University of Colorado **Boulder**

CSCI 5622: Machine Learning

Lecture 10

Overview

- Information entropy
- Decision Trees
  - Terminology (e.g. nodes, etc) & intuition
  - Entropy node splitting criterion
  - Algorithm Outline
  - Pruning
  - Regression Trees
- Random Forests

Overview

- Information entropy
- Decision Trees
  - Terminology (e.g. nodes, etc) & intuition
  - Entropy node splitting criterion
  - Algorithm Outline
  - Pruning
  - Regression Trees
- Random Forests

Information entropy

AAAAAAAA

Bucket 1

Suppose that we have a bucket with 8 letters, all being ‘A’. Suppose that we draw a letter from the bucket and the result is ‘A’. How surprised are you?

Information entropy

AAAAAAAA

Bucket 1

Suppose that we have a bucket with 8 letters, all being 'A'. Suppose that we draw a letter from the bucket and the result is 'A'. How surprised are you?

Not at all (100% chance of getting 'A')

Information entropy

AAAABBCD

Bucket 2

Suppose that we have a bucket with 8 letters, 4 'A', 2 'B', 1 'C', and 1 'D'. Suppose that we draw a letter from the bucket and the result is 'A'. How surprised are you?

Information entropy

AAAABBCD

Bucket 2

Suppose that we have a bucket with 8 letters, 4 ‘A’, 2 ‘B’, 1 ‘C’, and 1 ‘D’. Suppose that we draw a letter from the bucket and the result is ‘A’. How surprised are you?

A little bit (50% chance of getting ‘A’)

Information entropy

AABBCCDD

Bucket 3

Suppose that we have a bucket with 8 letters, 2 ‘A’, 2 ‘B’, 2 ‘C’, and 2 ‘D’. Suppose that we draw a letter from the bucket and the result is ‘A’. How surprised are you?

Information entropy

AABBCCDD

Bucket 3

Suppose that we have a bucket with 8 letters, 2 'A', 2 'B', 2 'C', and 2 'D'. Suppose that we draw a letter from the bucket and the result is 'A'. How surprised are you?

A lot (25% chance of getting 'A')

Information entropy

- Information theory is a field of study concerned with quantifying information for communication
- Information entropy is a concept from information theory that measures the uncertainty or randomness in a set of data or information
- Information entropy quantifies the average amount of information contained in a message, signal, or data source

Information entropy

AAAAAAAA	AAAABBCD	AABBCCDD
Bucket 1	Bucket 2	Bucket 3

Suppose that we draw a letter from the above buckets.
On average what is the minimum number of questions do we need to ask to find out what letter it is?

Information entropy

AAAAAAAA	AAAABBCD	AABBCCDD
Bucket 1	Bucket 2	Bucket 3

For Bucket 1:

- We don't need to ask any questions, since it only contains the letter A.
- Average Number of Questions $= 0$

Information entropy

AAAAAAAA	AAAABBCD	AABBCCDD
Bucket 1	Bucket 2	Bucket 3

For Buckets 2 and 3:

- Naively we could ask 4 questions
- Is the letter A? Is the letter B? Is the letter C? Is the letter D?
- **Can we do better than that?**

Information entropy

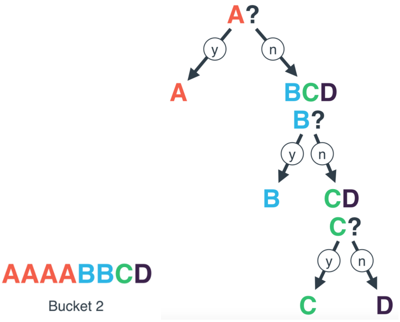
For Bucket 2:

- **We notice that 50% of letters are A**
- We can take advantage of this by asking "Is the letter A" in our first question.
- If the answer to the first question is "Yes", then we will have found the letter in 1 question (instead of 4).

Information entropy

For Bucket 2:

- If the letter is A, we can find out in 1 question
- If the letter is B, we can find out in 2 questions
- If the letter is C, we can find out in 3 questions
- Average Number of Questions $= \frac{1}{2} \cdot 1 + \frac{1}{4} \cdot 2 + \frac{1}{8} \cdot 3 + \frac{1}{8} \cdot 3 = 1.75$



Information entropy

For Bucket 3:

- We need 2 questions for any letter

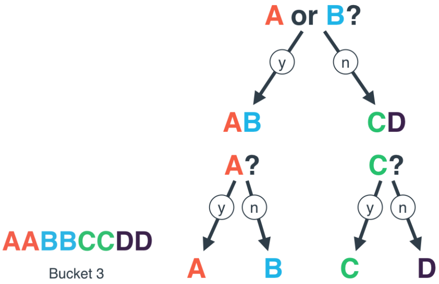
AABBCCDD

Bucket 3

Information entropy

For Bucket 3:

