Decision Trees - Part II

Machine Learning, A.Y. 2022/23, Padova

Fabio Aiolli

October 19th, 2022

DTs learning: Inductive Bias

Like other inductive algorithms, ID3 can be seen as a search in a hypothesis space for a hypothesis that best fits the data.

- The hypothesis space is the set of possible decision trees. Note that the corresponding rules are a subset of possible DNF defined on the attribute of the instances. Restrictions?
- The search is of type *hill climbing*. ID3 starts from the empty tree and continues with more and more elaborate trees. Then, it stops when a tree that is consistent with the examples is found. Hence, applying the information gain criterion, it follows that:
  - shorter trees are preferred to larger trees (Breadth First Search approximation)
  - attributes with high informative gain are closer to the root.

Real-valued Attributes

Up to now, we have only considered attributes with discrete values.
What happens when one or more attributes contain continuous values?

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	90 (Hot)	High	Weak	No
D2	Sunny	80 (Hot)	High	Strong	No
D3	Overcast	72 (Hot)	High	Weak	Yes
D4	Rain	60 (Mild)	High	Weak	Yes
D5	Rain	40 (Cool)	Normal	Weak	Yes
D6	Rain	48 (Cool)	Normal	Strong	No
D7	Overcast	40 (Cool)	Normal	Strong	Yes
D8	Sunny	60 (Mild)	High	Weak	Yes
D9	Sunny	40 (Cool)	Normal	Weak	Yes
...	...	...	...	...	...

Real-valued Attributes

Solution: starting from the real-valued attribute A, a new Boolean (pseudo-)attribute is dinamically created as

$$A_c = \begin{cases} \text{true} & \text{if } A < c \\ \text{false} & \text{otherwise} \end{cases}$$

How is it possible to select the "right" value for c?
One possibility is to select the value c that provides the maximal information gain!
The optimal value of c (the one which maximizes the gain) can be proved to be always localized between two values with a different label.

Instances	{D5,D7,D9}		{D6}		{D4,D8}	{D3}		{D2}	{D1}
Values	40		48		60	72		80	90
Target	yes		no		yes	yes		no	no
		↑		↑			↑		
Thresholds		(44)		(54)			(76)		

Real-valued Attributes

Hence, it is sufficient to compute the gain for only a bunch of values:

$$c = 44 \quad \underbrace{\{D5, D7, D9\}}_{Temp<44} \quad \underbrace{\{D1, D2, D3, D4, D6, D8\}}_{Temp\geq 44} \qquad \text{Gain} \simeq \boxed{0.379}$$

