Decision Tree
Training

M. R. Hasan
Readings

• Bishop: 14.4
• Murphy: 16, 16.1, 16.2.1, 16.2.2, 16.2.3, 16.2.4
• Geron: chapter 6
What We Will Cover

• Training (Growing) a Decision Tree
• Classification and Regression Trees (CART) Algorithm
• Impurity Measures (Classification)
  - Misclassification Rate
  - Entropy & Information Gain
  - Gini Index
Algorithm for Growing (Training) a Decision Tree
We want to predict the MPG.

Consider the following example to understand the main idea of the algorithm for training decision trees.

<table>
<thead>
<tr>
<th>mpg</th>
<th>cylinders</th>
<th>displacement</th>
<th>horsepower</th>
<th>weight</th>
<th>acceleration</th>
<th>modelyear</th>
<th>maker</th>
</tr>
</thead>
<tbody>
<tr>
<td>good</td>
<td>4 low</td>
<td>low</td>
<td>low</td>
<td>high</td>
<td>75to78</td>
<td>asia</td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>6 medium</td>
<td>medium</td>
<td>medium</td>
<td>medium</td>
<td>70to74</td>
<td>america</td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>4 medium</td>
<td>medium</td>
<td>medium</td>
<td>low</td>
<td>75to78</td>
<td>europe</td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>8 high</td>
<td>high</td>
<td>high</td>
<td>low</td>
<td>70to74</td>
<td>america</td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>6 medium</td>
<td>medium</td>
<td>medium</td>
<td>medium</td>
<td>70to74</td>
<td>america</td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>4 low</td>
<td>medium</td>
<td>low</td>
<td>medium</td>
<td>70to74</td>
<td>asia</td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>4 low</td>
<td>medium</td>
<td>low</td>
<td>low</td>
<td>70to74</td>
<td>asia</td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>8 high</td>
<td>high</td>
<td>high</td>
<td>low</td>
<td>75to78</td>
<td>america</td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>8 high</td>
<td>high</td>
<td>high</td>
<td>low</td>
<td>75to78</td>
<td>america</td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>4 low</td>
<td>low</td>
<td>low</td>
<td>low</td>
<td>79to83</td>
<td>america</td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>6 medium</td>
<td>medium</td>
<td>medium</td>
<td>high</td>
<td>75to78</td>
<td>america</td>
<td></td>
</tr>
<tr>
<td>good</td>
<td>4 medium</td>
<td>low</td>
<td>low</td>
<td>low</td>
<td>79to83</td>
<td>america</td>
<td></td>
</tr>
<tr>
<td>good</td>
<td>4 low</td>
<td>low</td>
<td>medium</td>
<td>high</td>
<td>79to83</td>
<td>america</td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>8 high</td>
<td>high</td>
<td>high</td>
<td>low</td>
<td>70to74</td>
<td>america</td>
<td></td>
</tr>
<tr>
<td>good</td>
<td>4 low</td>
<td>medium</td>
<td>low</td>
<td>medium</td>
<td>75to78</td>
<td>europe</td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>5 medium</td>
<td>medium</td>
<td>medium</td>
<td>medium</td>
<td>75to78</td>
<td>europe</td>
<td></td>
</tr>
</tbody>
</table>

40 Records
Training Decision Tree

- A **Decision Stump** (tree with depth 1).

![Decision Tree Diagram](image)
The final tree

mpg values: bad  good

root
22 18

Cylinders = 3
0 0
Predict bad

Cylinders = 4
4 17
Predict bad

Cylinders = 5
1 0
Predict bad

Cylinders = 6
8 0
Predict bad

Cylinders = 8
9 1
Predict bad

Maker = america
0 10
Predict good

Maker = asia
2 5
Predict good

Maker = europe
2 2
Predict good

Horsepower = low
0 0
Predict bad

Horsepower = medium
0 1
Predict bad

Horsepower = high
9 0
Predict bad

Acceleration = low
0 0
Predict bad

Acceleration = medium
1 0
Predict bad

Acceleration = high
1 1
Predict good

Model Year = 70 to 74
0 0
Predict bad

Model Year = 75 to 78
1 0
Predict bad

Model Year = 79 to 83
0 0
Predict bad

Table:

