Chair of Process and Data Science

RWTH AACHEN UNIVERSITY

Elements of Machine Learning & Data Science

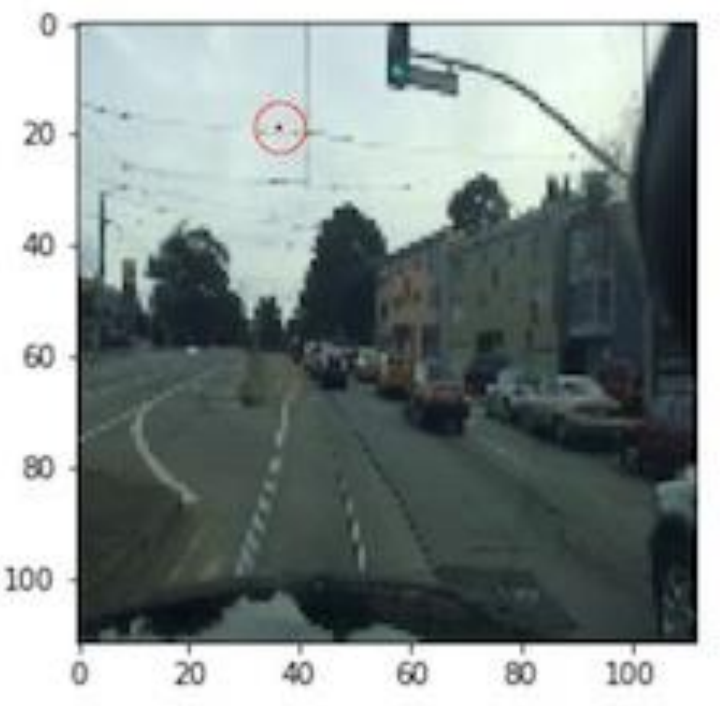
Decision Trees

Lecture 7

Prof. Wil van der Aalst

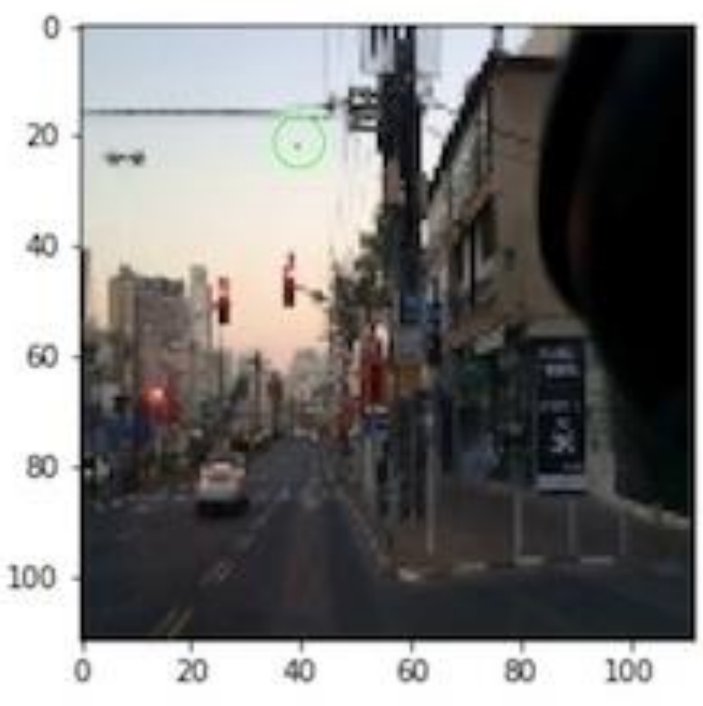
Marco Pegoraro, M.Sc.
Harry Beyel, M.Sc.

Classification problem: Red or Green?



Green light classified as red after one pixel change

Green light classified as red after one pixel change



Red light classified as green after one pixel change.

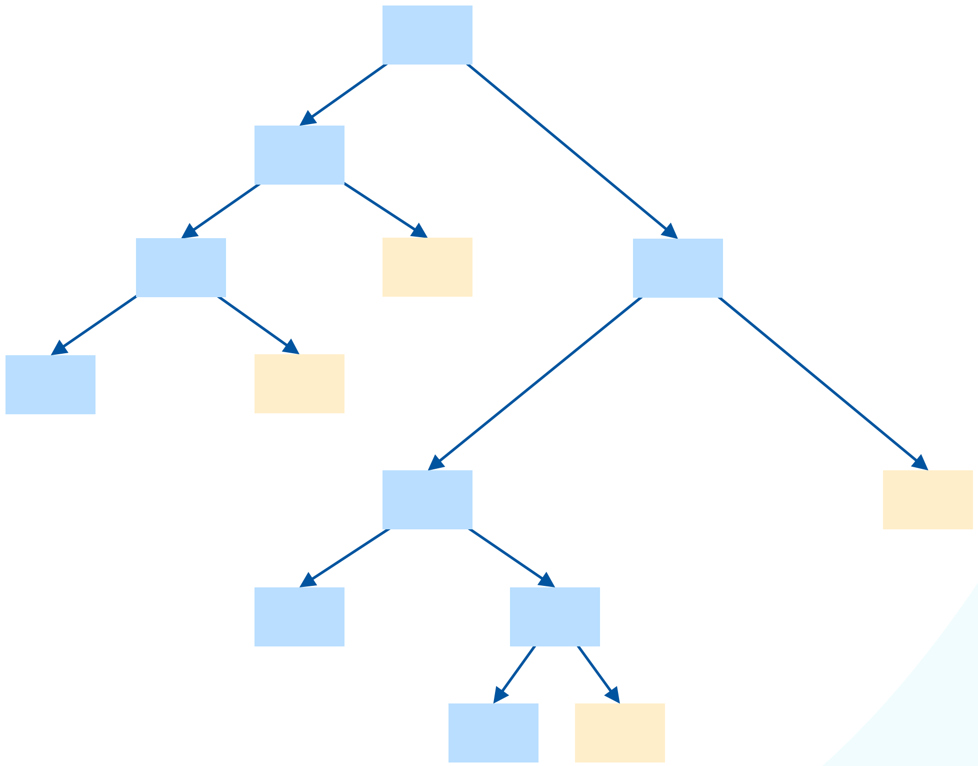
Winner Nexar traffic light challenge: On average, it takes only 3 pixels to turn red into green or green into red!

Wicker, M., Huang, X., Kwiatkowska, M. (2018). Feature-Guided Black-Box Safety Testing of Deep Neural Networks. TACAS 2018. https://doi.org/10.1007/978-3-319-89960-2_22

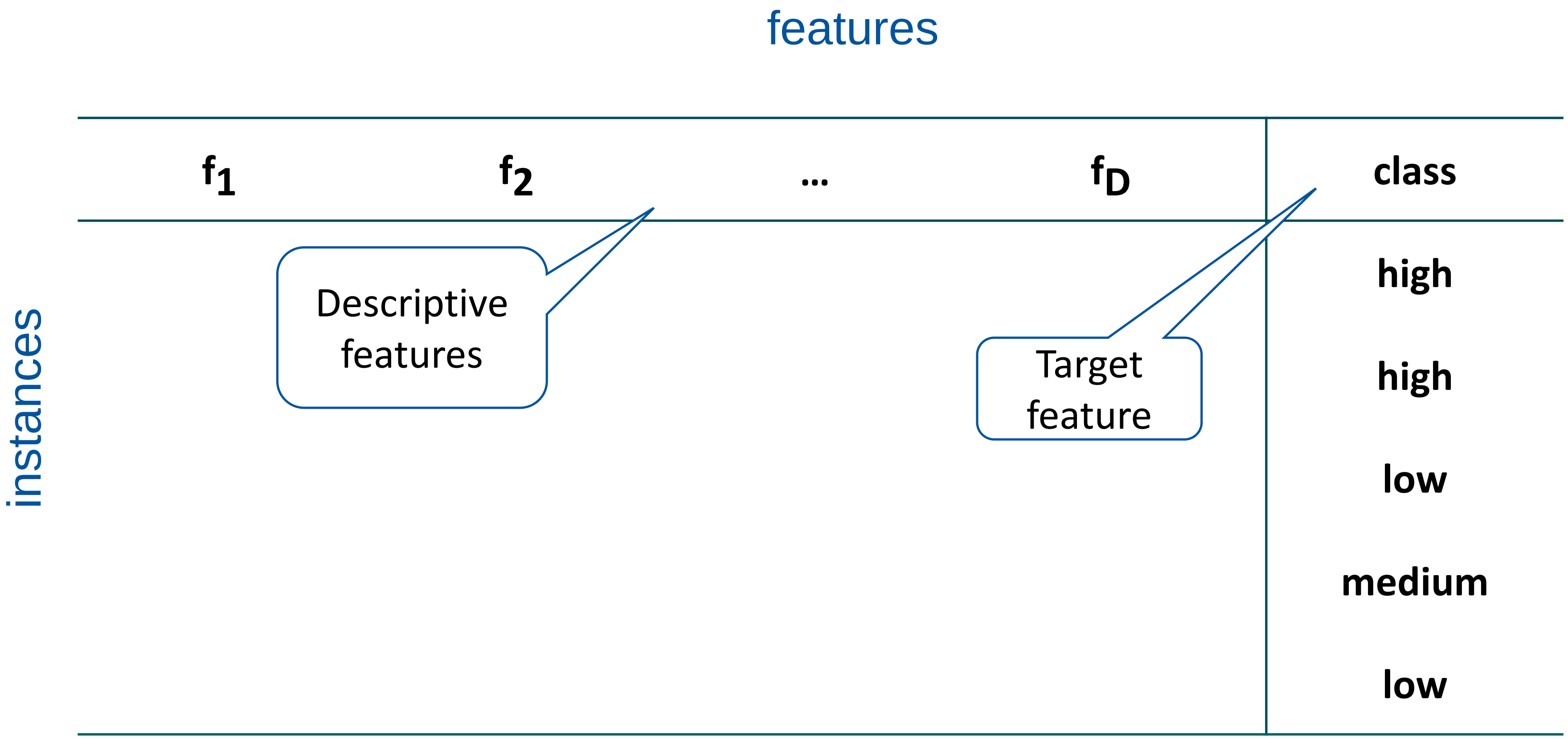
We start with simple tabular data and models that are easy to interpret!

Outline

1. **Introduction to Decision Trees**
2. Entropy
3. ID3 Algorithm
4. Quantifying Information Gain
5. Pruning
6. Ensembles
7. Continuous Data

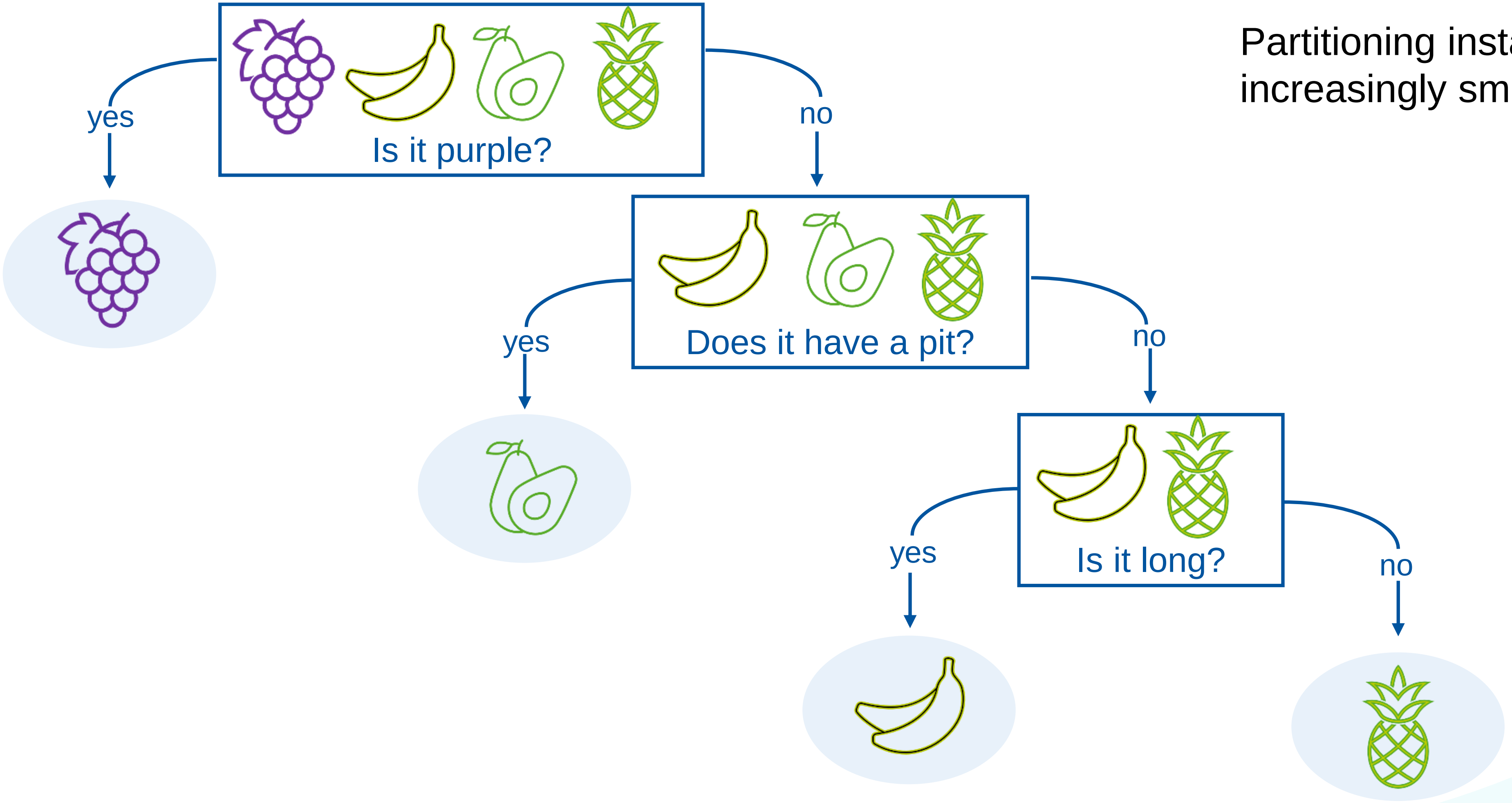


Intuition and Interpretation



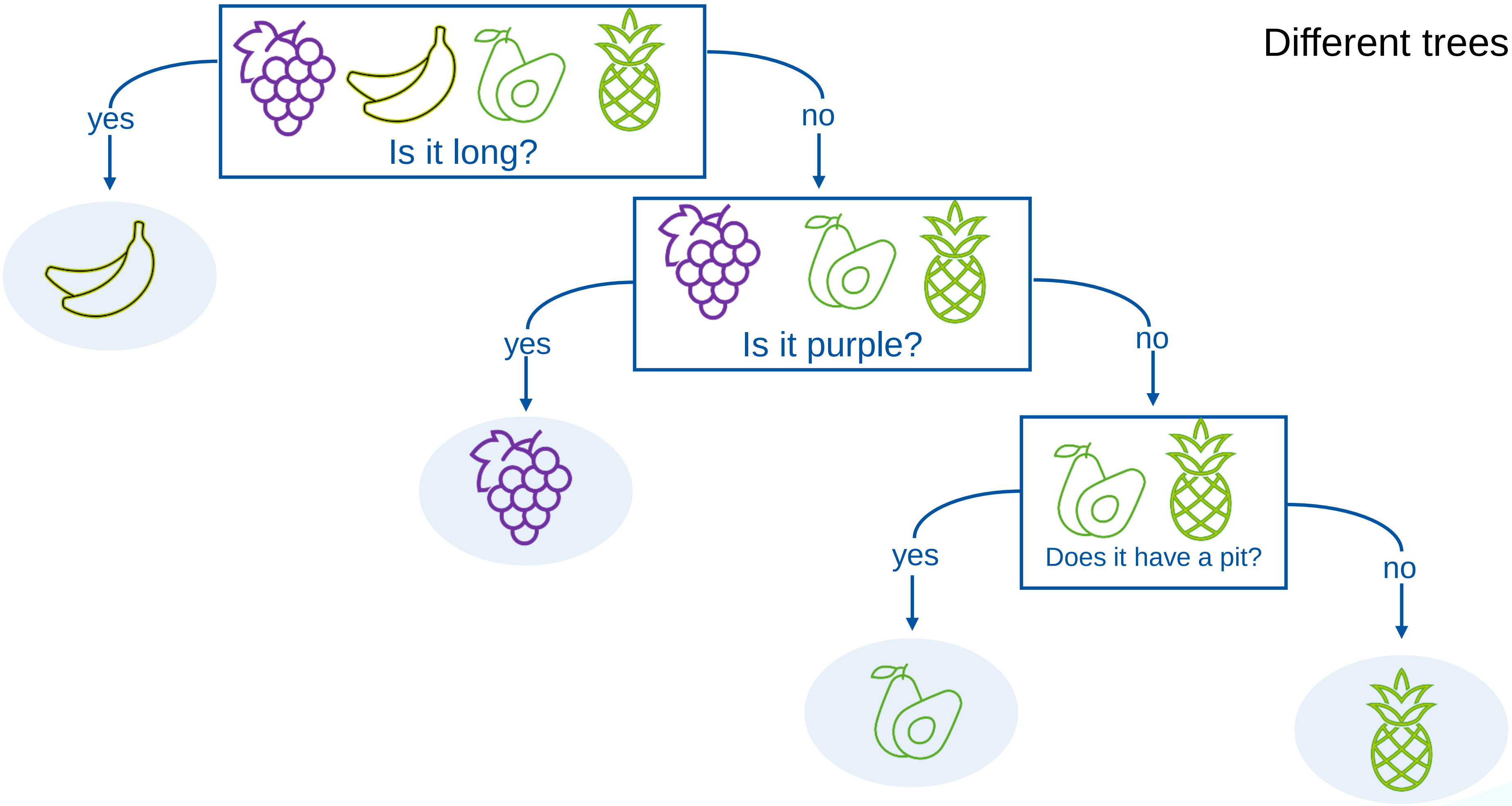
A decision tree aims to explain the target feature in terms of the descriptive features.

Fruity Example



Partitioning instances in increasingly smaller groups

Fruity Example



Different trees are possible

Example 2

Rain	Wind	Temperature (◦C)	Play tennis
Yes	Yes	15	**No**
No	No	34	**Yes**
Yes	No	23	**Yes**
Yes	Yes	20	**Yes**
No	Yes	28	**No**
...	...	...	...

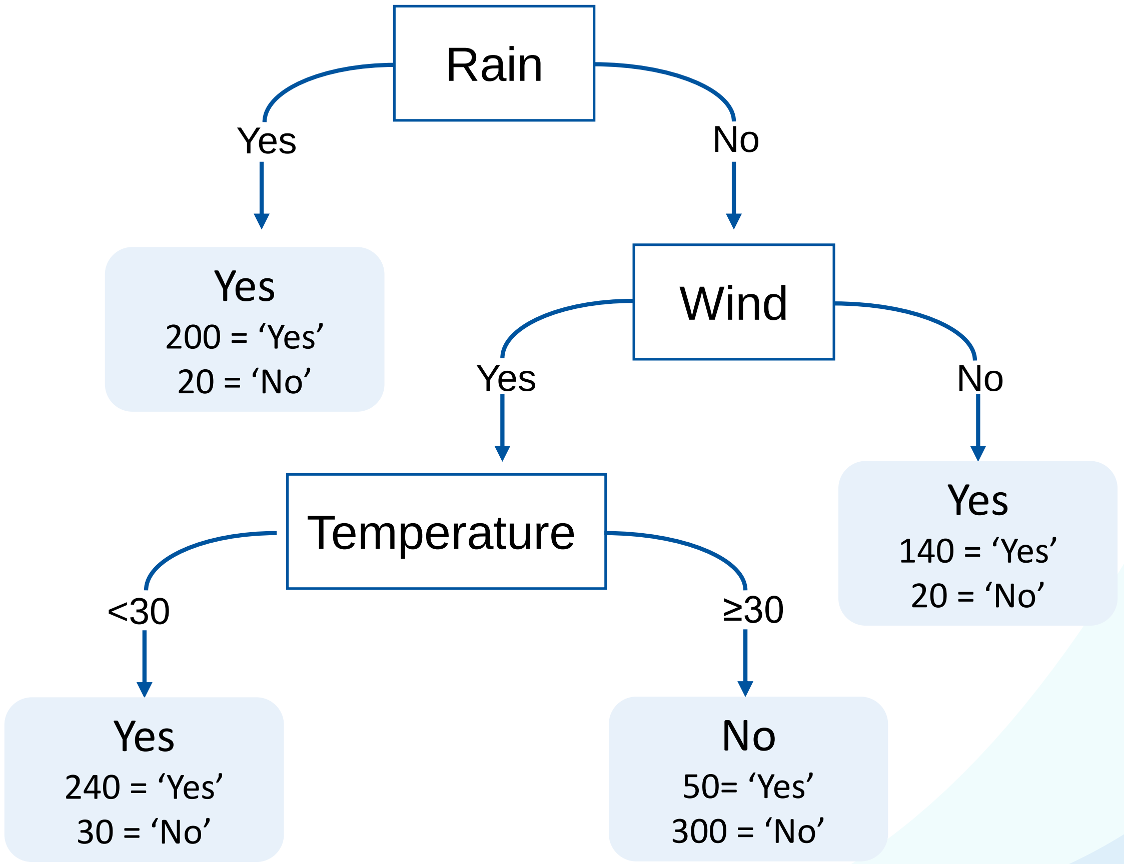
Target feature

Descriptive features

Example 2

Rain	Wind	Temperature (◦C)	Play tennis
Yes	Yes	15	**No**
No	No	34	**Yes**
Yes	No	23	**Yes**
Yes	Yes	20	**Yes**
No	Yes	28	**No**
…	…	…	…

