Decision Tree and Ensemble learning

Application of Machine Learning in Astronomy
01 Decision Tree

02 Ensemble learning
Decision Tree
1. Decision Tree

Recursion / Recursive algorithms

```python
def some_fun(x):
    if x == "" :
        return 0
    else:
        return 1 + some_fun(x[1:])
```

What dose this function do?
1. Decision Tree

Divide-and-Conquer algorithm

def quicksort(array):
    if len(array) < 2:
        return array
    else:
        pivot = array[0]
        smaller, bigger = [], []
        for ele in array[1:]:
            if ele <= pivot:
                smaller.append(ele)
            else:
                bigger.append(ele)
        return quicksort(smaller) + [pivot] + quicksort(bigger)

Cite: Sebastian Raschka, Machine Learning
1. Decision Tree

Divide-and-Conquer algorithm

```python
def quicksort(array):
    if len(array) < 2:
        return array
    else:
        pivot = array[0]
        smaller, bigger = [], []
        for ele in array[1:]:
            if ele <= pivot:
                smaller.append(ele)
            else:
                bigger.append(ele)
        return quicksort(smaller) + [pivot] + quicksort(bigger)
```

Time complexity (average): \(O(n \log n)\)
1. Decision Tree

Array Sorting Algorithms

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Time Complexity</th>
<th>Space Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Best</td>
<td>Average</td>
</tr>
<tr>
<td>Quicksort</td>
<td>Ω(n log(n))</td>
<td>θ(n log(n))</td>
</tr>
<tr>
<td>Mergesort</td>
<td>Ω(n log(n))</td>
<td>θ(n log(n))</td>
</tr>
<tr>
<td>Timsort</td>
<td>Ω(n)</td>
<td>θ(n log(n))</td>
</tr>
<tr>
<td>Heapsort</td>
<td>Ω(n log(n))</td>
<td>θ(n log(n))</td>
</tr>
<tr>
<td>Bubble Sort</td>
<td>Ω(n)</td>
<td>θ(n^2)</td>
</tr>
<tr>
<td>Insertion Sort</td>
<td>Ω(n)</td>
<td>θ(n^2)</td>
</tr>
<tr>
<td>Selection Sort</td>
<td>Ω(n^2)</td>
<td>θ(n^2)</td>
</tr>
<tr>
<td>Tree Sort</td>
<td>Ω(n log(n))</td>
<td>θ(n log(n))</td>
</tr>
<tr>
<td>Shell Sort</td>
<td>Ω(n log(n))</td>
<td>θ(n log(n))</td>
</tr>
<tr>
<td>Bucket Sort</td>
<td>Ω(n+k)</td>
<td>θ(n+k)</td>
</tr>
<tr>
<td>Radix Sort</td>
<td>Ω(nk)</td>
<td>θ(nk)</td>
</tr>
<tr>
<td>Counting Sort</td>
<td>Ω(n+k)</td>
<td>θ(n+k)</td>
</tr>
<tr>
<td>Cubesort</td>
<td>Ω(n)</td>
<td>θ(n log(n))</td>
</tr>
</tbody>
</table>

Cite: https://www.bigocheatsheet.com
1. Decision Tree

1.1 Decision tree terminology

- A nested sequence of “if-else” decisions based on the features.
- A class label as a return value at the end of each sequence.

Cite: Sebastian Raschka, Machine Learning
1. Decision Tree

1.1 Decision tree terminology

Decision Tree in Pseudocode

TreeGenerate($\mathcal{D}$):

if $y = 1$, $\forall \langle X, y \rangle \in \mathcal{D}$ or $y = 0$, $\forall \langle X, y \rangle \in \mathcal{D}$:

return Tree

else:

Pick best feature $x_j$:

$\mathcal{D}_0$ at Child$_0$ : $x_j = 0 \ \forall \langle X, y \rangle \in \mathcal{D}$

$\mathcal{D}_1$ at Child$_1$ : $x_j = 1 \ \forall \langle X, y \rangle \in \mathcal{D}$

return Node($x_j$, TreeGenerate($\mathcal{D}_0$), TreeGenerate($\mathcal{D}_1$))
1. Decision Tree

1.1 Decision tree terminology

TreeGenerate($D$, $T$, $A$)

$D$ : set of training examples;
$T$ : target attribute;
$A$ : set of descriptive attributes;

Create a Root node for the tree.

If $T$ have the same target attribute value $t_i$

Then Return the single-node tree, (i.e., Root or Leaf node), with target attribute $= t_i$;

If $A = \emptyset$ (i.e. there is no descriptive attributes available)

Then Return the single-node tree, (i.e., Root or Leaf node), with most common value of target in $T$

Otherwise

Select attribute $a^*$ from $A$ that best classify $D$ on splitting measure

Set $a^*$ the attribute for Root

For each value of $a^*$, $v_i$, do

Add a branch below Root, corresponding to $a^* = v_i$;

Let $D_v$ be the subset of $D$ that have $a^* = v_i$;

If $D_v = \emptyset$

Then add a leaf node below the branch with target value $=$ most common value of target in $T$

else

Add the subtree below the branch learned by TreeGenerate($D_v$, $T$, $A - \{a^*\}$)

Return (Root)
1. Decision Tree

1.1 Decision tree terminology

- **When to stop?**
  - if leaf nodes contain only examples of the same class;
  - feature values are all the same for all examples;
  - statistical significance test;
  - User-defined maximum depth.

- **How to split?**
  - what measurement/criterion as measure of goodness;
  - binary vs multi-category split;
1. Decision Tree

1.2 Example: simple training data set

<table>
<thead>
<tr>
<th>ID</th>
<th>Grayscale (Intensity)</th>
<th>Latitude</th>
<th>Location</th>
<th>Tilt Angle</th>
<th>Barb Number</th>
<th>Quiescent Filament</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>gray</td>
<td>high</td>
<td>northern hemisphere</td>
<td>small</td>
<td>many</td>
<td>Yes</td>
</tr>
<tr>
<td>2</td>
<td>dark gray</td>
<td>high</td>
<td>northern hemisphere</td>
<td>medium</td>
<td>many</td>
<td>Yes</td>
</tr>
<tr>
<td>3</td>
<td>dark gray</td>
<td>high</td>
<td>northern hemisphere</td>
<td>medium</td>
<td>many</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>gray</td>
<td>high</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>many</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>light gray</td>
<td>middle</td>
<td>southern hemisphere</td>
<td>small</td>
<td>some</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>gray</td>
<td>middle</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>some</td>
<td>Yes</td>
</tr>
<tr>
<td>7</td>
<td>dark gray</td>
<td>middle</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>some</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>dark gray</td>
<td>middle</td>
<td>northern hemisphere</td>
<td>large</td>
<td>few</td>
<td>No</td>
</tr>
<tr>
<td>9</td>
<td>light gray</td>
<td>middle</td>
<td>northern hemisphere</td>
<td>medium</td>
<td>few</td>
<td>No</td>
</tr>
<tr>
<td>10</td>
<td>gray</td>
<td>high</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>some</td>
<td>No</td>
</tr>
<tr>
<td>11</td>
<td>light gray</td>
<td>low</td>
<td>northern hemisphere</td>
<td>medium</td>
<td>few</td>
<td>No</td>
</tr>
<tr>
<td>12</td>
<td>dark gray</td>
<td>middle</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>many</td>
<td>No</td>
</tr>
<tr>
<td>13</td>
<td>gray</td>
<td>low</td>
<td>southern hemisphere</td>
<td>large</td>
<td>few</td>
<td>No</td>
</tr>
</tbody>
</table>
1. Decision Tree

1.2 Example: simple training data set

Problem setting

- **Set of possible samples** $X$
  - Each sample $x$ in $X$ is a feature vector;
  - e.g., $<$Grayscale=gray, Latitude=high, …, Barb number=many$>$;

- **Unknown target function** $f: X \rightarrow Y$
  - $Y = 1$ if the filament is quiescent filament, else $Y = 0$;

- **Set of function hypotheses** $H = \{h: X \rightarrow Y \}$
  - Each hypothesis $h$ is a decision tree;
  - Trees sorts $x$ to leaf, which assigns $y$. 
1. Decision Tree

1.3 Learning a decision stump

**Decision stump**: simple decision tree with 1 splitting rule based on thresholding 1 feature.

<table>
<thead>
<tr>
<th>ID</th>
<th>Grayscale (Intensity)</th>
<th>Latitude</th>
<th>Barb Number</th>
<th>Quiescent Filament</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>200</td>
<td>60</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>185</td>
<td>70</td>
<td>8</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>145</td>
<td>65</td>
<td>12</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>195</td>
<td>30</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>120</td>
<td>60</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>175</td>
<td>35</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

How to find the best “rule” (best classify measurement/criterion) ?
1. Define a **score** for the rule;
2. **Search** for the rule with the best score.

Most intuitive score: **classification accuracy**
If we use this rule, how many examples do we label correctly?

---

Cite: Mark Schmidt, Machine Learning and Data Mining
1. Decision Tree

1.3 Learning a decision stump

Computing classification accuracy for the rule (grayscale > 150):

- Find most common labels if we use this rule:
  - when (grayscale > 150), we get “yes” 2 times out of 4;
  - when (grayscale ≤ 150), we get “no” 1 times out of 2.

- Compute accuracy:
  - the accuracy (“score”) of the rule (grayscale=dark gray) is 3 times out of 6.

This “score” evaluates quality of a rule. We “learn” a decision stump by find the rule with the best score.
1. Decision Tree

1.1 Learning a decision stump

Search for the decision stump maximizing classification score:

- “baseline rule” of predicting mode (no split):
  - 3/6 accuracy
- If (grayscale>150):
  - predict “yes” (2/4) else predict “no” (1/2): 1/2 accuracy
- If (latitude>40):
  - predict “yes” (3/4) else predict “no” (2/2): 5/6 accuracy
- If (Barb>5):
  - predict “yes” (3/3) else predict “no” (3/3): 6/6 accuracy

Highest-scoring rule: (Barb>5) with leaves “yes” and “no”.