<table>
<thead>
<tr>
<th>mpg</th>
<th>cylinders</th>
<th>displacement</th>
<th>horsepower</th>
<th>weight</th>
<th>acceleration</th>
<th>modelyear</th>
<th>maker</th>
</tr>
</thead>
<tbody>
<tr>
<td>good</td>
<td>4</td>
<td>low</td>
<td>low</td>
<td>low</td>
<td>high</td>
<td>79078</td>
<td>asia</td>
</tr>
<tr>
<td>bad</td>
<td>6</td>
<td>medium</td>
<td>medium</td>
<td>medium</td>
<td>medium</td>
<td>70074</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>4</td>
<td>medium</td>
<td>medium</td>
<td>low</td>
<td>75078</td>
<td>europe</td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>8</td>
<td>high</td>
<td>high</td>
<td>low</td>
<td>70074</td>
<td>america</td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>6</td>
<td>medium</td>
<td>medium</td>
<td>medium</td>
<td>75078</td>
<td>america</td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>4</td>
<td>low</td>
<td>low</td>
<td>low</td>
<td>79083</td>
<td>america</td>
<td></td>
</tr>
<tr>
<td>good</td>
<td>4</td>
<td>low</td>
<td>low</td>
<td>low</td>
<td>79083</td>
<td>america</td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>6</td>
<td>medium</td>
<td>medium</td>
<td>high</td>
<td>75078</td>
<td>america</td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>4</td>
<td>low</td>
<td>medium</td>
<td>low</td>
<td>79083</td>
<td>america</td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>8</td>
<td>high</td>
<td>high</td>
<td>low</td>
<td>70074</td>
<td>america</td>
<td></td>
</tr>
<tr>
<td>good</td>
<td>4</td>
<td>low</td>
<td>medium</td>
<td>high</td>
<td>75078</td>
<td>europe</td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>5</td>
<td>medium</td>
<td>medium</td>
<td>medium</td>
<td>75078</td>
<td>europe</td>
<td></td>
</tr>
</tbody>
</table>

40 Records
Training Decision Tree

• Let’s discuss **how to train or grow** a decision tree.
• We will consider the following **two issues**:
  - How to choose the best feature/attribute?
  - When to stop?

```
<table>
<thead>
<tr>
<th>mpg</th>
<th>cylinders</th>
<th>displacement</th>
<th>horsepower</th>
<th>weight</th>
<th>acceleration</th>
<th>modelyear</th>
<th>maker</th>
</tr>
</thead>
<tbody>
<tr>
<td>good</td>
<td>4</td>
<td>low</td>
<td>low</td>
<td>high</td>
<td>75to78</td>
<td>asia</td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>6</td>
<td>medium</td>
<td>medium</td>
<td>medium</td>
<td>70to74</td>
<td>america</td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>4</td>
<td>medium</td>
<td>medium</td>
<td>low</td>
<td>75to78</td>
<td>europe</td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>8</td>
<td>high</td>
<td>high</td>
<td>low</td>
<td>70to74</td>
<td>america</td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>6</td>
<td>medium</td>
<td>medium</td>
<td>medium</td>
<td>70to74</td>
<td>america</td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>4</td>
<td>low</td>
<td>medium</td>
<td>low</td>
<td>70to74</td>
<td>asia</td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>4</td>
<td>medium</td>
<td>low</td>
<td>low</td>
<td>70to74</td>
<td>asia</td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>8</td>
<td>high</td>
<td>high</td>
<td>low</td>
<td>75to78</td>
<td>america</td>
<td></td>
</tr>
</tbody>
</table>
```
40 Records
Training Decision Tree

• Resort to a **greedy heuristic**:
  - Start from an empty decision tree.
  - Split on **next best attribute** (feature).
  - Recurse.

Classification and Regression Trees (CART) Algorithm

- Iterative Dichotomizer (ID3)
- C4.5 (ID3 + improvements)
Learning Decision Tree: CART

- The **Classification and Regression Trees** (CART) algorithm was proposed by *Breiman* et al. in 1984.
Learning Decision Tree: CART

- In CART, the sequential decision making process corresponds to the traversal of a **binary tree**.

