*** Applied Machine Learning Fundamentals ***
Decision Trees and Ensembles

Daniel Wehner, M.Sc.

SAP SE / DHBW Mannheim

Winter term 2023/2024

Find all slides on GitHub (DaWe1992/Applied_ML_Fundamentals)

Lecture Overview

Agenda for this Unit

Section:

Introduction

What are Decision Trees?

- Decision trees are induced in a **supervised fashion**
- The algorithm was originally proposed by *Ross Quinlan* in 1986

John Ross Quinlan is a computer science researcher in data mining and decision theory. He has contributed extensively to the development of decision tree algorithms, including inventing the canonical C4.5 and ID3 algorithms. He is currently running the company RuleQuest Research which he founded in 1997.

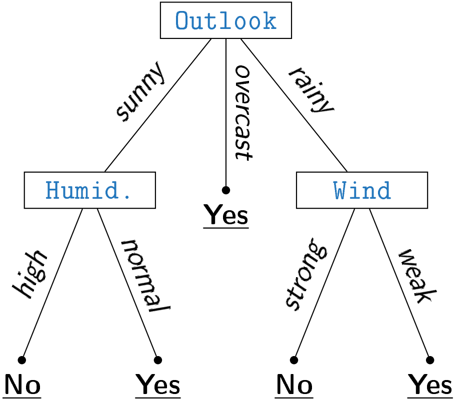
What are Decision Trees? (Ctd.)

- Decision trees are grown **recursively** $\rightarrow$ *'divide-and-conquer'*
- Decision trees are **easily interpretable** (unlike other methods like e. g. neural networks)
- A decision tree consists of:

Nodes	Each node corresponds to an **attribute test**
Edges	One edge per possible test outcome
Leaves	Class label to predict

What we want...

Outlook	Temperature	Humidity	Wind	PlayGolf
sunny	hot	high	weak	no
sunny	hot	high	strong	no
overcast	hot	high	weak	yes
rainy	mild	high	weak	yes
rainy	cool	normal	weak	yes
rainy	cool	normal	strong	no
overcast	cool	normal	strong	yes
sunny	mild	high	weak	no
sunny	cool	normal	weak	yes
rainy	mild	normal	weak	yes
sunny	mild	normal	strong	yes
overcast	mild	high	strong	yes
overcast	hot	normal	weak	yes
rainy	mild	high	strong	no
rainy	mild	normal	strong	???

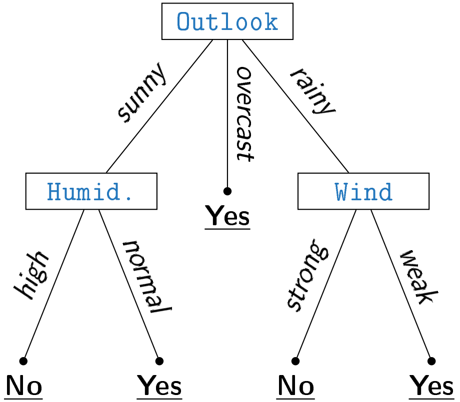


Classification of new Instances

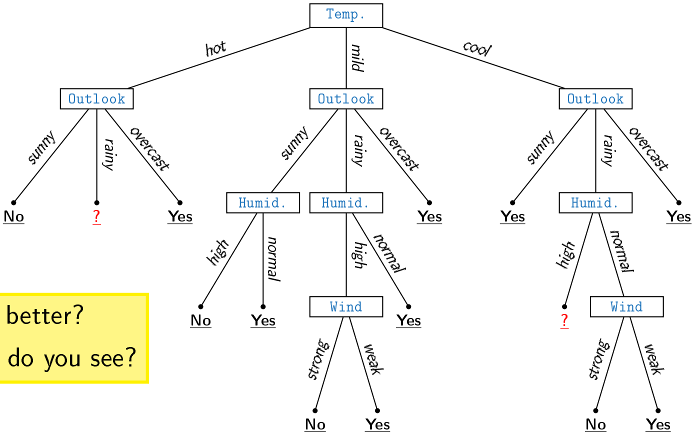
- Suppose we get a new instance:

`Outlook`	rainy
`Temperature`	mild
`Humidity`	normal
`Wind`	strong

- **What is its class?**
- Answer: **No**



Another Decision Tree...



Is this one better?

What problems do you see?