- Notice we only need to test feature thresholds that happen in the data:
  - There is no point in testing the rule (Barb>10), it gets the “baseline” score;
  - There is no point in testing the rule (Barb>0.5), it gets the (Barb>1) score;
  - We do not need to test “<”, since it would give equivalent rules.
1. Decision Tree

1.1 Learning a decision stump

How much does this cost?

Assume we have:
- “n” examples (filaments that we detected);
- “d” features (filaments features that we detected);
- “k” thresholds (>0, >10, …) for each feature.

Computing the score of one rule costs $O(n)$:
- We need to go through all “n” examples to find most common labels;
- We need to go through all “n” examples again to compute the accuracy.

We compute score for up to “k * d” rules.

So the total cost of $O(ndk)$. 

Cite: Mark Schmidt, Machine Learning and Data Mining
1. Decision Tree

1.1 Learning a decision stump

**Is a cost of** $O(ndk)$ **good?**

- **Assume we have:**
  - “$n$” examples (filaments that we detected);
  - “$d$” features (filaments features that we detected);
  - “$k$” thresholds (>0, >10, …) for each feature.

- **Size of the input data is** $O(nd)$:
  - If “$k$” is small then the cost is roughly the same as loading the data;
    - E.g., if all features are binary then $k=1$, just test (feature>0). The cost of fitting decision stump is is $O(nd)$, so we can fit huge dataset.
  - If “$k$” is large then this could be too slow for large datasets;
    - E.g., if all features are numerical with unique values then $k = n$. The cost of fitting decision stump is is $O(n^2d)$.
1. Decision Tree

Shouldn’t we just use accuracy score?

For leaves: yes, just maximize accuracy.

For internal nodes: ?

Which **score function** should a decision tree used?
1. Decision Tree

1.2 The splitting criterion

Entropy as Measure of Randomness

For a categorical variable that can take \( k \) values, entropy is defined by:

\[
\text{entropy} = -\sum_{c=1}^{k} p_c \log p_c
\]

where \( p_c \) is the proportion of times you have value \( c \).

- Roughly, it’s another measure of the “spread” of values.
- Minimum value is 0, maximum value is \( \log(k) \).
  --We use the convention that \( 0 \log 0 = 0 \).
1. Decision Tree

1.2 The splitting criterion

Entropy as Measure of Randomness

Low entropy means “very predictable”

High entropy means “very random”

For categorical features: uniform distribution has highest entropy.
For continuous densities with fixed mean and variance: Normal distribution has highest entropy.
1. Decision Tree

1.2 The splitting criterion

Information Entropy as measure of Purity

Low entropy means “high purity”

High entropy means “low purity”

\[
\text{Ent}(D) = - \sum_{k=1}^{K} p_k \log_2 p_k, \quad k = 1, 2, \ldots, K
\]
1. Decision Tree

1.2 The splitting criterion

**Information Entropy**

$$\text{Ent}(D) = - \sum_{k=1}^{K} p_k \log_2 p_k, \; k = 1, 2, \ldots, K$$

**Information Gain**

$$\text{Gain}(D) = \text{Ent}(D) - \sum_{v=1}^{V} \frac{|D^v|}{|D|} \text{Ent}(D^v), \; v = 1, 2, \ldots, V$$

$$A = \{a_1, a_2, \ldots, a_u\}, \; a_* = \{a_*^1, a_*^2, \ldots, a_*^V\}$$
1. Decision Tree

1.2 The splitting criterion

<table>
<thead>
<tr>
<th>ID</th>
<th>Grayscale (Intensity)</th>
<th>Latitude</th>
<th>Location</th>
<th>Tilt Angle</th>
<th>Barb Number</th>
<th>Quiescent Filament</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>gray</td>
<td>high</td>
<td>northern hemisphere</td>
<td>small</td>
<td>many</td>
<td>Yes</td>
</tr>
<tr>
<td>2</td>
<td>dark gray</td>
<td>high</td>
<td>northern hemisphere</td>
<td>medium</td>
<td>many</td>
<td>Yes</td>
</tr>
<tr>
<td>3</td>
<td>dark gray</td>
<td>high</td>
<td>northern hemisphere</td>
<td>medium</td>
<td>many</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>gray</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>light gray</td>
<td>middle</td>
<td>southern hemisphere</td>
<td>small</td>
<td>some</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>gray</td>
<td>middle</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>some</td>
<td>yes</td>
</tr>
<tr>
<td>7</td>
<td>dark gray</td>
<td>middle</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>some</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>dark gray</td>
<td>middle</td>
<td>northern hemisphere</td>
<td>large</td>
<td>few</td>
<td>No</td>
</tr>
<tr>
<td>9</td>
<td>light gray</td>
<td>middle</td>
<td>northern hemisphere</td>
<td>medium</td>
<td>few</td>
<td>No</td>
</tr>
<tr>
<td>10</td>
<td>gray</td>
<td>high</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>some</td>
<td>No</td>
</tr>
<tr>
<td>11</td>
<td>light gray</td>
<td>low</td>
<td>northern hemisphere</td>
<td>medium</td>
<td>few</td>
<td>No</td>
</tr>
<tr>
<td>12</td>
<td>dark gray</td>
<td>middle</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>many</td>
<td>No</td>
</tr>
<tr>
<td>13</td>
<td>gray</td>
<td>low</td>
<td>southern hemisphere</td>
<td>large</td>
<td>few</td>
<td>No</td>
</tr>
</tbody>
</table>
1. Decision Tree

1.2 The splitting criterion

<table>
<thead>
<tr>
<th>ID</th>
<th>Grayscale (Intensity)</th>
<th>Latitude</th>
<th>Location</th>
<th>Tilt Angle</th>
<th>Barb Number</th>
<th>Quiescent Filament</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>gray</td>
<td>high</td>
<td>northern hemisphere</td>
<td>small</td>
<td>many</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>gray</td>
<td>high</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>many</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>gray</td>
<td>middle</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>some</td>
<td>yes</td>
</tr>
<tr>
<td>10</td>
<td>gray</td>
<td>high</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>some</td>
<td>No</td>
</tr>
<tr>
<td>13</td>
<td>gray</td>
<td>low</td>
<td>southern hemisphere</td>
<td>large</td>
<td>few</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>dark gray</td>
<td>high</td>
<td>northern hemisphere</td>
<td>medium</td>
<td>many</td>
<td>Yes</td>
</tr>
<tr>
<td>3</td>
<td>dark gray</td>
<td>high</td>
<td>northern hemisphere</td>
<td>medium</td>
<td>many</td>
<td>Yes</td>
</tr>
<tr>
<td>7</td>
<td>dark gray</td>
<td>middle</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>some</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>dark gray</td>
<td>middle</td>
<td>northern hemisphere</td>
<td>large</td>
<td>few</td>
<td>No</td>
</tr>
<tr>
<td>12</td>
<td>dark gray</td>
<td>middle</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>many</td>
<td>No</td>
</tr>
<tr>
<td>5</td>
<td>light gray</td>
<td>middle</td>
<td>southern hemisphere</td>
<td>small</td>
<td>some</td>
<td>Yes</td>
</tr>
<tr>
<td>9</td>
<td>light gray</td>
<td>middle</td>
<td>northern hemisphere</td>
<td>medium</td>
<td>few</td>
<td>No</td>
</tr>
<tr>
<td>11</td>
<td>light gray</td>
<td>low</td>
<td>northern hemisphere</td>
<td>medium</td>
<td>few</td>
<td>No</td>
</tr>
</tbody>
</table>
1. Decision Tree

1.2 The splitting criterion

![Decision Tree Diagram]

- **Grayscale=?**
  - 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13
  - **Gray**: 1, 4, 6, 10, 13
  - **Dark Gray**: 2, 3, 7, 8, 12
  - **Light Gray**: 5, 9, 11
1. Decision Tree

1.2 The splitting criterion

Information Entropy

Grayscale=?

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13

Gray

1, 4, 6, 10, 13

Dark Gray

2, 3, 7, 8, 12

Light Gray

5, 9, 11

For $D^1 = \{1, 4, 6, 10, 13\}$, $p_1 = \frac{3}{5}$, $p_2 = \frac{2}{5}$, $\text{Ent}(D^1) = -\sum_{k=1}^{2} p_k \log_2 p_k = -\left(\frac{3}{5} \log_2 \frac{3}{5} + \frac{2}{5} \log_2 \frac{2}{5}\right) = 0.971$

For $D^2 = \{2, 3, 7, 8, 12\}$, $p_1 = \frac{2}{5}$, $p_2 = \frac{3}{5}$, $\text{Ent}(D^2) = -\sum_{k=1}^{2} p_k \log_2 p_k = -\left(\frac{2}{5} \log_2 \frac{2}{5} + \frac{3}{5} \log_2 \frac{3}{5}\right) = 0.971$

For $D^3 = \{5, 9, 11\}$, $p_1 = \frac{1}{3}$, $p_2 = \frac{2}{3}$, $\text{Ent}(D^3) = -\sum_{k=1}^{2} p_k \log_2 p_k = -\left(\frac{1}{3} \log_2 \frac{1}{3} + \frac{2}{3} \log_2 \frac{2}{3}\right) = 0.918$
1. Decision Tree

1.2 The splitting criterion

Information Gain

\[ \text{Gain}(D) = \text{Ent}(D) - \sum_{v=1}^{V} \frac{D^v}{D} \text{Ent}(D^v), v = 1, 2, \ldots, V \]

\[ \text{Ent}(D) = -\left( \frac{6}{13} \log_2 \frac{6}{13} + \frac{7}{13} \log_2 \frac{7}{13} \right) = 0.996 \]

\[ \text{Ent}(D^1) = -\left( \frac{3}{5} \log_2 \frac{3}{5} + \frac{2}{5} \log_2 \frac{2}{5} \right) = 0.971 \]

\[ \text{Ent}(D^2) = -\left( \frac{2}{5} \log_2 \frac{2}{5} + \frac{3}{5} \log_2 \frac{3}{5} \right) = 0.971 \]

\[ \text{Ent}(D^3) = -\left( \frac{1}{3} \log_2 \frac{1}{3} + \frac{2}{3} \log_2 \frac{2}{3} \right) = 0.918 \]

\[ \text{Gain}(D, \text{Grayscale}) = \text{Ent}(D) - \sum_{v=1}^{3} \frac{|D^v|}{|D|} \text{Ent}(D^v) \]

\[ = 0.996 - \left( \frac{5}{13} \times 0.971 + \frac{5}{13} \times 0.971 + \frac{3}{13} \times 0.918 \right) = 0.037 \]
1. Decision Tree

1.2 The splitting criterion

Information Gain

Grayscale=?