ID3 generalizes CART by producing **more than two children**.
Learning Decision Tree: CART

• The first node asks if $x_1$ is less than some threshold $t_1$.
• If yes, we then ask if $x_2$ is less than some other threshold $t_2$.
• If yes, we are in the bottom left quadrant of space, $R_1$.
• If no, we ask if $x_1$ is less than $t_3$.
• And so on.
Learning Decision Tree: CART

• The result of these **axis parallel splits** is to partition 2d space into 5 regions.

• We can now **associate a mean response with each of these regions**, resulting in the piecewise constant surface.
Training Decision Tree

- Original Dataset
- mpg values: bad good
  - root
  - 22 18

- Cylinders = 3: Predict bad
- Cylinders = 4: Predict good
- Cylinders = 5: Predict bad
- Cylinders = 6: Predict bad
- Cylinders = 8: Predict bad

- Records in which cylinders = 4
- Records in which cylinders = 5
- Records in which cylinders = 6
- Records in which cylinders = 8

- And partition it according to the value of the attribute we split on.
Training Decision Tree: Recursion Step
Recursively build a tree from the seven records in which there are four cylinders and the maker was based in Asia.
Training Decision Tree

The final tree
Learning Decision Tree: CART

• Finding the optimal partitioning of the data is \textit{NP-complete}.

• So it is common to use the \textit{greedy procedure}.
Training Decision Tree

- At each step of the CART, we need to use the **best attribute/feature** to make the split.
- How do we choose a **good attribute/feature**?
- What’s the best feature?
Training Decision Tree

• Best feature *reduces impurity/uncertainty*!
• The idea is really quite simple:
• The algorithm first splits the training set in two subsets using a **single feature** $j$ and a **threshold** $t$ (e.g., “petal length $\leq 2.45$ cm”).

---

Diagram:
- $X_1 \leq t_1$
- $X_2 \leq t_2$
- $X_1 \leq t_3$
- $X_2 \leq t_4$
- $R_1$
- $R_2$
- $R_3$
- $R_4$
- $R_5$

**Feature $j$:**
- petal length

**Threshold $t$:**
- $2.45$

**Children nodes**:
- **True**:
  - petal length (cm) $\leq 2.45$
  - gini = 0.6667
  - samples = 150
  - value = [50, 50, 50]
  - class = setosa
- **False**:
  - petal width (cm) $\leq 1.75$
  - gini = 0.5
  - samples = 100
  - value = [0, 50, 50]
  - class = versicolor

---

**Leaf nodes**:
- $R_1$
  - gini = 0.0
  - samples = 50
  - value = [50, 0, 0]
  - class = setosa
- $R_2$
  - gini = 0.168
  - samples = 54
  - value = [0, 49, 5]
  - class = versicolor
- $R_3$
  - gini = 0.0425
  - samples = 46
  - value = [0, 1, 45]
  - class = virginica
Training Decision Tree

• How does it choose feature $j$ and threshold $t$?
• It searches for the pair $(j, t)$ that produces the purest subsets (weighted by their size).
• The algorithm tries to minimize a cost function.

\[
\text{cost}(j, t) = \frac{|D_L|}{|D|} \text{cost}(D_L) + \frac{|D_R|}{|D|} \text{cost}(D_R)
\]

$|D|_{L/R}$: number of instances in the left/right leaf & $|D| = |D|_L + |D|_R$

Cost($D_{L/R}$): measures the cost of the left/right leaf.
Training Decision Tree

- Typically we define the **gain** of using a feature to be a normalized measure of the **reduction in cost**.

\[ cost(j, t) = \frac{|D_L|}{|D|} \text{cost}(D_L) + \frac{|D_R|}{|D|} \text{cost}(D_R) \]

- \(|D|_{L/R}\): **number of instances** in the left/right leaf & \( |D| = |D|_L + |D|_R \)

- Cost\((D_{L/R})\): measures the **cost** of the left/right leaf.
Training Decision Tree

- Once it has successfully splitted the training set in two, **it splits the subsets using the same logic**, then the sub-subsets and so on, recursively.
- It **stops recursing** once it reaches the **maximum depth**, or if it cannot find a split that will reduce impurity.

\[
\text{cost}(j, t) = \frac{|D_L|}{|D|} \text{cost}(D_L) + \frac{|D_R|}{|D|} \text{cost}(D_R)
\]

- \(|D|_L/R\): **number of instances** in the left/right leaf & \(|D| = |D|_L + |D|_R\)
- \(\text{Cost}(D_{L/R})\): measures the **cost** of the left/right leaf.
How do we specify the cost measure used to evaluate the quality of a proposed split?