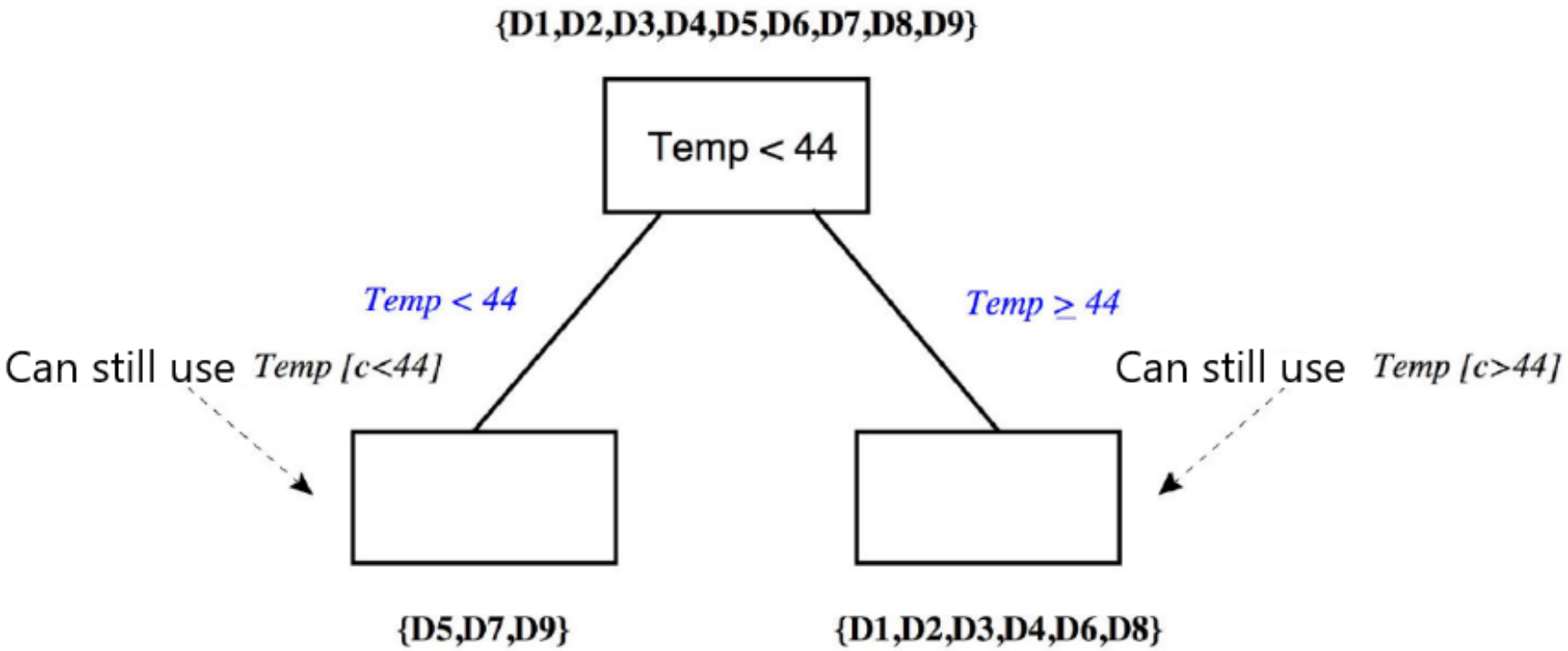
$$c = 54 \quad \underbrace{\{D5, D6, D7, D9\}}_{Temp<54} \quad \underbrace{\{D1, D2, D3, D4, D8\}}_{Temp\geq 54} \qquad \text{Gain} \simeq 0.091$$

$$c = 76 \quad \underbrace{\{D3, D4, D5, D6, D7, D9\}}_{Temp<76} \quad \underbrace{\{D1, D2, D8\}}_{Temp\geq 76} \qquad \text{Gain} \simeq 0.093$$

The threshold $c = 44$ is selected as it corresponds to the threshold with maximal information gain for this attribute! The information gain thus obtained will be compared with the gains of other attributes to choose the optimal attribute.

Real-valued Attributes

Be careful! In the case of real-valued attributes, differently from the case of discrete-valued ones, the same attribute can be used multiple times on the same path (in such a case, the optimal threshold will change).



Attributes with missing values

Problem: In practical applications it can happen that, for some instances, some attributes don't have a value (missing value).

Example: Medical Diagnosis

- For the patient 38 the CT scan report is missing;
- For the patient 45 blood tests and x-rays are missing.

Possible solutions: Let be given a set of examples $\hat{Tr}$. When for an example(x, y) the value of an attribute A is missing, we can:

a) Use the most frequent value for the attribute A in $\hat{Tr}$;

b) As in a) but just considering those examples with label y;

c) Consider all the values $v_i \in V(A)$, and their probability of occurrence $p(v_i|\hat{Tr})$, estimated on $\hat{Tr}$. Then, substitute the example (x, y) with $|V(A)|$ "fractionary examples", one for each possible value v_i and weight equal to $p(v_i|\hat{Tr})$.