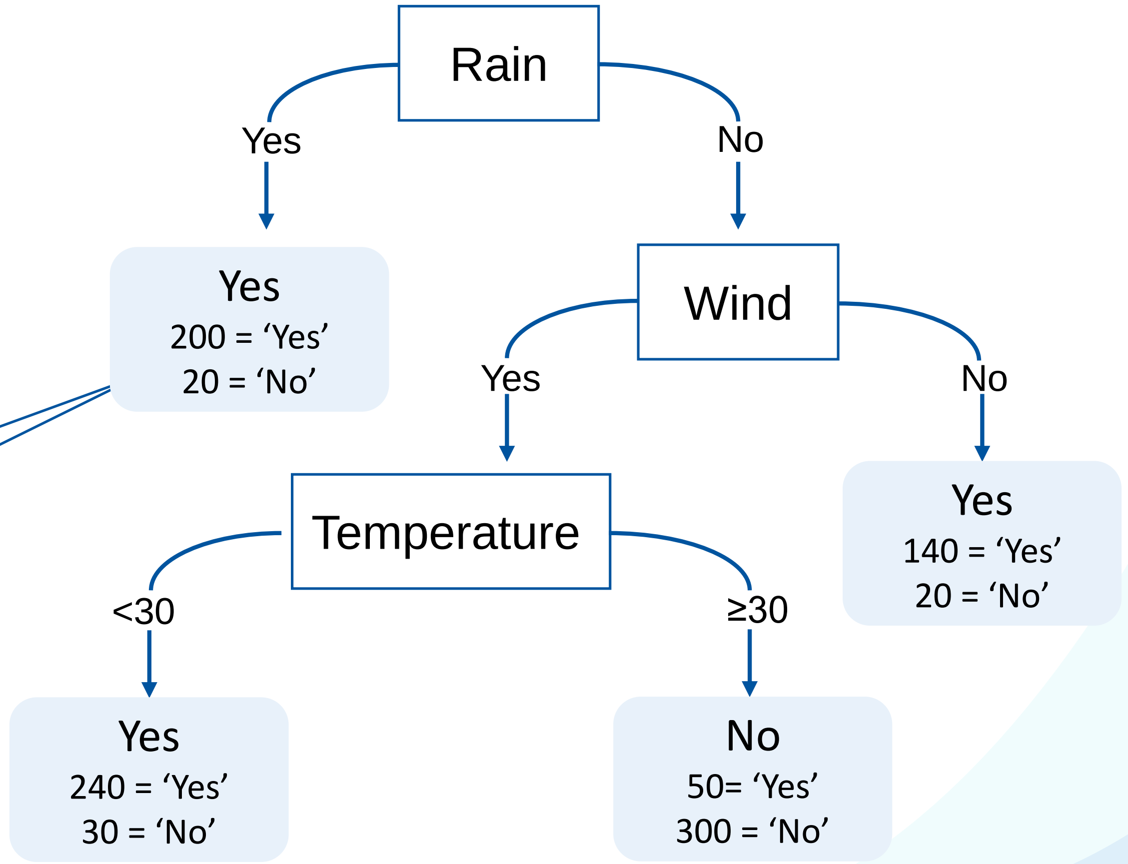
1000 Instances



Example 2

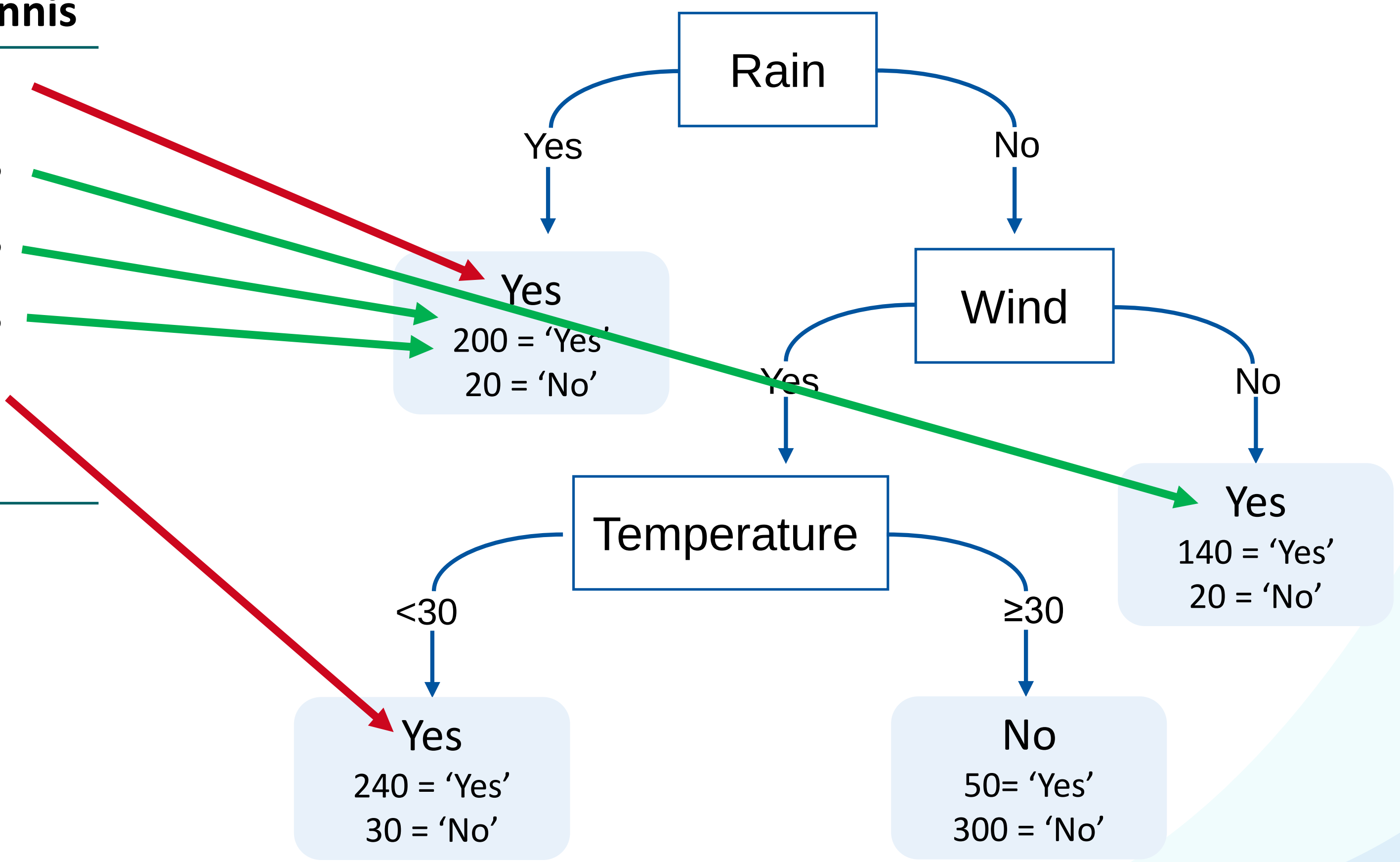
Rain	Wind	Temperature (◦C)	Play tennis
Yes	Yes	15	**No**
No	No	34	**Yes**
Yes	No	23	**Yes**
Yes	Yes	20	**Yes**
No	Yes	28	**No**
…	…	…	…

220 cases with *Rain = Yes* are classified as 'Yes' (Play tennis), but 20 are classified incorrectly



Example 2

Rain	Wind	Temperature (◦C)	Play tennis
Yes	Yes	15	**No**
No	No	34	**Yes**
Yes	No	23	**Yes**
Yes	Yes	20	**Yes**
No	Yes	28	**No**
…	…	…	…



Decision Tree Construction

Tree Structure

- Three types of nodes: root node, interior nodes and leaf nodes
- Root node refers to all instances
- Interior nodes partition the set of instances based on a descriptive feature
- Leaf nodes have a label (target feature value)
  (usually based on the label of the majority of instances in this node)

Decision Tree Construction

Tree Structure

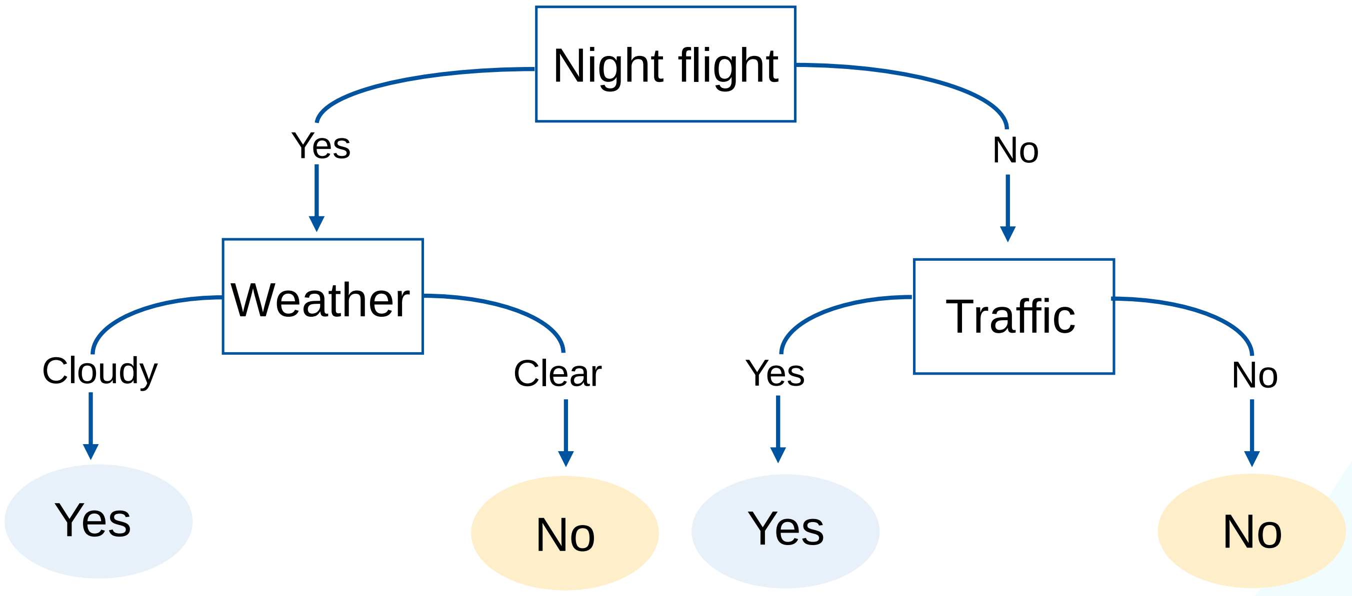
- Three types of nodes: root node, interior nodes and leaf nodes
- Root node refers to all instances
- Interior nodes partition the set of instances based on a descriptive feature
- Leaf nodes have a label (target feature value)
  (usually based on the label of the majority of instances in this node)

There are two goals (often conflicting)

- The tree is small and simple
- The leaves are homogeneous in terms of the target feature

Comparing Decision Trees (1/2)

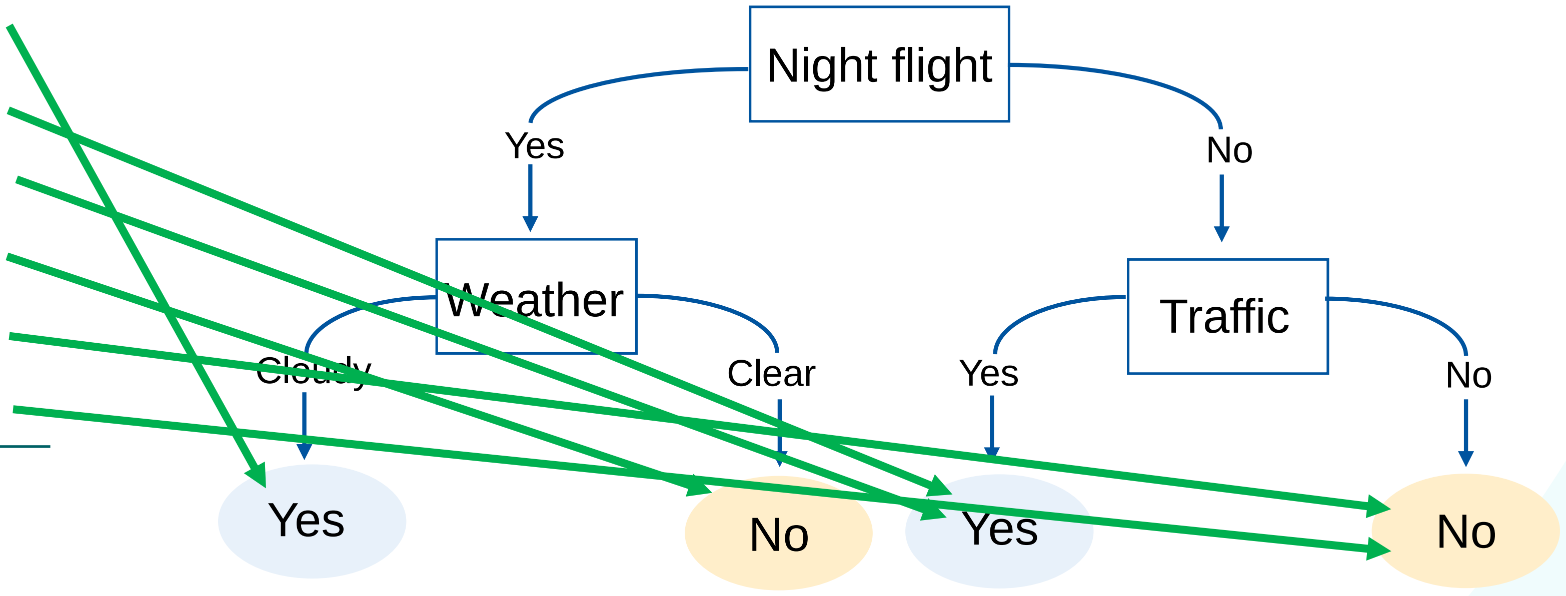
Weather	Traffic	Night flight	Flight delayed
Cloudy	No	Yes	**Yes**
Cloudy	Yes	No	**Yes**
Cloudy	Yes	No	**Yes**
Clear	Yes	Yes	**No**
Clear	No	No	**No**
Clear	No	No	**No**



Comparing Decision Trees (1/2)

Weather	Traffic	Night flight	Flight delayed
Cloudy	No	Yes	**Yes**
Cloudy	Yes	No	**Yes**
Cloudy	Yes	No	**Yes**
Clear	Yes	Yes	**No**
Clear	No	No	**No**
Clear	No	No	**No**

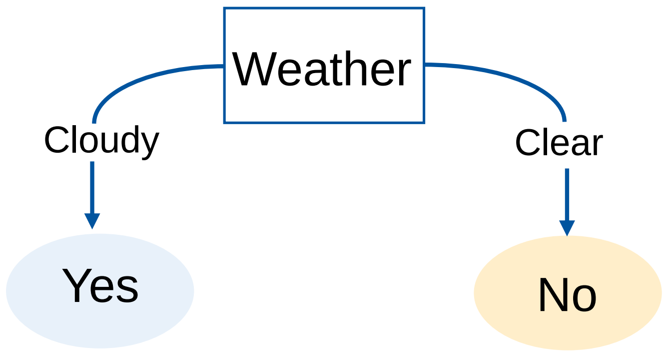
All instances correctly classified



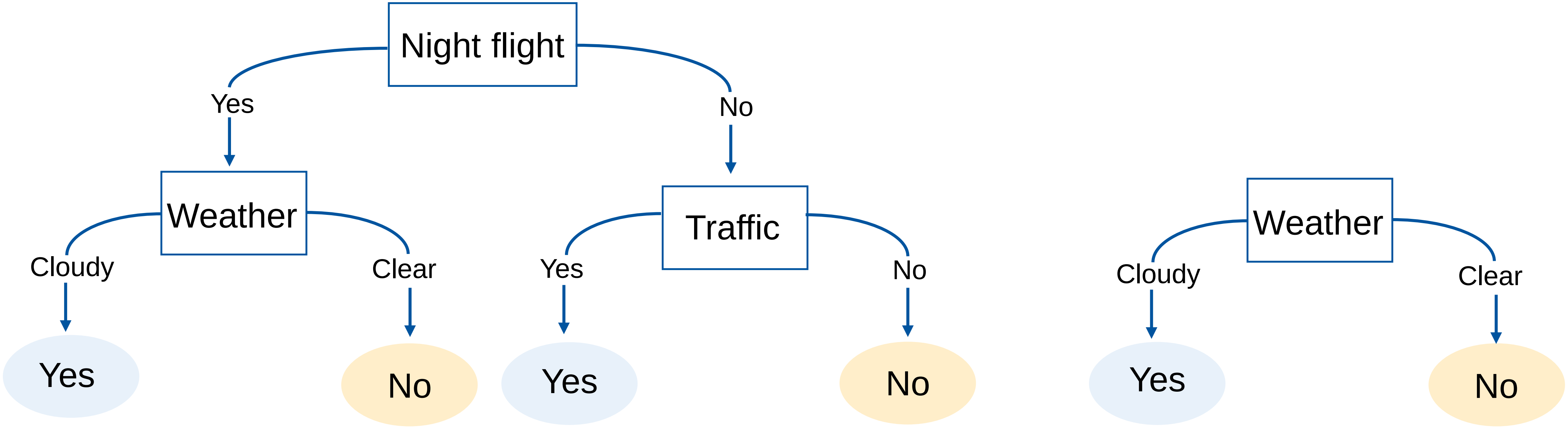
Comparing Decision Trees (2/2)

Weather	Traffic	Night flight	Flight delayed
Cloudy	No	Yes	**Yes**
Cloudy	Yes	No	**Yes**
Cloudy	Yes	No	**Yes**
Clear	Yes	Yes	**No**
Clear	No	No	**No**
Clear	No	No	**No**

All instances correctly classified



Comparing Decision Trees



Both trees correctly classify all observed instances, but the ‘simpler’ one seems ‘better’.

Key concepts:

- avoid overfitting
- apply Occam's razor
- prefer shallow trees

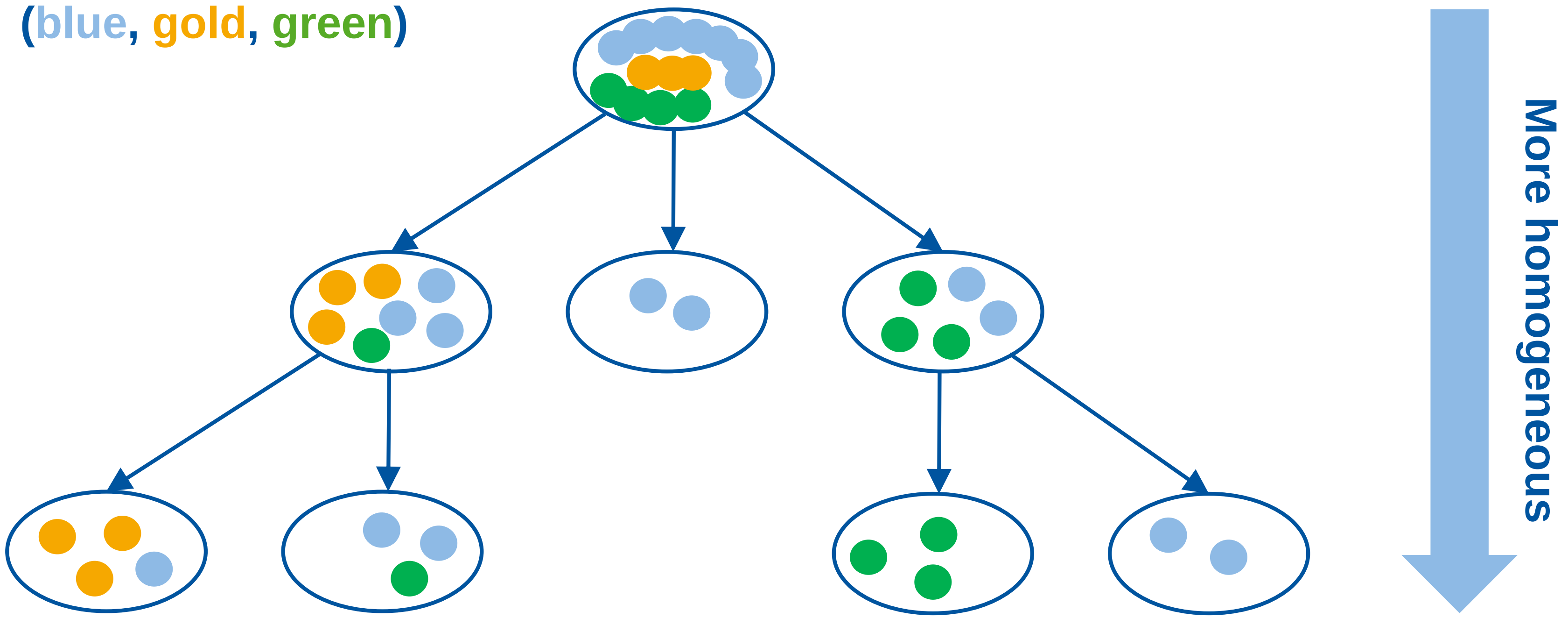
Characteristics Decision Trees

- A very simple model!
- In some cases, preferable to more complex and modern models (such as neural networks):
  - Fewer data points/attributes (managing overfitting is easier)
  - In domains where explainability and transparency are required
  - The choices of a tree are very easy to explain and show!

- There are extensions of decision trees that aim to combine simplicity and transparency with the ability to handle more complex data

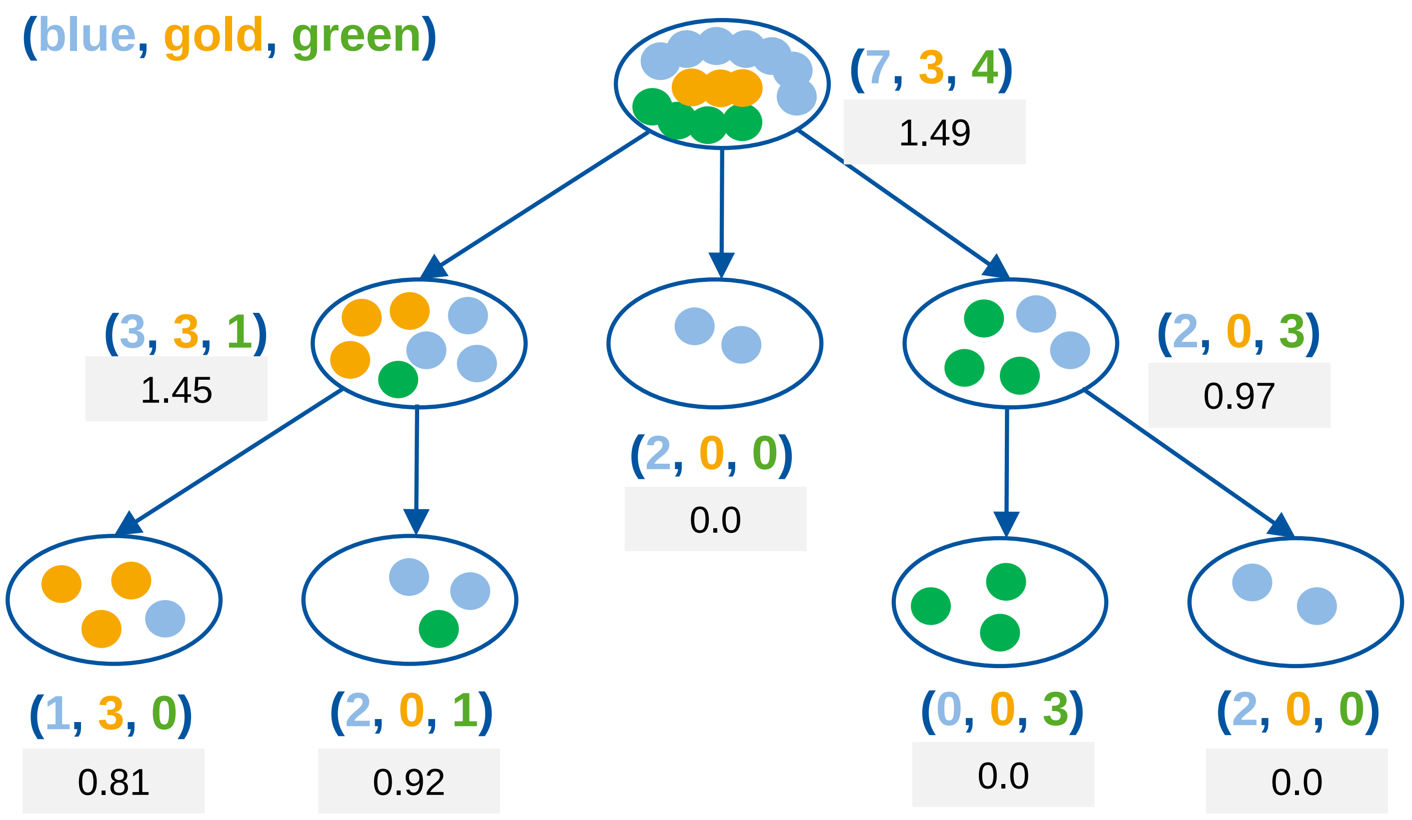
Information Gain

(blue, gold, green)



Information gain = improvement in knowledge
(predictability of target label in nodes)

Entropy - Intuition



Idea

- Measure of impurity
- Uncertainty when guessing
- Incompressibility

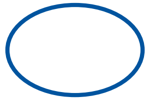
Worst case entropy for 3 values: $\log_2 3 \approx 1.58$

Entropy - Formula

$$H(t) = -\sum_{k=1}^{K}(P(t=k) \cdot log_s(P(t=k)))$$

(7, 3, 4)

$$H(color) = -(\frac{7}{14} \cdot log_2(\frac{7}{14}) + \frac{3}{14} \cdot log_2(\frac{3}{14}) + \frac{4}{14} \cdot log_2(\frac{4}{14})) \approx 1.49$$



t: examined target feature ($color$ in the example)

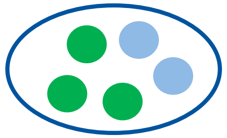
K: number of possible values of the target feature ($K = |\{blue, gold, green\}| = 3$ in the example)

$P(t = k) \in [0, 1]$: probability that a random value in t equals the kth value in the set of possible values

s: logarithm base (we use $s = 2$ by convention)

Entropy - Formula

$$H(t) = -\sum_{k=1}^{K}(P(t=k) \cdot log_s(P(t=k)))$$

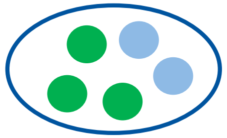


(2, 0, 3)

$$H(color) = -(\frac{2}{5} \cdot log_2(\frac{2}{5}) + \frac{0}{5} \cdot log_2(\frac{0}{5}) + \frac{3}{5} \cdot log_2(\frac{3}{5})) \approx 0.97$$

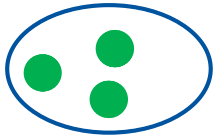
Entropy - Formula

$$H(t) = -\sum_{k=1}^{K}(P(t=k) \cdot log_s(P(t=k)))$$



(2, 0, 3)

$$H(color) = -(\frac{2}{5} \cdot log_2(\frac{2}{5}) + \frac{0}{5} \cdot log_2(\frac{0}{5}) + \frac{3}{5} \cdot log_2(\frac{3}{5})) \approx 0.97$$

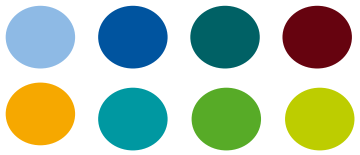


(0, 0, 3)

$$H(color) = -(\frac{0}{3} \cdot log_2(\frac{0}{3}) + \frac{0}{3} \cdot log_2(\frac{0}{3}) + \frac{3}{3} \cdot log_2(\frac{3}{3})) = 0$$

Questions

Suppose that we have *K* possible values (colors) and *N* instances (balls).

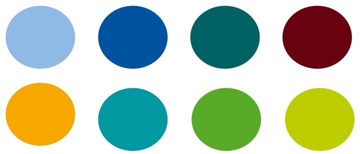


$$H(t) = -\sum_{k=1}^{K}(P(t=k) \cdot log_s(P(t=k)))$$

What distribution of the *N* instances over the *K* possible values yields the lowest entropy?

Questions

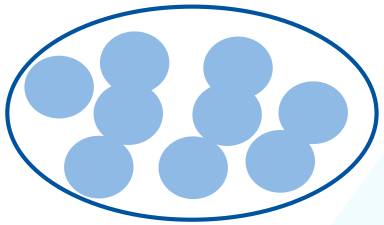
Suppose that we have *K* possible values (colors) and *N* instances (balls).



$$H(t) = -\sum_{k=1}^{K}(P(t=k) \cdot log_s(P(t=k)))$$

What distribution of the *N* instances over the *K* possible values yields the lowest entropy?