This depends on whether our goal is regression or classification.

\[
\text{cost}(j, t) = \frac{|D_L|}{|D|} \text{cost}(D_L) + \frac{|D_R|}{|D|} \text{cost}(D_R)
\]

\(|D|_L/R: \text{number of instances}\) in the left/right leaf & \(|D| = |D|_L + |D|_R\)

\(\text{Cost}(D_{L/R}): \text{measures the cost of the left/right leaf.}\)
How do we specify the cost measure used to evaluate the quality of a proposed split?

\[
\text{cost}(j, t) = \frac{|D_L|}{|D|} \text{cost}(D_L) + \frac{|D_R|}{|D|} \text{cost}(D_R)
\]

|\( |D|_{L/R} \): number of instances in the left/right leaf & \( |D| = |D|_L + |D|_R \)

Cost(\(D_{L/R}\)): measures the cost of the left/right leaf.
Training Decision Tree: Regression Cost

- **Regression Cost**: 
- In the *regression* setting, we define the cost as follows:

\[
\text{cost}(D) = \sum_{i \in D} (y_i - \bar{y})^2
\]

Here \(\bar{y} = \frac{1}{|D|} \sum_{i \in D} y_i\)

D: number of instances in the leaf nodes

It is the **mean of the response variable** in the specified set of data.
Training Decision Tree: Classification Cost

• **Classification Cost:**

• In the classification setting, there are several ways to measure the quality of a split.
  - Misclassification Rate
  - Entropy and/or Information Gain
  - Gini Index
Training Decision Tree: Classification Cost

- **Classification Cost**: Misclassification Rate

- Misclassification rate = \( \frac{\text{Incorrect Predictions}}{\text{All Predictions}} = \frac{FP+FN}{TN+FN+FP+TP} \)
• However, there is a **problem with misclassification rate**.
• Consider a two-class (X & Y) problem with **400 cases in each class**.
• Suppose one split created the nodes (300, 100) and (100, 300).
• The other split created the nodes (200, 400) and (200, 0).
• Both splits produce a **misclassification rate of 0.25**.

\[
\text{Misclassification rate} = \frac{\text{Incorrect Predictions}}{\text{All Predictions}} = \frac{FP+FN}{TN+FN+FP+TP}
\]
Training Decision Tree: Classification Cost

• **Classification Cost**: Entropy or Information Gain
• We fit a **Multinoulli model** to the data in the leaf satisfying the test $X_j < t$ by estimating the **class-conditional probabilities** as follows:

$$\hat{p}_c = \frac{1}{|D|} \sum_{i \in D} \text{No. of data points belonging to class } c$$

Here, $D$ is the **data in the leaf**

- Light boxes: WillWait is true
- Dark boxes: WillWait is false
Training Decision Tree: Classification Cost

- **Classification Cost**: Entropy or Information Gain
- Then, **entropy** is defined as follows:

\[
\mathcal{H}(\hat{p}) = - \sum_{c=1}^{C} \hat{p}_c \log \hat{p}_c
\]

\[\hat{p}_c = \frac{1}{|D|} \sum_{i \in D} \text{No. of data points belonging to class } c\]

It vanishes for \( p_c = 0 \) and \( p_c = 1 \) and have a maximum at \( p_c = 0.5 \).

Note: minimizing entropy is equivalent to **maximizing information gain**.
Mutual Information or Information Gain

• Another metric to choose the best feature in a decision tree is **mutual information** or **information gain**.

• What is information gain?
• Recall Entropy $H(X) = -\mathbb{E} \left[ \log P(X) \right] = -\sum_x P(X = x) \log P(X = x)$
• It measures how **uncertain are we about $X$**?