<table>
<thead>
<tr>
<th>Grayscale</th>
<th>Gain(D, Grayscale) = 0.037</th>
</tr>
</thead>
<tbody>
<tr>
<td>1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13</td>
<td>Ent(D) = 0.996</td>
</tr>
</tbody>
</table>

gray

<table>
<thead>
<tr>
<th>1, 4, 6, 10, 13</th>
<th>Ent(D₁) = 0.971</th>
</tr>
</thead>
</table>

dark gray

<table>
<thead>
<tr>
<th>2, 3, 7, 8, 12</th>
<th>Ent(D²) = 0.971</th>
</tr>
</thead>
</table>

light gray

<table>
<thead>
<tr>
<th>5, 9, 11</th>
<th>Ent(D³) = 0.918</th>
</tr>
</thead>
</table>
1. Decision Tree

1.2 The splitting criterion

**Information Gain**

<table>
<thead>
<tr>
<th>Latitude=?</th>
<th>Gain($D$, Latitude) = 0.291</th>
</tr>
</thead>
<tbody>
<tr>
<td>1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13</td>
<td>Ent($D$) = 0.996</td>
</tr>
</tbody>
</table>

- **High**
  - 1, 2, 3, 4, 10
  - Ent($D^1$) = 0.723

- **Middle**
  - 5, 6, 7, 8, 9, 12
  - Ent($D^2$) = 0.918

- **Low**
  - 11, 13
  - Ent($D^3$) = 0
1. Decision Tree

1.2 The splitting criterion

Information Gain

<table>
<thead>
<tr>
<th>Location=?</th>
<th>Gain(D, Location) = 0.004</th>
</tr>
</thead>
<tbody>
<tr>
<td>1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13</td>
<td>Ent(D) = 0.996</td>
</tr>
</tbody>
</table>

northern hemisphere

| 1, 2, 3, 8, 9, 11 | Ent(D^1) = 1 |

southern hemisphere

| 4, 5, 6, 7, 10, 12, 13 | Ent(D^2) = 0.985 |
1. Decision Tree

1.2 The splitting criterion

Information Gain

<table>
<thead>
<tr>
<th>Tilt angle=?</th>
<th>Gain($D$, Tilt angle) = 0.310</th>
</tr>
</thead>
<tbody>
<tr>
<td>1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13</td>
<td>Ent($D$) = 0.996</td>
</tr>
</tbody>
</table>

small

| 1, 5 | Ent($D^1$) = 0 |

medium

| 2, 3, 4, 6, 7, 9, 10, 11, 12 | Ent($D^2$) = 0.991 |

large

| 8, 13 | Ent($D^3$) = 0 |
1. Decision Tree

1.2 The splitting criterion

Information Gain

Barb number? = ?

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13

Ent(D) = 0.996

Gain(D, Barb number) = ?

many

1, 2, 3, 4, 12

Ent(D^1) = ?

some

5, 6, 7, 10

Ent(D^2) = ?

few

8, 9, 11, 13

Ent(D^3) = ?
1. Decision Tree

1.2 The splitting criterion

Information Gain

Barb number=?

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13

Gain(D, Barb number) = 0.410

Ent(D) = 0.996

many

1, 2, 3, 4, 12

Ent(D^1) = 0.723

some

5, 6, 7, 10

Ent(D^2) = 1

few

8, 9, 11, 13

Ent(D^3) = 0
1. Decision Tree

1.2 The splitting criterion

Information Gain

Gain($D$, Grayscale) = 0.037
Gain($D$, Latitude) = 0.291
Gain($D$, Location) = 0.004
Gain($D$, Barb number) = 0.410
Gain($D$, Tilt angle) = 0.310

Barb number = ?

Ent($D$) = 0.996

- many
  - 1, 2, 3, 4, 12
  - Ent($D^1$) = 0.723

- some
  - 5, 6, 7, 10
  - Ent($D^2$) = 1

- few
  - 8, 9, 11, 13
  - Ent($D^3$) = 0

$A_1 = A - \{"Barb number"\}$
1. Decision Tree

1.2 The splitting criterion

- **Barb number=?**
  - 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13
  - $\text{Ent}(D) = 0.996$

- **Latitude=?**
  - 1, 2, 3, 4, 12
  - $\text{Ent}(D^1) = 0.723$

- **Grayscale=?**
  - 5, 6, 7, 10
  - $\text{Ent}(D^2) = 1$

- **Not QF**
  - 8, 9, 11, 13
  - $\text{Ent}(D^3) = 0$

- **QF**
  - 1, 2, 3, 4
  - $\text{Ent}(D^{11}) = 0$

- **Not QF**
  - 12
  - $\text{Ent}(D^{12}) = 0$

- **Latitude=?**
  - 6, 10
  - $\text{Ent}(D^{21}) = 1$

- **Not QF**
  - 7
  - $\text{Ent}(D^{22}) = 0$

- **QF**
  - 5
  - $\text{Ent}(D^{23}) = 0$

- **Not QF**
  - 8, 9, 11, 13
  - $\text{Ent}(D^3) = 0$

- **QF**
  - 1, 2, 3, 4
  - $\text{Ent}(D^{11}) = 0$

- **Not QF**
  - 12
  - $\text{Ent}(D^{12}) = 0$
1. Decision Tree

Why growing decision trees via entropy instead of misclassification error (or classification accuracy)?
1. Decision Tree

Barb number=?

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13

Ent(D) = 0.996

Gain(D) = I(D) - \sum_{v=1}^{V} \frac{|D^v|}{|D|} I(D^v)
1. Decision Tree

<table>
<thead>
<tr>
<th>Barb number=?</th>
<th>1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ent(D)</td>
<td>0.996</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>many</th>
<th>1, 2, 3, 4, 12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ent(D^1)</td>
<td>0.723</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>some</th>
<th>5, 6, 7, 10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ent(D^2)</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>few</th>
<th>8, 9, 11, 13</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ent(D^3)</td>
<td>0</td>
</tr>
</tbody>
</table>

Gain(D, Barb number) = Ent(D) - \( \sum_{v=1}^{3} \frac{|D^v|}{|D|} \) Ent(D^v)

= 0.996 - (\( \frac{5}{13} \times 0.723 + \frac{4}{13} \times 1 + \frac{4}{13} \times 0 \)) = 0.410
1. Decision Tree

Barb number=?

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13

Err(D) = 7/13

many

1, 2, 3, 4, 12

Err(D^1) = 1/5

some

5, 6, 7, 10

Err(D^2) = 2/4

few

8, 9, 11, 13

Err(D^3) = 0/4

Gain(D, Barb number) = Err(D) - \( \sum_{v=1}^{3} \frac{|D^v|}{|D|} \operatorname{Err}(D^v) \)

= \( \frac{7}{13} - \left( \frac{5}{13} \times \frac{1}{5} + \frac{4}{13} \times \frac{2}{4} + \frac{4}{13} \times 0 \right) \) = 0
## 1. Decision Tree

### 1.2 The splitting criterion

<table>
<thead>
<tr>
<th>ID</th>
<th>Grayscale (Intensity)</th>
<th>Latitude</th>
<th>Location</th>
<th>Tilt Angle</th>
<th>Barb Number</th>
<th>Quiescent Filament</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>gray</td>
<td>high</td>
<td>northern hemisphere</td>
<td>small</td>
<td>many</td>
<td>Yes</td>
</tr>
<tr>
<td>2</td>
<td>dark gray</td>
<td>high</td>
<td>northern hemisphere</td>
<td>medium</td>
<td>many</td>
<td>Yes</td>
</tr>
<tr>
<td>3</td>
<td>dark gray</td>
<td>high</td>
<td>northern hemisphere</td>
<td>medium</td>
<td>many</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>gray</td>
<td>high</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>many</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>light gray</td>
<td>middle</td>
<td>southern hemisphere</td>
<td>small</td>
<td>some</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>gray</td>
<td>middle</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>some</td>
<td>Yes</td>
</tr>
<tr>
<td>7</td>
<td>dark gray</td>
<td>middle</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>some</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>dark gray</td>
<td>middle</td>
<td>northern hemisphere</td>
<td>large</td>
<td>few</td>
<td>No</td>
</tr>
<tr>
<td>9</td>
<td>light gray</td>
<td>middle</td>
<td>northern hemisphere</td>
<td>medium</td>
<td>few</td>
<td>No</td>
</tr>
<tr>
<td>10</td>
<td>gray</td>
<td>high</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>some</td>
<td>No</td>
</tr>
<tr>
<td>11</td>
<td>light gray</td>
<td>low</td>
<td>northern hemisphere</td>
<td>medium</td>
<td>few</td>
<td>No</td>
</tr>
<tr>
<td>12</td>
<td>dark gray</td>
<td>middle</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>many</td>
<td>No</td>
</tr>
<tr>
<td>13</td>
<td>gray</td>
<td>low</td>
<td>southern hemisphere</td>
<td>large</td>
<td>few</td>
<td>No</td>
</tr>
</tbody>
</table>
1. Decision Tree

1.2 The splitting criterion

Gain(D, Grayscale) = \( \text{Ent}(D) - \sum_{v=1}^{13} \frac{D^v}{D} \text{Ent}(D^v) \)