Section:

Iterative Dichotomizer (ID3)

Inductive Bias
Split Heuristics: Entropy and Information Gain
ID3 Algorithm

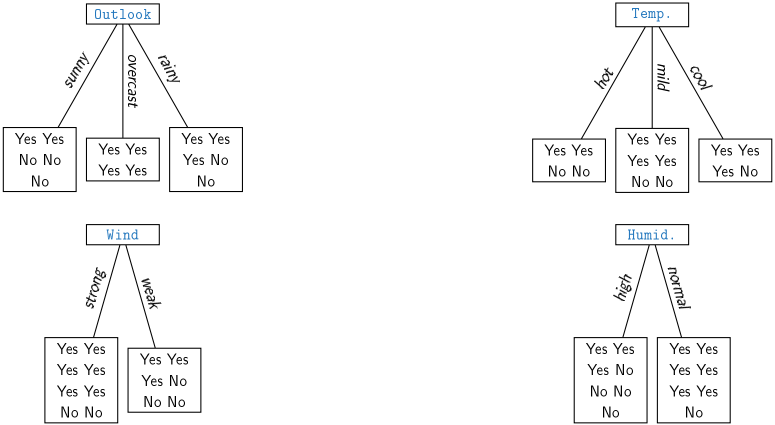
Important

Inductive Bias of Decision Trees

Complex models tend to **overfit** the data and **do not generalize well**. Therefore: Prefer the simplest hypothesis that fits the data!
Occam's razor: **'More things should not be used than are necessary.'**

William of Ockham (circa 1287 – 1347) was an English Franciscan friar who is believed to have been born in Ockham, a small village in Surrey. He is considered to be one of the major figures of medieval thought. He is commonly known for Occam's razor, the methodological principle that bears his name, and also produced significant works on logic, physics and theology.

The Root of all Evil... Which Attribute to choose?



Finding a proper Split Attribute

- Simple and small trees are preferred
  - Data in successor nodes should be **as pure as possible**
  - This means nodes containing only one class are preferable
- To learn small trees we have to split by attributes which **provide the most information** and produce the least successor nodes

Question:

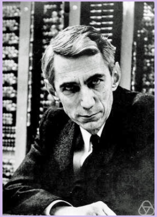
How can we express this thought as a mathematical formula?

Measure of Impurity: Entropy

Answer:

- **Entropy** (*Claude E. Shannon*)
- Originates in the field of **information theory**

Claude Elwood Shannon (April 30, 1916 – February 24, 2001) was an American mathematician, electrical engineer, computer scientist and cryptographer known as the "father of information theory". Shannon contributed to the field of cryptanalysis for national defense of the United States during World War II, and his mathematical theory of information became very well cited and laid the foundation for the field of information theory.



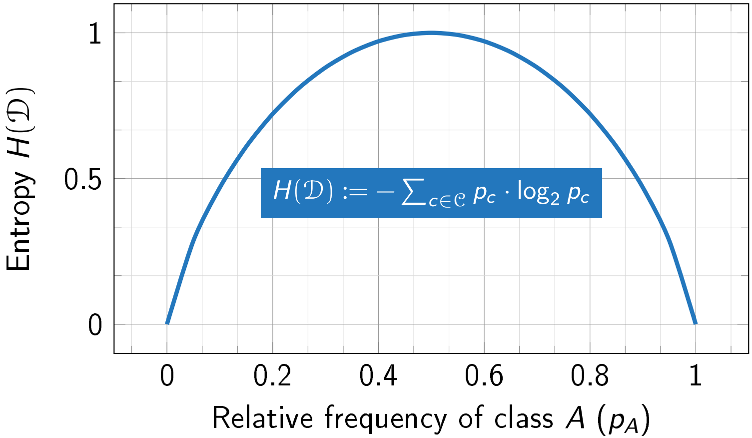
Measure of Impurity: Entropy (Ctd.)

- Entropy H (capital η) is a measure of chaos in the data (measured in bits)
- **Example:** Consider two classes A and B ($\mathcal{C} = \{A, B\}$)

$$H(\{\boldsymbol{A}, \boldsymbol{A}, \boldsymbol{A}, \boldsymbol{A}, \boldsymbol{A}, \boldsymbol{A}, \boldsymbol{A}, \boldsymbol{A}\}) \quad \to 0 \quad \textit{Bits}$$
$$H(\{\boldsymbol{A}, \boldsymbol{A}, \boldsymbol{A}, \boldsymbol{A}, \boldsymbol{A}, \boldsymbol{A}, B, B\}) \quad \to 0.81 \quad \textit{Bits}$$
$$H(\{\boldsymbol{A}, \boldsymbol{A}, \boldsymbol{A}, \boldsymbol{A}, B, B, B, B\}) \quad \to 1 \quad \textit{Bit}$$
$$H(\{\boldsymbol{A}, \boldsymbol{A}, B, B, B, B, B, B\}) \quad \to 0.81 \quad \textit{Bits}$$
$$H(\{B, B, B, B, B, B, B, B\}) \quad \to 0 \quad \textit{Bits}$$

If both classes are equally distributed, the entropy function H reaches its maximum. Pure datasets have minimal entropy.