**Classification Cost:**
• In the classification setting, there are several ways to measure the quality of a split.
  - Misclassification Rate
  - Entropy and/or Information Gain
  - Gini Index
Mutual Information or Information Gain

• If X has a **joint distribution** with Y, then we want to know whether observing Y reduces our uncertainty about X?
• In other words, after observing Y, **do we gain information** about observing X?
• We use information gain as a measure for this.

After observing Y (Patrons), do we **gain information** about observing X (WillWait)?

Light boxes: WillWait is true

Dark boxes: WillWait is false
Information Gain

- We will use the following two concepts to define Information gain:
  - Specific Conditional Entropy
  - Conditional Entropy
Information Gain

- **Specific Conditional Entropy:**
- It measures how uncertain are we about X if we know that Y takes a **specific state y**.
- \[ H(X | Y = y) = -\sum_x P(X = x | Y = y)\log P(X = x | Y = y) \]

After observing a **specific state** of Y (Patrons = *None*), does our uncertainty about observing X (WillWait) reduce?
Information Gain

- **Conditional Entropy:**
- It measures how uncertain are we about \( X \) if we know that we observed **some random values of** \( Y \) (**not just a specific state** of \( Y \)).
- I.e., if we conduct an experiment in which outcomes \( Y = y \), then what will our confusion about \( X \) be?

![Diagram showing the relationship between patrons and whether they will wait]

Y: Patrons
X: WillWait

Light boxes: WillWait is true
Dark boxes: WillWait is false

After **observing** \( Y \) (Patrons), does our uncertainty about observing \( X \) (WillWait) reduce?
Information Gain

• **Conditional Entropy:**

• In other words, conditional entropy measures the expectation of the specific conditional entropy $H(X \mid Y = y)$:

\[ H(X \mid Y) = E \left[ H(X \mid Y = y) \right] \]

\[ H(X \mid Y) = -\sum_y P(Y = y)H(X \mid Y = y) \]

Specific Conditional Entropy:

\[ H(X \mid Y = y) = -\sum_x P(X = x \mid Y = y)\log P(X = x \mid Y = y) \]
Information Gain

• Now based on entropy and conditional entropy, we define **mutual information** or **information gain** as

\[ I(X, Y) = H(X) - H(X | Y) \]

• \( H(X) \): Uncertainty in X
• \( H(X | Y) \): Uncertainty in X **if Y is known**

I(X, Y) represents the **reduction in uncertainty (entropy)** about X as a consequence of the new observation Y

If X and Y are independent, then there is no reduction in entropy (**no info gain**)!
Information Gain

- $I(X, Y) = H(X) - H(X | Y)$
- $H(X)$: Uncertainty in $X$
- $H(X | Y)$: Uncertainty in $X$ if $Y$ is known

$I(X, Y)$ represents the reduction in uncertainty (entropy) about $X$ as a consequence of the new observation $Y$.

Y: Patrons
X: WillWait

After observing $Y$ (Patrons), do we gain information about observing $X$ (WillWait)?

Light boxes: WillWait is true
Dark boxes: WillWait is false
Information Gain

• The concept of information gain can also be approached by using KL divergence.
• How do we determine whether X and Y are independent?
• Now consider the joint distribution between two sets of variables x and y given by p(x, y).
• If the sets of variables are independent, then their joint distribution will factorize into the product of their marginals p(x, y) = p(x)p(y).

If the KL divergence between p(x, y) and p(x)p(y) is zero, then X and Y are independent.
Information Gain

However, if the variables are not independent, we can gain some idea of whether they are “close” to being independent by considering the KL divergence between the joint distribution and the product of the marginals, given by:

\[ KL(p \parallel q) = \int_{-\infty}^{\infty} p(x) \log \frac{p(x)}{q(x)} \, dx \]

\[ I[x, y] \equiv KL(p(x, y) \parallel p(x)p(y)) \]
\[ = -\iint p(x, y) \ln \left( \frac{p(x)p(y)}{p(x, y)} \right) \, dx \, dy \]
Information Gain