= 0.996 - \( \left( \frac{1}{13} \times 0 + \frac{1}{13} \times 0 + \cdots + \frac{1}{13} \times 0 + \frac{1}{13} \times 0 \right) = 0.996 \)
1. Decision Tree

1.2 The splitting criterion

Information Gain

$$\text{Gain}(D) = \text{Ent}(D) - \sum_{v=1}^{V} \frac{D^v}{D} \text{Ent}(D^v), v = 1, 2, ..., V$$

$$A = \{a_1, a_2, ..., a_u\}, a_* = \{a^1_*, a^2_*, ..., a^V_*\}$$

Gain Ratio

$$\text{GainRatio}(D) = \frac{\text{Gain}(D)}{\text{SplitInfo}(D)}$$

$$\text{SplitInfo}(D) = - \sum_{v=1}^{V} \frac{|D^v|}{|D|} \log_2 \frac{|D^v|}{|D|}, v = 1, 2, ..., V$$

C4.5
1. Decision Tree

1.2 The splitting criterion

**Gini Impurity**

\[
\text{Gini}(D) = \sum_{k=1}^{K} \sum_{k' \neq k} p_k p_{k'} = 1 - \sum_{k=1}^{K} p_k^2, k = 1, 2, \ldots, K
\]

**Gini Index**

\[
\text{GiniIndex}(D) = \sum_{v=1}^{V} \frac{|D^v|}{|D|} \text{Gini}(D^v), v = 1, 2, \ldots, V
\]

\[
A = \{a_1, a_2, \ldots, a_u\}, a_* = \{a_1^*, a_2^*, \ldots, a_*^v\}
\]
1. Decision Tree

1.2 The splitting criterion

Cite: Sebastian Raschka, Machine Learning
### 1. Decision Tree

#### 1.3 Pruning

<table>
<thead>
<tr>
<th>ID</th>
<th>Grayscale (Intensity)</th>
<th>Latitude</th>
<th>Location</th>
<th>Tilt Angle</th>
<th>Barb Number</th>
<th>Quiescent Filament</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>gray</td>
<td>high</td>
<td>northern hemisphere</td>
<td>small</td>
<td>many</td>
<td>Yes</td>
</tr>
<tr>
<td>2</td>
<td>dark gray</td>
<td>high</td>
<td>northern hemisphere</td>
<td>medium</td>
<td>many</td>
<td>Yes</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>11</td>
<td>light gray</td>
<td>low</td>
<td>northern hemisphere</td>
<td>medium</td>
<td>few</td>
<td>No</td>
</tr>
<tr>
<td>12</td>
<td>dark gray</td>
<td>middle</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>many</td>
<td>No</td>
</tr>
<tr>
<td>13</td>
<td>gray</td>
<td>low</td>
<td>southern hemisphere</td>
<td>large</td>
<td>few</td>
<td>No</td>
</tr>
<tr>
<td>14</td>
<td>gray</td>
<td>low</td>
<td>cross equator</td>
<td>medium</td>
<td>some</td>
<td>Yes</td>
</tr>
</tbody>
</table>

What effect on earlier tree?
1. Decision Tree

1.3 Pruning

Overfitting in decision tree

Barb number=?
1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13
Ent(D) = 0.996

many
Latitude=?
1, 2, 3, 4, 12
Ent(D^1) = 0.723

high
QF
1, 2, 3, 4
Ent(D^{11}) = 0

middle
Not QF
12
Ent(D^{12}) = 0

few
Not QF
8, 9, 11, 13
Ent(D^3) = 0

Overfitting in decision tree

some
Grayscale=?
5, 6, 7, 10
Ent(D^2) = 1

dark gray
Latitude=?
6, 10
Ent(D^{21}) = 1

high
QF
5
Ent(D^{23}) = 0

middle
Not QF
7
Ent(D^{22}) = 0

light gray
gray
Not QF
8, 9, 11, 13
Ent(D^3) = 0

Overfitting in decision tree

low
cross equator
medium
some
Yes

Overfitting in decision tree

14
gray
low
cross equator
medium
some
Yes
1. Decision Tree

1.3 Pruning

Overfitting in decision tree

Accuracy

Tree depth

Training accuracy

Test accuracy
## 1. Decision Tree

### 1.3 Pruning

<table>
<thead>
<tr>
<th>ID</th>
<th>Grayscale (Intensity)</th>
<th>Latitude</th>
<th>Location</th>
<th>Tilt Angle</th>
<th>Barb Number</th>
<th>Quiescent Filament</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>gray</td>
<td>high</td>
<td>northern hemisphere</td>
<td>small</td>
<td>many</td>
<td>Yes</td>
</tr>
<tr>
<td>3</td>
<td>dark gray</td>
<td>high</td>
<td>northern hemisphere</td>
<td>medium</td>
<td>many</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>light gray</td>
<td>middle</td>
<td>southern hemisphere</td>
<td>small</td>
<td>some</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>gray</td>
<td>middle</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>some</td>
<td>Yes</td>
</tr>
<tr>
<td>7</td>
<td>dark gray</td>
<td>middle</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>some</td>
<td>No</td>
</tr>
<tr>
<td>9</td>
<td>light gray</td>
<td>middle</td>
<td>northern hemisphere</td>
<td>large</td>
<td>few</td>
<td>No</td>
</tr>
<tr>
<td>10</td>
<td>gray</td>
<td>high</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>some</td>
<td>No</td>
</tr>
<tr>
<td>12</td>
<td>dark gray</td>
<td>middle</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>many</td>
<td>No</td>
</tr>
</tbody>
</table>

Split data into training and validation set.
1. Decision Tree

1.3 Pruning

**Pre-pruning**

- Set a depth cut-off (maximum tree depth) a priori;

- Cost-complexity pruning:
  - where is an impurity measure,
  - is a tuning parameter,
  - is the total number of nodes;

- **Stop growing if split is not statistically significant** (e.g. $\chi^2$);

- Set a minimum number of data points for each node.
1. Decision Tree

1.3 Pruning

Post-pruning

- **Grow full tree first, then remove nodes**, in C4.5

- Reduced-error pruning, remove nodes via validation set:
  - Greedily remove the node that most improves validation set accuracy
  - Problematic for limited data

- Can also convert trees to rules first and then prune the rules:
  - Convert tree to equivalent set of rules;
  - Prune each rule independently of others;
  - Sort final rules into desired sequence for use.
1. Decision Tree

1.4 Extensions to decision tree

Continuous Valued Attributes

<table>
<thead>
<tr>
<th>Spine length</th>
<th>Spine length</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&lt; 50</td>
</tr>
<tr>
<td>50.08</td>
<td>0</td>
</tr>
<tr>
<td>59.09</td>
<td>0</td>
</tr>
<tr>
<td>103.15</td>
<td>0</td>
</tr>
<tr>
<td>38.06</td>
<td>1</td>
</tr>
<tr>
<td>170.25</td>
<td>0</td>
</tr>
<tr>
<td>47.07</td>
<td>1</td>
</tr>
<tr>
<td>74.11</td>
<td>0</td>
</tr>
<tr>
<td>103.15</td>
<td>0</td>
</tr>
<tr>
<td>42.06</td>
<td>1</td>
</tr>
</tbody>
</table>

Turn numerical data into categorical: Only need consider 3 values.
1. Decision Tree

1.4 Extensions to decision tree

Multivariate decision tree

How would the decision tree split look like this?
1. Decision Tree

1.4 Extensions to decision tree

Regression trees
1. Decision Tree

1.4 Extensions to decision tree

**Unknown/missing Attribute Values**

Use training example anyway, sort through tree:

- If node $n$ tests $A$, assign most common value of $A$ among other examples sorted to node $n$;
- Assign most common value of $A$ among other examples with same target value;
- Assign probability to each possible value of $A$: assign fraction of example to each descendant in tree;

Classify new examples in same fashion.
1. Decision Tree

1.5 Pros and Cons

ID3 – Iterative Dichotomizer 3

One of the earliest decision tree algorithms.


- Cannot handle numeric features;
- No pruning, prone to overfitting;
- Short and wide trees (compared to CART);
- Maximizing information gain or minimizing entropy;
- Discrete features, binary and multi-category features.
1. Decision Tree

1.5 Pros and Cons

**C4.5**


- Continuous and discrete features;
- Continuous is very expensive, because must consider all possible ranges, bi-partition;
- Can handle missing attributes (ignores them in gain compute);
- Post-pruning (bottom-up pruning);
- Adopt gain ratio.
1. Decision Tree

1.5 Pros and Cons

**CART – Classification and Regression Tree**


- Continuous and discrete features;
- Strictly binary splits (taller trees than ID3 and C4.5);
- Binary splits can generate better trees than C4.5, but tend to be larger and harder to interpret;
- Variance reduction in regression trees;
- Adopt Gini impurity.
- Cost complexity pruning.
1. Decision Tree

1.5 Pros and Cons

😊 Easy to interpret and communicate;
😊 Can represent “complete” hypothesis space;
😊 Easy to overfit;
😊 Elaborate pruning required;
😊 Expensive to just fit a “diagonal line”;
😊 Output range is bounded (depend on training example) in regression trees.

