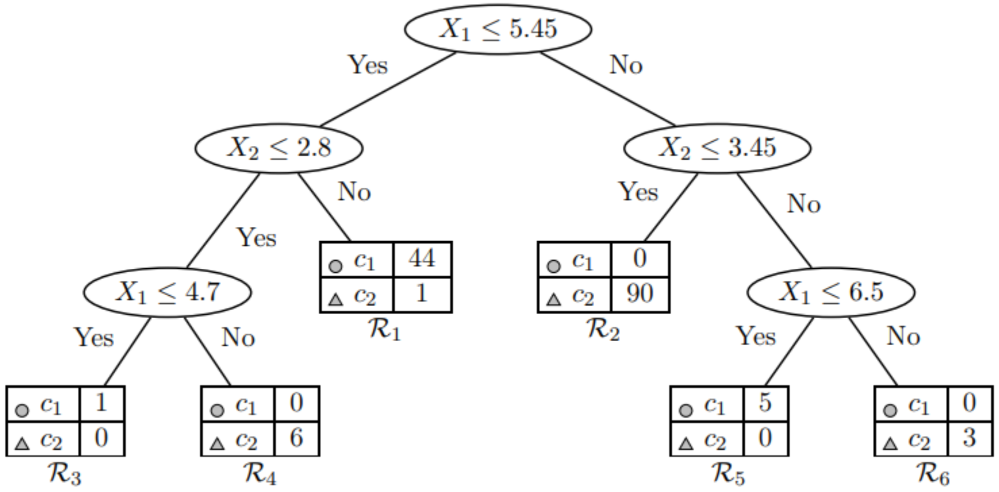
- You are given a training dataset D of n points. $D = \{x_1, x_2, \dots, x_n\}$
- Each point $x_i \in D$ is in d dimensional space.
- Attributes of each point can be either categorical or numerical.
- You are also given class labels of each data point x_i, denoted as y_i.
- $y = \{y_1, y_2, \dots, y_n\} \quad \textit{this is the label column}$
- $y_i \in \{c_1, c_2, \dots, c_k\}$ $\textit{each label is from one of these k valeus}$
- The classification model predicts the class $\hat{y}_i$ for the point x_i
- Let $\mathcal{R}$ denote the data space that encompasses the set of input points D.
- Decision tree classifier uses axis-parallel hyperplane to split the input space $\mathcal{R}$ into two resulting half-spaces or regions, say $\mathcal{R}_1, \mathcal{R}_2$, which includes partition of points into D_1 and D_2.