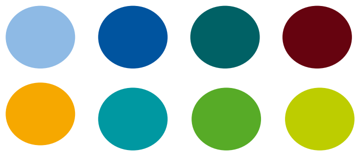
$$H(color) = -(1 \cdot log_2(1)) = 0$$



→ all instances have the same value

Questions

Suppose that we have *K* possible values (colors) and *N* instances (balls).

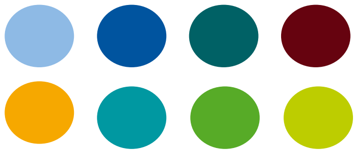


$$H(t) = -\sum_{k=1}^{K}(P(t=k)\cdot log_s(P(t=k)))$$

What distribution of the *N* instances over the *K* possible values yields the highest entropy?

Questions

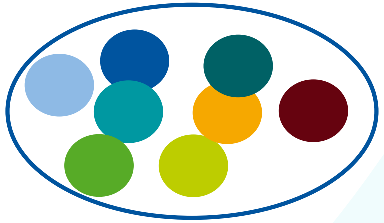
Suppose that we have *K* possible values (colors) and *N* instances (balls).



$$H(t) = -\sum_{k=1}^{K}(P(t=k) \cdot log_s(P(t=k)))$$

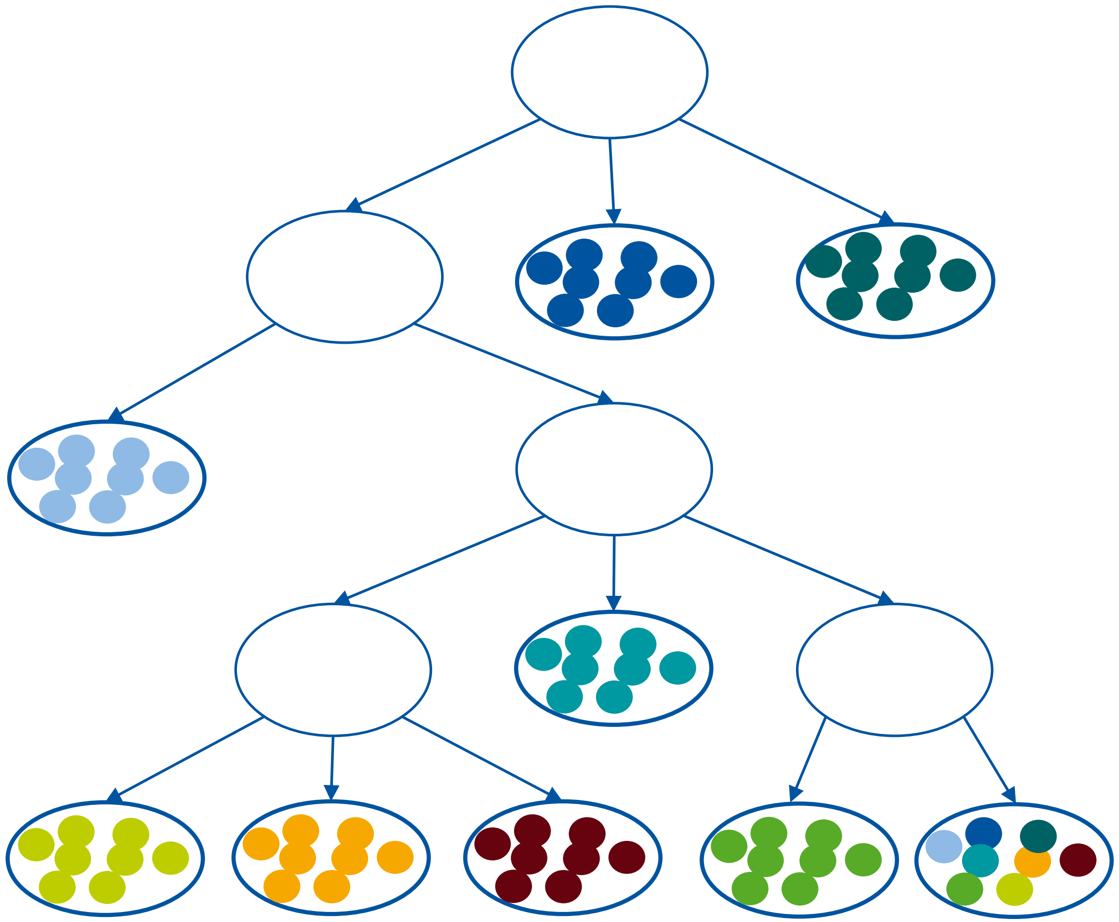
What distribution of the *N* instances over the *K* possible values yields the highest entropy?

$$H(color) = -\sum_{k=1}^{K}(\frac{1}{K} \cdot log_2(\frac{1}{K}))$$
$$= -(K \cdot \frac{1}{K} \cdot log_2(\frac{1}{K}))$$
$$= -log_2(\frac{1}{K})$$
$$= log_2(K)$$



→ Even distribution over all possible values

Overall Entropy

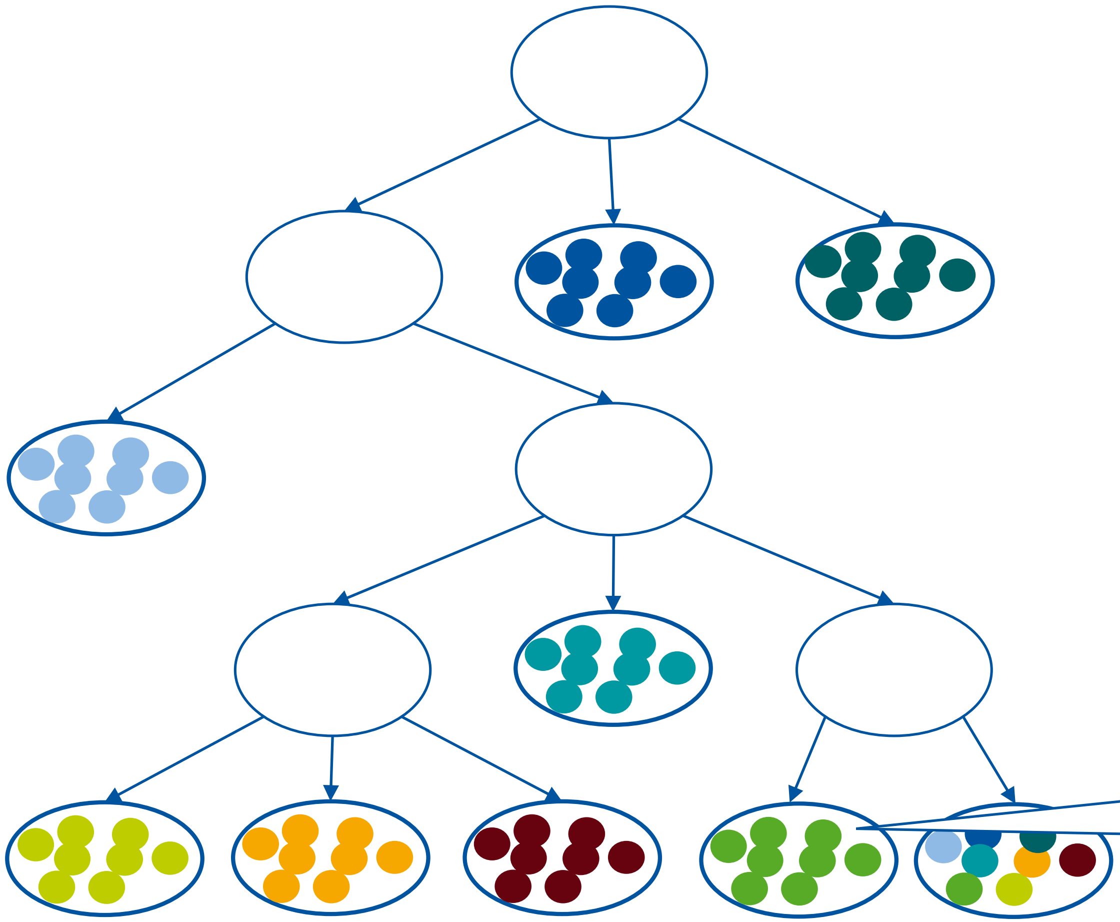


Overall entropy H_W is the weighted average of the individual entropies:

$$H_W(t) = \sum_{node \in nodes} \left(\frac{|node|}{N} \cdot H^{node}(t)\right)$$

Example: $N = 72, K = 8$

Overall Entropy



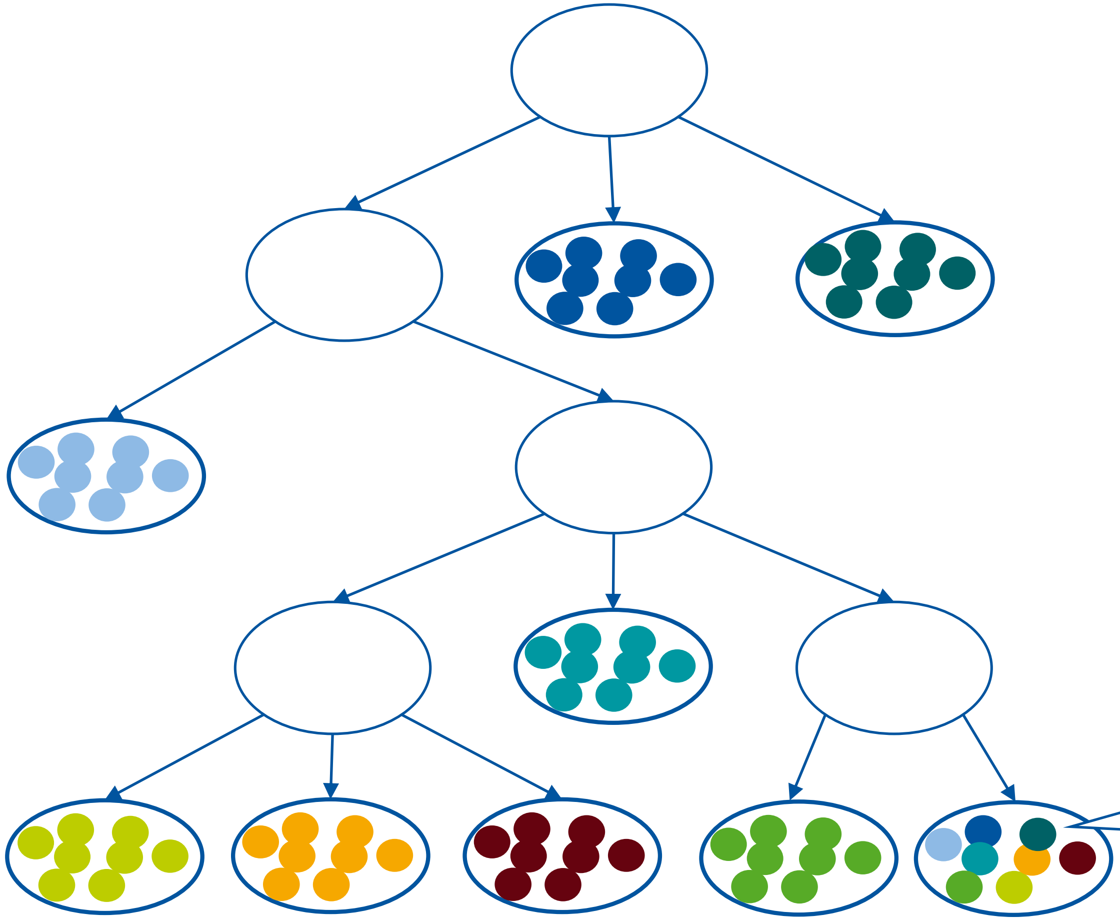
Overall entropy H_W is the weighted average of the individual entropies:

$$H_W(t) = \sum_{node \in nodes} \left(\frac{|node|}{N} \cdot H^{node}(t) \right)$$

Example: $N = 72, K = 8$

8 homogeneously colored balls:
$H^{node}(color) = -(\frac{8}{8} \cdot \log_2(\frac{8}{8})) = 0$

Overall Entropy



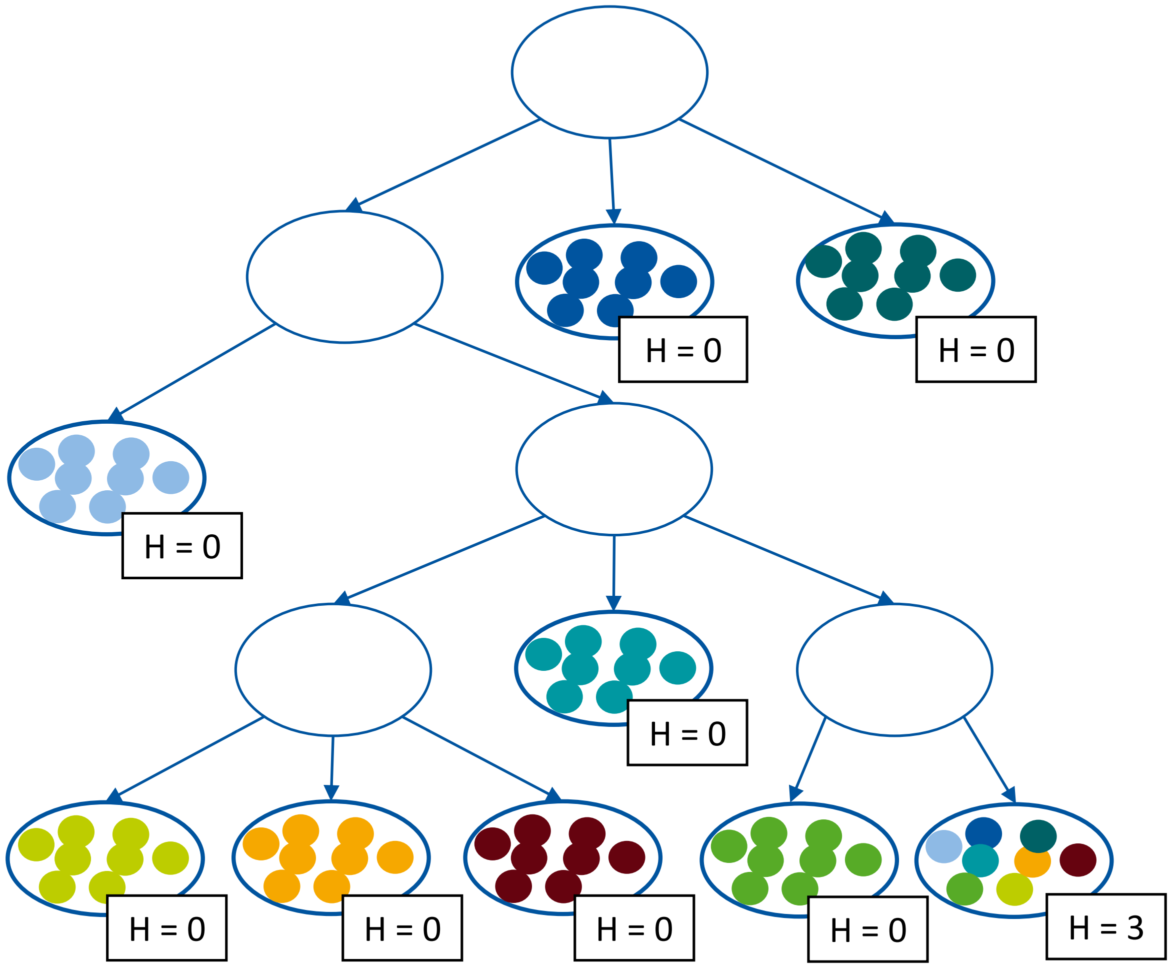
Overall entropy H_W is the weighted average of the individual entropies:

$$H_W(t) = \sum_{node \in nodes} \left(\frac{|node|}{N} \cdot H^{node}(t) \right)$$

Example: $N = 72, K = 8$

Even distribution of 8 colors over 8 balls:
$H^{node}(color) = -\sum_{k=1}^{8} \frac{1}{8} \cdot log_2(\frac{1}{8}) = \log_2(8) = 3$

Overall Entropy

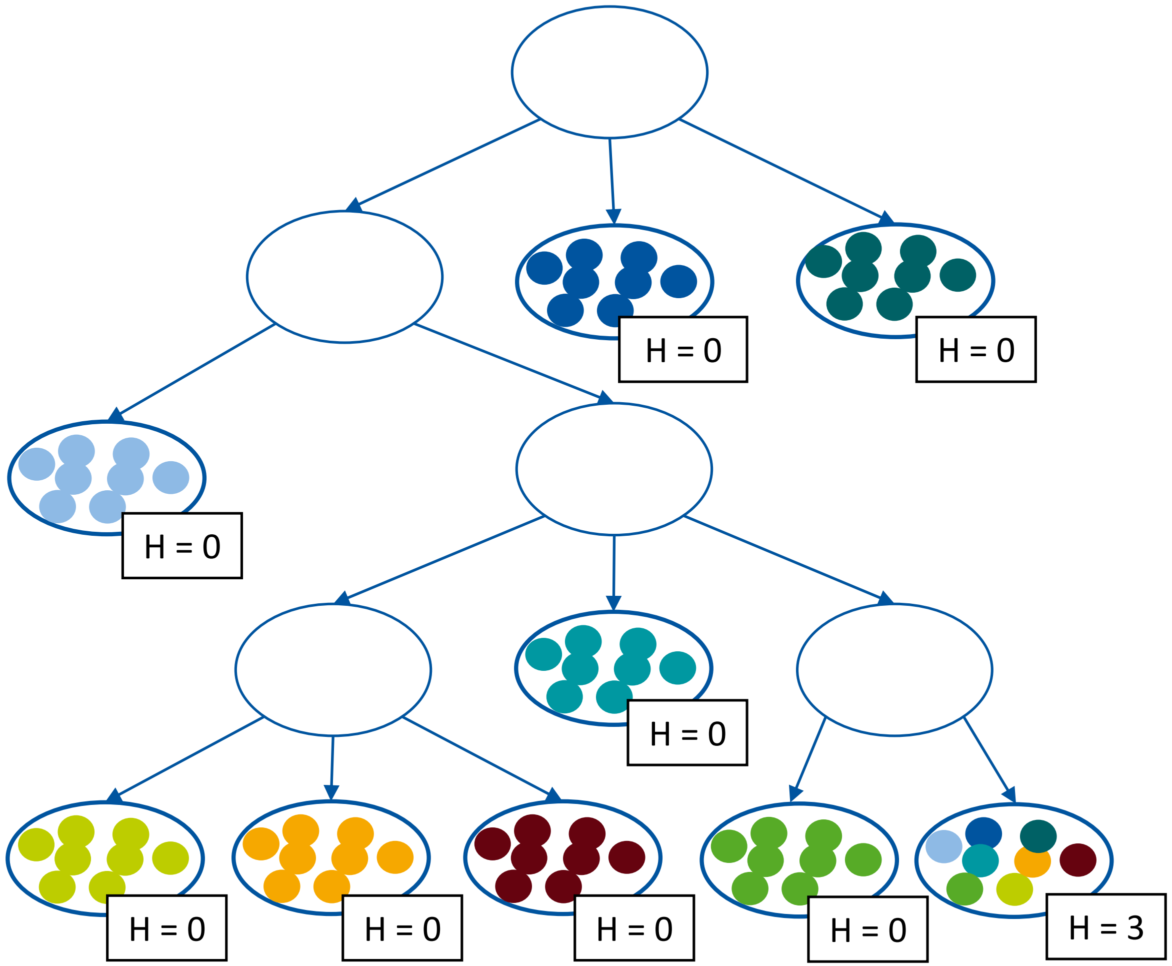


Overall entropy H_W is the weighted average of the individual entropies:

$$H_W(t) = \sum_{node \in nodes} \left(\frac{|node|}{N} \cdot H^{node}(t) \right)$$

Example: $N = 72, K = 8$

Overall Entropy



Overall entropy H_W is the weighted average of the individual entropies:

$$H_W(t) = \sum_{node \in nodes} \left(\frac{|node|}{N} \cdot H^{node}(t) \right)$$

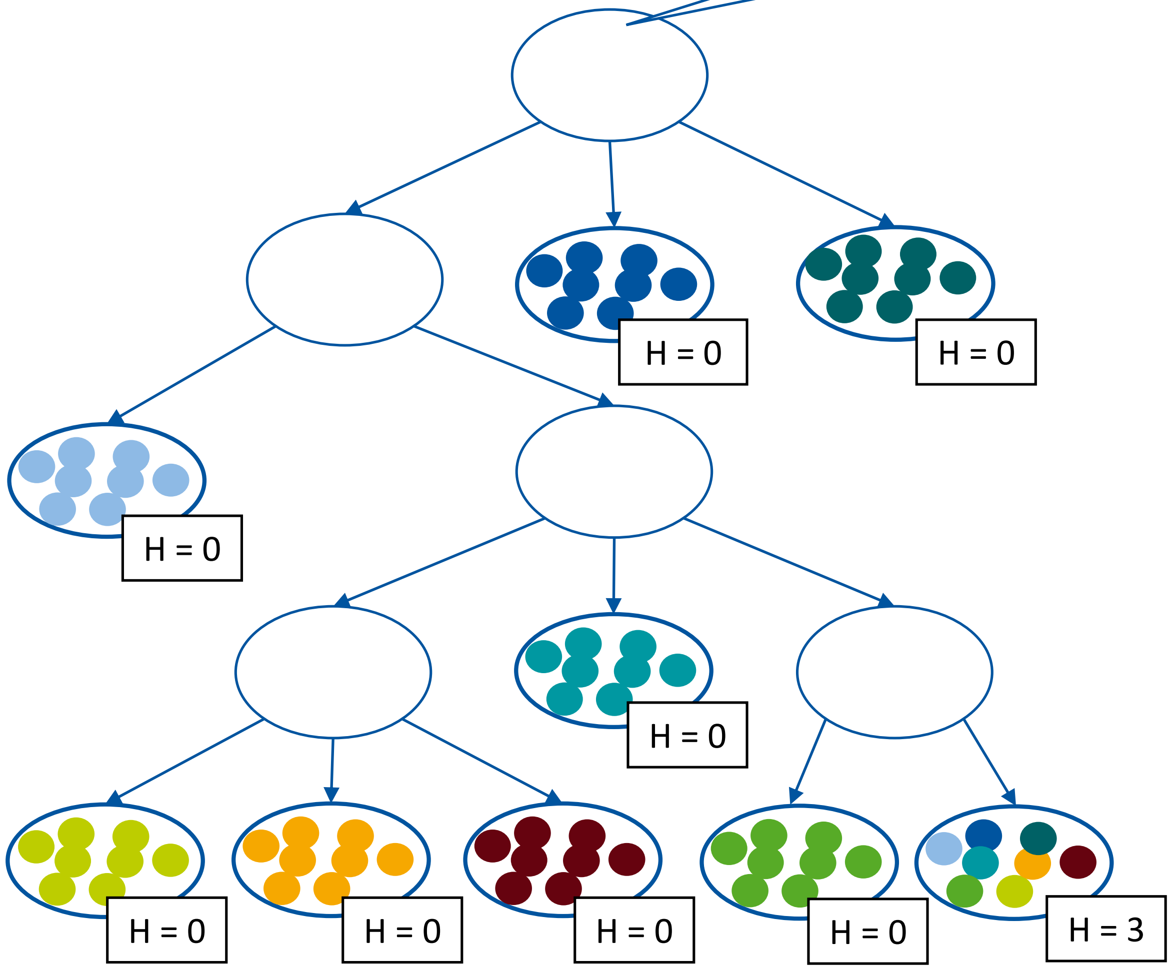
Example: $N = 72, K = 8$

$$H_W(color) = \frac{8}{72} \cdot 0 + \frac{8}{72} \cdot 0 + \frac{8}{72} \cdot 0 + \frac{8}{72} \cdot 0 + \frac{8}{72} \cdot 0 + \frac{8}{72} \cdot 0 + \frac{8}{72} \cdot 0 + \frac{8}{72} \cdot 0 + \frac{8}{72} \cdot 3 = \frac{24}{72} \approx 0.33$$

Overall Entropy

Even distribution of 8 colors over 72 balls:

$$H_W(color) = \frac{72}{72} \cdot \left(-\sum_{k=1}^{8}\left(\frac{9}{72} \cdot log_2(\frac{9}{72})\right)\right) = \log_2(8) = 3$$