- It is called the **information gain** between the variables $x$ and $y$.
- From the properties of the KL divergence, we see that $I(x, y) \geq 0$ with equality if, and only if, $x$ and $y$ are independent.

$$I[x, y] = \frac{KL(p(x, y)\|p(x)p(y))}{\int \int p(x, y) \ln \left( \frac{p(x)p(y)}{p(x, y)} \right) \, dx \, dy}$$

$I[x, y] = 0$ represents the fact the $p(y)$ and $p(x)$ are independent and that observing $y$ does not change our uncertainty about $x$. 
Information Gain

- Thus, we can view the mutual information as the reduction in the uncertainty about \( x \) by virtue of being told the value of \( y \) (or vice versa).

\[
I[x, y] \equiv KL(p(x, y) \| p(x)p(y)) \\
= - \iint p(x, y) \ln \left( \frac{p(x)p(y)}{p(x, y)} \right) \, dx \, dy
\]
Training Decision Tree: Classification Cost

- **Summary:**
- Information Gain: $I(Y, X) = H(Y) - H(Y | X)$
- $H(Y)$: Uncertainty in $Y$
- $H(Y | X)$: Uncertainty in $Y$ *if $X$ is known*.
- $I \rightarrow$ drop in uncertainty
- A good feature will have **max. info gain or min conditional entropy**.
- Let’s see an example.
Training Decision Tree: Classification Cost

- Recall the Restaurant example: the random variable $Y$ is "WillWait" which is either "yes" or "no"
- The entropy of the random variable $Y$ is: $H(Y) = -\sum_i P(y_i) \log P(y_i)$
- $H(WillWait) = -P(\text{yes}) \log P(\text{yes}) - P(\text{no}) \log P(\text{no})$
- $H(WillWait) = -\frac{6}{12} \log \frac{6}{12} - \frac{6}{12} \log \frac{6}{12} = \frac{1}{2}(-1) - \frac{1}{2}(-1) = 0.5 + 0.5 = 1$

Light boxes:
WillWait = Yes

Dark boxes:
WillWait = No
Training Decision Tree: Classification Cost

- **Conditional Entropy for the attribute “Patrons”:**
  - $H(Y \mid X) = -\sum_x P(X = x)H(Y \mid X = x)$
  - $H(\text{WillWait} \mid \text{Patrons}) = -\{P(\text{None})H(\text{Yes, No}) + P(\text{Some})H(\text{Yes, No}) + P(\text{Full})H(\text{Yes, No}) \}$
  - $H(\text{WillWait} \mid \text{Patrons}) = -\{2/12 H(0, 2)_{\text{None}} + 4/12 H(4, 0)_{\text{Some}} + 6/12 H(2, 4)_{\text{Full}} \}$

![Diagram](image)
Training Decision Tree: Classification Cost

- Conditional Entropy for the attribute “Patrons”:
  \[
  H(\text{WillWait} \mid \text{Patrons}) = -\{ \frac{2}{12}[0/2 \log(0/2) + 2/2 \log(2/2)]_{\text{None}} + \frac{4}{12}[4/4 \log(4/4) + 0/4\log(0/4)]_{\text{Some}} + \frac{6}{12}[2/6 \log(2/6) + 4/6\log(4/6)]_{\text{Full}} \} = -\{ \frac{1}{6}[0 + 0]_{\text{None}} + 1[0 + 0]_{\text{Some}} + \frac{1}{2}[-0.53 - 0.39]_{\text{Full}} \} = -\{\frac{1}{2}(-0.92)\} = 0.46
  \]

- \( I(\text{WillWait, Patrons}) = H(\text{WillWait}) - H(\text{WillWait} \mid \text{Patrons}) \)
  \[
  = 1 - 0.46 = 0.54
  \]
Training Decision Tree: Classification Cost

• Conditional Entropy for the attribute “Type”:

\[ H(Y | X) = -\sum_x P(X = x)H(Y | X = x) \]

\[ H(\text{WillWait} | \text{Type}) = -\{ P(\text{French})H(Y, N) + P(\text{Italian})H(Y, N) + P(\text{Thai})H(Y, N) + P(\text{Burger})H(Y, N) \} \]

\[ H(\text{WillWait} | \text{Type}) = -\{2/12 H(1, 1)_{\text{French}} + 2/12 H(1, 1)_{\text{Italian}} + 4/12 H(2, 2)_{\text{Thai}} + 4/12 H(2, 2)_{\text{Burger}} \} \]