- We need 2 questions for any letter
- Average Number of Questions $= \frac{1}{4} \cdot 2 + \frac{1}{4} \cdot 2 + \frac{1}{4} \cdot 2 + \frac{1}{4} \cdot 2 = 2$



Information entropy

AAAAAAAA	AAAABBCD	AABBCCDD
Bucket 1	Bucket 2	Bucket 3
Avg No. Questions = 0	**Avg No. Questions = 1.75**	**Avg No. Questions = 2**

- If we want to find out a letter drawn out of a bucket, the average number of questions we must ask to find out what is this letter, is the entropy of the set (if we ask our questions in the smartest possible way).
- Entropy is a measure of uncertainty
- Entropy $= 0 \rightarrow$ no uncertainty

[Watch this: `https://www.youtube.com/watch?v=fRed0Xmc2Wg`]

Entropy for discrete distribution

Entropy is typically measured in bits and is calculated using the probability distribution of the possible outcomes of the data.
Let X be a discrete random variable with $\{x_1, \ldots, x_N\}$ outcomes, each occurring with probability $p(x_1), \ldots, p(x_N)$.
The information content of outcome x_n is inversely proportional to its probability, $h(x_n) = \log \frac{1}{p(x_n)} = -\log p(x_n)$
The entropy of the random variable X is the average information content of the outcomes:

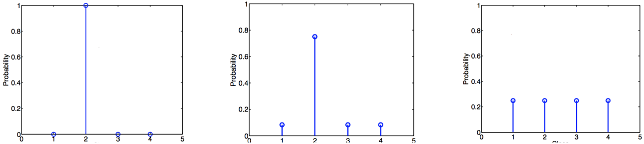
$$H(X) = \sum_{n=1}^{N} p(x_n) \log(\frac{1}{p(x_n)}) = -\sum_{n=1}^{N} p(x_n) \log(p(x_n))$$

[Watch this useful video: `https://www.khanacademy.org/computing/computer-science/informationtheory/moderninfotheory/v/information-entropy`]

Entropy for discrete distribution

Example

Assume three discrete probability distributions $p_1(x)$, $p_2(x)$, and $p_3(x)$ with 4 possible outcomes ($x_1 = 1$, $x_2 = 2$, $x_3 = 3$, $x_4 = 4$). Taking into account the definition of entropy, $H(X) = \sum_{n=1}^{N} p(x_n) \log(\frac{1}{p(x_n)}) = -\sum_{n=1}^{N} p(x_n) \log(p(x_n))$, which of the following statements is correct in regard to the entropy $H_1(X)$, $H_2(X)$, and $H_3(X)$ of the three distributions?



A) $H_1(X) = H_2(X) = H_3(X) = 0$

B) $H_1(X) < H_2(X) < H_3(X)$

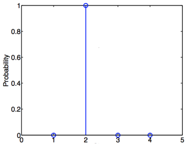
C) $H_1(X) > H_2(X) > H_3(X)$

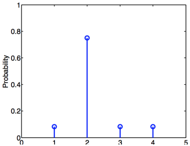
Entropy for discrete distribution

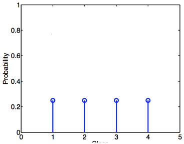
Entropy for discrete distribution

Example

Assume three discrete probability distributions $p_1(x)$, $p_2(x)$, and $p_3(x)$ with 4 possible outcomes ($x_1 = 1$, $x_2 = 2$, $x_3 = 3$, $x_4 = 4$). Taking into account the definition of entropy,
$H(X) = \sum_{n=1}^{N} p(x_n)\log(\frac{1}{p(x_n)}) = -\sum_{n=1}^{N} p(x_n)\log(p(x_n))$, what is the entropy $H_1(X)$, $H_2(X)$, and $H_3(X)$ for each of the three distributions?





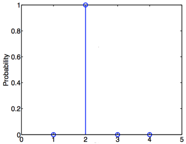


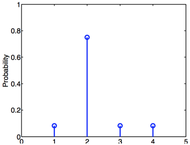
For the **first** distribution: $p(x_1) = 0,\ p(x_2) = 1,\ p(x_3) = 0,\ p(x_4) = 0$
$H_1(X) = -p(x_1)\log p(x_1) - p(x_2)\log p(x_2) - p(x_3)\log p(x_3) - p(x_4)\log p(x_4) =$
$0 + 1\log 1 + 0 + 0 = 0$

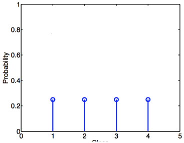
Entropy for discrete distribution

Example

Assume three discrete probability distributions $p_1(x)$, $p_2(x)$, and $p_3(x)$ with 4 possible outcomes ($x_1 = 1$, $x_2 = 2$, $x_3 = 3$, $x_4 = 4$). Taking into account the definition of entropy, $H(X) = \sum_{n=1}^{N} p(x_n) \log(\frac{1}{p(x_n)}) = -\sum_{n=1}^{N} p(x_n) \log_2(p(x_n))$, what is the entropy $H_1(X)$, $H_2(X)$, and $H_3(X)$ for each of the three distributions?