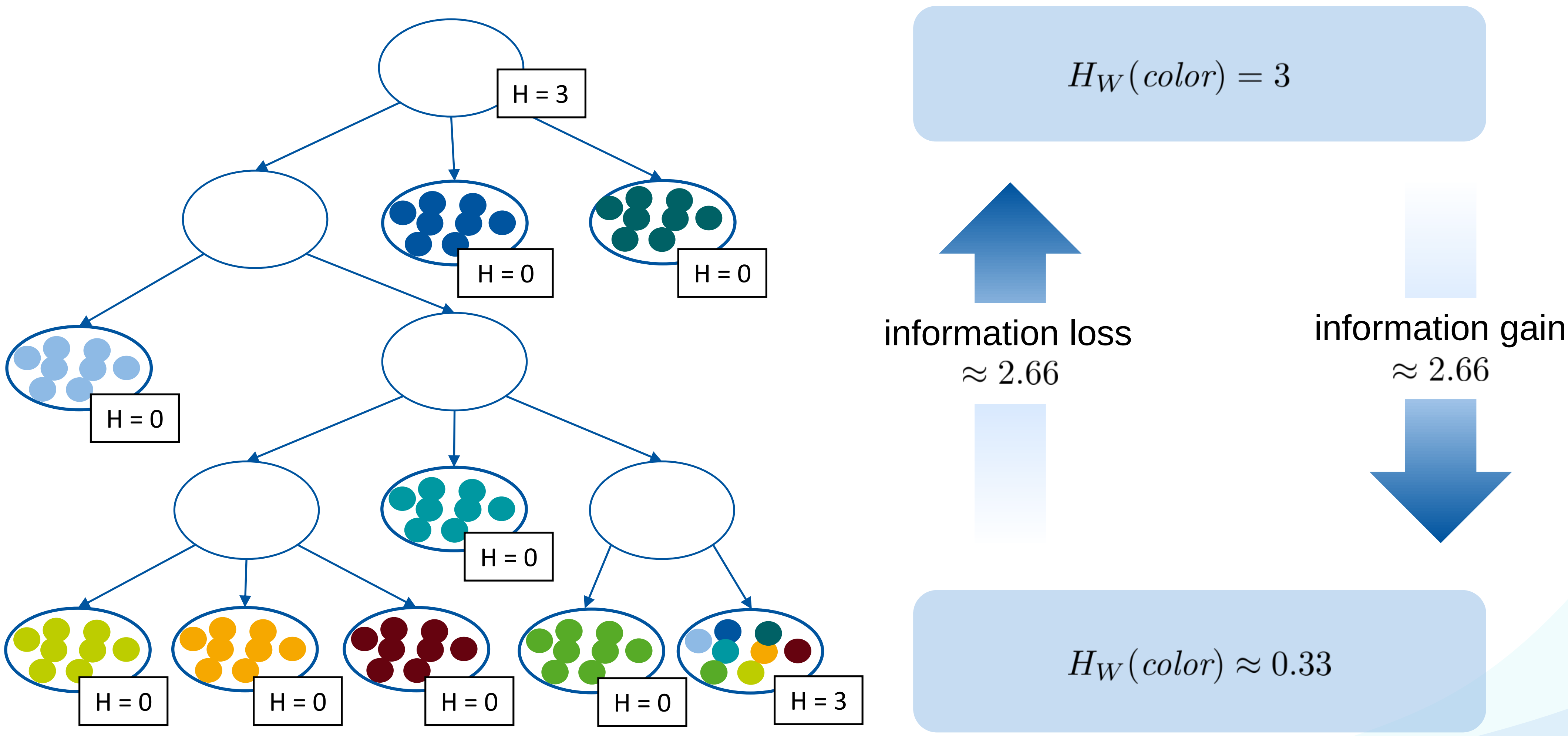
Overall entropy H_W is the weighted average of the individual entropies:

$$H_W(t) = \sum_{node \in nodes}(\frac{|node|}{N} \cdot H^{node}(t))$$

Example: $N = 72, K = 8$

$$H_W(color) = \frac{8}{72} \cdot 0 + \frac{8}{72} \cdot 0 + \frac{8}{72} \cdot 0 + \frac{8}{72} \cdot 0 + \frac{8}{72} \cdot 0 + \frac{8}{72} \cdot 0 + \frac{8}{72} \cdot 0 + \frac{8}{72} \cdot 0 + \frac{8}{72} \cdot 3 = \frac{24}{72} \approx 0.33$$

Information Gain



Information Gain - Example Revisited

$$H(\text{delayed}) = 1$$

Weather	Traffic	Night flight	Flight delayed
Cloudy	No	Yes	**Yes**
Cloudy	Yes	No	**Yes**
Cloudy	Yes	No	**Yes**
Clear	Yes	Yes	**No**
Clear	No	No	**No**
Clear	No	No	**No**

Information Gain - Example Revisited

$$H(\text{delayed}) = 1$$

Weather	Traffic	Night flight	Flight delayed
Cloudy	No	Yes	**Yes**
Cloudy	Yes	No	**Yes**
Cloudy	Yes	No	**Yes**
Clear	Yes	Yes	**No**
Clear	No	No	**No**
Clear	No	No	**No**

$$H^{cloudy}(\text{delayed}) = 0$$
$$H^{clear}(\text{delayed}) = 0$$

$$H_W^{weather}(\text{delayed}) = 0$$

Weather	Flight delayed
Cloudy	**Yes**
Cloudy	**Yes**
Cloudy	**Yes**
Clear	**No**
Clear	**No**
Clear	**No**

Information Gain - Example Revisited

$$H(\text{delayed}) = 1$$

Weather	Traffic	Night flight	Flight delayed
Cloudy	No	Yes	**Yes**
Cloudy	Yes	No	**Yes**
Cloudy	Yes	No	**Yes**
Clear	Yes	Yes	**No**
Clear	No	No	**No**
Clear	No	No	**No**

$$H^{cloudy}(\text{delayed}) = 0$$
$$H^{clear}(\text{delayed}) = 0$$

$$H_W^{weather}(\text{delayed}) = 0$$

Weather	Flight delayed
Cloudy	**Yes**
Cloudy	**Yes**
Cloudy	**Yes**
Clear	**No**
Clear	**No**
Clear	**No**

$$H^{traffic\_yes}(\text{delayed}) = 0.92$$
$$H^{traffic\_no}(\text{delayed}) = 0.92$$

$$H_W^{traffic}(\text{delayed}) = 0.92$$

Traffic	Flight delayed
No	**Yes**
Yes	**Yes**
Yes	**Yes**
Yes	**No**
No	**No**
No	**No**

Information Gain - Example Revisited

$$H(\text{delayed}) = 1$$

Weather	Traffic	Night flight	Flight delayed
Cloudy	No	Yes	**Yes**
Cloudy	Yes	No	**Yes**
Cloudy	Yes	No	**Yes**
Clear	Yes	Yes	**No**
Clear	No	No	**No**
Clear	No	No	**No**

$$H^{cloudy}(\text{delayed}) = 0$$
$$H^{clear}(\text{delayed}) = 0$$

$$H_W^{weather}(\text{delayed}) = 0$$

Weather	Flight delayed
Cloudy	**Yes**
Cloudy	**Yes**
Cloudy	**Yes**
Clear	**No**
Clear	**No**
Clear	**No**

$$H^{traffic\_yes}(\text{delayed}) = 0.92$$
$$H^{traffic\_no}(\text{delayed}) = 0.92$$

$$H_W^{traffic}(\text{delayed}) = 0.92$$

Traffic	Flight delayed
No	**Yes**
Yes	**Yes**
Yes	**Yes**
Yes	**No**
No	**No**
No	**No**

$$H^{night\_yes}(\text{delayed}) = 0$$
$$H^{night\_no}(\text{delayed}) = 1$$

$$H_W^{night\_flight}(\text{delayed}) \approx 0.67$$

Night flight	Flight delayed
Yes	**No**
No	**Yes**
No	**Yes**
Yes	**No**
No	**No**
No	**No**

Information Gain - Example Revisited

$H(\text{delayed}) = 1$

Weather	Traffic	Night flight	Flight delayed
Cloudy	No	Yes	**Yes**
…	…	…	**…**

$H^{cloudy}(\text{delayed}) = 0$
$H^{clear}(\text{delayed}) = 0$

$H_W^{weather}(\text{delayed}) = 0$

Weather	Flight delayed
Cloudy	**Yes**
…	**…**

$H^{traffic\_yes}(\text{delayed}) = 0.92$
$H^{traffic\_no}(\text{delayed}) = 0.92$

$H_W^{traffic}(\text{delayed}) = 0.92$

Traffic	Flight delayed
No	**Yes**
…	**…**

$H^{night\_yes}(\text{delayed}) = 0$
$H^{night\_no}(\text{delayed}) = 1$

$H_W^{night\_flight}(\text{delayed}) \approx 0.67$

Night flight	Flight delayed
Yes	**No**
…	**…**

$$IG(weather) = H(delayed) - H_W^{weather}(delayed) = 1 - 0 = 1$$

Information Gain - Example Revisited

$H(\text{delayed}) = 1$

Weather	Traffic	Night flight	Flight delayed
Cloudy	No	Yes	**Yes**
...	...	...	**...**

$H^{cloudy}(\text{delayed}) = 0$
$H^{clear}(\text{delayed}) = 0$

$H_W^{weather}(\text{delayed}) = 0$

Weather	Flight delayed
Cloudy	**Yes**
...	**...**

$H^{traffic\_yes}(\text{delayed}) = 0.92$
$H^{traffic\_no}(\text{delayed}) = 0.92$

$H_W^{traffic}(\text{delayed}) = 0.92$

Traffic	Flight delayed
No	**Yes**
...	**...**

$H^{night\_yes}(\text{delayed}) = 0$
$H^{night\_no}(\text{delayed}) = 1$

$H_W^{night\_flight}(\text{delayed}) \approx 0.67$

Night flight	Flight delayed
Yes	**No**
...	**...**

$$IG(weather) = H(delayed) - H_W^{weather}(delayed) = 1 - 0 = 1$$

$$IG(traffic) = H(delayed) - H_W^{traffic}(delayed) = 1 - 0.92 = 0.08$$

Information Gain - Example Revisited

$H(\text{delayed}) = 1$

Weather	Traffic	Night flight	Flight delayed
Cloudy	No	Yes	**Yes**
...	...	...	**...**

$H^{cloudy}(\text{delayed}) = 0$
$H^{clear}(\text{delayed}) = 0$

$H_W^{weather}(\text{delayed}) = 0$

Weather	Flight delayed
Cloudy	**Yes**
...	**...**

$H^{traffic\_yes}(\text{delayed}) = 0.92$
$H^{traffic\_no}(\text{delayed}) = 0.92$

$H_W^{traffic}(\text{delayed}) = 0.92$

Traffic	Flight delayed
No	**Yes**
...	**...**

$H^{night\_yes}(\text{delayed}) = 0$
$H^{night\_no}(\text{delayed}) = 1$

$H_W^{night\_flight}(\text{delayed}) \approx 0.67$

Night flight	Flight delayed
Yes	**No**
...	**...**

$$IG(weather) = H(delayed) - H_W^{weather}(delayed) = 1 - 0 = 1$$

$$IG(traffic) = H(delayed) - H_W^{traffic}(delayed) = 1 - 0.92 = 0.08$$

$$IG(night\_flight) = H(delayed) - H_W^{night\_flight}(delayed) = 1 - 0.67 = 0.33$$

Information Gain - Example Revisited

$H(\text{delayed}) = 1$

Weather	Traffic	Night flight	Flight delayed
Cloudy	No	Yes	**Yes**
...	...	...	**...**

$H^{cloudy}(\text{delayed}) = 0$
$H^{clear}(\text{delayed}) = 0$

$H_W^{weather}(\text{delayed}) = 0$

Weather	Flight delayed
Cloudy	**Yes**
...	**...**

$H^{traffic\_yes}(\text{delayed}) = 0.92$
$H^{traffic\_no}(\text{delayed}) = 0.92$

$H_W^{traffic}(\text{delayed}) = 0.92$

Traffic	Flight delayed
No	**Yes**
...	**...**

$H^{night\_yes}(\text{delayed}) = 0$
$H^{night\_no}(\text{delayed}) = 1$

$H_W^{night\_flight}(\text{delayed}) \approx 0.67$

Night flight	Flight delayed
Yes	**No**
...	**...**

$$IG(weather) = H(delayed) - H_W^{weather}(delayed) = 1 - 0 = 1$$ **good**

$$IG(traffic) = H(delayed) - H_W^{traffic}(delayed) = 1 - 0.92 = 0.08$$ **worst**

$$IG(night\_flight) = H(delayed) - H_W^{night\_flight}(delayed) = 1 - 0.67 = 0.33$$ **not so good**

ID3 (Iterative Dichotomiser 3) - Key Idea

Approach

1. For each feature: calculate the resulting entropy splitting the dataset $\mathcal{X}$ using the selected feature
2. Split the set $\mathcal{X}$ into subsets using the feature for which the resulting entropy (after splitting) is minimal (equivalently, information gain is maximum)
3. Create a decision tree node based on that feature
4. Recurse on subsets using remaining features (until stopping criteria are reached)

When to Stop?

Three stopping criteria

- When all of the instances have the same classification (label = consensus value)
- When there are no features left (label = majority value)
- When the dataset is empty (label = majority parent)

Algorithm

ID3 algorithm:

1. **if** all the instances in X have the same classification
   (a) **return** a decision tree with one leaf node with consensus value as a label
2. **else if** there are no features left
   (a) **return** a decision tree with one leaf node with majority value as a label
3. **else if** the dataset is empty
   (a) **return** a decision tree with one leaf node with majority parent value as a label

three stopping criteria

4. **else**
   (a) pick a feature that maximizes information gain
   (b) once a feature is picked along a path from the root, it cannot be used again
   (c) create subproblems based on the selected feature

recursively constructing the tree

Example

$$H(\text{Customer})$$
$$= -(\frac{2}{7} \cdot log_2(\frac{2}{7}) + \frac{3}{7} \cdot log_2(\frac{3}{7}) + \frac{2}{7} \cdot log_2(\frac{2}{7}))$$
$$= 1.5567$$

ID	Insurance	Education	Employment	Customer
1	Yes	Bachelor	Employed	Basic
2	Yes	High school	Unemployed	Premium
3	Yes	Bachelor	Self-employed	Premium
4	No	Bachelor	Self-employed	Basic
5	No	Master	Employed	Economy
6	Yes	Bachelor	Retired	Economy
7	Yes	Bachelor	Employed	Premium

Example

ID	Insurance	Education	Employment	Customer
1	Yes	Bachelor	Employed	Basic
2	Yes	High school	Unemployed	Premium
3	Yes	Bachelor	Self-employed	Premium
4	No	Bachelor	Self-employed	Basic
5	No	Master	Employed	Economy
6	Yes	Bachelor	Retired	Economy
7	Yes	High school	Employed	Premium

$$H(\text{Customer}) = 1.5567$$

Split by feature	Possible Values	Instances	Entropy	Overall Entropy	Information Gain
Insurance	No	4, 5	1	1.265	1.5567 – 1.265 = **0.2917**
	Yes	1, 2, 3, 6, 7	1.3710		
Education	High school	2, 7	0	0.8571	1.5567 – 0.8571 = **0.6996**
	Master	5	0		
	Bachelor	1, 3, 4, 6	1.5		
Employment	Employed	1, 5, 7	1.5850	0.9650	1.5567 – 0.9650= **0.5917**
	Unemployed	2	0		
	Self-employed	3, 4	1		
	Retired	6	0		

Example

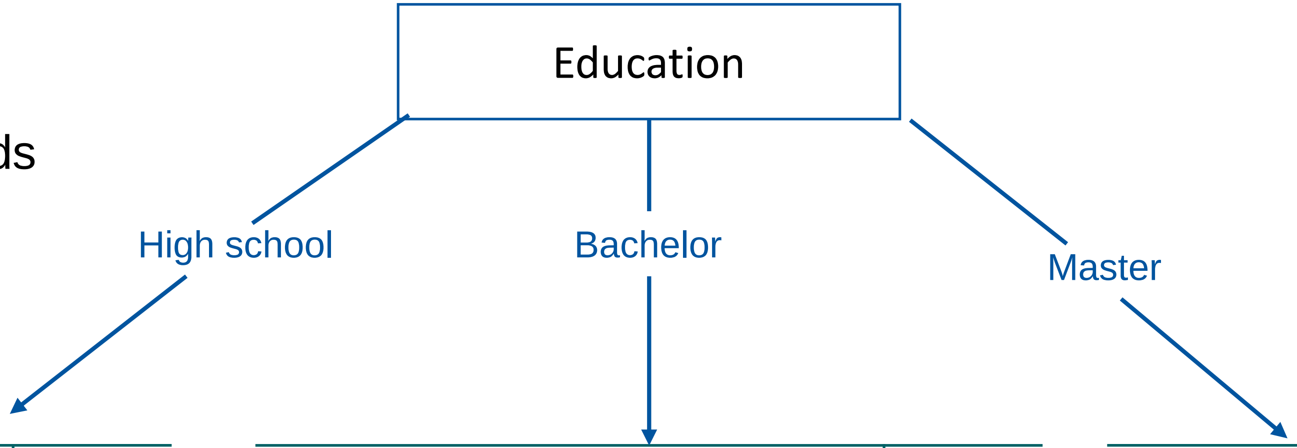
ID	Insurance	Education	Employment	Customer
1	Yes	Bachelor	Employed	Basic
2	Yes	High school	Unemployed	Premium
3	Yes	Bachelor	Self-employed	Premium
4	No	Bachelor	Self-employed	Basic
5	No	Master	Employed	Economy
6	Yes	Bachelor	Retired	Economy
7	Yes	High school	Employed	Premium

$$H(\text{Customer}) = 1.5567$$

Split by feature	Possible Values	Instances	Entropy	Overall Entropy	Information Gain
Insurance	No	4, 5	1	1.265	1.5567 – 1.265 = **0.2917**
	Yes	1, 2, 3, 6, 7	1.3710		
Education	High school	2, 7	0	0.8571	1.5567 – 0.8571 = **0.6996**
	Master	5	0		
	Bachelor	1, 3, 4, 6	1.5		
Employment	Employed	1, 5, 7	1.5850	0.9650	1.5567 – 0.9650= **0.5917**
	Unemployed	2	0		
	Self-employed	3, 4	1		
	Retired	6	0		

Example

Recursion until stopping criteria holds



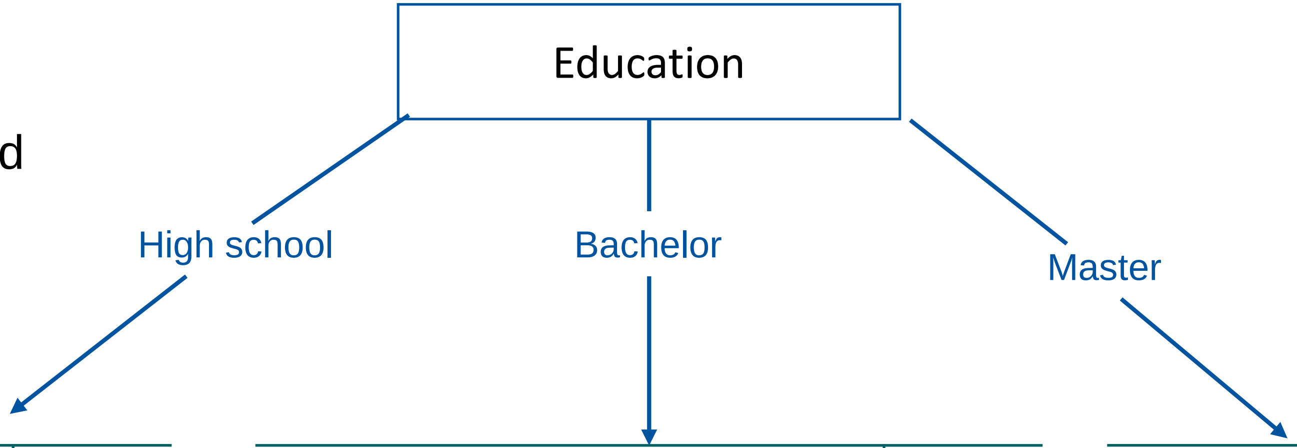
ID	Insurance	Education	Employment	Customer
2	Yes	High school	Unemployed	Premium
7	Yes	High school	Employed	Premium

ID	Insurance	Education	Employment	Customer
1	Yes	Bachelor	Employed	Basic
3	Yes	Bachelor	Self-employed	Premium
4	No	Bachelor	Self-employed	Basic
6	Yes	Bachelor	Retired	Economy

ID	Insurance	Education	Employment	Customer
5	No	Master	Employed	Economy

Example

Recursion until stopping criteria hold



ID	Insurance	Education	Employment	Customer
2	Yes	High school	Unemployed	Premium
7	Yes	High school	Employed	Premium

consensus

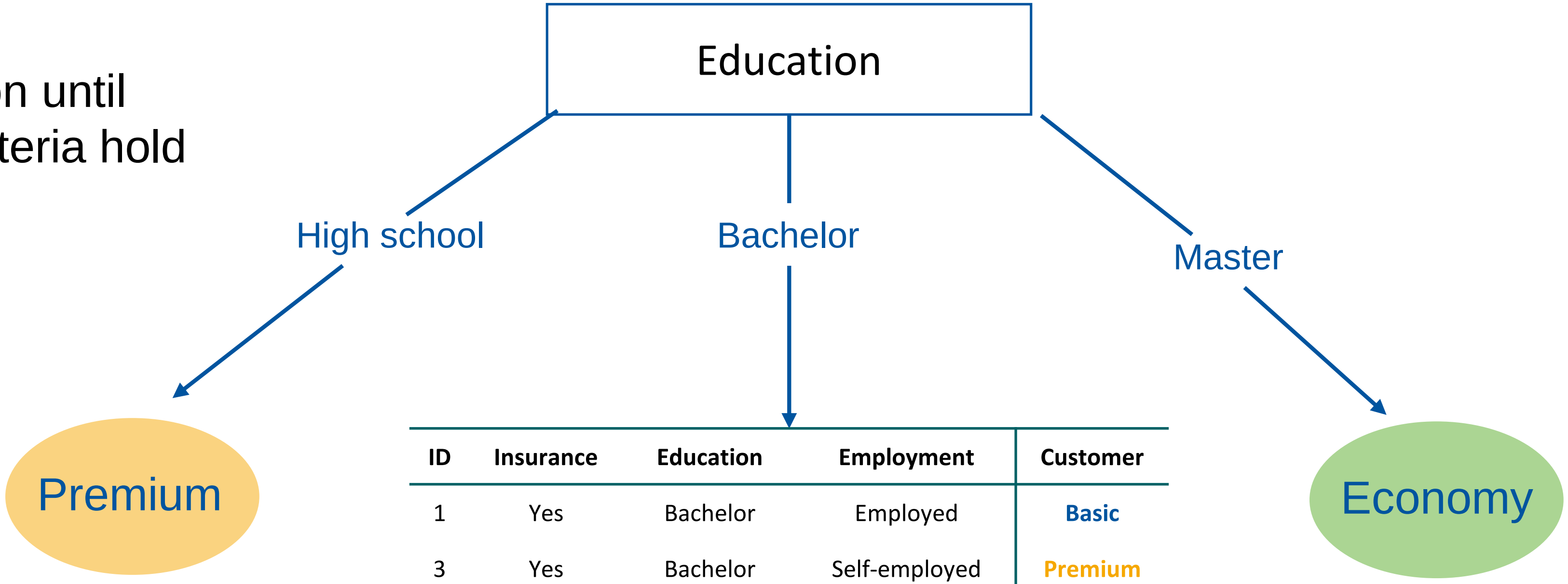
ID	Insurance	Education	Employment	Customer
1	Yes	Bachelor	Employed	Basic
3	Yes	Bachelor	Self-employed	Premium
4	No	Bachelor	Self-employed	Basic
6	Yes	Bachelor	Retired	Economy