Cite: Sebastian Raschka, Machine Learning
Ensemble learning
2. Ensemble learning

2.1 Voting

- **Unanimity**

- **Majority**

- **Plurality**
2. Ensemble learning

2.1 Voting

Majority Vote Classifier

\[
\hat{y}_f = \text{mode}\{h_1(x), h_2(x), \ldots, h_n(x)\}
\]

where \( h_i(x) = \hat{y}_i \)
2. Ensemble learning

2.1 Voting

Majority Vote Classifier

<table>
<thead>
<tr>
<th></th>
<th>T₁</th>
<th>T₂</th>
<th>T₃</th>
</tr>
</thead>
<tbody>
<tr>
<td>h₁</td>
<td>✔</td>
<td>✔</td>
<td>X</td>
</tr>
<tr>
<td>h₂</td>
<td>X</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>h₃</td>
<td>✔</td>
<td>X</td>
<td>✔</td>
</tr>
<tr>
<td>Ensemble</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
</tbody>
</table>

Example 1

<table>
<thead>
<tr>
<th></th>
<th>T₁</th>
<th>T₂</th>
<th>T₃</th>
</tr>
</thead>
<tbody>
<tr>
<td>h₁</td>
<td>✔</td>
<td>✔</td>
<td>X</td>
</tr>
<tr>
<td>h₂</td>
<td>X</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>h₃</td>
<td>✔</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Ensemble</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

Example 2

<table>
<thead>
<tr>
<th></th>
<th>T₁</th>
<th>T₂</th>
<th>T₃</th>
</tr>
</thead>
<tbody>
<tr>
<td>h₁</td>
<td>✔</td>
<td>✔</td>
<td>X</td>
</tr>
<tr>
<td>h₂</td>
<td>X</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>h₃</td>
<td>✔</td>
<td>X</td>
<td>✔</td>
</tr>
<tr>
<td>Ensemble</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

Example 3
2. Ensemble learning

2.1 Voting

Why Majority Vote?

- Assume \( n \) independent classifiers with a base error rate \( \epsilon \);
  - Independent means that the errors are uncorrelated;

- Assume a binary classification task;

- Assume the error rate is better than random guessing;
  - i.e., lower than 0.5 for binary classification.

- The probability that we make a wrong prediction via the ensemble if \( k \) classifiers predict the same class label:

\[
P(k) = \binom{n}{k} \epsilon^k (1 - \epsilon)^{n-k}, k > \lfloor n/2 \rfloor
\]
2. Ensemble learning

2.1 Voting

Why Majority Vote?

- The probability that we make a wrong prediction via the ensemble if \( k \) classifiers predict the same class label:

\[
P(k) = \binom{n}{k} \epsilon^k (1 - \epsilon)^{n-k}, k > \lceil n/2 \rceil
\]

- Ensemble error:

\[
\epsilon_{ens} = \sum_{k}^{n} \binom{n}{k} \epsilon^k (1 - \epsilon)^{n-k}, k > \lceil n/2 \rceil
\]
2. Ensemble learning

2.1 Voting

Why Majority Vote?

\[ \epsilon_{\text{ens}} = \sum_{k}^{n} \binom{n}{k} \epsilon^k (1 - \epsilon)^{n-k}, \ k > [n/2] \]
2. Ensemble learning

2.1 Voting

**Hard voting**

\[ \hat{y} = \arg \max_j \sum_{i=1}^{n} h_i^j(x) \]

\[ H(x) = \{h_1, h_2, \ldots, h_n\}, h_i^j(x) \in \{0, 1\} \]

\( h_i^j(x) \) is the predicted class membership of the \( i \) th classifier for class label \( j \)
2. Ensemble learning

2.1 Voting

**soft voting**

Use only for well-calibrated classifiers!

\[
\hat{y} = \arg \max_j \sum_{i=1}^n w_i h_i^j(x)
\]

\[
H(x) = \{h_1, h_2, \ldots, h_n\}, h_i^j(x) \in [0, 1]
\]

\(h_i^j(x)\) is the predicted class membership probability of the \(i\) th classifier for class label \(j\); \(w_i\) is the optional weighting parameter, \(w_i \geq 0, \sum_{i=1}^n w_i = 1\).
2. Ensemble learning

2.1 Voting

soft voting

Use only for well-calibrated classifiers!

Use only for well-calibrated classifiers!

<table>
<thead>
<tr>
<th>$H(x)$</th>
<th>$j = 0$</th>
<th>$j = 1$</th>
<th>$w_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$h_1$</td>
<td>0.9</td>
<td>0.1</td>
<td>0.2</td>
</tr>
<tr>
<td>$h_2$</td>
<td>0.8</td>
<td>0.2</td>
<td>0.3</td>
</tr>
<tr>
<td>$h_3$</td>
<td>0.3</td>
<td>0.7</td>
<td>0.5</td>
</tr>
</tbody>
</table>

$\hat{y} = \arg \max_j \sum_{i=1}^{n} w_i h_i^j(x)$

$j \in \{0, 1\}, i \in \{1, 2, 3\}$

$p(j = 0|x) = 0.2 \times 0.9 + 0.3 \times 0.8 + 0.3 \times 0.5 = 0.57$

$p(j = 1|x) = 0.2 \times 0.1 + 0.3 \times 0.2 + 0.3 \times 0.7 = 0.28$

$\hat{y} = \arg \max_j \sum_{i=1}^{n} w_i h_i^j(x) = \arg \max_j \{p(j = 0|x), p(j = 1|x)\} = 0$
2. Ensemble learning

2.2 Bagging (Bootstrap Aggregating)

Bagging in Pseudocode

Let \( n \) be the number of bootstrap samples

\[
\text{for } i = 1 \text{ to } n \text{ do } \\
\text{Draw bootstrap sample of size } m, D_i; \\
\text{Train base classifier } h_i \text{ on } D_i; \\
\hat{\gamma} = \text{mode}\{h_1, h_2, \ldots, h_n\}
\]
2. Ensemble learning

2.2 Bagging (Bootstrap Aggregating)

Bootstrap Sampling

Original Dataset: \(X_1 X_2 X_3 X_4 X_5 X_6 X_7 X_8 X_9 X_{10}\)

Bootstrap 1: \(X_8 X_6 X_2 X_9 X_5 X_8 X_1 X_4 X_8 X_2\)

Bootstrap 2: \(X_{10} X_1 X_3 X_5 X_1 X_7 X_4 X_2 X_1 X_8\)

Bootstrap 3: \(X_6 X_5 X_4 X_1 X_2 X_4 X_2 X_6 X_9 X_2\)

Training Sets:

\(X_3 X_7 X_{10}\)

\(X_6 X_9\)

\(X_3 X_7 X_8 X_{10}\)

Cite: Sebastian Raschka, Machine Learning
2. Ensemble learning

2.2 Bagging (Bootstrap Aggregating)

Bootstrap Sampling

\[ P(\text{not chosen}) = \left(1 - \frac{1}{n}\right)^n, \]

\[ \frac{1}{e} \approx 0.368, \quad n \to \infty. \]

\[ P(\text{chosen}) = 1 - \left(1 - \frac{1}{n}\right)^n \approx 0.632 \]
2. Ensemble learning

2.2 Bagging (Bootstrap Aggregating)

Bagging classifier

\[ \hat{y}_f = \text{mode}\{h_1(x), h_2(x), \ldots, h_n(x)\} \]

where \( h_i(x) = \hat{y}_i \)

Cite: Sebastian Raschka, Machine Learning
2. Ensemble learning

2.2 Bagging (Bootstrap Aggregating)
2. Ensemble learning

2.2 Bagging (Bootstrap Aggregating)

- **Black** curve: the true function $\hat{f}$
- **Red** curves: $f^*$ in 1000 training sets
- **Yellow** curve: the average of 1000 $f^* = \bar{f}$
2. Ensemble learning

2.2 Bagging (Bootstrap Aggregating)

Previous Lecture

\[ y = b + w_1 \cdot x_{date} \]

Small Variance

Large Variance
2. Ensemble learning

2.2 Bagging (Bootstrap Aggregating)

Error observed
- Error from bias
- Error from variance

Model Complexity

Underfitting
- Large Bias
- Small Variance

Overfitting
- Small Bias
- Large Variance
2. Ensemble learning

2.3 Boosting

Guarantee:
- If your ML algorithm can produce classifier with error rate smaller than 50% (accuracy larger than 50%) on training data;
- You can obtain 0% error rate classifier after boosting.
2. Ensemble learning

2.3 Boosting

General Boosting

- Obtain the first classifier $f_1(x)$;
- Find another function $f_2(x)$ to help $f_1(x)$;
  - However, if $f_2(x)$ is similar to $f_1(x)$, it will not help a lot;
  - We want $f_2(x)$ to be complementary with $f_1(x)$; **How?**
- Obtain the second classifier $f_2(x)$;
- Repeat until the stopping criterion; **When?**
- Finally, combining all the classifiers. **How?**

The classifiers are learned sequentially.
2. Ensemble learning

2.3 Boosting

How to obtain different classifier?

- Training on different training data sets.
- How to have different training data sets?
  - Re-sampling your training data to form a new set;
  - Re-weighting your training data to form a new set.

\[
L(f) = \sum_{n} l(f(x^n), \hat{y}^n)
\]

\[
L(f) = \sum_{n} w^n l(f(x^n), \hat{y}^n)
\]

In real implementation, you only have to change the cost (objective) function.
2. Ensemble learning

2.3 Boosting

- Adaptive Boosting
  - e.g., AdaBoost

- Gradient Boosting
  - LightGBM, XGBoost, scikit-learn’s GradientBoostingClassifier

- Differ mainly in terms of how
  - Weights are updated -- have different training data sets
  - Classifiers are combined
2. Ensemble learning

2.3 Boosting

**AdaBoost**

Idea: Training $f_2(x)$ on the new training set that fails $f_1(x)$

$\varepsilon_1$: the error rate of $f_1(x)$ on its training data,

$$\varepsilon_1 = \frac{\sum_{i=1}^{n} w_1^i \delta(f_1(x^i) \neq \hat{y}^i)}{Z_1}, \quad Z_1 = \sum_{i=1}^{n} w_1^i, \quad \varepsilon_1 < 0.5$$

Change the example weights from $w_1^i$ to $w_2^i$ by

$$\sum_{i=1}^{n} w_2^i \delta(f_1(x^i) \neq \hat{y}^i) = 0.5$$

Training $f_2(x)$ based on the new weights $w_2^i$.