Measure of Impurity: Entropy (Ctd.)



Important

Measure of Impurity: Entropy (Ctd.)

Entropy formula:

$$H(\mathcal{D}) := -\sum_{c \in \mathcal{C}} p_c \cdot \log_2 p_c \tag{1}$$

- p_c denotes the relative frequency of class $c \in \mathcal{C}$
- **Weather data:**

$$\mathcal{C} := \{yes, no\} \qquad \text{i. e.} \qquad p_{yes} = {}^{9}/_{14} \quad \text{and} \quad p_{no} = {}^{5}/_{14}$$

$$H(\mathcal{D}) = -\sum_{c \in \mathcal{C}} p_c \cdot \log_2 p_c = -({}^{9}/_{14} \cdot \log_2 {}^{9}/_{14} + {}^{5}/_{14} \cdot \log_2 {}^{5}/_{14}) = 0.9403$$

Quality of the Split: Average Entropy

- We still don't know which attribute to use for the split
- Calculate the entropy after each potential split
- **Average Entropy** after splitting by attribute $\mathtt{A}$:

$$H(\mathcal{D}|\mathtt{A}) := \sum_{v \in \mathrm{dom}(\mathtt{A})} \frac{|\mathcal{D}_{\mathtt{A}=v}|}{|\mathcal{D}|} \cdot H(\mathcal{D}_{\mathtt{A}=v}) \tag{2}$$

- Legend:

$\mathtt{A}$	Attribute
$\mathrm{dom}(\mathtt{A})$	Possible values attribute $\mathtt{A}$ can take (domain of $\mathtt{A}$)
$\|\mathcal{D}_{\mathtt{A}=v}\|$	Number of examples satisfying $\mathtt{A} = v$

Quality of the Split: Average Entropy (Ctd.)

Example: Weather data, attribute `Outlook`

$$H(\mathcal{D}|\texttt{Outlook}) = \sum_{v \in \mathrm{dom}(\texttt{Outlook})} \frac{|\mathcal{D}_{\texttt{Outlook}=v}|}{|\mathcal{D}|} \cdot H(\mathcal{D}_{\texttt{Outlook}=v})$$

$$= 5/14 \cdot 0.9710 + 5/14 \cdot 0.9710 + 4/14 \cdot 0 \qquad = 0.6936$$

$$H(\mathcal{D}_{\texttt{Outlook}=sunny}) = -\left(2/5 \cdot \log_2(2/5) + 3/5 \cdot \log_2(3/5)\right) \qquad = 0.9710$$

$$H(\mathcal{D}_{\texttt{Outlook}=rainy}) = -\left(3/5 \cdot \log_2(3/5) + 2/5 \cdot \log_2(2/5)\right) \qquad = 0.9710$$

$$H(\mathcal{D}_{\texttt{Outlook}=overcast}) = -\left(4/4 \cdot \log_2(4/4) + 0/4 \cdot \log_2(0/4)\right) \qquad = 0$$

Information Gain

- We have calculated the entropy before and after the split (average entropy)
- The difference of both is called the **information gain (IG)**
- Select the attribute with the highest IG

Attribute	H_{before}	H_{after}	IG
`Outlook`	0.9403	0.6936	0.2464
`Temperature`	0.9403	0.9111	0.0292
`Humidity`	0.9403	0.7885	0.1518
`Wind`	0.9403	0.8922	0.0481