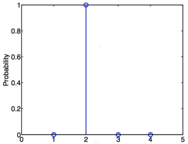
For the **third** distribution: $p(x_1) = 0.25$, $p(x_2) = 0.25$, $p(x_3) = 0.25$, $p(x_4) = 0.25$

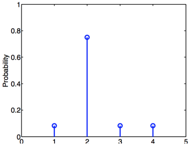
$$H_3(X) = -p(x_1)\log_2 p(x_1) - p(x_2)\log_2 p(x_2) - p(x_3)\log_2 p(x_3) - p(x_4)\log_2 p(x_4) =$$
$$0.25 \cdot \log_2 \frac{1}{0.25} + 0.25 \cdot \log_2 \frac{1}{0.25} + 0.25 \cdot \log_2 \frac{1}{0.25} + 0.25 \cdot \log_2 \frac{1}{0.25} = 2$$

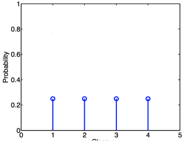
Entropy for discrete distribution

Example

Assume three discrete probability distributions $p_1(x)$, $p_2(x)$, and $p_3(x)$ with 4 possible outcomes ($x_1 = 1$, $x_2 = 2$, $x_3 = 3$, $x_4 = 4$). Taking into account the definition of entropy, $H(X) = \sum_{n=1}^{N} p(x_n) \log(\frac{1}{p(x_n)}) = -\sum_{n=1}^{N} p(x_n) \log_2(p(x_n))$, what is the entropy $H_1(X)$, $H_2(X)$, and $H_3(X)$ for each of the three distributions?







For the **second** distribution: $p(x_1) = 0.1$, $p(x_2) = 0.7$, $p(x_3) = 0.1$, $p(x_4) = 0.1$

$$H_2(X) = -p(x_1)\log_2 p(x_1) - p(x_2)\log_2 p(x_2) - p(x_3)\log_2 p(x_3) - p(x_4)\log_2 p(x_4) =$$

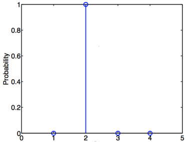
$$0.1 \cdot \log_2 \frac{1}{0.1} + 0.7 \cdot \log_2 \frac{1}{0.7} + 0.1 \cdot \log_2 \frac{1}{0.1} + 0.1 \cdot \log_2 \frac{1}{0.1} = 1.35677$$

Entropy for discrete distribution

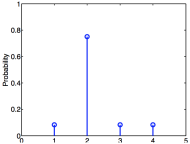
Example

Assume three discrete probability distributions $p_1(x)$, $p_2(x)$, and $p_3(x)$ with 4 possible outcomes (1, 2, 3, 4). Taking into account the definition of entropy, $H(X) = \sum_{n=1}^{N} p(x_n) \log(\frac{1}{p(x_n)}) = -\sum_{n=1}^{N} p(x_n) \log(p(x_n))$, what is the entropy $H_1(X)$, $H_2(X)$, and $H_3(X)$ for each of the three distributions?

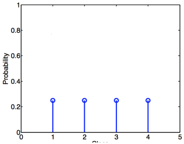
no uncertainty



some uncertainty



high uncertainty



$H_1(X) < H_2(X) < H_3(X)$

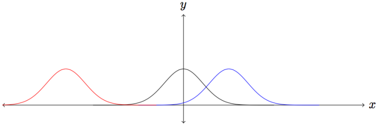
Entropy for continuous distribution

Let X be a continuous random variable with $x \in \Omega$. Its entropy is defined as follows:

$$H(X) = -\int_{x \in \Omega} p(x) \log(p(x)) dx$$

Example

If $X \sim \mathcal{N}(\mu, \sigma^2)$ its entropy is $H(X) = \frac{1}{2}(1 + \log(2\pi\sigma^2))$.
The entropy depends on the variance of the Gaussian.
i.e. higher variance $\rightarrow$ higher uncertainty, and vice-versa.



Gaussians with the same σ, therefore same entropy.