The performance of $f_1$ for new weights would be random.
2. Ensemble learning

2.3 Boosting

**AdaBoost**

**Idea:** Training $f_2(x)$ on the new training set that fails $f_1(x)$

\[
\begin{align*}
(x^1, \hat{y}^1, w^1) w^1 &= 1 & w^1 &= 1/\sqrt{3} & \checkmark \\
(x^2, \hat{y}^2, w^2) w^2 &= 1 & w^2 &= \sqrt{3} & \times \\
(x^3, \hat{y}^3, w^3) w^3 &= 1 & w^3 &= 1/\sqrt{3} & \checkmark \\
(x^4, \hat{y}^4, w^4) w^4 &= 1 & w^4 &= 1/\sqrt{3} & \checkmark \\
\end{align*}
\]

$\varepsilon_1 = 0.25$ \quad $f_1(x)$ \quad $\varepsilon_1 = 0.5$ \quad $f_2(x)$ \quad $\varepsilon_2 < 0.5$
2. Ensemble learning

2.3 Boosting

AdaBoost

Idea: Training $f_2(x)$ on the new training set that fails $f_1(x)$

If $x^i$ misclassified by $f_1$ ($f_1(x^i) \neq \hat{y}^i$)

$$w_2^i \leftarrow w_1^i \text{ multiplying } d_1$$

Increase

If $x^i$ correctly classified by $f_1$ ($f_1(x^i) = \hat{y}^i$)

$$w_2^i \leftarrow w_1^i \text{ divided by } d_1$$

Decrease

What is the value of $d_1$?

$f_2$ will be learned based on sample weights $w_2^i$. 
2. Ensemble learning

2.3 Boosting

**AdaBoost**

Idea: Training $f_2(x)$ on the new training set that fails $f_1(x)$

$$
\varepsilon_1 = \frac{\sum_{i=1}^{n} w_1^i \delta(f_1(x^i) \neq \hat{y}^i)}{Z_1}, Z_1 = \sum_{i=1}^{n} w_1^i, \varepsilon_1 < 0.5
$$

$$
\sum_{i=1}^{n} w_2^i \delta(f_1(x^i) \neq \hat{y}^i) = 0.5, Z_2 = \sum_{i=1}^{n} w_2^i
$$

$$
\sum_{i=1}^{n} w_2^i \delta(f_1(x^i) \neq \hat{y}^i) = \sum_{f_1(x^i) \neq \hat{y}^i} w_1^i d_1
$$

$$
\sum_{f_1(x^i) = \hat{y}^i} w_2^i = \sum_{f_1(x^i) \neq \hat{y}^i} w_2^i + \sum_{f_1(x^i) = \hat{y}^i} w_1^i / d_1
$$

$$
f_1(x^i) \neq \hat{y}^i, w_2^i \leftarrow w_1^i \text{ multiplying } d_1\nonumber
$$

$$
f_1(x^i) = \hat{y}^i, w_2^i \leftarrow w_1^i \text{ divided by } d_1\nonumber$$

2. Ensemble learning

2.3 Boosting

**AdaBoost**

Idea: Training $f_2(x)$ on the new training set that fails $f_1(x)$

$$\frac{\sum_{i=1}^{n} w_2^i \delta(f_1(x^i) \neq \hat{y}^i)}{Z_2} = 0.5,$$

$$\frac{\sum_{f_1(x^i) \neq \hat{y}^i} w_1^i d_1 + \sum_{f_1(x^i) = \hat{y}^i} w_1^i/d_1}{\sum_{f_1(x^i) \neq \hat{y}^i} w_1^i d_1} = 2,$$

$$\frac{\sum_{f_1(x^i) = \hat{y}^i} w_1^i/d_1}{\sum_{f_1(x^i) \neq \hat{y}^i} w_1^i d_1} = 1$$

$$\sum_{f_1(x^i) = \hat{y}^i} w_1^i/d_1 = \sum_{f_1(x^i) \neq \hat{y}^i} w_1^i d_1,$$

$$\frac{1}{d_1} \sum_{f_1(x^i) = \hat{y}^i} w_1^i = d_1 \sum_{f_1(x^i) \neq \hat{y}^i} w_1^i,$$

$$\varepsilon_1 = \frac{\sum_{i=1}^{n} w_1^i \delta(f_1(x^i) \neq \hat{y}^i)}{Z_1} = \frac{\sum_{f_1(x^i) \neq \hat{y}^i} w_1^i}{Z_1}, \sum_{f_1(x^i) \neq \hat{y}^i} w_1^i = Z_1 \varepsilon_1$$
2. Ensemble learning

2.3 Boosting

AdaBoost

Idea: Training $f_2(x)$ on the new training set that fails $f_1(x)$

$$
\sum_{f_1(x^i)=\hat{y}^i} w_1^i/d_1 = \sum_{f_1(x^i)\neq \hat{y}^i} w_1^i d_1, \quad \frac{1}{d_1} \sum_{f_1(x^i)=\hat{y}^i} w_1^i = d_1 \sum_{f_1(x^i)\neq \hat{y}^i} w_1^i,
$$

$$
\varepsilon_1 = \frac{\sum_{i=1}^n w_1^i \delta(f_1(x^i) \neq \hat{y}^i)}{Z_1} = \frac{\sum_{f_1(x^i) \neq \hat{y}^i} w_1^i}{Z_1},
$$

$$
\sum_{f_1(x^i)\neq \hat{y}^i} w_1^i = Z_1 \varepsilon_1 \quad \sum_{f_1(x^i)=\hat{y}^i} w_1^i = Z_1 (1 - \varepsilon_1)
$$

$$
Z_1 (1 - \varepsilon_1) / d_1 = Z_1 \varepsilon_1 d_1 \quad d_1 = \sqrt{(1 - \varepsilon_1) / \varepsilon_1} > 1
$$
2. Ensemble learning

2.3 Boosting

Initialize $k$ : the number of AdaBoost rounds
Initialize $D$ : the training dataset, $D = \{(x^1, \hat{y}^1), (x^2, \hat{y}^2) \ldots (x^n, \hat{y}^n)\}$
Initialize $w^i_1$: $w^i_1 = \frac{1}{n}, i = 1, 2, \ldots, n$

For $r = 1$ to $k$ do

For all $i$ : $w^i_r = \frac{w^i_r}{\sum_i w^i_r}$ Normalize weights

$f_r(x)$: = FitWeakLearner($D, w_r$);

$\varepsilon_r$ = $\sum_i w^i_r \delta (f_1(x^i) \neq \hat{y}^i)$; Compute error

If $\varepsilon_r > 1/2$ then stop;

$d_r = \sqrt{(1 - \varepsilon_r)/\varepsilon_r}$; Small if error is large and vice versa

$w^i_{r+1} = w_r \times \begin{cases} d_r & \text{if } f_r(x^i) \neq \hat{y}^i \\ \frac{1}{d_r} & \text{if } f_r(x^i) = \hat{y}^i \end{cases}$

$\alpha_r = \frac{1}{2} \ln \sqrt{(1 - \varepsilon_r)/\varepsilon_r}$;

$w_{r+1} = w_r \times \begin{cases} e^{\alpha_r} & \text{if } f_r(x^i) \neq \hat{y}^i \\ e^{-\alpha_r} & \text{if } f_r(x^i) = \hat{y}^i \end{cases}$

Predict: $H(x) = \text{sign}\left(\sum_r^k \alpha_r f_r(x)\right)$
2. Ensemble learning

2.3 Boosting

**Stopping Criteria**

- Stop if classification is perfect;
- Stop if the error is at least as bad as random guessing (e.g. \( \frac{1}{2} \) for binary classification);
- Stop if the number of rounds meets \( k \);

**combining all the classifiers**

- Uniform weight: \( H(x) = \text{sign} \left( \sum_{r=1}^{k} f_r(x) \right) \);
- Non-uniform weight: \( H(x) = \text{sign} \left( \sum_{r}^{k} \alpha_r f_r(x) \right) \).
2. Ensemble learning

2.3 Boosting

AdaBoost

Smaller error $\varepsilon$, larger weight for final voting

$\varepsilon = 0.1 \rightarrow \alpha \approx 0.55$

$\varepsilon = 0.4 \rightarrow \alpha \approx 0.10$

$\varepsilon_r = 0.2 \times 0 + 0.2 \times 0 + 0.2 \times 0 + 0.2 \times 1 + 0.2 \times 1 = \frac{2}{5} = 0.4$, $r = 1$

$\alpha_r = \frac{1}{2} \ln \frac{1 - \varepsilon_r}{\varepsilon_r} = \frac{1}{2} \ln \frac{1 - 0.4}{0.4} \approx 0.1$

if $f_r(x^i) \neq \hat{y}^i$ $w_{r+1} = w_re^{\alpha_r} = 0.2 \times e^{0.1} \approx 0.22$

if $f_r(x^i) = \hat{y}^i$ $w_{r+1} = w_re^{-\alpha_r} = 0.2 \times e^{-0.1} \approx 0.18$

$\sum_{i} w^i_r = 3 \times 0.18 + 2 \times 0.22 = 0.98$

$\frac{0.18}{0.98} \approx 0.18$

$\frac{0.22}{0.98} = 0.22$
2. Ensemble learning

2.3 Boosting

AdaBoost

**Decision stump**: simple decision tree with 1 splitting rule based on thresholding 1 feature.

**Weak classifier**: Considering decision tree stump for binary classification problem with labels -1, 1
2. Ensemble learning

2.3 Boosting

AdaBoost

Cite: Sebastian Raschka, Machine Learning
2. Ensemble learning

2.3 Boosting

Conceptual overview based on decision trees

Gradient Boosting

- **Step 1**: Construct a base tree (just the root node)
- **Step 2**: Build next tree based on errors of the previous tree
- **Step 3**: Combine tree from step 1 with trees from step 2. Go back to step 2.
2. Ensemble learning