ID	Insurance	Education	Employment	Customer
5	No	Master	Employed	Economy

consensus

Example

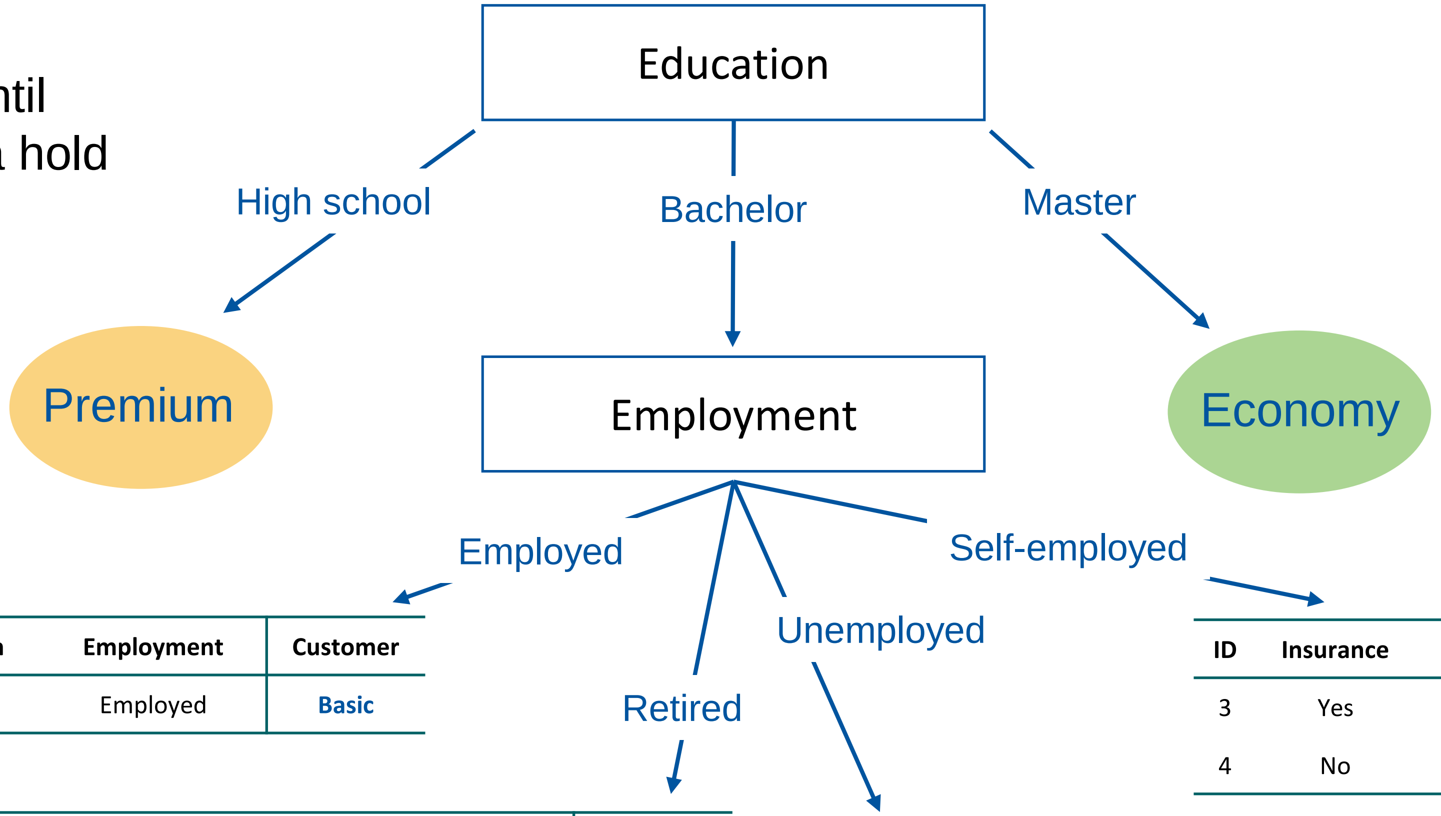
Recursion until stopping criteria hold



ID	Insurance	Education	Employment	Customer
1	Yes	Bachelor	Employed	**Basic**
3	Yes	Bachelor	Self-employed	**Premium**
4	No	Bachelor	Self-employed	**Basic**
6	Yes	Bachelor	Retired	**Economy**

Example

Recursion until stopping criteria hold



ID	Insurance	Education	Employment	Customer
1	Yes	Bachelor	Employed	**Basic**

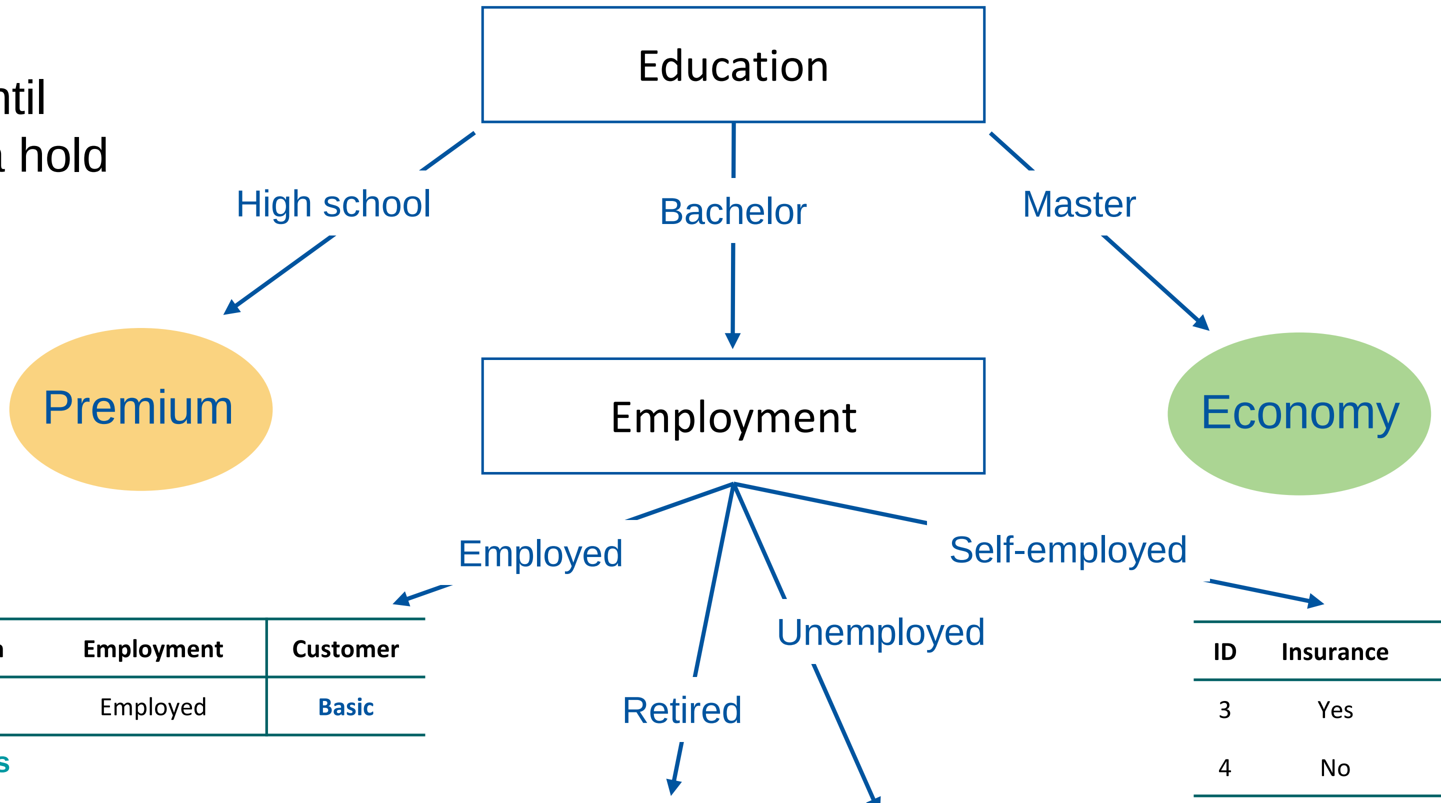
ID	Insurance	Education	Employment	Customer
6	Yes	Bachelor	Retired	**Economy**

ID	Insurance	Education	Employment	Customer
-	-	-	-	-

ID	Insurance	Education	Employment	Customer
3	Yes	Bachelor	Self-employed	**Premium**
4	No	Bachelor	Self-employed	**Basic**

Example

Recursion until stopping criteria hold



ID	Insurance	Education	Employment	Customer
1	Yes	Bachelor	Employed	**Basic**

consensus

ID	Insurance	Education	Employment	Customer
6	Yes	Bachelor	Retired	**Economy**

consensus

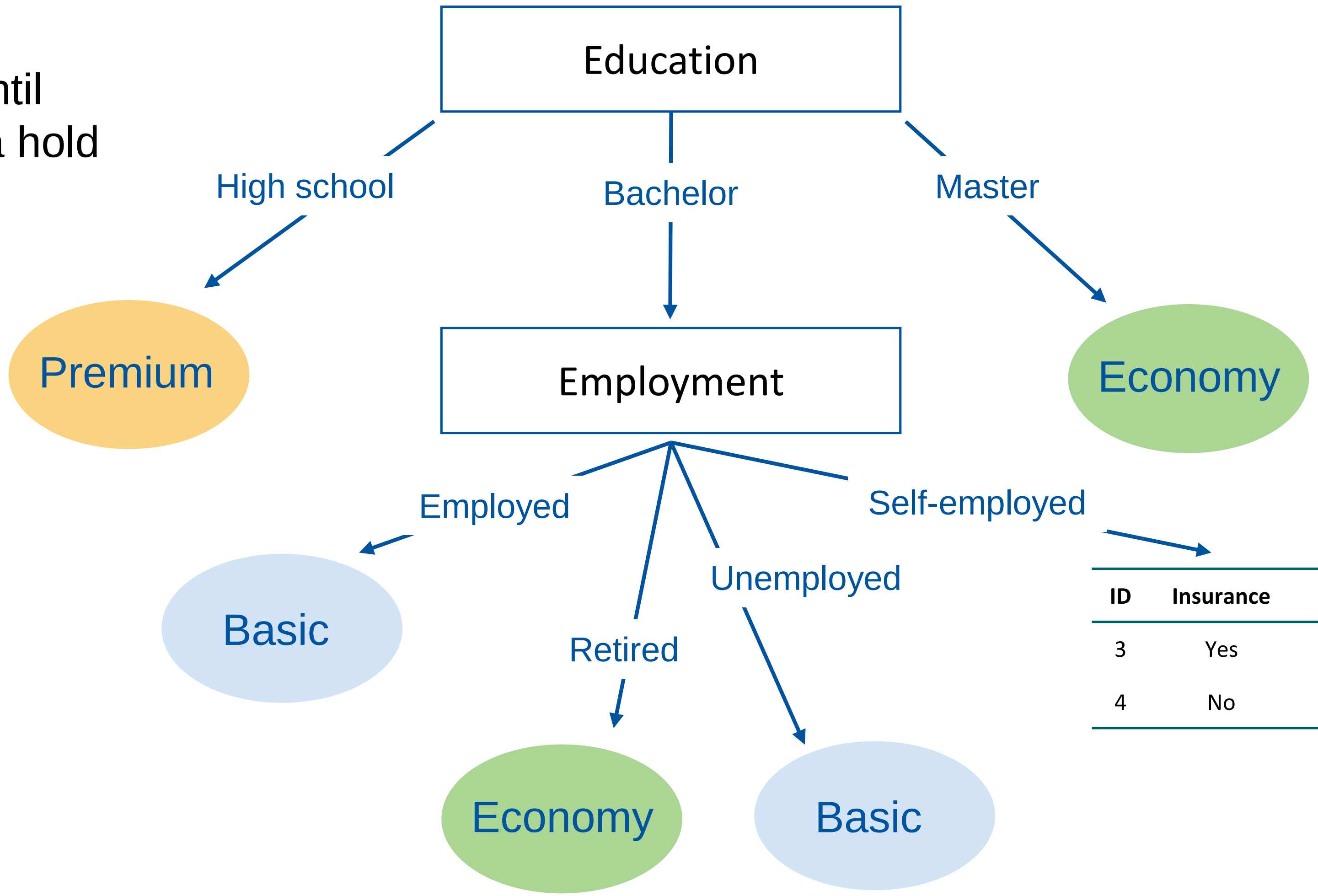
ID	Insurance	Education	Employment	Customer
-	-	-	-	-

empty (label parent majority = Basic)

ID	Insurance	Education	Employment	Customer
3	Yes	Bachelor	Self-employed	**Premium**
4	No	Bachelor	Self-employed	**Basic**

Example

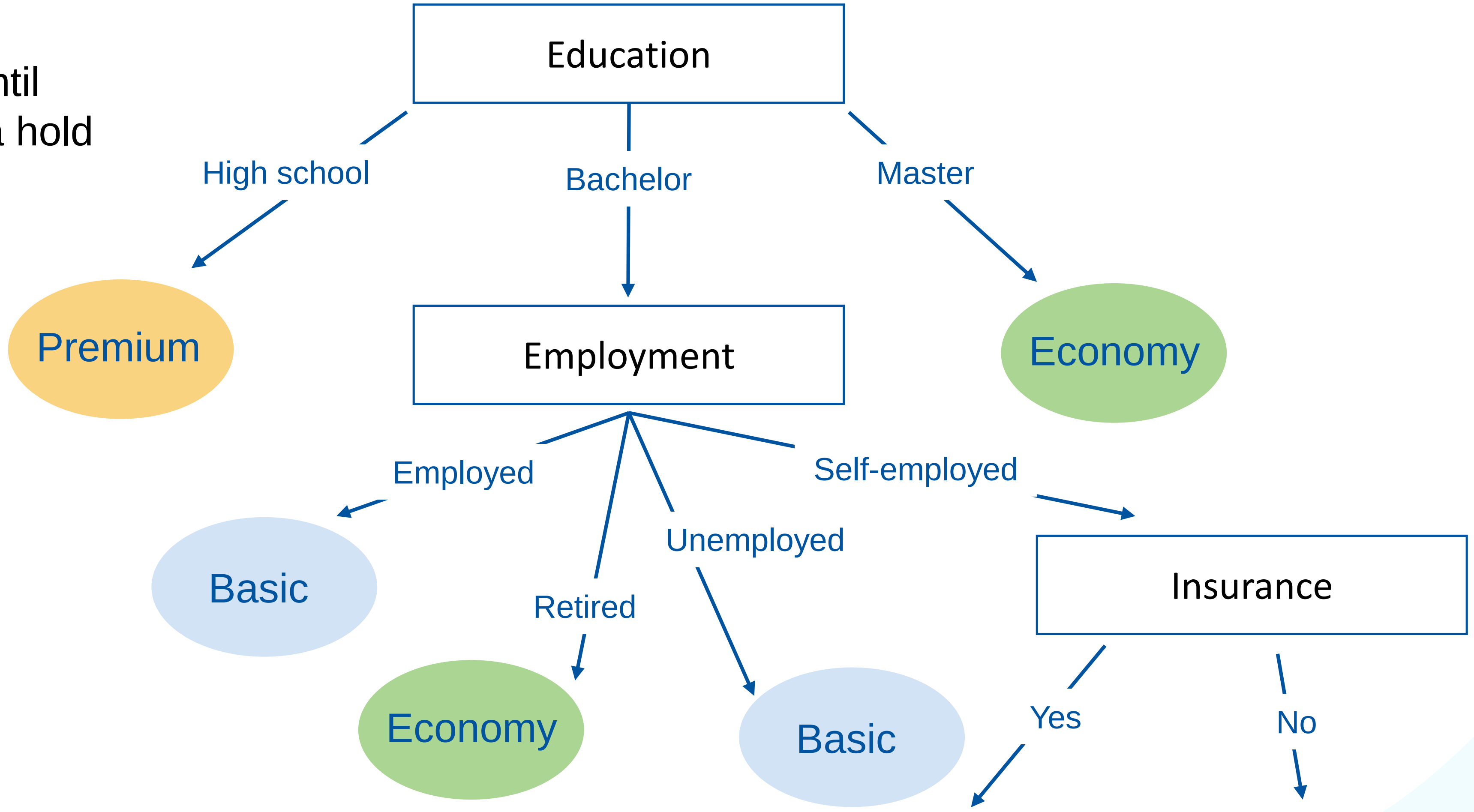
Recursion until stopping criteria hold



ID	Insurance	Education	Employment	Customer
3	Yes	Bachelor	Self-employed	Premium
4	No	Bachelor	Self-employed	Basic

Example

Recursion until stopping criteria hold



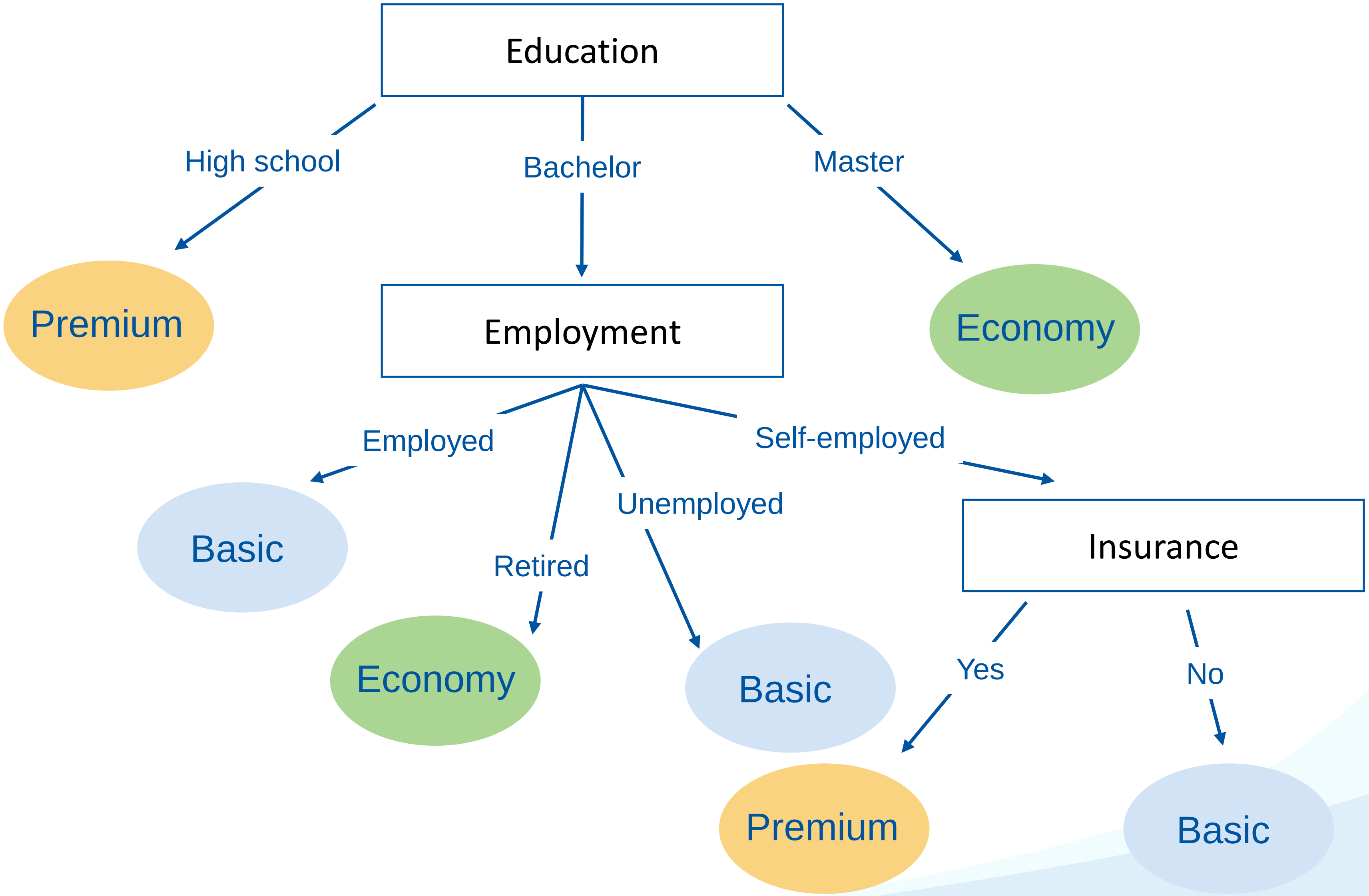
ID	Insurance	Education	Employment	Customer
3	Yes	Bachelor	Self-employed	Premium

consensus

ID	Insurance	Education	Employment	Customer
4	No	Bachelor	Self-employed	Basic

consensus

Example



(no feature exhaustion in this small example)

Alternative Information Gain Notions

- Information gain aims to measure the improvement in purity / predictability / compressibility

- Example approaches:
  - Entropy-based information gain (IG)
  - Information gain ratio (GR)
  - Gini index (Gini)

Alternative Information Gain Notions

- Information gain aims to measure the improvement in purity / predictability / compressibility

- Example approaches:
  - **Entropy-based information gain (IG)**
  - Information gain ratio (GR)
  - Gini index (Gini)

Entropy of target feature *t* before splitting

$$H(t) = -\sum_{k=1}^{K}(P(t=k) \cdot log_s(P(t=k)))$$

$$H_W(t) = \sum_{node \in nodes(d)}(\frac{|node|}{N} \cdot H_{node}(t))$$

Weighted entropy of target feature *t* after splitting based on *d*

$$IG(d) = H(t) - H_W^d(t)$$

(seen before)

Alternative Information Gain Notions

- Information gain aims to measure the improvement in purity / predictability / compressibility

- Example approaches:
  - Entropy-based information gain (IG)
  - **Information gain ratio (GR)**
  - **Gini index (Gini)**

Information Gain Ratio

- Entropy-based information gain favors features with many different values (split in many subsets decreases entropy)
- Information gain ratio addresses this issue:

Information gain when splitting based on descriptive feature *d*

Entropy of target feature *t*

Overall entropy of target feature *t* after splitting based on *d*

$$GR(d) = \frac{IG(d)}{H(d)} = \frac{H(t) - H_W^d(t)}{-\sum_{k=1}^{K}(P(d=k) \cdot log_2(P(d=k)))}$$

Entropy of descriptive feature *d*

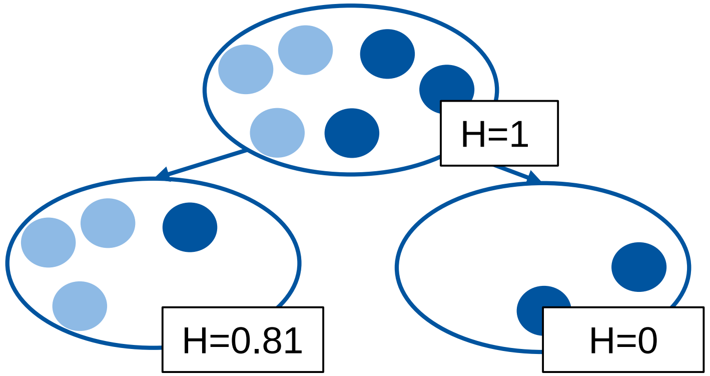
d can take *K* possible values

Probability of *d* taking the *k*th possible value

→ we can think of it as making an absolute value relative

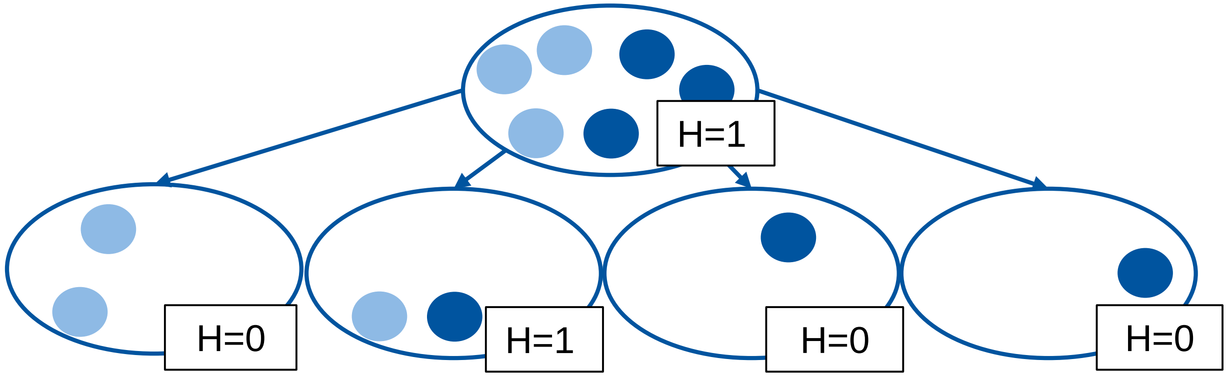
Information Gain Ratio - Example

split based on feature *d*



$H_W^d(color) = \frac{4}{6} \cdot 0.81 = 0.54$
$IG(d) = 0.46$

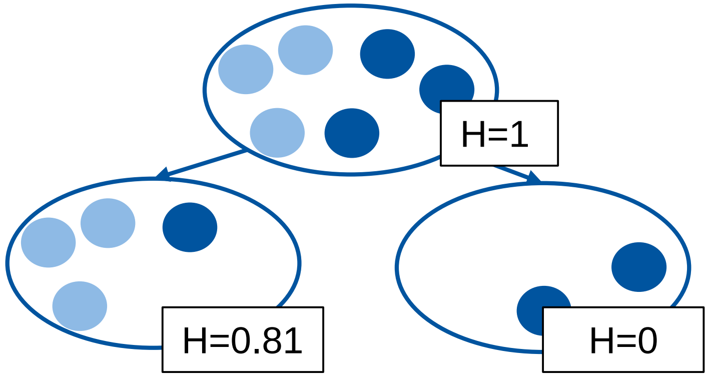
split based on feature *d'*



$H_W^{d'}(color) = \frac{2}{6} \cdot 1 = 0.33$
$IG(d') = 0.67$

Information Gain Ratio - Example

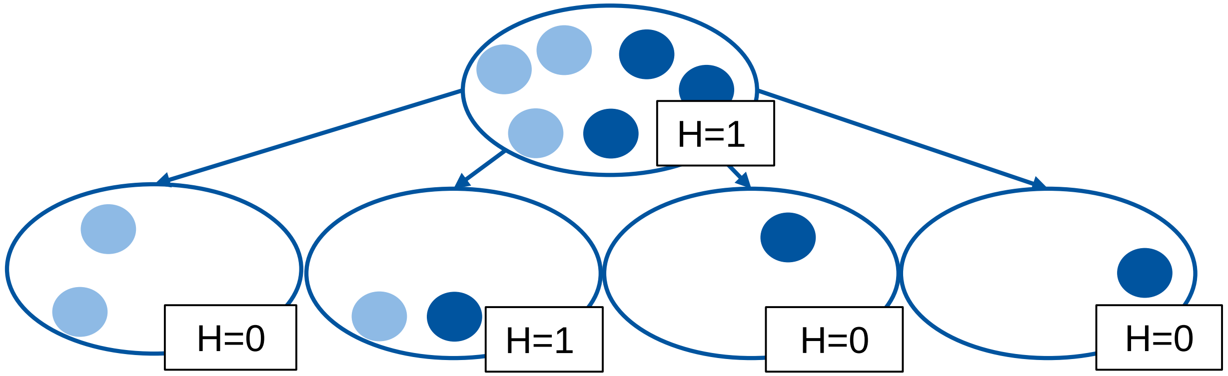
split based on feature *d*



$$IG(d) = 0.46$$

$$GR(d) = \frac{0.46}{-(\frac{4}{6} \cdot \log_2(\frac{4}{6}) + \frac{2}{6} \cdot \log_2(\frac{2}{6}))} = \frac{0.46}{0.92} = 0.5$$

split based on feature *d'*



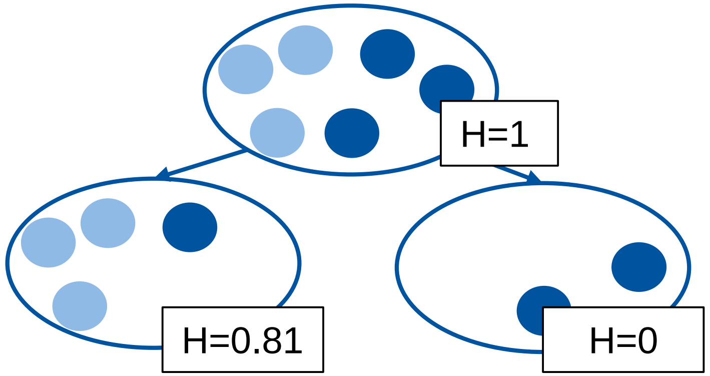
$$IG(d') = 0.67$$

$$GR(d') = \frac{0.67}{-(\frac{2}{6} \cdot \log_2(\frac{2}{6}) + \frac{2}{6} \cdot \log_2(\frac{2}{6}) + \frac{1}{6} \cdot \log_2(\frac{1}{6}) + \frac{1}{6} \cdot \log_2(\frac{1}{6}))} = \frac{0.67}{1.92} = 0.35$$

$$GR(d) = \frac{IG(d)}{H(d)} = \frac{H(t) - H_W^d(t)}{-\sum_{k=1}^{K}(P(d=k) \cdot log_2(P(d=k)))}$$