- Attribute `Outlook` maximizes IG

Training Data after the Split by Attribute `Outlook`

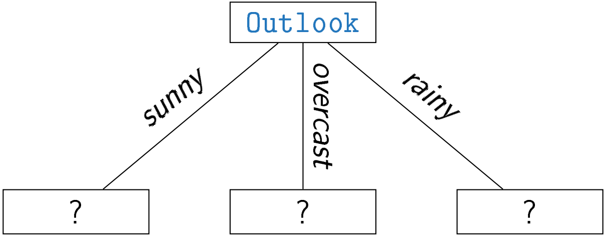
~~`Outlook`~~	`Temperature`	`Humidity`	`Wind`	`PlayGolf`
~~sunny~~	hot	high	weak	no
~~sunny~~	hot	high	strong	no
~~sunny~~	mild	high	weak	no
~~sunny~~	cool	normal	weak	yes
~~sunny~~	mild	normal	strong	yes
~~rainy~~	mild	high	weak	yes
~~rainy~~	cool	normal	weak	yes
~~rainy~~	cool	normal	strong	no
~~rainy~~	mild	normal	weak	yes
~~rainy~~	mild	high	strong	no
~~overcast~~	cool	normal	strong	yes
~~overcast~~	hot	high	weak	yes
~~overcast~~	mild	high	strong	yes
~~overcast~~	hot	normal	weak	yes

- The table on the left displays the dataset $\mathcal{D}$ after the split by attribute `Outlook`
- We obtain three subsets (one per attribute value)
- Attribute `Outlook` is removed in the current branch of the tree **(Why?)**

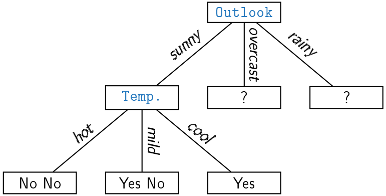
How to proceed?

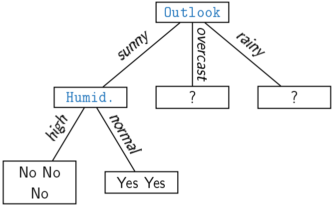
- The algorithm is recursively applied to the resulting subsets
  1. Calculate entropy (before and after the split)
  2. Calculate information gain for each attribute
  3. Choose the attribute with maximum information gain for the split
  4. In the current branch: Do not consider the attribute any more
  5. **Recursion** ↺ (`Go to 1`)
- Recursion stops as soon as the subset is pure **(Danger: ☠ overfitting ☠)**
- In the example above the subset $\mathcal{D}_{\texttt{Outlook}=\mathit{overcast}}$ is already pure
- This algorithm is referred to as **ID3 (Iterative Dichotomizer)**

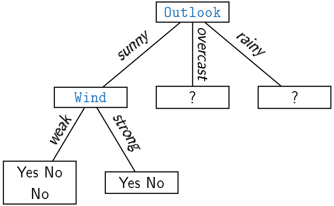
Step by Step: Construction of the Tree



Step by Step: Construction of the Tree (Ctd.)

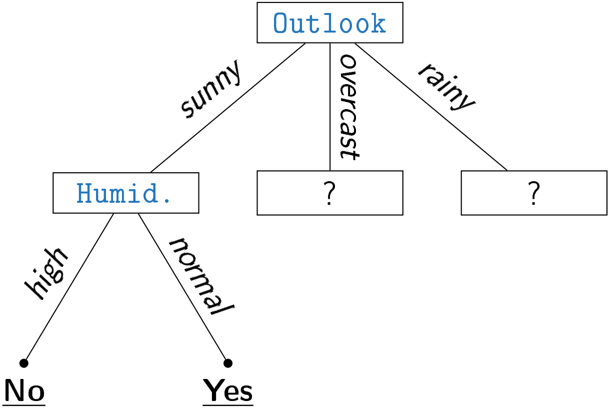




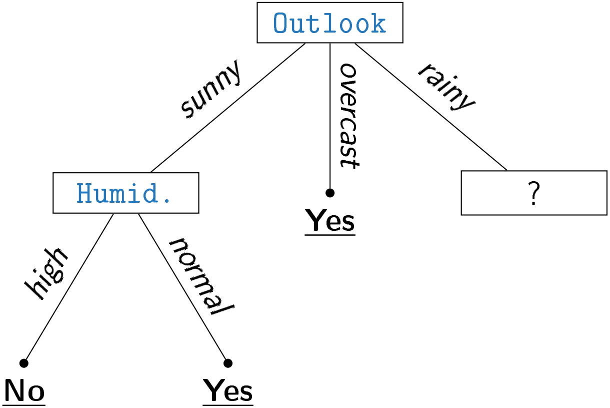


- $IG(\texttt{Temperature}) = 0.571$
- $IG(\texttt{Humidity}) = \mathbf{0.971}$
- $IG(\texttt{Wind}) = 0.020$