Conditional Entropy

We want to quantify how much uncertainty the realization of a random variable Y has if another random variable X is known. Let X take the values $\{x_1, \ldots, x_M\}$.
The conditional entropy is defined as:

$$
\begin{aligned}
H(Y|X) &= \sum_{m=1}^{M} p_X(x_m) H_{Y|X=x_m}(Y|X = x_m) \\
&= \sum_{m=1}^{M} p_X(x_m) \left(- \sum_{n=1}^{N} p_{Y|X}(y_n|x_m) \log(p_{Y|X}(y_n|x_m)) \right) \\
&= - \sum_{m=1}^{M} \sum_{n=1}^{N} p_X(x_m) p_{Y|X}(y_n|x_m) \log(p_{Y|X}(y_n|x_m))
\end{aligned}
$$

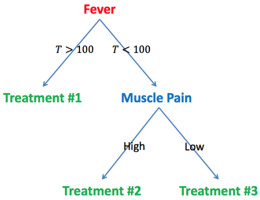
Overview

- Information entropy
- Decision Trees
  - Terminology (e.g. nodes, etc) & intuition
  - Entropy node splitting criterion
  - Algorithm Outline
  - Pruning
  - Regression Trees
- Random Forests

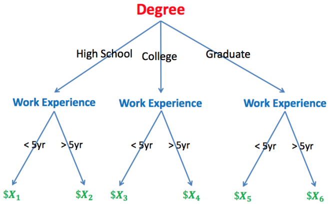
Decision Trees

Many decisions are tree-like structures

Medical treatment



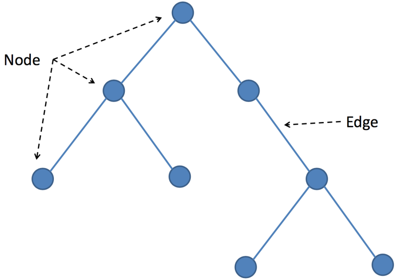
Salary in a company



Decision Trees

What is a decision tree

A hierarchical data structure implementing the divide-and-conquer strategy for decision making

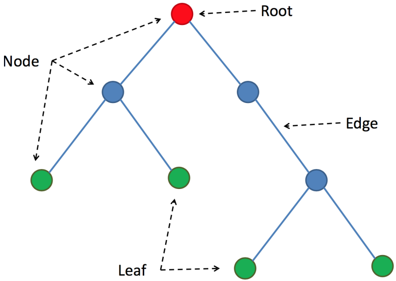


Can be used for both classification & regression

Decision Trees

What is a decision tree

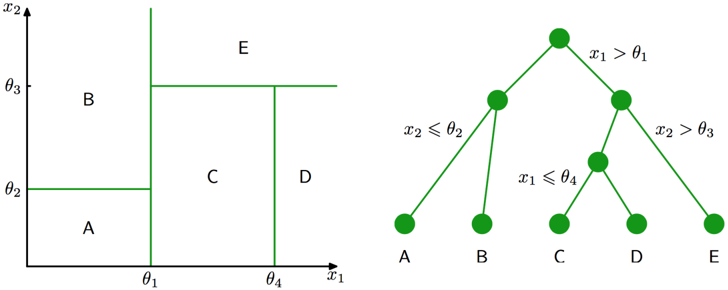
A hierarchical data structure implementing the divide-and-conquer strategy for decision making



Can be used for both classification & regression

Decision Trees

A decision tree partitions the feature space



Three things to learn

- The tree structure (i.e. attributes and #branches for splitting)
- The threshold values (i.e. θ_i)
- The values of the leaves (i.e. $A, B, \ldots$)

Decision Trees

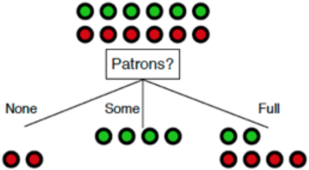
Example

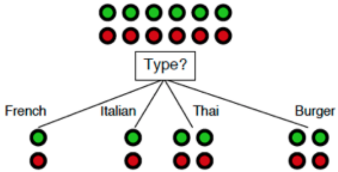
We want to find a decision process for choosing a restaurant

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
X_1	*T*	*F*	*F*	*T*	*Some*	*$$$*	*F*	*T*	*French*	*0–10*	*T*
X_2	*T*	*F*	*F*	*T*	*Full*	*$*	*F*	*F*	*Thai*	*30–60*	*F*
X_3	*F*	*T*	*F*	*F*	*Some*	*$*	*F*	*F*	*Burger*	*0–10*	*T*
X_4	*T*	*F*	*T*	*T*	*Full*	*$*	*F*	*F*	*Thai*	*10–30*	*T*
X_5	*T*	*F*	*T*	*F*	*Full*	*$$$*	*F*	*T*	*French*	*>60*	*F*
X_6	*F*	*T*	*F*	*T*	*Some*	*$$*	*T*	*T*	*Italian*	*0–10*	*T*
X_7	*F*	*T*	*F*	*F*	*None*	*$*	*T*	*F*	*Burger*	*0–10*	*F*
X_8	*F*	*F*	*F*	*T*	*Some*	*$$*	*T*	*T*	*Thai*	*0–10*	*T*
X_9	*F*	*T*	*T*	*F*	*Full*	*$*	*T*	*F*	*Burger*	*>60*	*F*
X_{10}	*T*	*T*	*T*	*T*	*Full*	*$$$*	*F*	*T*	*Italian*	*10–30*	*F*
X_{11}	*F*	*F*	*F*	*F*	*None*	*$*	*F*	*F*	*Thai*	*0–10*	*F*
X_{12}	*T*	*T*	*T*	*T*	*Full*	*$*	*F*	*F*	*Burger*	*30–60*	*T*