MONTANA STATE UNIVERSITY



(a) Recursive Splits

(a) Recursive Splits



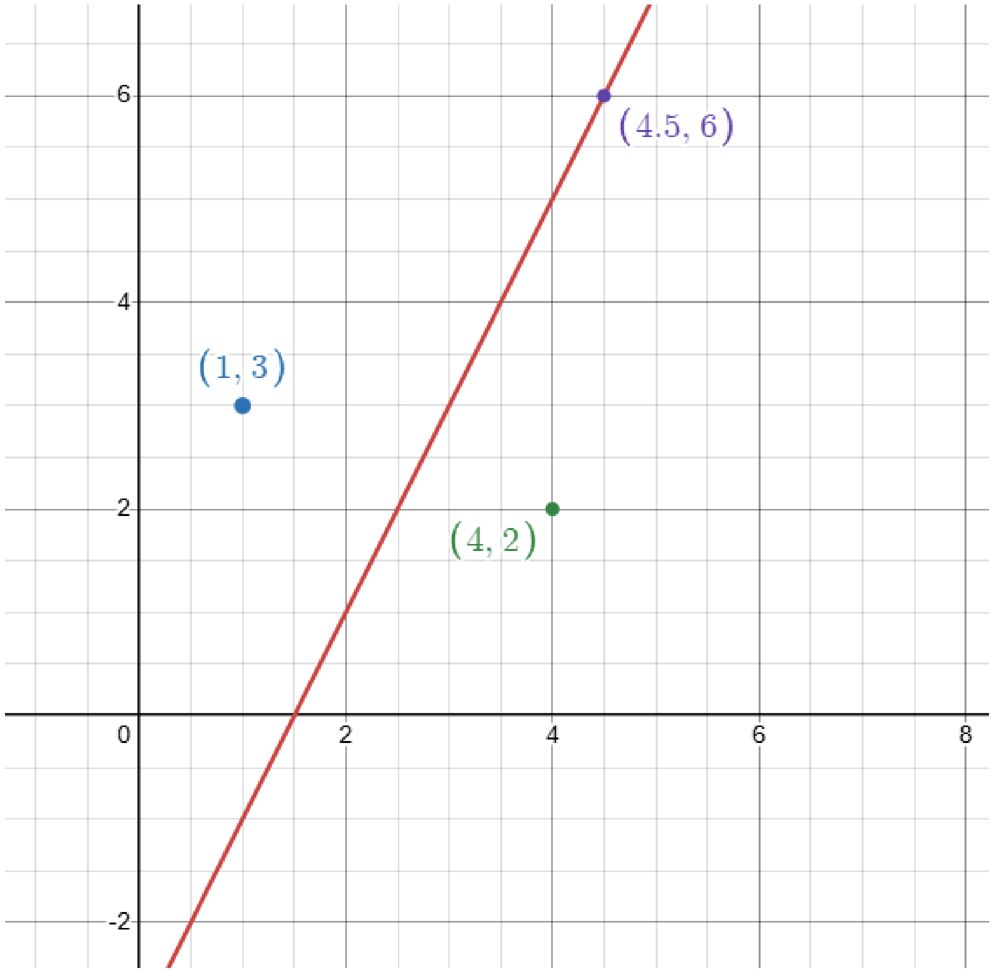
(b) Decision Tree

Decision Trees

- A decision tree consists of internal nodes that represents the decisions corresponding to the hyperplanes or split-points (i.e., which half space a given point lies in), and leaf nodes that represent regions or partitions of the data space, which are labelled with the majority of the class.
- Axis-Parallel hyperplanes
  - A hyperplane $h(x)$ is defined as the set of all points x that satisfy the following equation.
    - $h(x): w^T x + b = 0$
  - An axis-parallel hyperplane is a hyperplane where the weight vector is parallel to one of the original dimensions. (Hyperplane is parallel to one of the original dimensions)
  - Desmos..

Decision Trees

- Example: Consider a hyperplane in 2D
- $y \; - 2x + 3 = 0$
- Split-Points
- All points on one side of the hyperplane evaluates to a positive value and all points in other side to a negative values.
- Try this.
- This is **not** an axis-parallel hyperplane.



https://www.desmos.com/calculator/ejoj3cf8hi

Decision Trees

- Each split (using an axis-parallel line) of $\mathcal{R}$ into $\mathcal{R}_1$ and $\mathcal{R}_2$ induces binary partition corresponding to the input data points D.

$$D_1 = \{x \mid x \in D, x_j \leq v\}$$
$$D_2 = \{x \mid x \in D, x_j > v\}$$

- Purity

$$purity(D_j) = \max_i \left\{ \frac{n_{ji}}{n_j} \right\}$$

purity is the fraction of points with the majority label in D_j

$n_j = |D_j|$ and n_{ji} is the number of points in D_j with class label c_i.

MONTANA STATE UNIVERSITY

Decision Trees

- Decision trees can handle Categorical attributes as well.
- For categorical attributes split-points or decisions are of the $X_j \in V$, where $V \subset dom(X_j)$
- Now we have a binary decision for splitting the dataset.
- If the attribute j of the point x, $x_j \in V$, we put the x into $\mathcal{R}_1$ and otherwise into $\mathcal{R}_2$.

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`

  $c^* \leftarrow \arg\max_i \left\{\frac{n_i}{n}\right\}$ `// majority class`

  create leaf node, and label it with class c^*

  return

$(split\text{-}point^*, score^*) \leftarrow (\emptyset, 0)$ `// 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** $(split\text{-}point^*, score^*) \leftarrow (X_j \leq v, score)$

  else if *(X_j is categorical)* **then**

    $(V, score) \leftarrow$ Evaluate-Categorical-Attribute$(\mathbf{D}, X_j)$

    if $score > score^*$ **then** $(split\text{-}point^*, score^*) \leftarrow (X_j \in V, score)$

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

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

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

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

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

Split-point evaluation methods

- Intuitively, we want to select a split-point that gives the best separation or discrimination between the different class labels.