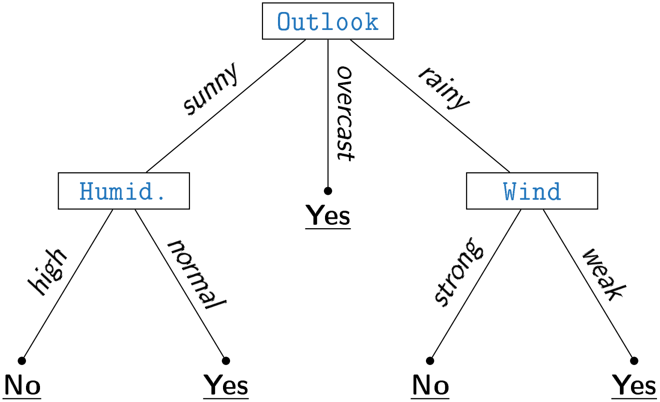
Step by Step: Construction of the Tree (Ctd.)



Step by Step: Construction of the Tree (Ctd.)



Step by Step: Construction of the Tree (Ctd.)



Section:
Extensions and Variants

- Other Measures of Impurity
- Highly-branching Attributes
- Numeric Attributes
- Regression Trees

An Alternative to Information Gain: Gini Index

Gini index:

$$Gini(\mathcal{D}) := \sum_{c \in \mathcal{C}} p_c \cdot (1 - p_c) = 1 - \sum_{c \in \mathcal{C}} p_c^2 \tag{3}$$

- Gini index and entropy always produce the same decision tree
- Often used as a default in machine learning libraries **(Why?)**
- Used e. g. in **CART (Classification and Regression Trees)**
- **Gini gain** could be defined analogously to IG *(usually not done)*

Why not use the Error as a splitting Criterion?

- The bias towards pure leaves is **not strong enough**
- **Example:**

Split 1 (rows) / **Split 2** (columns)

Split 1	40 of A	60 of A
	60 of A	40 of B

Split 2

Error without splitting:

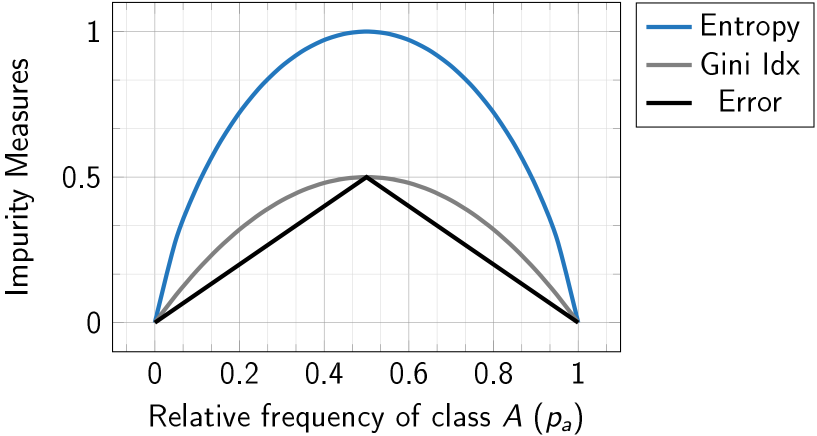
20 %

Error after splitting:

20 %

Both splits don't improve the error.
But together they give a perfect split!

Summary: Impurity Measures



Highly-branching Attributes

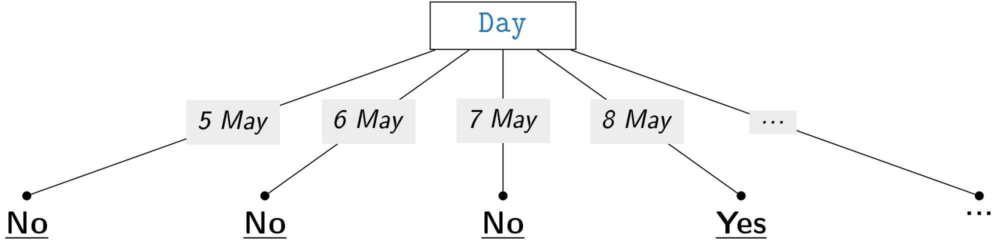
Attributes with a large number of values are problematic, since the leaves are not 'backed' with sufficient data examples.

In extreme cases only one example per node (e. g. IDs)

This may lead to:

- **Overfitting** (selection of attributes which are not optimal for prediction)
- **Fragmentation** (data is fragmented into too many small sets)

Highly-branching Attributes (Ctd.)



- Entropy before was 0.9403, Entropy after split is 0
- $\mathrm{IG}(\mathcal{D}, \texttt{Day}) = 0.9403$
- Attribute `Day` would be chosen for the split $\Rightarrow$ **Bad for prediction** ☠

Highly-branching Attributes (Ctd.)