Decision Trees

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F	T	Some	$$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	$$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	$$$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	$	F	F	Burger	30–60	T

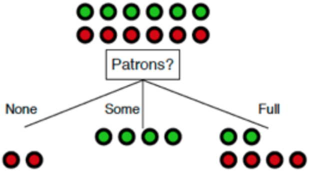


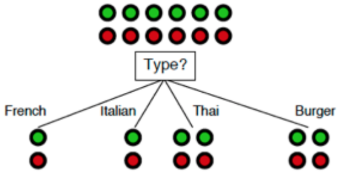


Which split is more useful?

Decision Trees

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
X_1	*T*	*F*	*F*	*T*	*Some*	*$$$*	*F*	*T*	*French*	*0–10*	*T*
X_2	*T*	*F*	*F*	*T*	*Full*	*$*	*F*	*F*	*Thai*	*30–60*	*F*
X_3	*F*	*T*	*F*	*F*	*Some*	*$*	*F*	*F*	*Burger*	*0–10*	*T*
X_4	*T*	*F*	*T*	*T*	*Full*	*$*	*F*	*F*	*Thai*	*10–30*	*T*
X_5	*T*	*F*	*T*	*F*	*Full*	*$$$*	*F*	*T*	*French*	*>60*	*F*
X_6	*F*	*T*	*F*	*T*	*Some*	*$$*	*T*	*T*	*Italian*	*0–10*	*T*
X_7	*F*	*T*	*F*	*F*	*None*	*$*	*T*	*F*	*Burger*	*0–10*	*F*
X_8	*F*	*F*	*F*	*T*	*Some*	*$$*	*T*	*T*	*Thai*	*0–10*	*T*
X_9	*F*	*T*	*T*	*F*	*Full*	*$*	*T*	*F*	*Burger*	*>60*	*F*
X_{10}	*T*	*T*	*T*	*T*	*Full*	*$$$*	*F*	*T*	*Italian*	*10–30*	*F*
X_{11}	*F*	*F*	*F*	*F*	*None*	*$*	*F*	*F*	*Thai*	*0–10*	*F*
X_{12}	*T*	*T*	*T*	*T*	*Full*	*$*	*F*	*F*	*Burger*	*30–60*	*T*





If we split the train samples with respect to the attribute "Patron", we will minimize uncertainty regarding the outcome (i.e., "Patrons" can find the correct outcome with less questions).

Decision Trees

How do we measure information gain?

- Intuitively, information gain tells us how important a given attribute is for predicting the outcome
- We will use it to decide the ordering of attributes in the nodes of a decision tree (i.e. tree structure)
- Main idea: Gaining information reduces uncertainty
- From information theory, we have a measure of uncertainty $\rightarrow$ entropy

Decision Trees

Example: Choosing a restaurant

Measuring the conditional entropy on each of the "Patrons" attributes

For "None" branch : H(Outcome | Patrons="None")

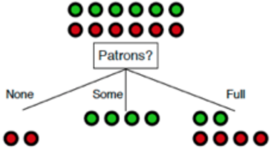
$$-\left(\frac{0}{0+2}\log\frac{0}{0+2}+\frac{2}{0+2}\log\frac{2}{0+2}\right)=0$$

For "Some" branch : H(Outcome | Patrons="Some")

$$-\left(\frac{4}{4+0}\log\frac{4}{4+0}+\frac{0}{4+0}\log\frac{0}{4+0}\right)=0$$

For "Full" branch : H(Outcome | Patrons="Full")

$$-\left(\frac{2}{2+4}\log\frac{2}{2+4}+\frac{4}{2+4}\log\frac{4}{2+4}\right)\approx 0.9$$



Measuring the conditional entropy on Patrons

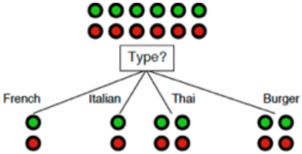
$H(Outcome|Patron) = \frac{2}{12} \times 0 + \frac{4}{12} \times 0 + \frac{6}{12} \times 0.9 = 0.45$

"How uncertain is the Outcome with respect to attribute Patrons"

Decision Trees

Example: Choosing a restaurant

Measuring the conditional entropy on each of the “Type” attributes



For “French” branch

$$-\left(\frac{1}{1+1}\log\frac{1}{1+1}+\frac{1}{1+1}\log\frac{1}{1+1}\right)=1$$