Light boxes:
WillWait = Yes

Dark boxes:
WillWait = No
Training Decision Tree: Classification Cost

• Conditional Entropy for the attribute “Type”:
  
  \[ H(\text{WillWait} \mid \text{Type}) = -\{1/6 \left[ 1/2 \log(1/2) + 1/2 \log(1/2)\right]_{\text{French}} + 1/6 \left[ 1/2 \log(1/2) + 1/2 \log(1/2)\right]_{\text{Italian}} + 1/3 \left[ 2/4 \log(2/4) + 2/4 \log(2/4)\right]_{\text{Thai}} + 1/3 \left[ 2/4 \log(2/4) + 2/4 \log(2/4)\right]_{\text{Burger}} \} \]

• \[ H(\text{WillWait} \mid \text{Type}) = -\{1/6 [-0.5 - 0.5]_{\text{French}} + 1/6 [-0.5 - 0.5]_{\text{Italian}} + 1/3 [-0.5 - 0.5]_{\text{Thai}} + 1/3 [-0.5 - 0.5]_{\text{Burger}} \} \]

• \[ H(\text{WillWait} \mid \text{Type}) = -\{-0.17 - 0.17 - 0.33 - 0.33\} = -(-1.0) = 1 \]

• \[ I(\text{WillWait, Type}) = H(\text{WillWait}) - H(\text{WillWait} \mid \text{Type}) = 1 - 1 = 0 \]
Training Decision Tree: Classification Cost

- **Conclusion:**
  - $I(\text{WillWait, Type}) = 0$
  - $I(\text{WillWait, Patrons}) = 0.54$
  - After observing “Patrons” our *uncertainty about “WillWait” reduces!*
  - Therefore, “**Patrons**” is a better attribute to split on.
Training Decision Tree

- Look at all information gains:

<table>
<thead>
<tr>
<th>mpg</th>
<th>cylinders</th>
<th>displacement</th>
<th>horsepower</th>
<th>weight</th>
<th>acceleration</th>
<th>model.year</th>
<th>maker</th>
</tr>
</thead>
<tbody>
<tr>
<td>good</td>
<td>4</td>
<td>low</td>
<td>low</td>
<td>low</td>
<td>high</td>
<td>75to78</td>
<td>asia</td>
</tr>
<tr>
<td>bad</td>
<td>6</td>
<td>medium</td>
<td>medium</td>
<td>medium</td>
<td>medium</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>4</td>
<td>medium</td>
<td>medium</td>
<td>medium</td>
<td>low</td>
<td>75to78</td>
<td>europe</td>
</tr>
<tr>
<td>bad</td>
<td>8</td>
<td>high</td>
<td>high</td>
<td>high</td>
<td>low</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>6</td>
<td>medium</td>
<td>medium</td>
<td>medium</td>
<td>low</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>4</td>
<td>low</td>
<td>medium</td>
<td>medium</td>
<td>low</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>4</td>
<td>low</td>
<td>medium</td>
<td>low</td>
<td>low</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>8</td>
<td>high</td>
<td>high</td>
<td>low</td>
<td>low</td>
<td>75to78</td>
<td>asia</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bad</td>
<td>8</td>
<td>high</td>
<td>high</td>
<td>low</td>
<td>low</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>8</td>
<td>high</td>
<td>medium</td>
<td>high</td>
<td>high</td>
<td>79to83</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>8</td>
<td>high</td>
<td>high</td>
<td>high</td>
<td>low</td>
<td>75to78</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>4</td>
<td>low</td>
<td>low</td>
<td>low</td>
<td>low</td>
<td>79to83</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>6</td>
<td>medium</td>
<td>medium</td>
<td>high</td>
<td>high</td>
<td>75to78</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>4</td>
<td>medium</td>
<td>medium</td>
<td>high</td>
<td>low</td>
<td>79to83</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>4</td>
<td>low</td>
<td>medium</td>
<td>high</td>
<td>low</td>
<td>79to83</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>8</td>
<td>high</td>
<td>high</td>
<td>low</td>
<td>low</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>4</td>
<td>low</td>
<td>medium</td>
<td>low</td>
<td>medium</td>
<td>75to78</td>
<td>europe</td>
</tr>
<tr>
<td>bad</td>
<td>5</td>
<td>medium</td>
<td>medium</td>
<td>medium</td>
<td>low</td>
<td>75to78</td>
<td>europe</td>
</tr>
</tbody>
</table>