- Calculate the **intrinsic information (IntI)**:

$$IntI(\mathfrak{D}, \mathtt{A}) := -\sum_{v \in \mathrm{dom}(\mathtt{A})} \frac{|\mathfrak{D}_{\mathtt{A}=v}|}{|\mathfrak{D}|} \cdot \log_2 \frac{|\mathfrak{D}_{\mathtt{A}=v}|}{|\mathfrak{D}|} \tag{4}$$

- Attributes with high *IntI* are **less useful** ($\Rightarrow$ high fragmentation)
- New splitting heuristic: **Gain ratio (GR)**

$$GR(\mathfrak{D}, \mathtt{A}) := \frac{IG(\mathfrak{D}, \mathtt{A})}{IntI(\mathfrak{D}, \mathtt{A})} \tag{5}$$

Highly-branching Attributes (Ctd.)

- Intrinsic information for attribute Day:

$$IntI(\mathcal{D}, \texttt{Day}) = 14 \cdot \left(-{}^{1}/_{14} \cdot \log_2({}^{1}/_{14})\right) = 3.807 \tag{6}$$

- Gain ratio for attribute Day:

$$GR(\mathcal{D}, \texttt{Day}) = \frac{0.9403}{3.807} = 0.246 \tag{7}$$

Remark: In this case the attribute Day would still be chosen. Be careful what features to include in the training dataset! **(Feature engineering is important!)**

Handling of numeric Attributes

- Usually, only **binary splits** are considered, e. g.:
  - $\texttt{Temperature} < 48$
  - $\texttt{CPU} > 24$
  - **Not:** $24 \leqslant \texttt{Temperature} \leqslant 31$ (produces three subsets)
- To support non-binary splits, the attribute is **not removed**
  (the same attribute can be used again for another split)
- **Problem:** There is an **infinite number** of possible splits!
- **Solution:** Discretize range (fixed step size, ...)
- **Splitting on numeric attributes is computationally demanding!**

Handling numeric Attributes: Example I

- Consider the attribute `Temperature`:
  Use **numerical values** instead of discrete values like *cool*, *mild*, *hot*:

64	65	68	69	70	71	72	72	75	75	80	81	83	85
Yes	No	Yes	Yes	Yes	No	No	Yes	Yes	Yes	No	Yes	Yes	No

- `Temperature` < 71.5
  yes: 4 | no: 2
- `Temperature` $\geqslant 71.5$
  yes: 5 | no: 3

$$H(\mathcal{D}|\texttt{Temp.}) = {}^{6}\!/_{14} \cdot H(\texttt{Temp.} < 71.5) + {}^{8}\!/_{14} \cdot H(\texttt{Temp.} \geqslant 71.5) = 0.939$$

Handling numeric Attributes: Example II

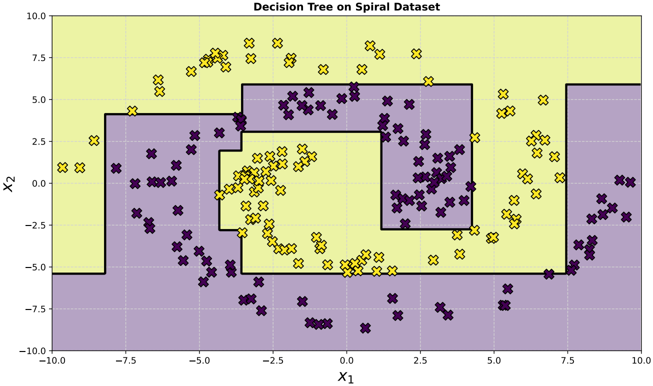
Dataset:

Taxable income	60	70	75	85	90	95	100	120	125	220
Class label	No	No	No	Yes	Yes	Yes	No	No	No	No

Evaluation of splits:

Split point	55		65		72		80		87		92		97		110		122		172		230	
	$\leqslant$	$>$	$\leqslant$	$>$	$\leqslant$	$>$	$\leqslant$	$>$	$\leqslant$	$>$	$\leqslant$	$>$	$\leqslant$	$>$	$\leqslant$	$>$	$\leqslant$	$>$	$\leqslant$	$>$	$\leqslant$	$>$
Yes	0	3	0	3	0	3	0	3	1	2	2	1	3	0	3	0	3	0	3	0	3	0
No	0	7	1	6	2	5	3	4	3	4	3	4	3	4	4	3	5	2	6	1	7	0
Gini index	0.420		0.400		0.375		0.343		0.417		0.400		**0.300**		0.343		0.375		0.400		0.420	