For “Italian” branch

$$-\left(\frac{1}{1+1}\log\frac{1}{1+1}+\frac{1}{1+1}\log\frac{1}{1+1}\right)=1$$

For “Thai” and “Burger” branches

$$-\left(\frac{2}{2+2}\log\frac{2}{2+2}+\frac{2}{2+2}\log\frac{2}{2+2}\right)=1$$

For choosing “Type”

Measuring the conditional entropy on Type

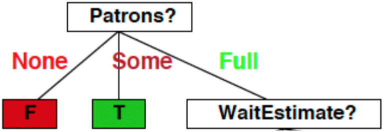
$$H(Outcome|Type)=\frac{2}{12}\times 1+\frac{2}{12}\times 1+\frac{4}{12}\times 1+\frac{4}{12}\times 1=1$$

“How uncertain is the Outcome with respect to attribute Type”

Decision Trees

Example: Choosing a restaurant

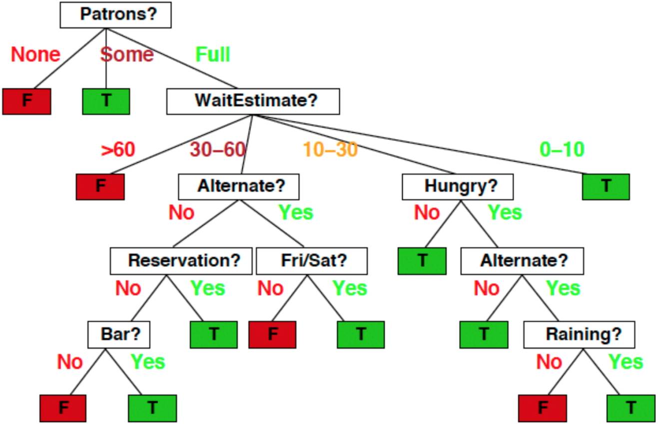
- $H(Outcome|Patron) < H(Outcome|Type)$,
  $H(Outcome|Patron) < H(Outcome|Price)$, ...
- The entropy of the Outcome conditioned on Patron is the lowest
- So the first split is performed with respect to Patron
- We do not split the "None" and "Some" nodes, since their decision is deterministic from the train data
- Next split? We will look only at the 6 instances assigned to the node "Full"



Decision Trees

Example: Choosing a restaurant

Greedily we build the tree and looks like this



Decision Trees: Algorithm Outline

GenerateTree($\mathcal{X}$) (Input $\mathcal{X}$: training samples)

1. $i := \mathit{SplitAttribute}(\mathcal{X})$ (find attribute with lowest uncertainty)
2. For each branch of $\mathbf{x_i}$ (for all values of the above attribute)
   - 2a Find $\mathcal{X}_i$ falling in branch
   - 2b GenerateTree($\mathcal{X}_i$)

SplitAttribute($\mathcal{X}$) (Input $\mathcal{X}$: training samples)

1. $\mathit{MinEnt} := \mathit{MAX}$
2. For all attributes X_i, $i = 1, \ldots, D$
   - 2a Compute $H(Y|\mathcal{X}_i)$ (entropy of outcome conditioned on attribute X_i)
   - 2b If $\mathit{MinEnt} > H(Y|\mathcal{X}_i)$ (current attribute X_i with lowest conditional entropy so far)
     - 2b.i $\mathit{MinEnt} := H(Y|\mathcal{X}_i)$
     - 2b.ii $\mathit{SplitAttr} := i$
3. Return $\mathit{SplitAttr}$

Decision Trees

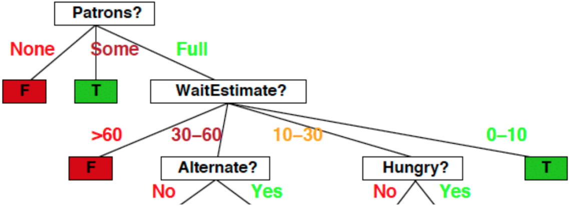
Should we continue to split until every training sample is classified correctly?