Information Gain Ratio - Example

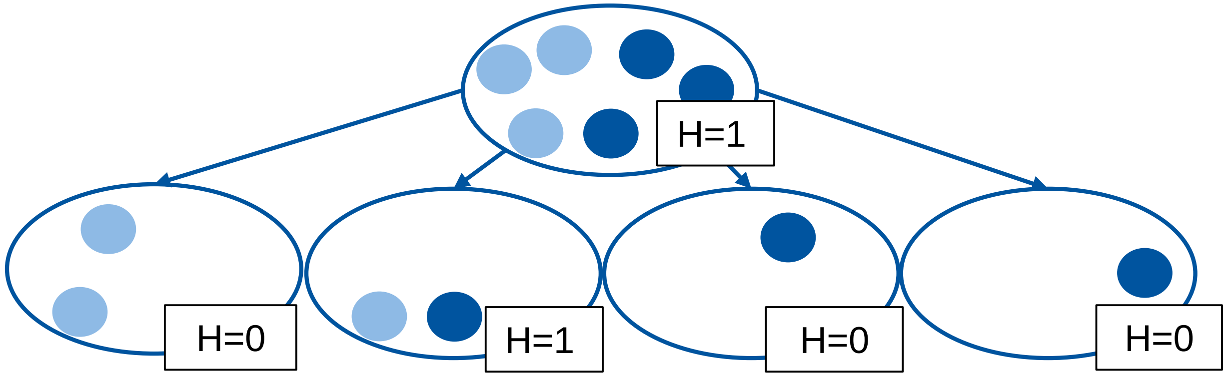
split based on feature *d*



$$IG(d) = 0.46$$

$$GR(d) = \frac{0.46}{-(\frac{4}{6}\cdot\log_2(\frac{4}{6}) + \frac{2}{6}\cdot\log_2(\frac{2}{6}))} = \frac{0.46}{0.92} = 0.5$$

split based on feature *d'*



$$IG(d') = 0.67$$

$$GR(d') = \frac{0.67}{-(\frac{2}{6}\cdot\log_2(\frac{2}{6}) + \frac{2}{6}\cdot\log_2(\frac{2}{6}) + \frac{1}{6}\cdot\log_2(\frac{1}{6}) + \frac{1}{6}\cdot\log_2(\frac{1}{6}))} = \frac{0.67}{1.92} = 0.35$$

$$GR(d) = \frac{IG(d)}{H(d)} = \frac{H(t) - H_W^d(t)}{-\sum_{k=1}^{K}(P(d=k)\cdot log_2(P(d=k)))}$$

Gini Index

- An alternative measure of impurity
- Expected misclassification rate when guessing based on the observed distribution

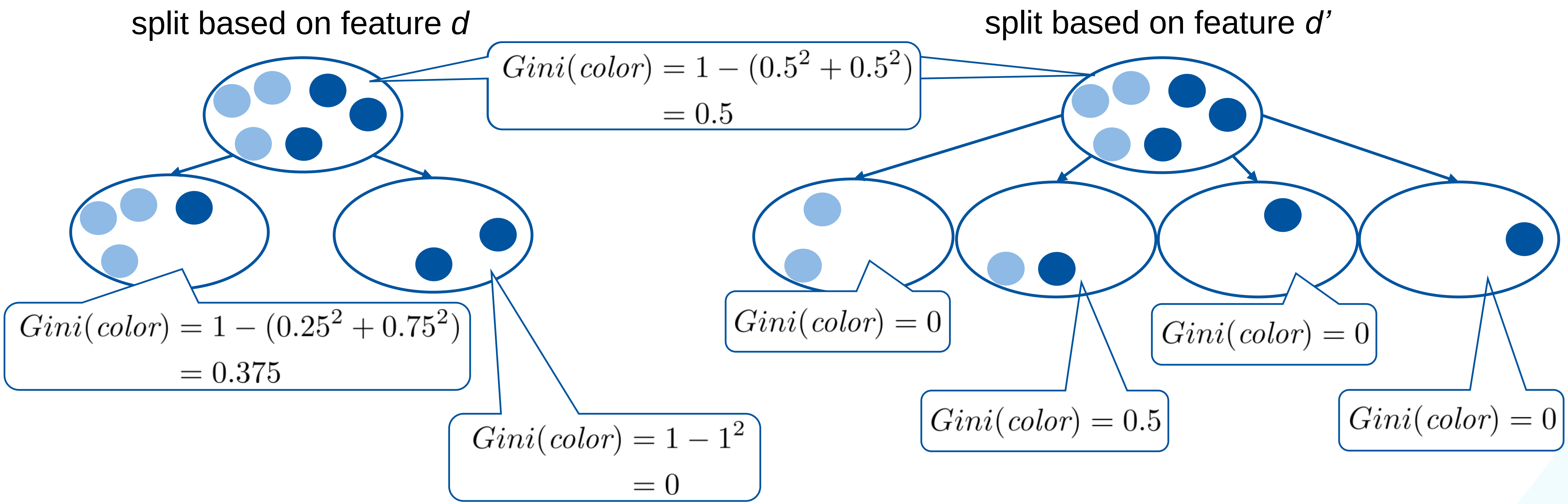
$$Gini(t) = 1 - \sum_{k=1}^{K} P(t = k)^2$$

t can take *K* possible values

Probability that *t* takes the *k*th value

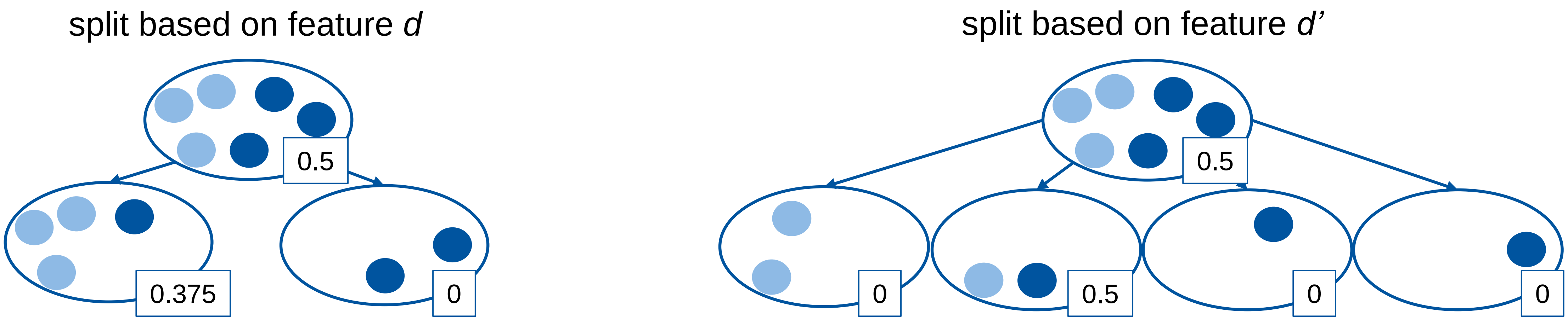
- With probability $P(t = k)$ we guess that *class* equals the *k*th possible value and with probability $P(t = k)$ this guess is correct
- Can be seen as the probability of guessing the wrong label

Gini Index - Example



$$Gini(t) = 1 - \sum_{k=1}^{K} P(t = k)^2$$

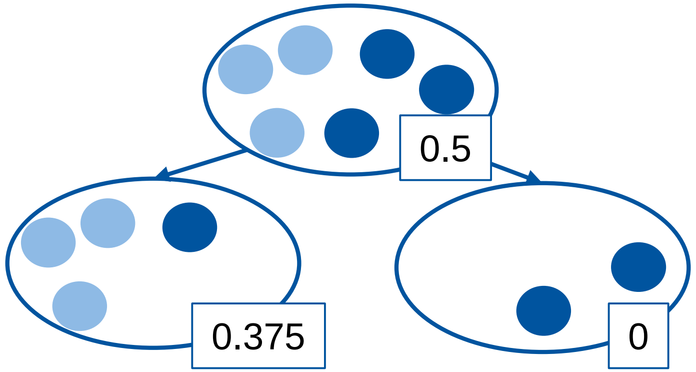
Gini Index - Example



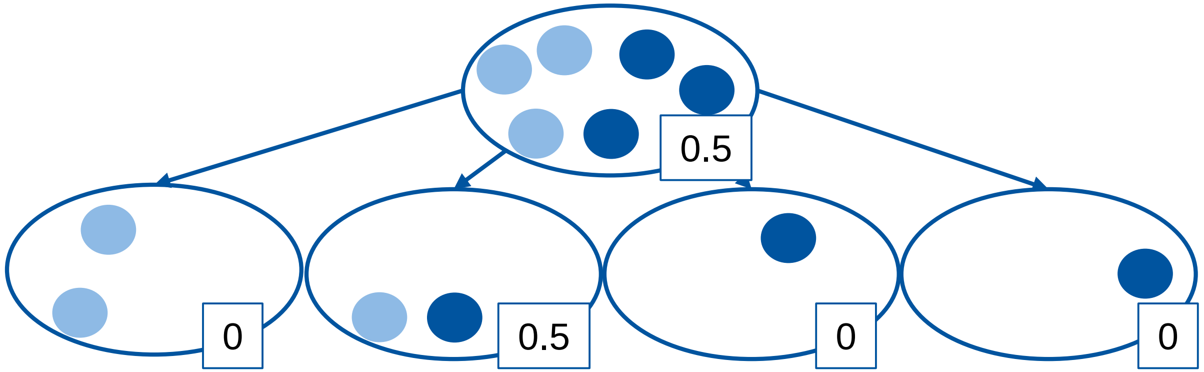
$$Gini(t) = 1 - \sum_{k=1}^{K} P(t = k)^2$$

Gini Index - Example

split based on feature *d*



split based on feature *d'*



$$Gini_W(color) = \frac{4}{6} \cdot 0.375 = 0.25$$
$$IG_{Gini}(d) = 0.5 - 0.25 = 0.25$$

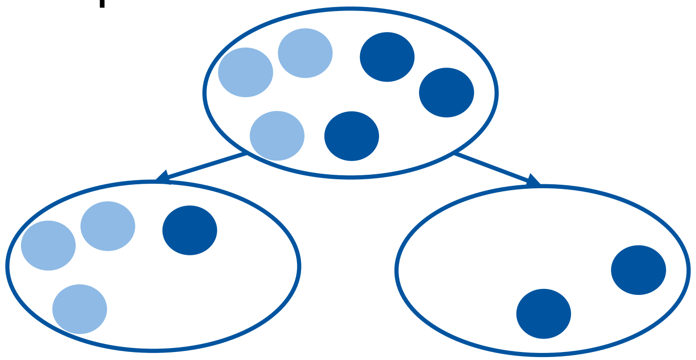
$$Gini_W(color) = \frac{2}{6} \cdot 0.5 = 0.166$$
$$IG_{Gini}(d') = 0.5 - 0.166 = 0.33$$

compute weighted average and information gain as before

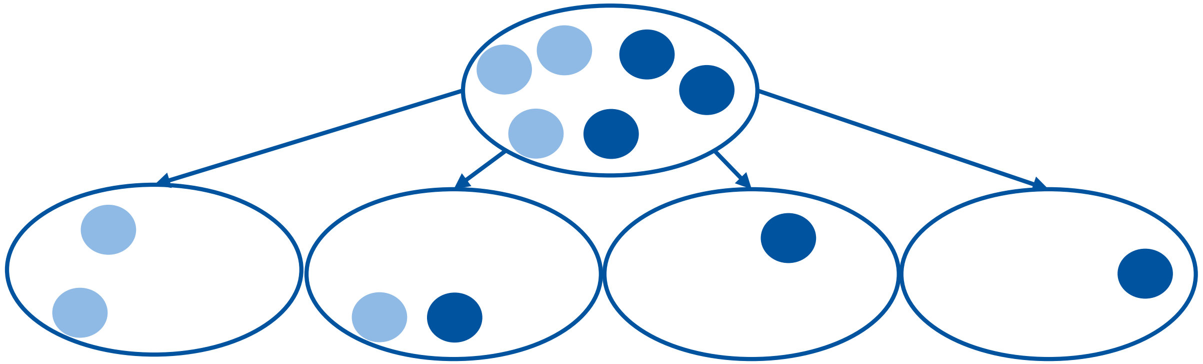
$$Gini(t) = 1 - \sum_{k=1}^{K} P(t = k)^2$$

Comparison

split based on feature *d*



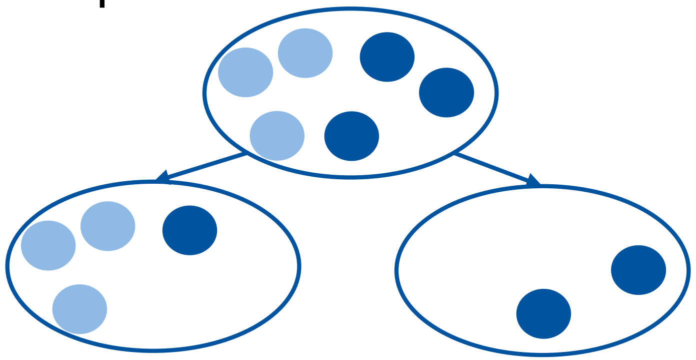
split based on feature *d'*



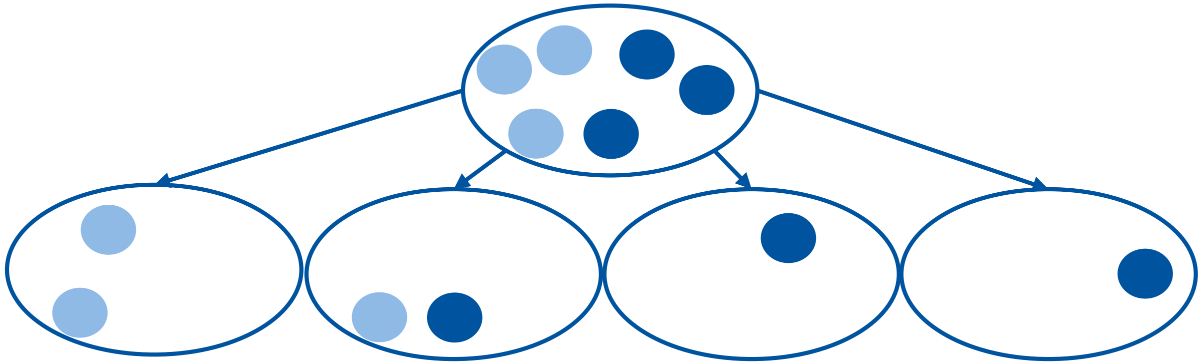
$IG_{Entropy}(d) = 0.4591$	Entropy-based information gain	$IG_{Entropy}(d') = 0.6667$
$GR(d) = 0.5$	Information gain ratio	$GR(d') = 0.34$
$IG_{Gini}(d) = 0.25$	Gini-based information gain ratio	$IG_{Gini}(d') = 0.33$

Comparison

split based on feature *d*



split based on feature *d'*



split based on feature *d*		split based on feature *d'*
$IG_{Entropy}(d) = 0.4591$	Entropy-based information gain	$IG_{Entropy}(d') = 0.6667$ 👍
$GR(d) = 0.5$ 👍	Information gain ratio	$GR(d') = 0.34$
$IG_{Gini}(d) = 0.25$	Gini-based information gain ratio	$IG_{Gini}(d') = 0.33$ 👍

Pruning Decision Trees

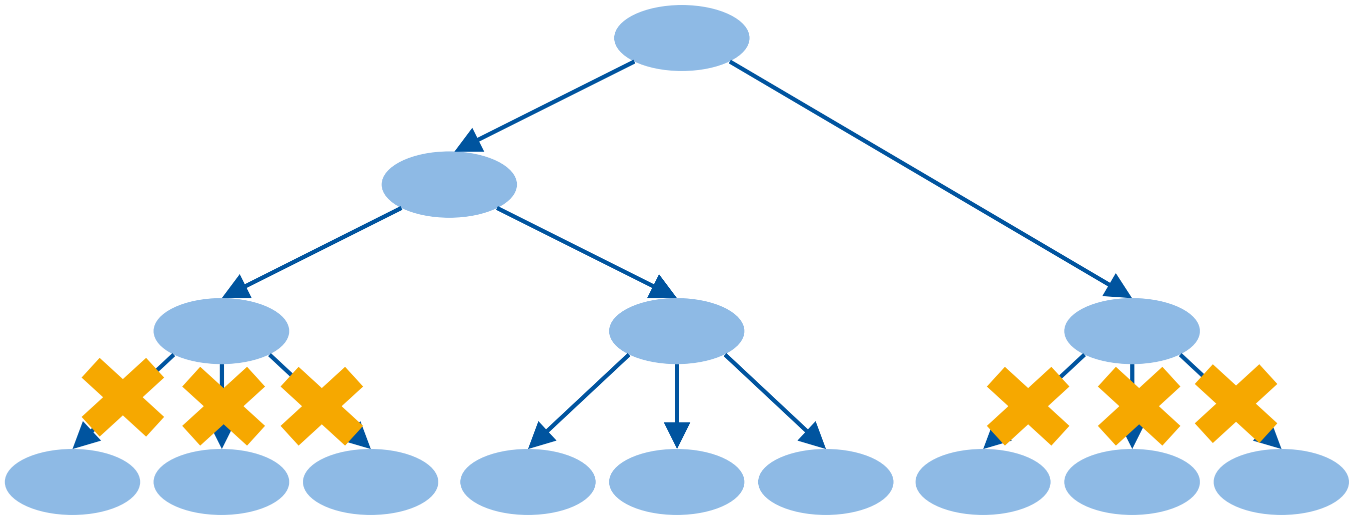
- Possible problems:
  - Decision tree is overfitting the data
  - Decision tree is too complex or too deep
- Two solution directions:
  - Pre-pruning (early stopping/forward)
  - Post-pruning (reduced error/backward)
- To generalize and avoid overfitting

Pre-pruning

- Stop creating subtrees and use majority vote to determine the label
- Many possible stopping criteria:
  - lower bound for number of instances
  - lower bound for information gain
  - …
- May create trees that are not consistent with respect to the data

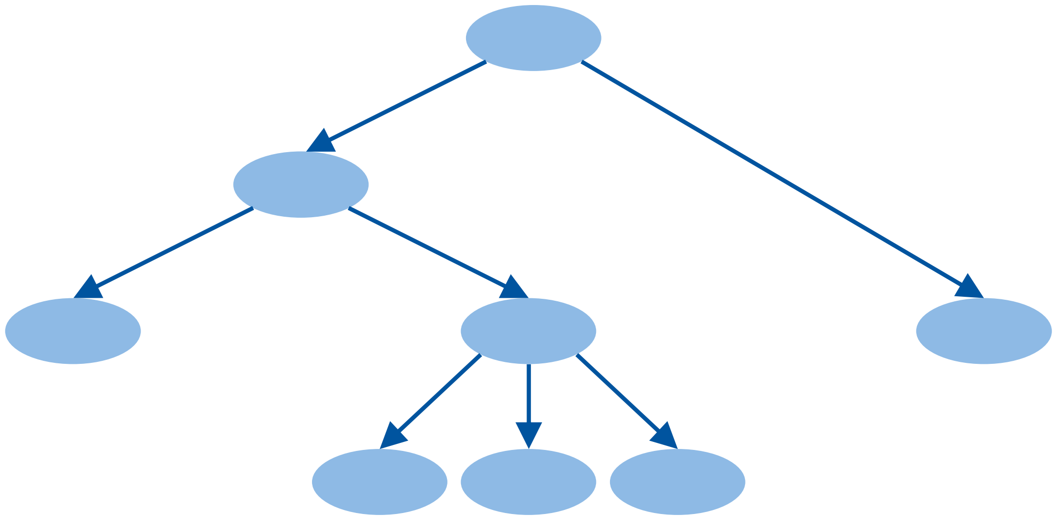
Pre-pruning

Example:
Information Gain = 0.0001
→ do not to split further



Example:
Only 10 instances left
→ do not to split further

Pre-pruning

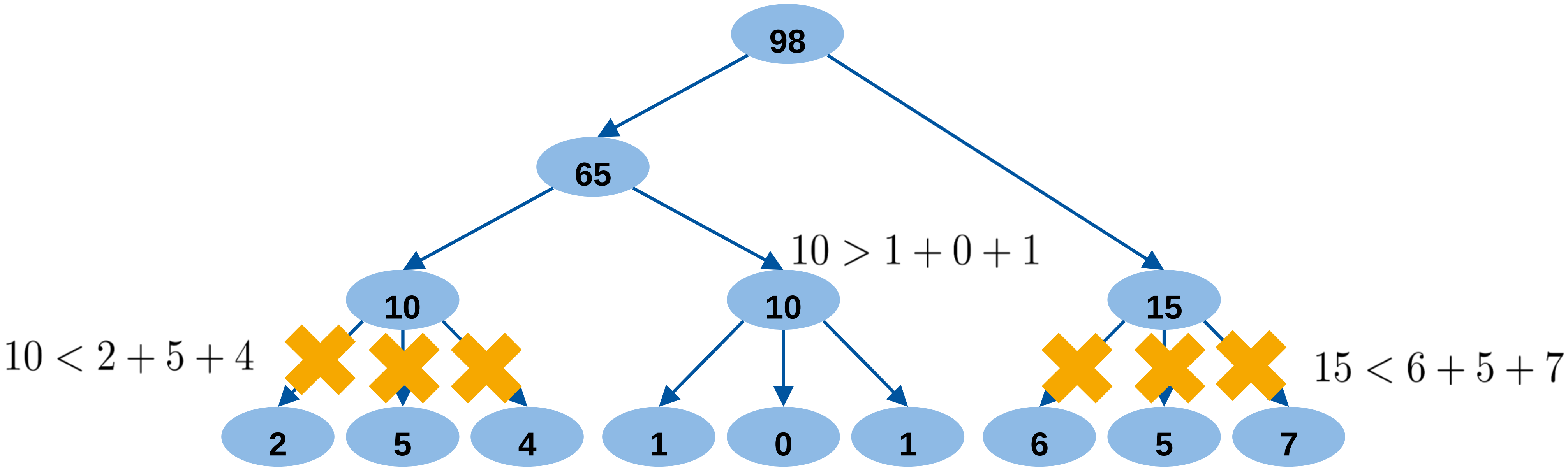


efficient, but we may miss strong dependencies at lower levels of trees

Post-pruning

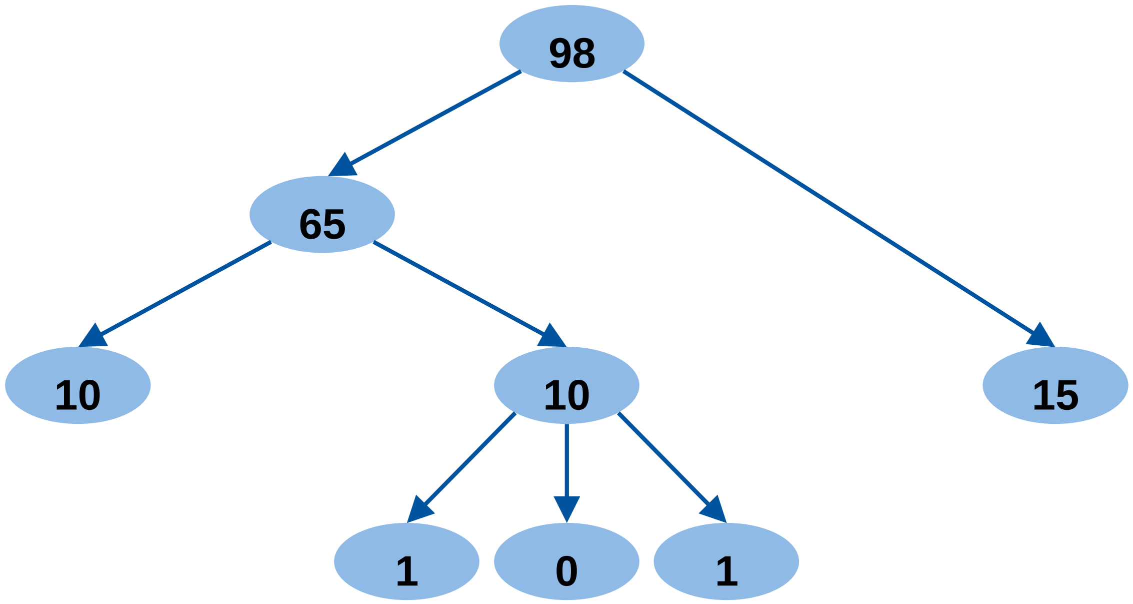
- First, build the whole decision tree; then cut off branches that do not add much
- Common approach is to split the data into a training set and a validation/test set
- Measure the performance of splits based on a validation/test set