Decision Tree on a Spiral Dataset



Regression Trees

- Prediction of **continuous target variables**
- Predict the average value of all examples in the leaf
- Split the data such that the variance in the leaves is minimized
- **Termination criterion is important, otherwise single point per leaf!**

Standard deviation reduction (SDR):

$$SDR(\mathcal{D}, \mathtt{A}) := SD(\mathcal{D}) - \sum_{v \in dom(\mathtt{A})} \frac{|\mathcal{D}_{\mathtt{A}=v}|}{|\mathcal{D}|} \cdot SD(\mathcal{D}_{\mathtt{A}=v}) \tag{8}$$

Section:
Ensemble Methods

Introduction Ensemble Methods

- **Key Idea**: Don't learn a single classifier, but a **set of classifiers**
- Combine the predictions of the single classifiers to obtain the final prediction

Problem: How can we induce multiple classifiers from a single dataset without getting the same classifier over and over again? **We want to have diverse classifiers, otherwise the ensemble is useless!**

- Basic techniques:
  - Bagging
  - Boosting (not covered)
  - Stacking (not covered)

What is the Advantage of an Ensemble?

- Let 25 **independent** base classifiers be given
- **Independence assumption:** The probability of a single classifier misclassifying an instance **does not** depend on other classifiers in the ensemble
- This condition is usually not fully satisified in practice **(Why?)**
- Each individual classifier in the ensemble is assumed to have an error rate of $\varepsilon := 0.35$
- **What is the error rate of the ensemble?**

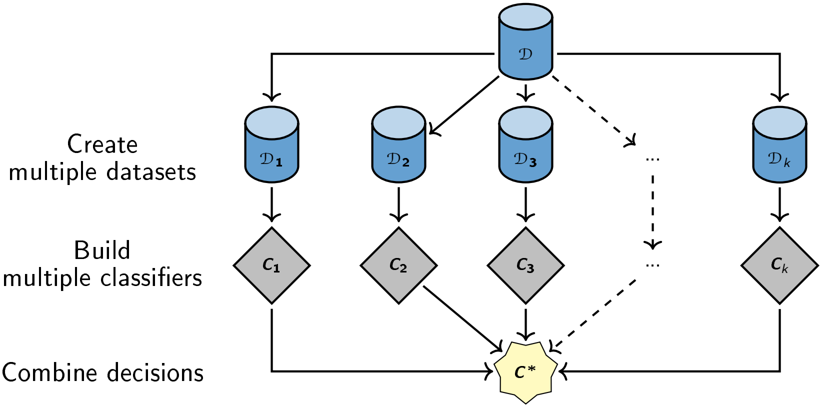
What is the Advantage of an Ensemble? (Ctd.)

- The ensemble's prediction is given by the **majority vote**
- The ensemble makes a wrong prediction **if the majority is wrong** ($\Rightarrow$ i. e. at least 13)
- This probability is computed according to the binomial distribution:

$$\varepsilon_{ensemble} := \sum_{k=13}^{25} \binom{25}{k} \cdot \varepsilon^k \cdot (1 - \varepsilon)^{25-k} \approx 0.06 \ll \varepsilon \tag{9}$$

Important

Bootstrap Aggregating (Bagging)



Creating the Bootstrap Samples

- How to generate multiple datasets which are different?
- **Solution:** Use sampling with replacement

Original Data	1	2	3	4	5	6	7	8	9	10
Bagging (Round 1)	7	8	10	8	2	5	10	10	5	9
Bagging (Round 2)	1	4	9	1	2	3	2	7	3	2
Bagging (Round 3)	1	8	5	10	5	5	9	6	3	7

- Some examples may appear **in more than one set**
- Some examples may appear **more than once** in one set
- Some examples may **not appear at all** in a set