- We should be very careful about the depth of the tree
- Eventually, we can get all training examples right
  - Is this what we want?
- The maximum depth of the tree is a hyperparameter

Decision Trees: Pruning

Example: Choosing a restaurant

We should prune some of the leaves of the tree to get a smaller depth

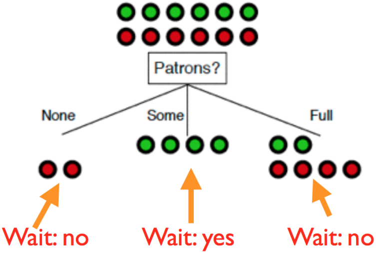


- If we stop here, not all training samples are classified correctly
- How do we classify a new instance?
  - We label the leaves of this smaller tree with the label of the majority of training samples

Decision Trees

Example: Choosing a restaurant

If we wanted to prune at this first node, we would take the following decisions



Decision Trees: Pruning

- Pre-Pruning
  - Stop growing the tree earlier, before it perfectly classifies the training set
  - Use a pre-specified max depth
- Post-Pruning
  - Grow the tree full until no training error
  - Trim the nodes of the decision tree in a bottom-up fashion
  - If generalization error improves after trimming, replace sub-tree by a leaf node
    - Class label of leaf node is determined from majority class of instances in the sub-tree

Decision Trees: Alternative splitting criteria

2-class problem

$\hat{p}$, $1-\hat{p}$: frequency of class 0 and 1

- Entropy:
  $\phi(\hat{p}) =$
  $-\hat{p}\log\hat{p} - (1-\hat{p})\log(1-\hat{p})$
- Gini index:
  $\phi(\hat{p}) = 2\hat{p}(1-\hat{p})$

C-class problem

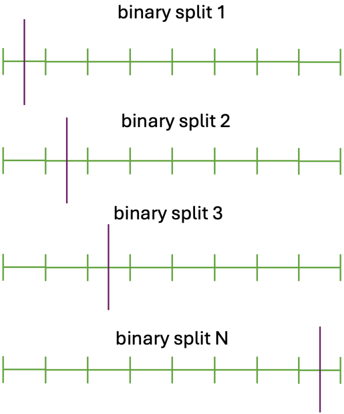
$\hat{p}_1$,...,$\hat{p}_C$: frequency of class $1, \ldots, C$

- Entropy:
  $\phi(\hat{p}_1, \ldots, \hat{p}_C) =$
  $-\sum_c \hat{p}_c \log \hat{p}_c$
- Gini index:
  $\phi(\hat{p}_1, \ldots, \hat{p}_C) =$
  $\sum_c \hat{p}_c(1-\hat{p}_c)$

- When the gini index is zero, it means that all elements belong to one class.
- Entropy and gini index are very similar. Entropy is slightly more expensive to calculate.

Decision Trees: Continuous features

- For continuous features, we can try multiple splits
  - e.g., if we have feature values in $[10, 14]$, we can split at $(10.5, 11.5, 13.5)$
- Among the split positions that are possible, we select the one that minimizes the entropy or gini index

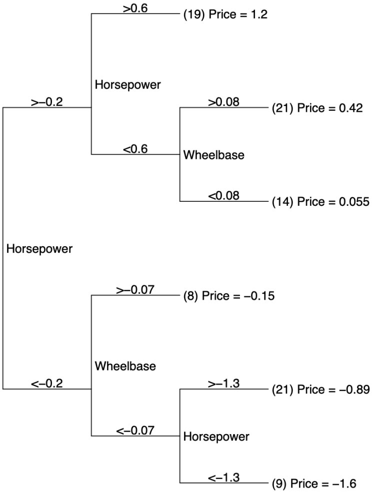


Regression Trees

- Similar to classification trees with some differences
- Split criterion
  - Mean square error between predicted and actual value of samples that have reached current node
- Leaf node value
  - Mean (or median) of samples that have reached the node
  - Leaf node is created (splitting stops) if the current node has "acceptable" error

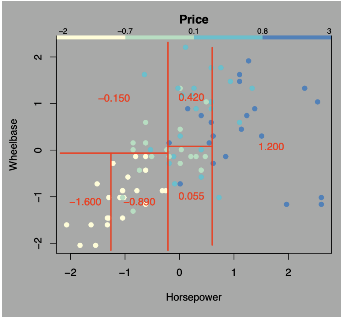
Regression Trees

Example: Regression tree



Regression Trees

Example: Regression tree feature split



Decision Trees

Advantages

- The models are transparent: easily interpretable by human (as long as the tree is not too big)
- It is compact, since we only need to store the splitting criteria and corresponding values
- Data can contain combination of continuous and discrete features
- Can handle missing data

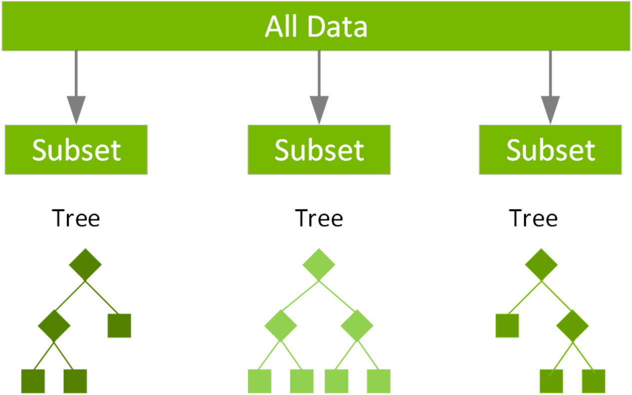
Overview

- Information Entropy
- Decision Trees
  - Terminology (e.g. nodes, etc) & intuition
  - Entropy node splitting criterion
  - Algorithm Outline
  - Pruning
  - Regression Trees
- Random Forests