Post-pruning



- Decision tree learned on a training set
- Numbers indicate misclassifications based on a validation set

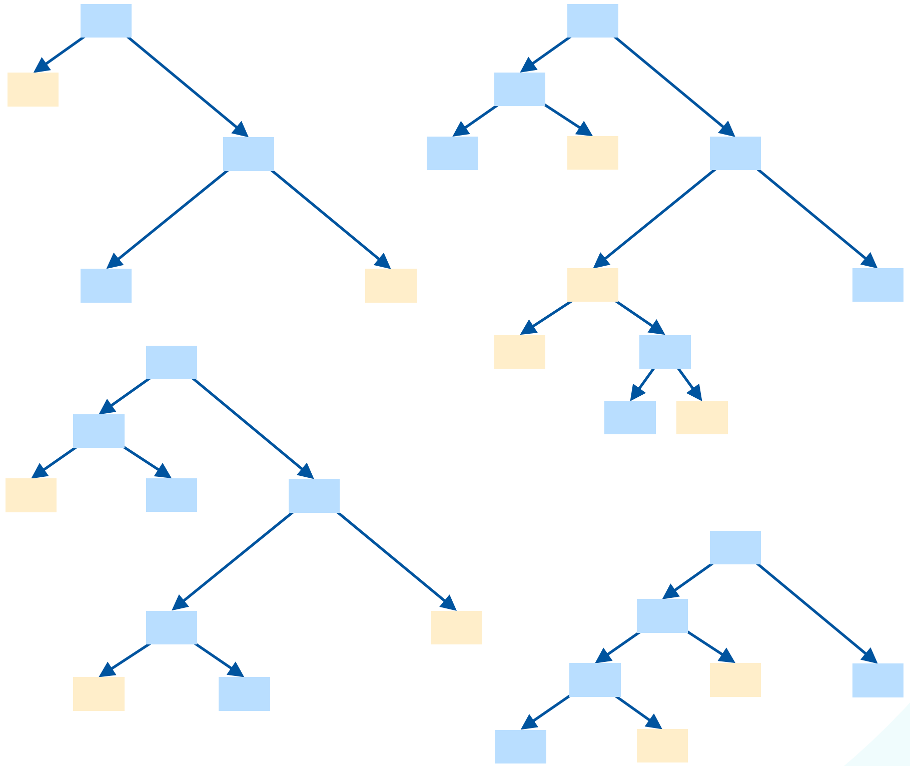
Post-pruning



Less efficient, but based on the complete tree

Idea

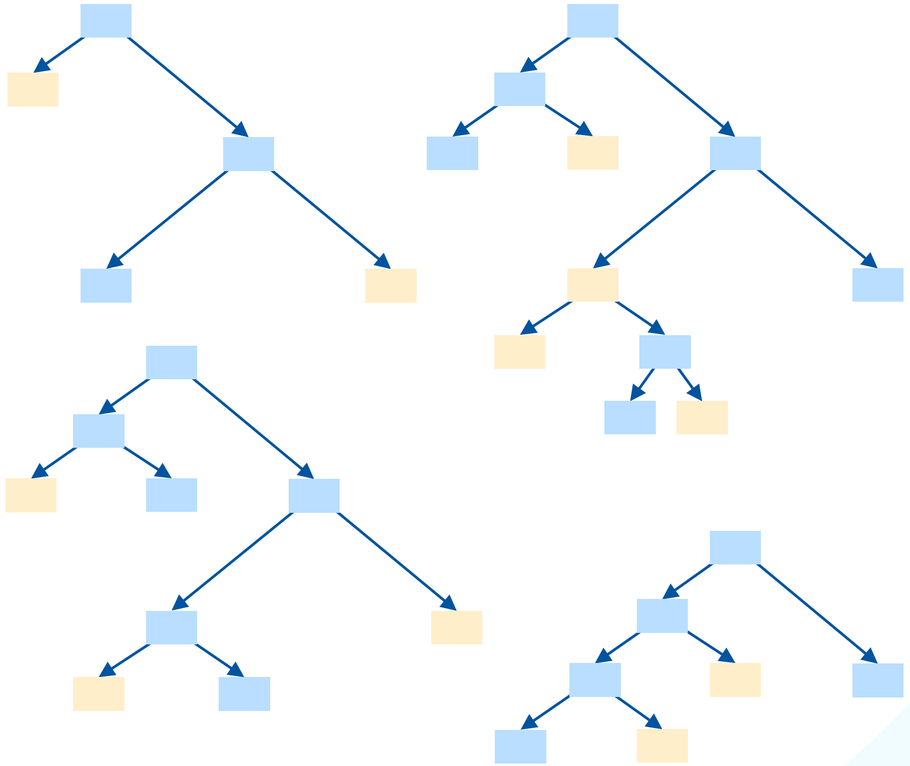
- Rather than creating a single decision tree, we aim to create a set of trees (called a model ensemble)
- Models should complement each other
- Different models can ”vote” on the label (votes may be weighted)
- Multiple trees may give different answers (select the most frequent value or the average)
- Many variations of the same idea…



Boosting

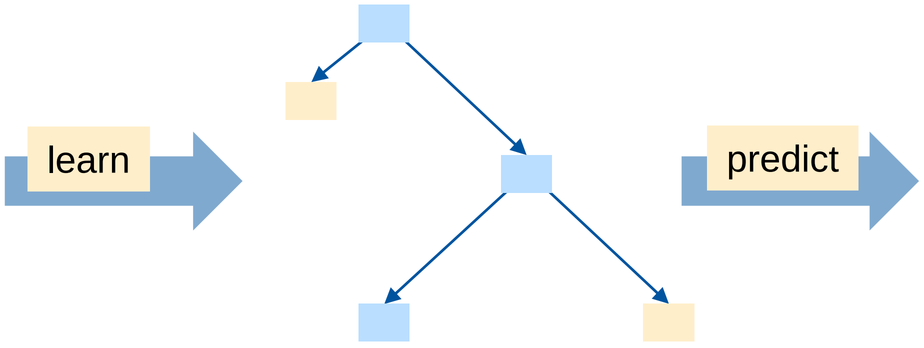
Correct iteratively

- Iteratively update the data set based on misclassifications
- Instances that are wrongly classified get a higher weight when learning the next model
- Each iteration new models are added to the ensemble



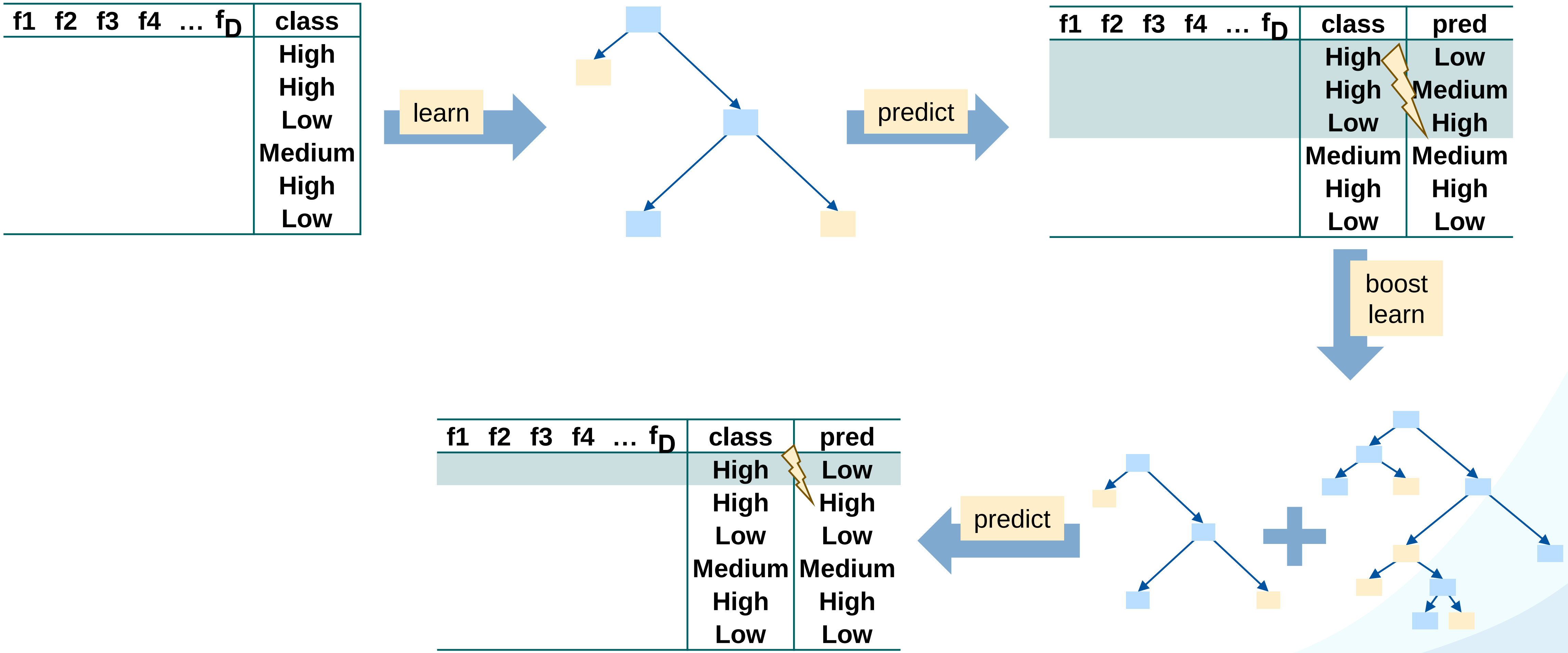
Boosting - Example

f1	f2	f3	f4	…	f_D	class
						High
						High
						Low
						Medium
						High
						Low

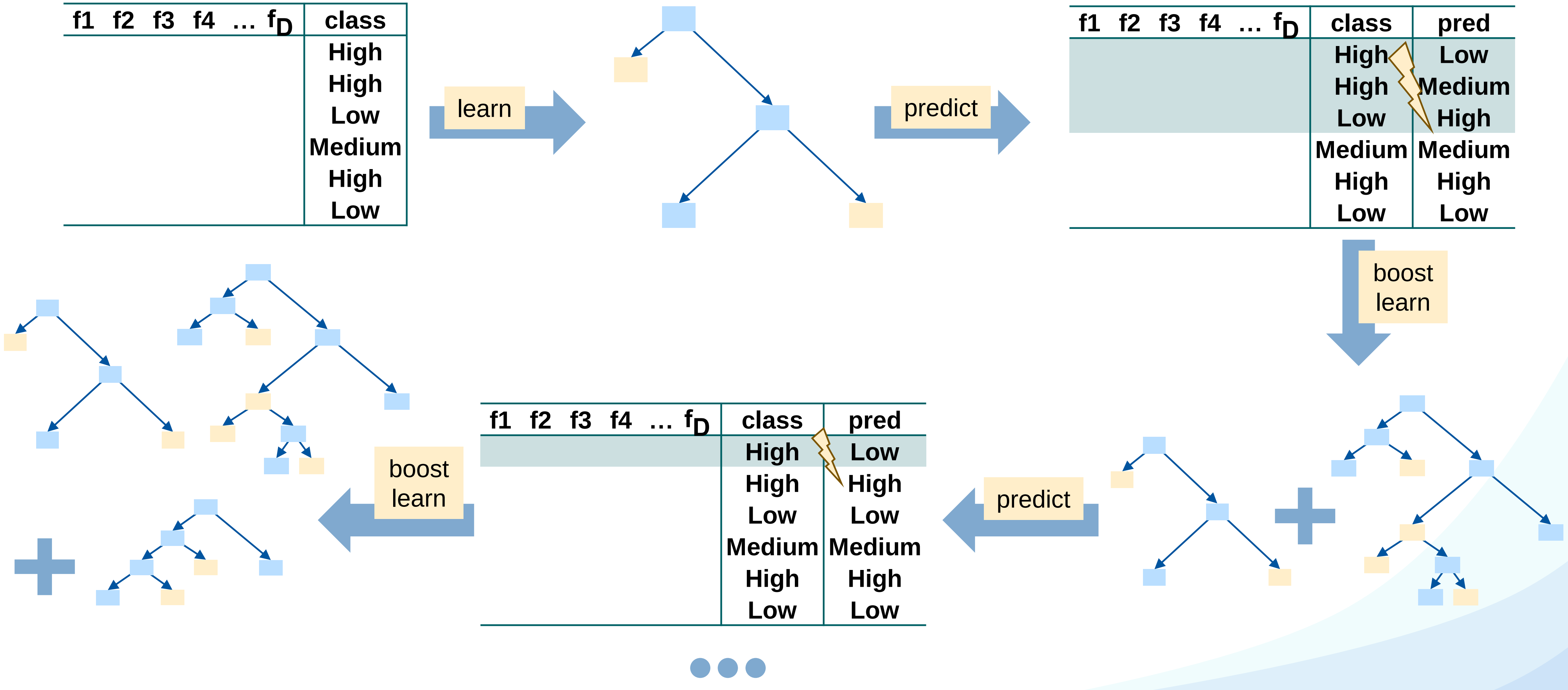


f1	f2	f3	f4	…	f_D	class	pred
						High	Low
						High	Medium
						Low	High
						Medium	Medium
						High	High
						Low	Low

Boosting - Example



Boosting - Example



f1 f2 f3 f4 … f_D	class
	High
	High
	Low
	Medium
	High
	Low

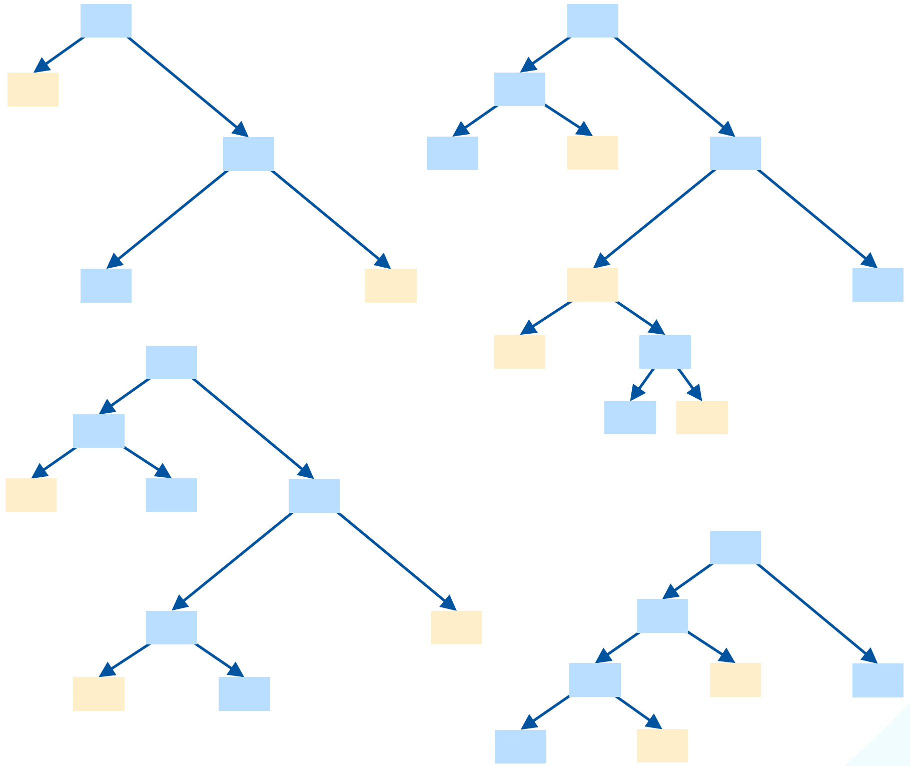
f1 f2 f3 f4 … f_D	class	pred
	High	Low
	High	Medium
	Low	High
	Medium	Medium
	High	High
	Low	Low

f1 f2 f3 f4 … f_D	class	pred
	High	Low
	High	High
	Low	Low
	Medium	Medium
	High	High
	Low	Low

Bagging

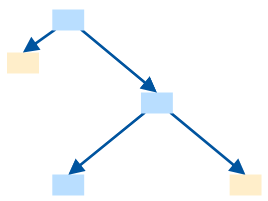
Split data upfront

- Each model is based on a random sample of the data set
- Avoids model depending on a specific sample of the data set (learning decision trees may be very sensitive to small variations)
- Many variants (e.g. remove some instances and duplicate others)

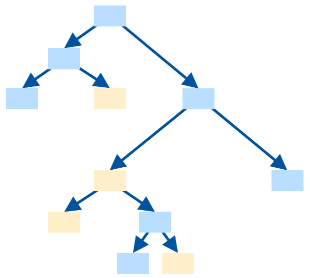


Bagging - Example

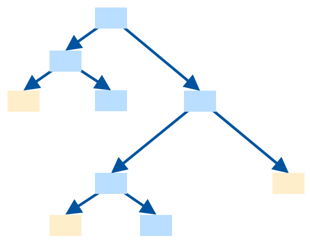
f1	f2	f3	f4	…	f_D	class
						High
		X				High
		X				Low
						Medium
						High
						Low



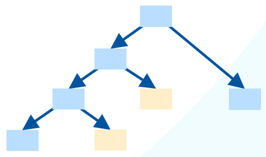
f1	f2	f3	f4	…	f_D	class
		X				High
						High
		X				Low
						Medium
						High
						Low



f1	f2	f3	f4	…	f_D	class
						High
						High
						Low
		X				Medium
		X				High
						Low

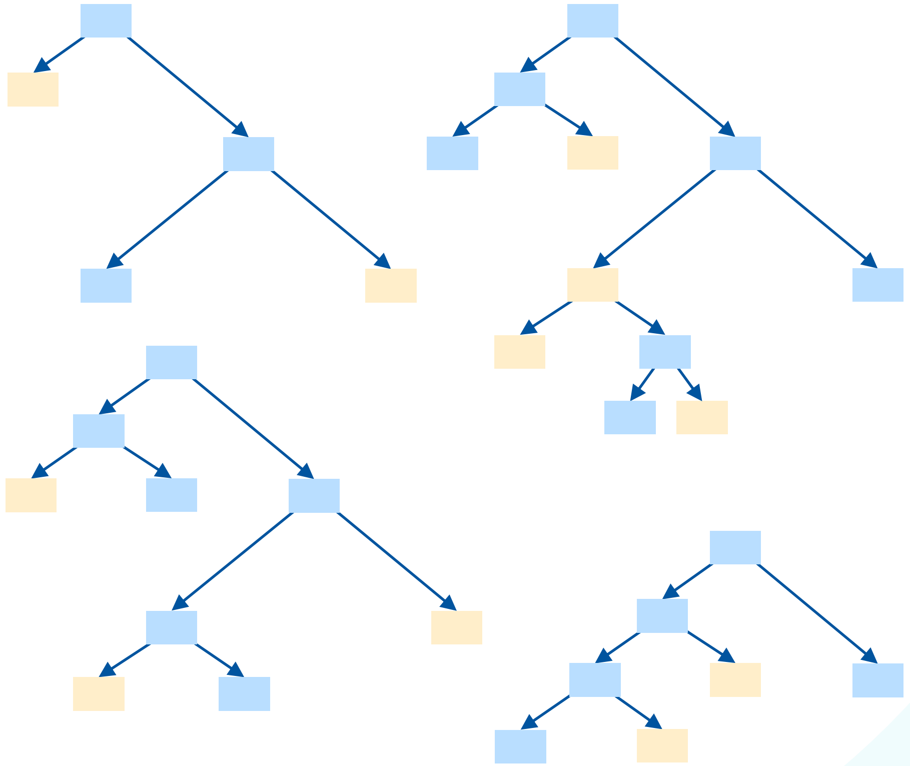


f1	f2	f3	f4	…	f_D	class
						High
		X				High
						Low
						Medium
						High
		X				Low

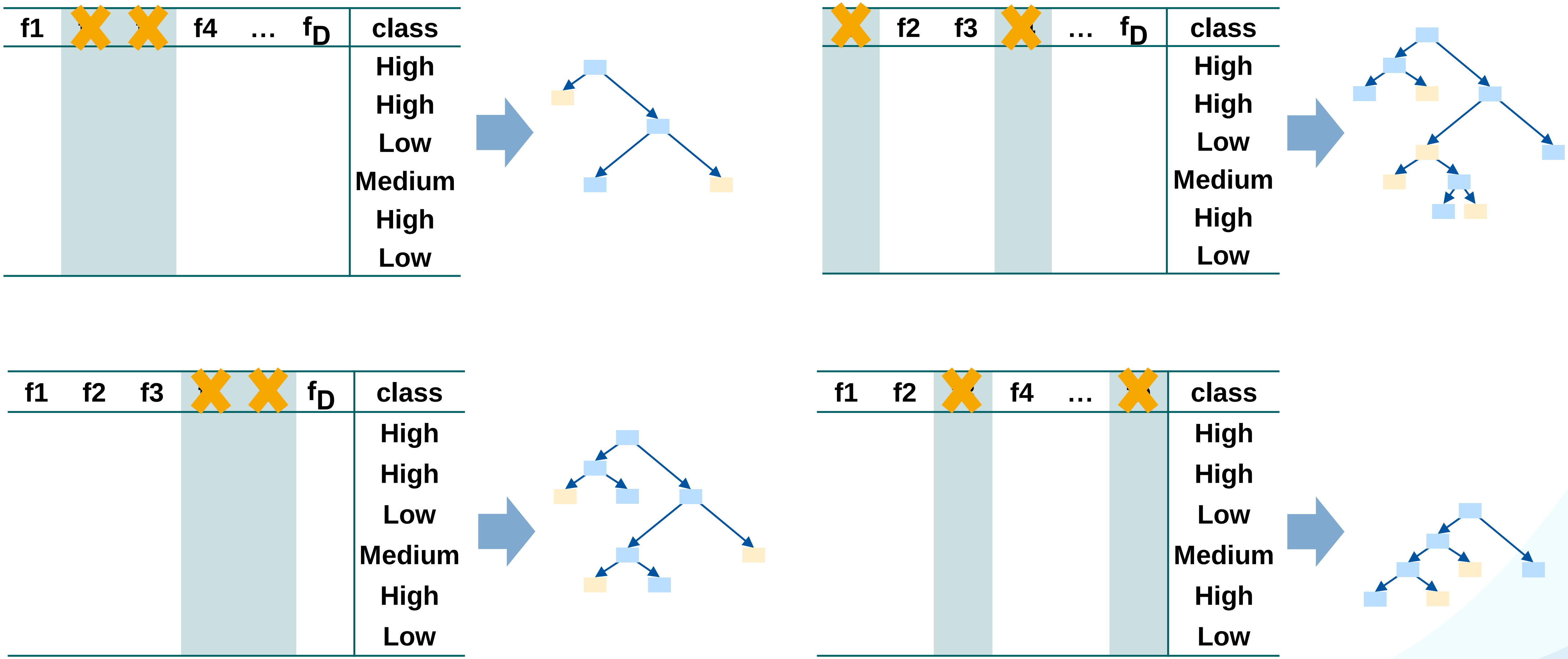


Subspace Sampling

- Each model is based on a random set of descriptive features
- More efficient and less likely to be overfitting when focusing on just a few features