Entropy

- Measures the amount of disorder or uncertainty in a system.
- A partition has lower entropy (or low disorder) if it is relatively pure, i.e., if most of the points have the same label.
- A partition has higher entropy (or more disorder) if the class labels are mixed.
- The entropy of a set of labeled points D is defined as follows:

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

Entropy

- The entropy of a set of labeled points D is defined as follows:

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

- If a region has points from one class (high purity): entropy would be zero.
- If the classes are all mixed up, and each appears with equal probability $P(c_i|\, D) = \frac{1}{k}$, then the entropy has the highest value, $H(D) = \log_2 k$

Split-Point Entropy

We can define entropy for the split point as well.

- Split point entropy is the weighted entropy of each of the resulting partitions:

$$H(D_Y, D_N) = \frac{n_Y}{n} \cdot H(D_Y) + \frac{n_N}{n} \cdot H(D_N)$$

- To see if the split point results in a reduced overall entropy, we can calculate the following (Information Gain):

$$Gain(D, D_Y, D_N) = H(D) \ - H(D_Y, D_N)$$

- Higher the information gain more the reduction in entropy and better the split point.

Gini-Index

- Another common measure to gauge the purity of a split-point is the Gini-index, defined as follows

$$G(D) = 1 - \sum_{i=1}^{k} P(c_i|\, D)^2$$

- If the partition is pure, then the probability of the majority class is 1 and the probability of all other classes is 0, Gini-Index is 0.
- If each class is equally probable, then $P(c_i|D) = \frac{1}{k}$, then the Gini-Index value is $\frac{k-1}{k}$
- Higher the Gini-index value, higher the disorder is.

Gini-Index

- Just like before we can define this for the split point as well:

$$G(D_Y, D_N) = \frac{n_Y}{n} \cdot G(D_Y) + \frac{n_N}{n} \cdot G(D_N)$$

CART (Classification and Regression Tree)

- This measure thus prefers a split-point that maximizes the difference between the class probability mass function for the two partitions.
- higher the CART measure, the better the split-point.

$$CART(D_Y, D_N) = 2\frac{n_Y}{n}\frac{n_N}{n}\sum_{i=1}^{k}|P(c_i|D_Y) - P(c_i|D_N)|$$

Evaluating split points

Numeric Attributes

- Problem: There are infinite choices to evaluate even if we restrict the values to a particular range.
- One reasonable approach is to only look at mid-points between two successive distinct values of the attribute X in the sample D.
- Since there are at most n distinct values in X, we only have to consider $n - 1$ mid-points.
- Let $\{v_1, v_2, \dots, v_m\}$ denote the set of all such mid points, such that $v_1 < v_2 < \cdots < v_m$.
- For each split point $X \leq v$, we have to estimate the class PMF
  - $\hat{P}(c_i | D_Y) = \hat{P}(c_i | X \leq v)$
  - $\hat{P}(c_i | D_N) = \hat{P}(c_i | X > v)$

MONTANA STATE UNIVERSITY

Evaluating split points

Numeric Attributes

- $\hat{P}(c_i|X \leq v) = \frac{\hat{P}(X \leq v|c_i)\,\hat{P}(c_i)}{\hat{P}(X \leq v)} = \frac{\hat{P}(X \leq v|c_i)\,\hat{P}(c_i)}{\sum_{j=1}^{k}\hat{P}(X \leq v|c_j)\,\hat{P}(c_j)}$
- $P(c_i) = \frac{1}{n}\sum_{j=1}^{n} I(y_j = c_i) = \frac{n_i}{n}$,
  $n_i\ is\ the\ number\ of\ points\ in\ D\ with\ class\ c_i$
- N_{vi} is the number of points $x_j \leq v$ with class c_i, where x_j is the value of data point $\boldsymbol{x_j}$ for the attribute X
  - $N_{vi} = \sum_{j=1}^{n} I(x_j \leq v\ and\ y_j = c_i)$
- We can estimate the $\hat{P}(c_i|X \leq v)$ as follows:
  - $\hat{P}(c_i|X \leq v) = \frac{\hat{P}(x_j \leq v\ and\ y_j = c_i)}{\hat{P}(c_i)} = \frac{N_{vi}}{n_i}$
- Now:
  - $\boldsymbol{\hat{P}(c_i|D_Y) = \hat{P}(c_i|X \leq v) = \frac{N_{vi}}{\sum_{j=1}^{k} N_{vj}}}$