Bagging Algorithm

Algorithm 1: Bagging Algorithm

Input: Training set $\mathcal{D}$, number of base classifiers k

Training phase:

forall $u \in \{1, 2, \dots, k\}$ **do**

  Draw a bootstrap sample $\mathcal{D}_u$ with replacement from $\mathcal{D}$

  Learn a base classifier C_u (e. g. a decision tree) from $\mathcal{D}_u$

  Add the classifier C_u to the ensemble

Prediction phase:

forall *unlabeled instances* **do**

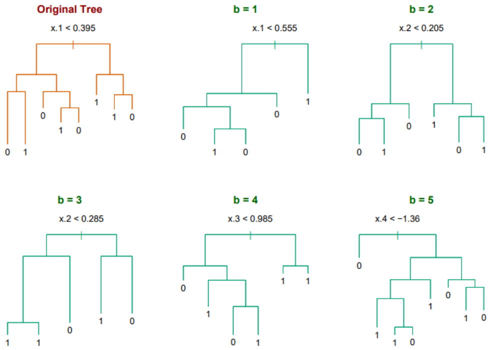
  Get predictions from all classifiers C_u $(1 \leqslant u \leqslant k)$

return *Class which receives the majority of votes (combined classifier C^*)*

Bagging Variations

- So far we have considered bootstrap samples of equal size which are drawn with replacement
- Also conceivable:
  1. **Varying the size** of the bootstrap samples
  2. Sampling **without replacement** $\Rightarrow$ **Pasting**
  3. **Sampling of features**, not instances
     - Not all features are available in all bootstrap samples
     - This is how **random forests** work (see below)
  4. Creating **heterogeneous ensembles** comprising different types of base classifiers (neural networks, decision trees, support vector machines, ...)

Bagged Decision Trees



cf. Hastie.2008, page 284

Randomization

- Why not **randomizing the algorithm** instead of the data?
- Some algorithms already do that: E. g. neural networks (random initialization of weights)
- Especially greedy algorithms can be randomized:
  - Pick from the options **randomly**, instead of picking the best one
  - E. g. decision trees: Do not choose attribute with the highest information gain, but select a split attribute randomly

A random forest combines randomization and bagging.

Random Forest Algorithm

- Ensemble of decision trees
- Combines **bagging** and **random attribute subset selection**
- Grow a decision tree from a bootstrap sample
- Select the best split attribute among a random subset of attributes

At each step a random forest selects the best splitting attribute from a randomly chosen subset of features, but the globally best feature **may not** be available.

Important

Random Forest Algorithm

Algorithm 2: Random Forest Algorithm

Input: Training set $\mathcal{D}$, number of base classifiers k

Training phase:
for $u \in \{1, 2, \ldots, k\}$ **do**
  Create a bootstrap sample from $\mathcal{D}$ (e. g. with replacement) $\Rightarrow$ **Bagging**
  begin
    Grow the tree
    At every node: Randomly choose a subset of attributes to be considered for the split
    $\Rightarrow$ **Randomization**
  Add tree C_u to the ensemble

Prediction phase:
forall *unlabeled instances* **do**
  Get predictions from all classifiers C_u $(1 \leqslant u \leqslant k)$

return *Class which receives the majority of votes (combined classifier C^*)*

ExtraTrees (Randomization 2.0)

- One more step of randomization ⇒ **Ext**remely **Ra**ndomized **Trees** (ExtraTrees)
- The general approach is the same as for random forests, **but:**
  - Instead of choosing the optimal split point...
  - ...it is selected randomly
  - The decision tree is grown without having to calculate entropy
- It is **much faster** (due to less computation)

The large number of classifiers compensates for suboptimal splits.

Section:

Wrap-Up

Summary

- **Decision trees:**
  - The construction of decision trees is guided by an **impurity measure**, e. g. entropy or Gini
  - Recursively select features which **maximize the information gain**
  - Decision trees can handle **numeric attributes** and **continuous output**
- **Ensembles:**
  - Usually, ensembles allow for a significant error reduction
  - **Bagging:** Sample diverse datasets from underlying data
  - **Random forests** combine bagging and randomization

Self-Test Questions

1. What is an inductive bias? What is the inductive bias of decision trees?
2. Explain what Occam's razor is.
3. What does entropy measure? How do you compute the information gain?
4. True or false? *'Pure datasets have maximal entropy.'*
5. What is the advantage of ensemble methods?
6. What is bagging?
7. Explain what a random forest does.

What's next...?

Unit I	Machine Learning Introduction
Unit II	Mathematical Foundations
Unit III	Bayesian Decision Theory
Unit IV	Regression
Unit V	Classification I
Unit VI	**Evaluation**
Unit VII	Classification II
Unit VIII	Clustering
Unit IX	Dimensionality Reduction

Thank you very much for the attention!

Topic: *** Applied Machine Learning Fundamentals *** Decision Trees and Ensembles
Term: Winter term 2023/2024

Contact:

Daniel Wehner, M.Sc.
SAP SE / DHBW Mannheim
daniel.wehner@sap.com

Do you have any questions?