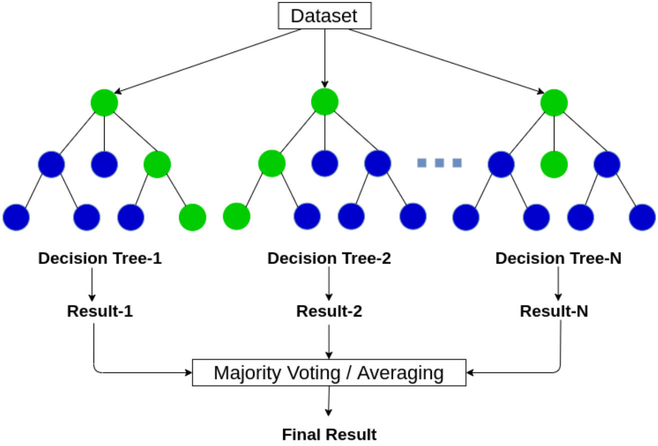
Random Forests

- We grow many classification trees through bagging & randomization
- Bagging (**B**ootstrap **agg**regat**ing**)
  - Generate independently bootstrap datasets from original data
  - Run a decision tree in each one of them
- Randomize over the set of attributes
  - Before growing a bootstrap decision tree
  - When splitting an interior node of the classification tree
- It is recommended to build small trees
- For each sample, each tree "votes" for a class and we perform majority voting for final decision

Random Forests

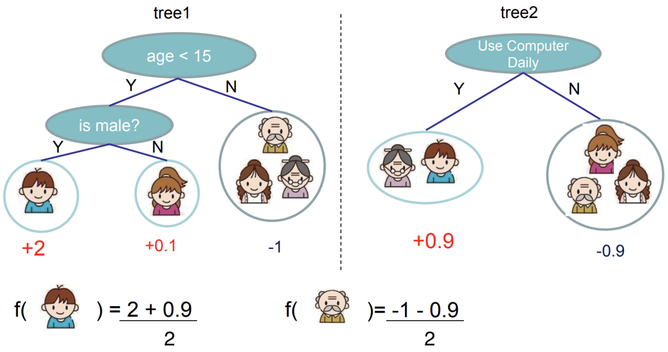


Random Forests



Random Forests

Example: Likelihood of one playing video games



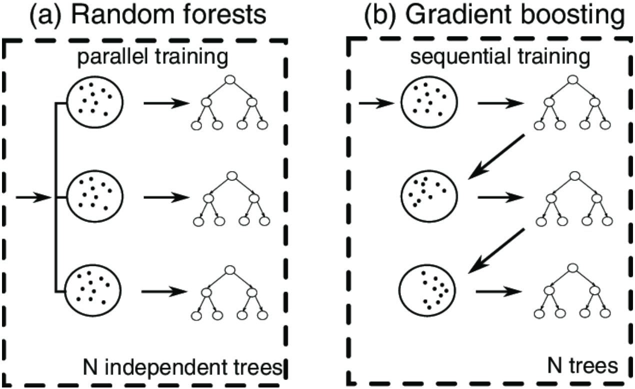
Random Forests

Advantages

- Good performance in practice
- Runs efficiently on large data bases
- Runs efficiently on large feature sets
- Gives estimates of the most relevant variables for the problem

Random Forests

Can we do better? We can use boosting algorithms (e.g., AdaBoost, Gradient Boosting) to build trees sequentially (rather than in parallel)



Kowalek, P., Loch-Olszewska, H., & Szwabiński, J. (2019). Classification of diffusion modes in single-particle tracking data: Feature-based versus deep-learning approach. Physical Review E, 100(3), 032410.

Random Forests

- **Gradient boosting:** each iteration uses the error residuals of the previous model to fit the next model
- **XGBoost:** a scalable and highly accurate implementation of gradient boosting.
  - Trees are built in parallel, instead of sequentially like in gradient boosting
  - Level-wise strategy, scanning across gradient values and using these partial sums to evaluate the quality of splits at every possible split

What have we learnt so far

Decision Trees

- Hierarchical (tree-like) structure to perform classification/regression
- Tree structure determined by splitting criterion
  - Entropy (measure of uncertainty), gini index, etc.
- Pruning
  - Prevent overfitting by limiting the depth of the tree
  - Avoids perfect performance on train set
  - Pre/Post-pruning
- Main advantage: interpretability

Random Forests

- Tree ensemble
- Bagging & Randomization
- Good peformance in practice

Readings: Alpaydin 9.1-9.4