Evaluating split points

Numeric Attributes

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

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

Algorithm 19.2: Evaluate Numeric Attribute (Using Gain)

Evaluate-Numeric-Attribute $(\mathbf{D}, X)$:

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

Time complexity

- Initial sorting takes $O(n \cdot \log n)$
- The cost of computing the mid-points and class specific counts take $O(nk)$
- Cost of computing the score is also bounded by $O(nk)$
- The total cost of evaluating the numeric attribute is $O(n \cdot \log n + nk)$.
  - Since k is usually a small value $O(n \cdot \log n)$

Evaluating split points

Categorical Attributes

- If X is a categorical attribute, we evaluate the split-points of the form $X \in V$, where $V \subset dom(X)\ and\ V \neq \emptyset$.
- All distinct partitions of the set values of X are considered.
- The total number of distinct partitions is
  - $\sum_{i=1}^{m} \binom{m}{i}$?
  - It is actually $\sum_{i=1}^{\left\lfloor \frac{m}{2} \right\rfloor} \binom{m}{i}$ which is $O(2^{m-1})$
  - $m = |dom(X)|$ basically the number of unique categorical values.
  - Number of split points to consider is exponential in m.
  - One restriction that we can do is to consider V of size 1.

Evaluating split points

Categorical Attributes

- To evaluate the split point $X \in V$ we need to compute:
  - $P(c_i|\, D_Y) = P(c_i \mid X \in V)$ and $P(c_i|\, D_N) = P(c_i \mid X \notin V)$
  - We can use the bayes theorem.
  - $\frac{P(X \in V \mid c_i)P(c_i)}{P(X \in V)} = \frac{P(X \in V \mid c_i)P(c_i)}{\sum_{j=1}^{k} P(X \in V \mid c_j)P(c_j)}$
  - Note that given point x can only take one value in the domain of X. Therefore, the values $v \in dom(X)$ are mutually exclusive.
    - $P(X \in V \mid c_i) = \sum_{v \in V} P(X = v|c_i)$
    - We can write $P(c_i|D_Y)$ as
    - $P(c_i|D_Y) = \frac{\sum_{v \in V} P(X=v|c_i)P(c_i)}{\sum_{j=1}^{k} \sum_{v \in V} P(X=v|c_i)P(c_i)}$
    - $\hat{P}(X = v|c_i) = \frac{n_{vi}}{n_i}$, n_{vi} is the number of points $x_j \in D$, with value $x_j = v$ for attribute X and having class $y_j = c_i$
  - $\hat{P}(c_i|D_Y) = \frac{\sum_{v \in V} n_{vi}}{\sum_{j=1}^{k} \sum_{v \in V} n_{vj}}$ $\hat{P}(c_i|D_N) = \frac{\sum_{v \notin V} n_{vi}}{\sum_{j=1}^{k} \sum_{v \notin V} n_{vj}}$

Algorithm 19.3: Evaluate Categorical Attribute (Using Gain)

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

for $i = 1, \ldots, k$ **do**
    $n_i \leftarrow 0$
    forall $v \in dom(X)$ **do** $n_{vi} \leftarrow 0$

for $j = 1, \ldots, n$ **do**
    if $x_j = v$ *and* $y_j = c_i$ **then** $n_{vi} \leftarrow n_{vi} + 1$ `// frequency statistics`

`// evaluate split-points of the form` $X \in V$
$V^* \leftarrow \emptyset$; $score^* \leftarrow 0$ `// initialize best split-point`
forall $V \subset dom(X)$, *such that* $1 \leq |V| \leq l$ **do**
    for $i = 1, \ldots, k$ **do**
        $\hat{P}(c_i|\mathbf{D}_Y) \leftarrow \frac{\sum_{v \in V} n_{vi}}{\sum_{j=1}^{k} \sum_{v \in V} n_{vj}}$
        $\hat{P}(c_i|\mathbf{D}_N) \leftarrow \frac{\sum_{v \notin V} n_{vi}}{\sum_{j=1}^{k} \sum_{v \notin V} n_{vj}}$
    $score(X \in V) \leftarrow Gain(\mathbf{D}, \mathbf{D}_Y, \mathbf{D}_N)$ `// use` (19.5)
    if $score(X \in V) > score^*$ **then**
        $V^* \leftarrow V; score^* \leftarrow score(X \in V)$

return $(V^*, score^*)$