2.3 Boosting

Gradient Boosting

Conceptual overview based on decision trees

<table>
<thead>
<tr>
<th>$X_0$= ID</th>
<th>$X_1$= Grayscale</th>
<th>$X_2$= Length</th>
<th>$X_3$= Barb Number</th>
<th>$Y$= Latitude</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>200</td>
<td>500</td>
<td>10</td>
<td>60</td>
</tr>
<tr>
<td>2</td>
<td>185</td>
<td>450</td>
<td>8</td>
<td>70</td>
</tr>
<tr>
<td>3</td>
<td>145</td>
<td>620</td>
<td>12</td>
<td>65</td>
</tr>
<tr>
<td>4</td>
<td>195</td>
<td>150</td>
<td>2</td>
<td>30</td>
</tr>
</tbody>
</table>

Step 1: Construct a base tree (just the root node):

$$y_1^* = \frac{1}{n} \sum_{i=1}^{n} y^i = \frac{1}{4} (60 + 70 + 65 + 30) = 56.25$$
2. Ensemble learning

2.3 Boosting

Gradient Boosting

Step 2: Build next tree based on errors of the previous tree:

First, compute (pseudo) residuals, suppose \( r_1 = y_1 - y_1^* \)

<table>
<thead>
<tr>
<th></th>
<th>( X_0 = ) ID</th>
<th>( X_1 = ) Grayscale</th>
<th>( X_2 = ) Length</th>
<th>( X_3 = ) Barb Number</th>
<th>( Y = ) Latitude</th>
<th>( r_1 = ) residual</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>200</td>
<td>500</td>
<td>10</td>
<td>60</td>
<td>60-56.25=3.75</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>185</td>
<td>450</td>
<td>8</td>
<td>70</td>
<td>70-56.25=13.75</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>145</td>
<td>620</td>
<td>12</td>
<td>65</td>
<td>65-56.25=8.75</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>195</td>
<td>150</td>
<td>2</td>
<td>30</td>
<td>30-56.75=-26.75</td>
<td></td>
</tr>
</tbody>
</table>
2. Ensemble learning

2.3 Boosting

Gradient Boosting

Conceptual overview based on decision trees

Step 2: Build next tree based on errors of the previous tree:

Then, create a tree based on $x_1, x_2, \ldots, x_m$ to fit the residuals

<table>
<thead>
<tr>
<th>ID</th>
<th>ID</th>
<th>Grayscale</th>
<th>Length</th>
<th>Barb Number</th>
<th>Latitude</th>
<th>residual</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>200</td>
<td>500</td>
<td>10</td>
<td>60</td>
<td>3.75</td>
</tr>
<tr>
<td>2</td>
<td>185</td>
<td>450</td>
<td>8</td>
<td>70</td>
<td>13.75</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>145</td>
<td>620</td>
<td>12</td>
<td>65</td>
<td>8.75</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>195</td>
<td>150</td>
<td>2</td>
<td>30</td>
<td>-26.75</td>
<td></td>
</tr>
</tbody>
</table>

Barb Number ≥ 5

- Yes
- No

Length ≥ 500

- Yes
- No

6.25
- 3.75
- 8.75
- 13.75
- -26.75
2. Ensemble learning

2.3 Boosting

Gradient Boosting

Step 3: Combine tree from step 1 with tree:

\[ y_1^* = \frac{1}{n} \sum_{i=1}^{n} y_i^i = 56.25 \]

E.g. predict ID=3: \( 56.25 + \alpha \times 6.25 \)

where \( \alpha \) learning rate between 0 and 1 (If \( \alpha = 1 \), low bias but high variance)
2. Ensemble learning

2.3 Boosting

**Gradient Boosting**

**Initial function** \( g_0(x) = 0 \)

**For** \( t = 1 \) to \( T \) **do**

Find a function \( f_t(x) \) and \( \alpha_t \) to improve \( g_{t-1}(x) \)

\[
\begin{align*}
g_{t-1}(x) &= \sum_{i=1}^{t-1} \alpha_i f_i(x) \\
g_t(x) &= g_{t-1}(x) + \alpha_t f_t(x)
\end{align*}
\]

**Predict:** \( H(x) = \text{sign}(g_T(x)) \)

What is the learning target of \( g(x) \)?

**Maximize** \( \hat{y}^n g(x^n) \)

**Minimize** \( L(g) = \sum_{n} l(\hat{y}^n, g(x^n)) = \sum_{n} \exp(-\hat{y}^n g(x^n)) \)

Cite: Hung-yi Lee, Machine Learning
2. Ensemble learning

2.3 Boosting

Gradient Boosting

Find $g(x)$, minimize $L(g) = \sum_n \exp(-\hat{y}^n g(x^n))$

If we already have $g(x) = g_{t-1}(x)$, how to update $g(x)$?

Gradient Descent:

$$g_t \leftarrow g_{t-1} - \eta \nabla L(g_{t-1}) - \sum_n \exp(-\hat{y}^n g_{t-1}(x^n))(-\hat{y}^n)$$

$$g_t(x) = g_{t-1}(x) + \alpha_t f_t(x)$$
2. Ensemble learning

2.3 Boosting

Gradient Boosting

We want to find $f_t(x)$ maximizing

$$
\sum_n \exp(-\hat{y}^n g_t(x^n))(\hat{y}^n)
$$

```
Cite: Hung-yi Lee, Machine Learning
```
2. Ensemble learning

2.3 Boosting

Gradient Boosting

Find $g(x)$, minimize $L(g) = \sum_n \exp(-\hat{y}^n g(x^n))$

$$g_t(x) = g_{t-1}(x) + \alpha_t f_t(x) \quad \alpha_t \text{ is something like learning rate}$$

Find $\alpha_t$ minimizing $L(g)$

$$L(g) = \sum_n \exp(-\hat{y}^n g(x^n)) = \sum_n \exp \left( -\hat{y}^n (g_{t-1}(x^n) + \alpha_t f_t(x^n)) \right) =$$

$$\sum_n \exp \left( -\hat{y}^n (g_{t-1}(x^n)) \right) \exp \left( -\hat{y}^n (\alpha_t f_t(x^n)) \right) =$$

$$\sum_{f_t(x^n) \neq \hat{y}^n} \exp(-\hat{y}^n g_{t-1}(x^n)) \exp(\alpha_t) + \sum_{f_t(x^n) = \hat{y}^n} \exp(-\hat{y}^n g_{t-1}(x^n)) \exp(-\alpha_t)$$

Let $\partial L(g)/\partial \alpha_t = 0$

$$\alpha_t = \ln \sqrt{\frac{1 - \varepsilon_r}{\varepsilon_r}}$$

Cite: Hung-yi Lee, Machine Learning
2. Ensemble learning

2.4 Random Forests

Trees VS Forests

- A single tree may over fit to the training data;
- Instead train multiple trees and combine their predictions;
- Each tree can differ in both training data and node tests;
- Achieve this by injecting randomness into training algorithm.
2. Ensemble learning

2.4 Random Forests

Random Forests =

Bagging of decision trees +

random feature subsets
## 2. Ensemble learning

### 2.4 Random Forests

**Bootstrap Sampling**

<table>
<thead>
<tr>
<th>ID</th>
<th>Grayscale (Intensity)</th>
<th>Latitude</th>
<th>Location</th>
<th>Tilt Angle</th>
<th>Barb Number</th>
<th>Quiescent Filament</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>gray</td>
<td>high</td>
<td>northern hemisphere</td>
<td>small</td>
<td>many</td>
<td>Yes</td>
</tr>
<tr>
<td>2</td>
<td>dark gray</td>
<td>high</td>
<td>northern hemisphere</td>
<td>medium</td>
<td>many</td>
<td>Yes</td>
</tr>
<tr>
<td>3</td>
<td>dark gray</td>
<td>high</td>
<td>northern hemisphere</td>
<td>medium</td>
<td>many</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>gray</td>
<td>high</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>many</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>light gray</td>
<td>middle</td>
<td>southern hemisphere</td>
<td>small</td>
<td>some</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>gray</td>
<td>middle</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>some</td>
<td>yes</td>
</tr>
<tr>
<td>7</td>
<td>dark gray</td>
<td>middle</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>some</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>dark gray</td>
<td>middle</td>
<td>northern hemisphere</td>
<td>large</td>
<td>few</td>
<td>No</td>
</tr>
<tr>
<td>9</td>
<td>light gray</td>
<td>middle</td>
<td>northern hemisphere</td>
<td>medium</td>
<td>few</td>
<td>No</td>
</tr>
<tr>
<td>10</td>
<td>gray</td>
<td>high</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>some</td>
<td>No</td>
</tr>
<tr>
<td>11</td>
<td>light gray</td>
<td>low</td>
<td>northern hemisphere</td>
<td>medium</td>
<td>few</td>
<td>No</td>
</tr>
<tr>
<td>12</td>
<td>dark gray</td>
<td>middle</td>
<td>southern hemisphere</td>
<td>medium</td>
<td>many</td>
<td>No</td>
</tr>
<tr>
<td>13</td>
<td>gray</td>
<td>low</td>
<td>southern hemisphere</td>
<td>large</td>
<td>few</td>
<td>No</td>
</tr>
</tbody>
</table>
2. Ensemble learning

2.4 Random Forests

Bootstrap Sampling

Original Dataset: 1 2 3 4 5 6 7 8 9 10 11 12 13

Bootstrap 1: 1 2 6 3 5 10 7 13

Bootstrap 2: 2 6 4 4 3 6 11 9

Bootstrap 3: 5 3 3 13 5 12 7 10

Training set
2. Ensemble learning

2.4 Random Forests

Random feature subset for each tree of node:

Original Features: ID, Grayscale, Latitude, Location, Tilt Angle, Barb Number

Subset 1: ID, Grayscale, Latitude, Location

Subset 2: Latitude, Location, Tilt Angle, Barb Number

Subset 3: ID, Grayscale, Location, Barb Number

Splitting feature set

Number of features in subset = \( \log_2 m + 1 \), where \( m \) is the number of input features.
2. Ensemble learning

2.4 Random Forests

😊 Random forests average a set of deep decision trees;

- Tend to be one of the best “out of the box” classifiers;
- Often close to the best performance of any method on the first run;