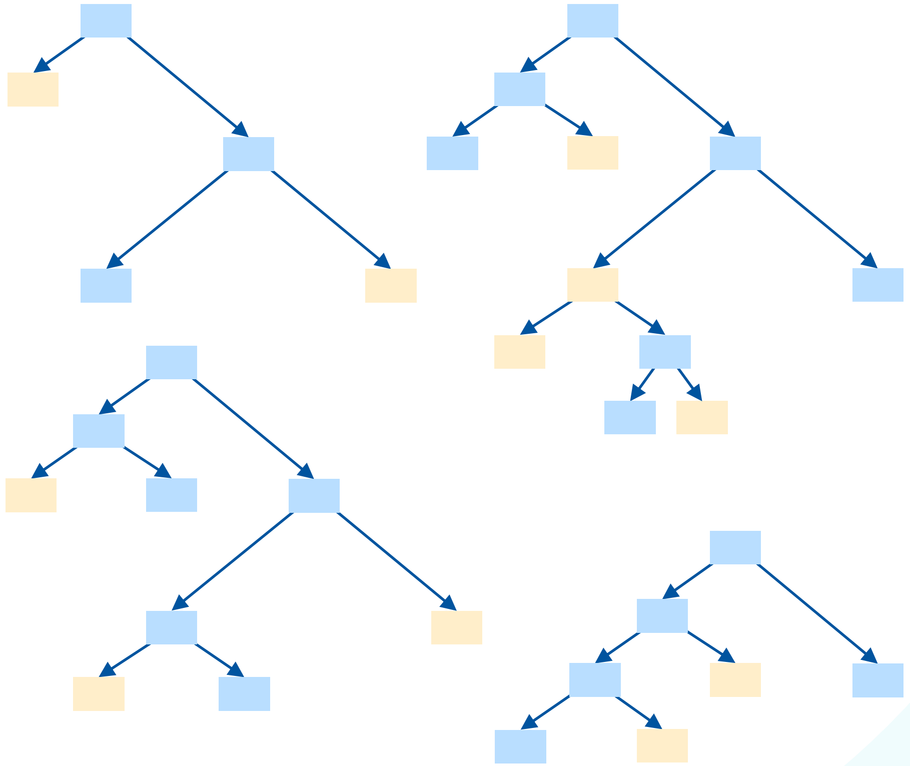
Subspace sampling



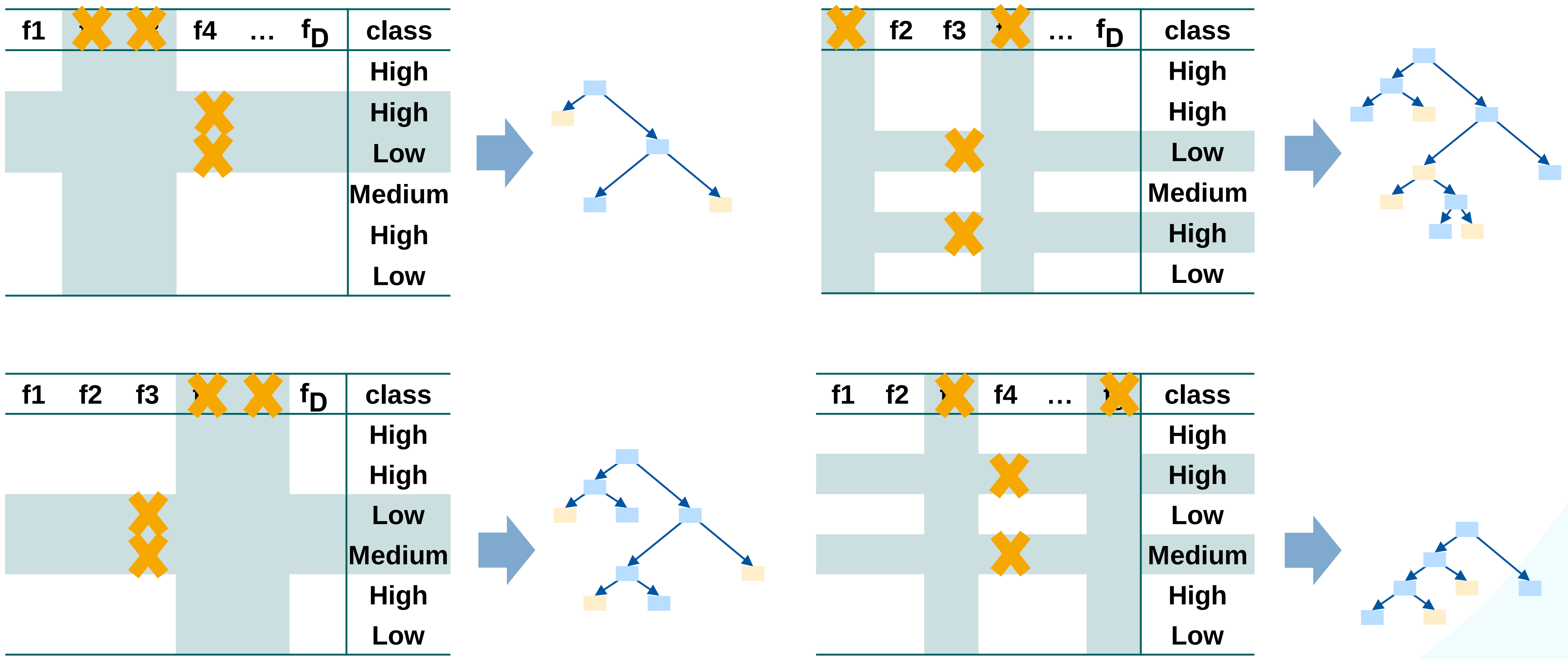
Random Forest

Combine of bagging and subspace sampling

- Split data twice
  - random sample of instances (bagging)
  - random set of descriptive features (subspace sampling)
- Find a model for each subset of data created this way



Random Forest



Dealing with Continuous Variables

- Thus far we assumed features were categorical
- We can use binning to make continuous features categorical

continuous target feature

features

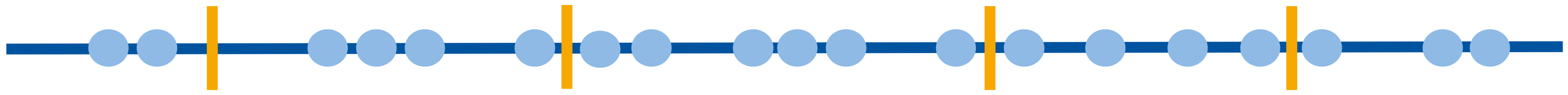
f_1	f_2	...	f_D	class
high	88		59.99	**5043**
high	76		50.00	**4598**
low	32		39.50	**3248**
low	89		49.99	**5466**
high	21		59.99	**7682**

instances

continuous descriptive features

Continuous Descriptive Features

- Challenge: determine suitable boundaries (infinite number of thresholds is possible)
- Idea:
  - sort instances based on the continuous descriptive feature
  - look for changes in target feature labels
- Change points are candidate thresholds
- Select the threshold with the highest information gain



Continuous Descriptive Features - Example

ID	Insurance	Income	Employment	Customer
1	Yes	3500	Employed	Basic
2	Yes	0	Unemployed	Premium
3	Yes	1000	Self-employed	Premium
4	No	2000	Self-employed	Basic
5	No	5000	Employed	Economy
6	Yes	5100	Retired	Economy
7	Yes	3000	Employed	Premium

sort

Continuous Descriptive Features - Example

ID	Insurance	Income	Employment	Customer
2	Yes	0	Unemployed	Premium
3	Yes	1000	Self-employed	Premium
4	No	2000	Self-employed	Basic
7	Yes	3000	Employed	Premium
1	Yes	3500	Employed	Basic
5	No	5000	Employed	Economy
6	Yes	5100	Retired	Economy

sort ↑

Continuous Descriptive Features - Example

ID	Insurance	Income	Employment	Customer
2	Yes	0	Unemployed	Premium
3	Yes	1000	Self-employed	Premium
4	No	2000	Self-employed	Basic
7	Yes	3000	Employed	Premium
1	Yes	3500	Employed	Basic
5	No	5000	Employed	Economy
6	Yes	5100	Retired	Economy

Thresholds: middle values of continuous feature in between changed target features

ID	Insurance	Income	Employment	Customer
2	Yes	0	Unemployed	Premium
3	Yes	1000	Self-employed	Premium
4	No	2000	Self-employed	Basic
7	Yes	3000	Employed	Premium
1	Yes	3500	Employed	Basic
5	No	5000	Employed	Economy
6	Yes	5100	Retired	Economy

Continuous Descriptive Features - Example

ID	Insurance	Income	Employment	Customer
2	Yes	0	Unemployed	Premium
3	Yes	1000	Self-employed	Premium
	1500			
4	No	2000	Self-employed	Basic
	2500			
7	Yes	3000	Employed	Premium
	3250			
1	Yes	3500	Employed	Basic
	4250			
5	No	5000	Employed	Economy
6	Yes	5100	Retired	Economy

Four candidate thresholds

Thresholds: middle values of continuous feature in between changed target features

Continuous Descriptive Features - Example

Threshold	Instances	Partition Entropy	Overall Entropy	Information Gain
≥1500	2, 3 1, 4, 5, 6, 7	0 1.5219	1.0871	0.1981
≥2500	2, 3, 4 1, 5, 6, 7	0.9183 1.5	1.2507	0.306
≥3250	2, 3, 4, 7 1, 5, 6	0.8113 0.9183	0.8572	0.6995
≥4250	1, 2, 3, 4, 7 5, 6	0.9710 0	0.6935	0.8631

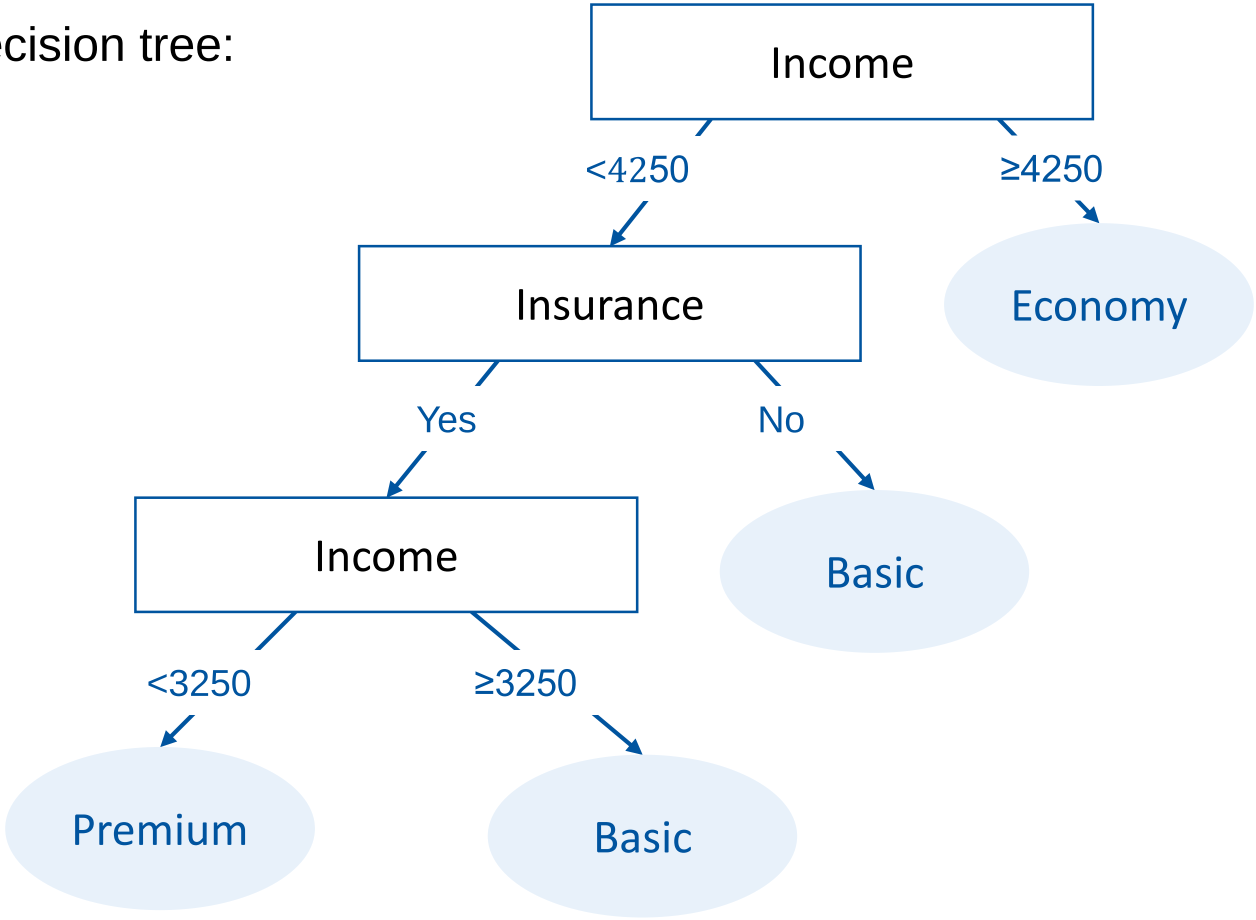
Compute as usual

Continuous Descriptive Features - Example

Threshold	Instances	Partition Entropy	Overall Entropy	Information Gain
≥1500	2, 3 1, 4, 5, 6, 7	0 1.5219	1.0871	0.1981
≥2500	2, 3, 4 1, 5, 6, 7	0.9183 1.5	1.2507	0.306
≥3250	2, 3, 4, 7 1, 5, 6	0.8113 0.9183	0.8572	0.6995
≥4250	1, 2, 3, 4, 7 5, 6	0.9710 0	0.6935	0.8631 (best)

Continuous Descriptive Features - Example

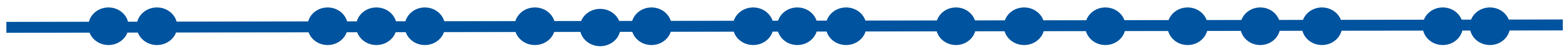
Resulting decision tree:



The same feature can now be used twice along a path!

Continuous Target Features

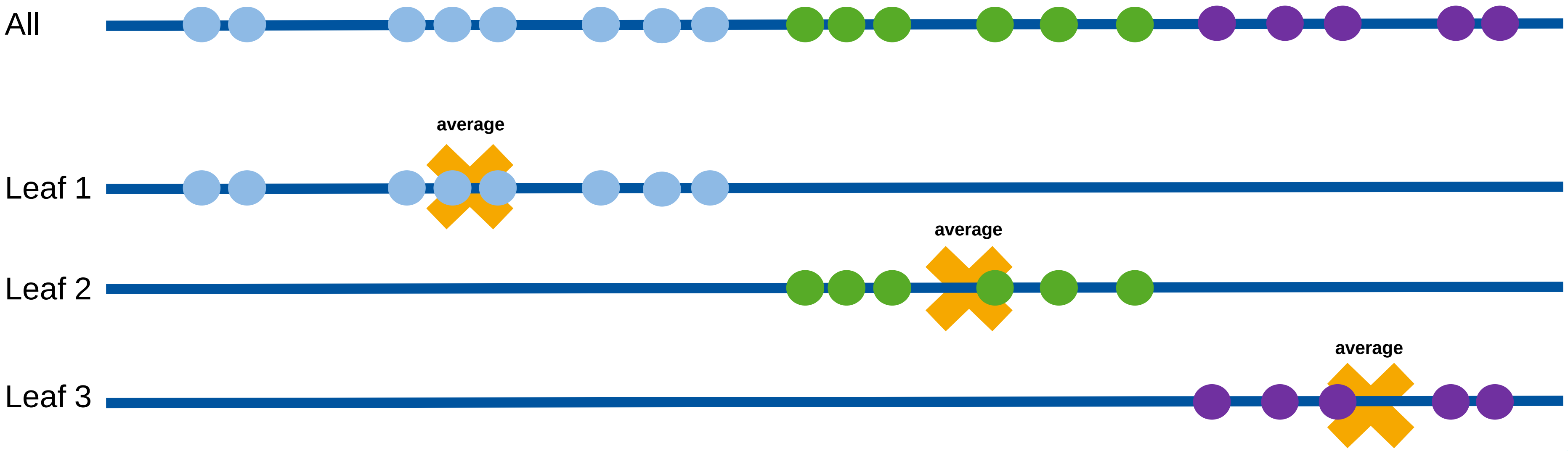
- Goal: find descriptive features that ‘nicely’ partition the target feature axis
- Impurity = Variance within a partition
- We cannot use the target feature itself
- We ‘color the dots’ based on a selected descriptive feature



Continuous Target Features

Good Classification

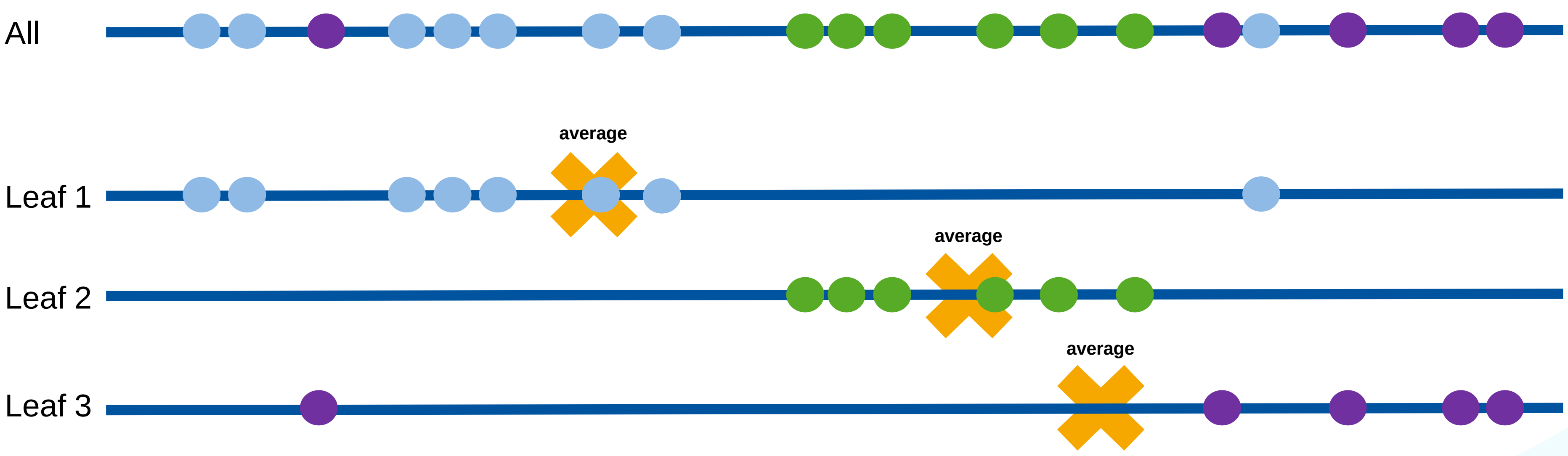
- Three leaves (purple, green, blue show mapping based on descriptive feature)
- Impurity as measure of quality: variance within a leaf of the decision tree



Continuous Target Features

Reasonable Classification

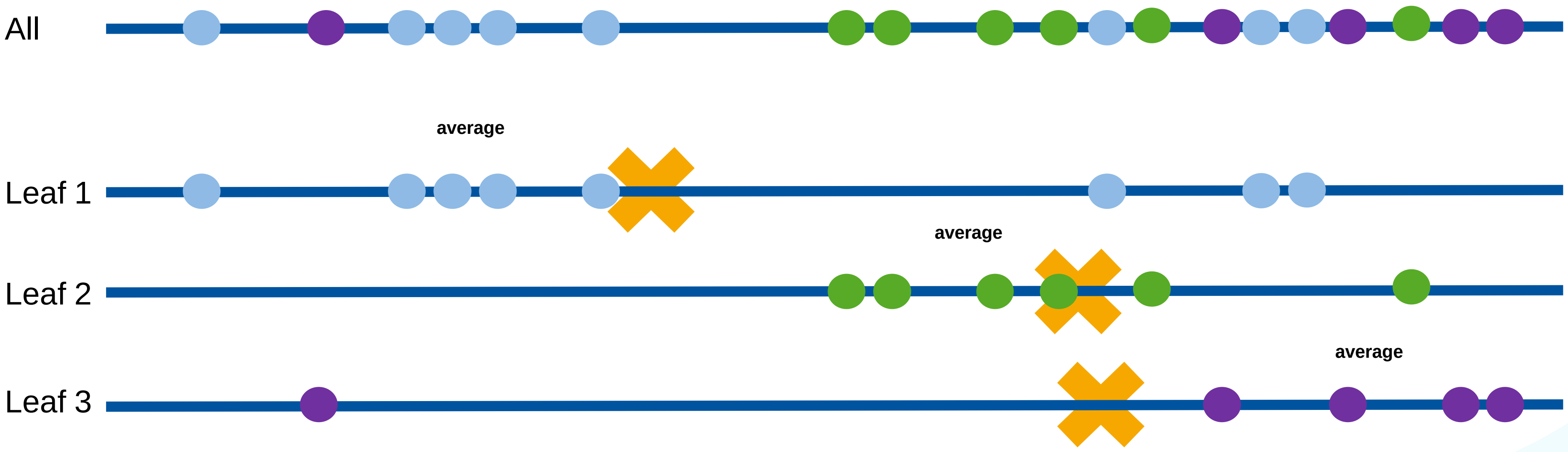
Variance within Leaf 1 and Leaf 3 increased with respect to the ‘good classification’



Continuous Target Features

Poor Classification

Variance within all leaves is high compared to the ‘good classification’



Impurity

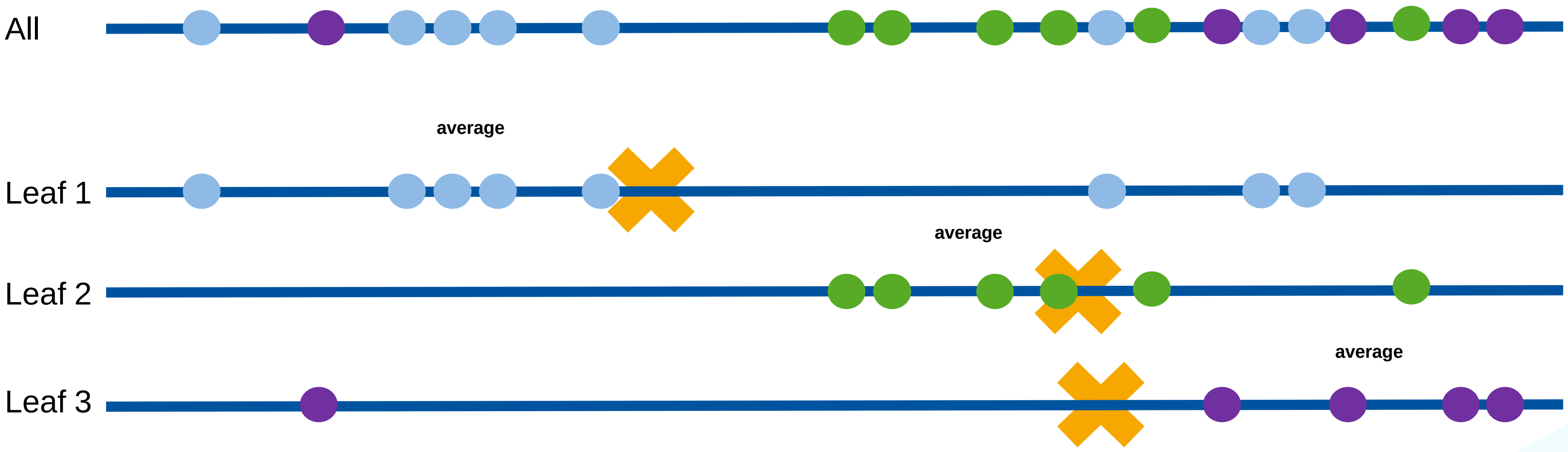
Variance in a Node/Leaf

Number of instances

Target value of instance *i*

Mean of target values

$$Var(t) = \frac{\sum_{i=1}^{N}(t_i - \bar{t})^2}{N-1}$$



Adapting the ID3 Algorithm

ID3 algorithm:

1. **if** all the instances in the dataset have the same classification
   (a) **return** a decision tree with one leaf node with consensus value as a label
2. **else if** there are no features left
   (a) **return** a decision tree with one leaf node with majority value as a label
3. **else if** the dataset is empty
   (a) **return** a decision tree with one leaf node with majority parent value as a label
4. **else**
   (a) pick a feature that lowers the weighted variance most within the subtrees
   (b) once a feature is picked along a path from the root, it cannot be used again
   (c) create subproblems based on the selected feature

Stopping criteria (add pruning strategies to avoid overfitting)

Instead of maximizing information gain

Note that we presented a toolbox! (Not one specific algorithm.)

Many variations are possible by combining ideas

There is no best solution, it all depends on your data

Performance on unseen test data is what counts

Avoid overfitting the data!

Split data into training and test data

Topics such as accuracy and confusion matrix be discussed later

Decision Trees - Conclusion

- Supervised learning aims to explain the target feature in terms of descriptive features
- Decision trees are easy to understand and interpret
- Focus on categorical variables but extensions to continuous data are possible
- Many variations based on the basic ID3 algorithm
  - Pruning
  - Ensembles
  - Information gain definitions
  - …

Next: Clustering techniques