40 Records
Training Decision Tree

• Illustration: Use of information gain to **make the best split**: 

Observe the **Info. Gain** from both horizontal (split 1) and vertical (split 2) splits.
Training Decision Tree: Classification Cost

- **Another impurity measure**: Gini Index
- A node is split by selecting a threshold that reduces the Gini index (impurity).
Training Decision Tree: Classification Cost

- Gini Index:
- We fit a Multinoulli model to the data in the leaf satisfying the test $X_j < t$ by estimating the class-conditional probabilities as follows:

\[ \hat{p}_c = \frac{1}{|D|} \sum_{i \in D} \text{No. of data points belonging to class c} \]

Here, $D$ is the data in the leaf.
Training Decision Tree: Classification Cost

- **Classification Cost**: Gini Index
- Then, Gini impurity index is given by:

\[
Gini\ Index = \sum_{c=1}^{C} \hat{p}_c (1 - \hat{p}_c) = \sum_{c=1}^{C} \hat{p}_c - \sum_{c=1}^{C} \hat{p}_c^2
\]

\[
Gini\ Index = 1 - \sum_{c=1}^{C} \hat{p}_c^2
\]

\[
\hat{p}_c = \frac{1}{|D|} \sum_{i \in D} No.\ of\ data\ points\ belonging\ to\ class\ c
\]
Training Decision Tree: Classification Cost

- **Classification Cost**: Gini Index
- Just like Entropy, gini index vanishes for $p_c = 0$ and $p_c = 1$ and have a maximum at $p_c = 0.5$.

Both the entropy and the Gini index encourage the formation of regions in which a high proportion of the data points are assigned to one class.

\[
Gini\ Index = 1 - \sum_{c=1}^{C} \hat{p}_c^2
\]

\[
\mathcal{H}(\hat{p}) = -\sum_{c=1}^{C} \hat{p}_c \log \hat{p}_c
\]
**Decision Tree**

- Gini attribute measures **impurity of a node**.

A node is “pure” (**gini = 0**) if all training instances it applies to **belong to the same class**.

\[
Gini \text{ Index} = 1 - \sum_{c=1}^{C} \hat{p}_c^2
\]

Gini =
1 – (50/50)^2 – (0/50)^2 – (0/50)^2 = 0

Gini =
1 – (0/54)^2 – (49/54)^2 – (5/54)^2 = 0.168

\[
\hat{p}_c = \frac{1}{|D|} \sum_{i \in D} \text{No. of data points belonging to class } c
\]
Comparison of Three Impurity Measures for Classification
Comparison of Three Impurity Measures for Classification

• Comparison of **three impurity measures** for a **two-class** case.
• The misclassification rate is $1 - \max(p, 1 - p)$, the entropy is $H(p)$, and the Gini index is $2p(1 - p)$.

![Graph showing comparison of impurity measures with X-axis: node probability]
Comparison of Three Impurity Measures for Classification

- The **entropy and the Gini index are better measures** than the misclassification rate for growing the tree.
- Because they are **more sensitive to the node probabilities**.
Gini Impurity vs Entropy

- Should we use Gini impurity or entropy?
- Most of the time it **does not make a big difference.**
- They lead to **similar trees.**

Gini impurity is **slightly faster** to compute, so it is a good default.

However, when they differ, Gini impurity tends to **isolate the most frequent class** in its own branch of the tree, while entropy tends to produce **slightly more balanced trees.**

X-axis: node probability
Training Decision Tree

• Next question that we need to resolve is how do we decide when to stop growing the tree?