😊 Predictions are very fast;

- E.g. compare with Bagging considering all features, random forest only considering a subset;

☹ Many parameters: depth of tree, number of trees, type of node tests, random sampling;

☹ Requires a lot of training data and large memory footprint.
2. Ensemble learning

2.4 Random Forests

Application--Body tracking in Microsoft Kinect for XBox 360

Input data for each frame

RGB image

Depth image

Multi-class classification

Inferred body parts

Output

Left hand

Neck

Right shoulder

Left elbow

Fit stickman model and track skeleton

2. Ensemble learning

2.4 Random Forests

Application – Body tracking in Microsoft Kinect for XBox 360

Input data point: $v = (x, y)$ (pixel position in 2D image)

Output:
- $c \in \{\text{left-hand, right-hand, head, r.-foot, r.-shoulder...}\}$

Feature response:
$$f(v, \theta_j) = I(v) - I(p_j) \quad \text{(depth difference between two points)}$$

Nodes tests

2. Ensemble learning

2.4 Random Forests

**Application—Predict solar flares**

<table>
<thead>
<tr>
<th>SHARP Keyword</th>
<th>Formula</th>
<th>Unit</th>
<th>RF Importance</th>
<th>B Class (n = 128)</th>
<th>C Class (n = 552)</th>
<th>M Class (n = 142)</th>
<th>X Class (n = 23)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TOTUSJH</td>
<td>(H_{\text{total}} \propto \sum</td>
<td>B_z \cdot J_z</td>
<td>)</td>
<td>(10^5 \text{ G}^2 \text{ m}^{-1})</td>
<td>37.4</td>
<td>4.8 ± 3.1</td>
<td>13.9 ± 9.9</td>
</tr>
<tr>
<td>TOTBSQ</td>
<td>(F \propto \sum B^2)</td>
<td>(10^{10} \text{ G}^2)</td>
<td>17.9</td>
<td>1.0 ± 0.9</td>
<td>2.6 ± 1.9</td>
<td>4.6 ± 3.0</td>
<td>10.7 ± 8.6</td>
</tr>
<tr>
<td>TOTPOT</td>
<td>(\rho_{\text{tot}} \propto \sum (B_{\text{obs}} - B_{\text{pot}})^2 dA)</td>
<td>(10^{23} \text{ ergs cm}^{-3})</td>
<td>21.1</td>
<td>1.0 ± 1.4</td>
<td>2.7 ± 2.7</td>
<td>6.7 ± 5.7</td>
<td>19.6 ± 18.0</td>
</tr>
<tr>
<td>TOTUSIJZ</td>
<td>(J_{\text{total}} = \sum</td>
<td>J_z</td>
<td>dA)</td>
<td>(10^{12} \text{ A})</td>
<td>50.6</td>
<td>9.5 ± 6.4</td>
<td>30.3 ± 21.4</td>
</tr>
<tr>
<td>ABSNIZH</td>
<td>(H_{\text{abs}} \propto \sum</td>
<td>B_z \cdot J_z</td>
<td>)</td>
<td>(10 \text{ G}^2 \text{ m}^{-1})</td>
<td>19.9</td>
<td>6.1 ± 7.0</td>
<td>14.3 ± 17.0</td>
</tr>
<tr>
<td>SAVNCPP</td>
<td>(J_{\text{sum}} \propto</td>
<td>\sum B_z^2 J_z dA</td>
<td>+</td>
<td>\sum B_z J_z dA</td>
<td>)</td>
<td>(10^{12} \text{ A})</td>
<td>24.6</td>
</tr>
<tr>
<td>USFLUX</td>
<td>(\Phi = \sum</td>
<td>B_z</td>
<td>dA)</td>
<td>(10^{21} \text{ Mx})</td>
<td>14.2</td>
<td>7.1 ± 5.5</td>
<td>19.9 ± 14.7</td>
</tr>
<tr>
<td>AREA_ACR</td>
<td>(\text{Area} = \sum \text{Pixels})</td>
<td>(10^2 \text{ pixels})</td>
<td>23.7</td>
<td>3.0 ± 2.4</td>
<td>8.2 ± 6.1</td>
<td>13.3 ± 7.7</td>
<td>29.2 ± 22.3</td>
</tr>
<tr>
<td>TOTFZ</td>
<td>(F_z \propto \sum (B_z^2 + B_y^2 - B_z^2) dA)</td>
<td>(-10^{23} \text{ dyne})</td>
<td>13.9</td>
<td>1.2 ± 1.3</td>
<td>2.7 ± 2.7</td>
<td>3.9 ± 3.7</td>
<td>6.1 ± 6.2</td>
</tr>
<tr>
<td>MEANPOT</td>
<td>(\bar{\rho} \propto \frac{1}{N} \sum (B_{\text{obs}} - B_{\text{pot}})^2)</td>
<td>(10^3 \text{ ergs cm}^{-3})</td>
<td>19.8</td>
<td>6.5 ± 5.8</td>
<td>5.9 ± 3.7</td>
<td>8.9 ± 4.2</td>
<td>12.1 ± 4.2</td>
</tr>
<tr>
<td>R_VALUE</td>
<td>(\Phi = \sum</td>
<td>B_{\text{LOS}}</td>
<td>dA \text{ within } R \text{ mask})</td>
<td>(\text{Mx})</td>
<td>31.4</td>
<td>3.2 ± 0.7</td>
<td>3.8 ± 0.6</td>
</tr>
<tr>
<td>EPSZ</td>
<td>(\delta F_z \propto \sum (B_z^2 + B_y^2 - B_z^2) / \sum B^2)</td>
<td>(-10^{-1})</td>
<td>15.4</td>
<td>2.1 ± 1.3</td>
<td>2.0 ± 1.3</td>
<td>1.7 ± 1.2</td>
<td>1.2 ± 1.1</td>
</tr>
<tr>
<td>SHRGT45</td>
<td>(\text{Area with shear } &gt; 45^\circ / \text{Total Area})</td>
<td></td>
<td>12.7</td>
<td>0.23 ± 0.17</td>
<td>0.27 ± 0.14</td>
<td>0.34 ± 0.13</td>
<td>0.40 ± 0.11</td>
</tr>
</tbody>
</table>

2. Ensemble learning

2.4 Random Forests

**Application**—Predict solar flares

2. Ensemble learning

2.4 Random Forests

Application — Predict solar flares

Fit another classifier that uses the predictions as features.
2. Ensemble learning

2.5 Stacking

Cite: Sebastian Raschka, Machine Learning
2. Ensemble learning

2.5 Stacking

Diagram:

- Input: $X = [x^1, x^2, x^3, ..., x^n]$
- Models:
  - Model 1
  - Model 2
  - Model 3
  - Model 4
- Outputs:
  - $y_1$
  - $y_2$
  - $y_3$
  - $y_4$

Combination:

- Majority Vote
2. Ensemble learning

2.5 Stacking

![Diagram of ensemble learning]

- **Model 1**: $y_1$
- **Model 2**: $y_2$
- **Model 3**: $y_3$
- **Model 4**: $y_4$

- **Final Classifier**

- **Training Data**
- **Validation Data**

- **X**: Features $x^1, x^2, x^3, \ldots, x^n$

- **y**: Predictions $y_1, y_2, y_3, y_4$

- **As new feature**

- Holdout Validation
- Leave-One-Out CV
- k-fold CV
To learn more

Mlxtend (machine learning extensions)

http://rasbt.github.io/mlxtend/#examples
http://rasbt.github.io/mlxtend/installation/
https://anaconda.org/conda-forge/mlxtend

Open Terminal:

```bash
>>Anaconda prompt
>>conda install -c conda-forge mlxtend
```
To learn more

Mlxtend (machine learning extensions)

```python
from sklearn import datasets
import numpy as np
iris = datasets.load_iris()
X = iris.data[:, [2, 3]]
y = iris.target

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.3, random_state=1, stratify=y)

%matplotlib inline
import matplotlib.pyplot as plt
from sklearn.tree import DecisionTreeClassifier
from mlxtend.plotting import plot_decision_regions

tree = DecisionTreeClassifier(criterion='entropy',
                               max_depth=2,
                               random_state=1)
tree.fit(X_train, y_train)

plot_decision_regions(X_train, y_train, tree)

plt.xlabel('petal length [cm]')
plt.ylabel('petal width [cm]')
plt.legend(loc='upper left')
plt.tight_layout()
plt.savefig('images/mlxtend_tree.png', dpi=300)
plt.savefig('images/mlxtend_tree.pdf')
plt.show()
```
Ensemble Demos

- http://rasbt.github.io/mlxtend/user_guide/classifier/EnsembleVoteClassifier/
- http://rasbt.github.io/mlxtend/user_guide/classifier/StackingClassifier/
- http://rasbt.github.io/mlxtend/user_guide/classifier/StackingCVClassifier/

Cite: Sebastian Raschka, Machine Learning
To learn more

LightGBM, Light Gradient Boosting Machine  https://github.com/Microsoft/LightGBM

- Faster training speed and higher efficiency.
- Lower memory usage.
- Better accuracy.
- Support of parallel and GPU learning.
- Capable of handling large-scale data.

sklearn.ensemble

- **Major Feature** Add two new implementations of gradient boosting trees: `ensemble.HistGradientBoostingClassifier` and `ensemble.HistGradientBoostingRegressor`. The implementation of these estimators is inspired by LightGBM and can be orders of magnitude faster than `ensemble.GradientBoostingRegressor` and `ensemble.GradientBoostingClassifier` when the number of samples is larger than tens of thousands of samples. The API of these new estimators is slightly different, and some of the features from `ensemble.GradientBoostingClassifier` and `ensemble.GradientBoostingRegressor` are not yet supported.

https://scikit-learn.org/stable/whats_new.html#version-0-21-0

Cite: Sebastian Raschka, Machine Learning
Decision Tree and Ensemble learning

Hao, Qi
School of Astronomy and Space Science

THANKS