Attributes with missing values (example solution a)

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	-	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes

Considering the attribute `Outlook` and the example `D5`:

$$\texttt{D5} \rightarrow (\texttt{[Sunny, Cool, Normal, Weak],Yes})$$

since $P(\texttt{Sunny}|\hat{Tr}) = \frac{1}{2}$, $P(\texttt{Overcast}|\hat{Tr}) = P(\texttt{Rain}|\hat{Tr}) = \frac{1}{4}$

Attributes with missing values (example solution b)

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	-	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes

Considering the attribute `Outlook` and the example `D5`:

$$\texttt{D5} \rightarrow (\texttt{[Overcast, Cool, Normal, Weak]},\texttt{Yes})$$

since $P(\texttt{O}|y = Yes, \hat{Tr}) = \frac{1}{2}$, $P(\texttt{S}|y = Yes, \hat{Tr}) = P(\texttt{R}|y = Yes, \hat{Tr}) = \frac{1}{4}$

Attributes with missing values (example solution c)

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	_	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes

Considering the attribute `Outlook` and the example `D5`:

$$\texttt{D5} \rightarrow \begin{cases} \texttt{D5}_{\texttt{S}} = \texttt{[Sunny, Cool, Normal, Weak]} & p_S = 1/2 \\ \texttt{D5}_{\texttt{O}} = \texttt{[Overcast, Cool, Normal, Weak]} & p_O = 1/4 \\ \texttt{D5}_{\texttt{R}} = \texttt{[Rain, Cool, Normal, Weak]} & p_R = 1/4 \end{cases}$$

Attributes with missing values (example solution c)

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5'	-	Cool	Normal	-	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes

Considering the attribute `Outlook` and the example `D5'`:

If multiple attributes have missing values, the fractionated examples are fractionated again, and the associated weight will be the product of the weights obtained by considering the single attributes in turn.

Attributes with missing values (example solution c)

Learning with missing values (using solution c)

- Modify the concept of cardinality of a set such that the fractionary weights are considered. Unfractionated examples will have weight 1.
- Modify the definition of Information Gain, accordingly.

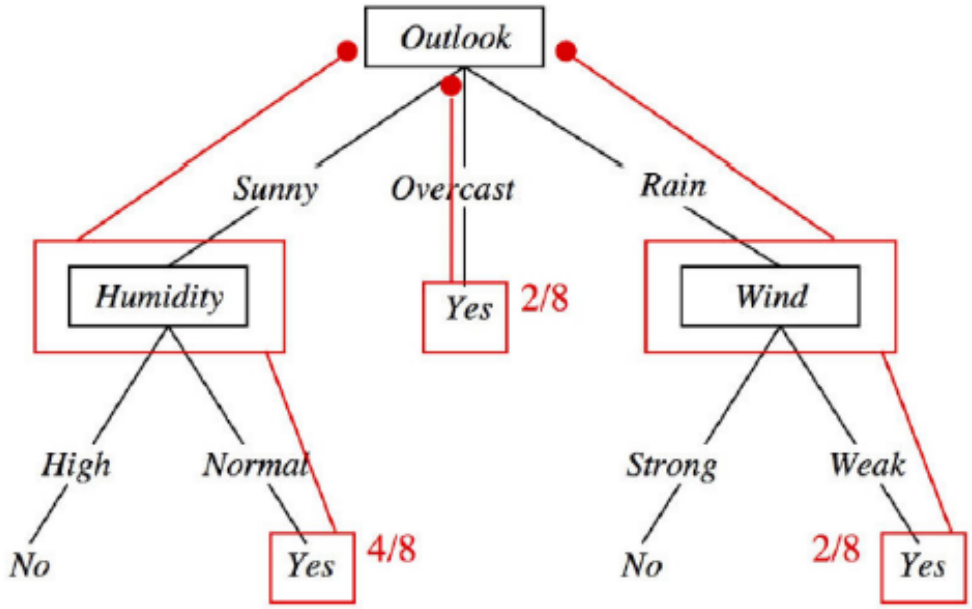
$$G(S, a) = I(S) - \sum_{v \in V(a)} \frac{|S_{a=v}|}{|S|} I(S_{a=v})$$

$$|Q| \rightarrow \sum_{q \in Q} w_q$$

Attributes with missing values (example solution c)

Consider now the classification of a new instance that has missing values. In this case, each fractionated example is classified, and, for each label, the weights of the examples classified with that label are cumulated.

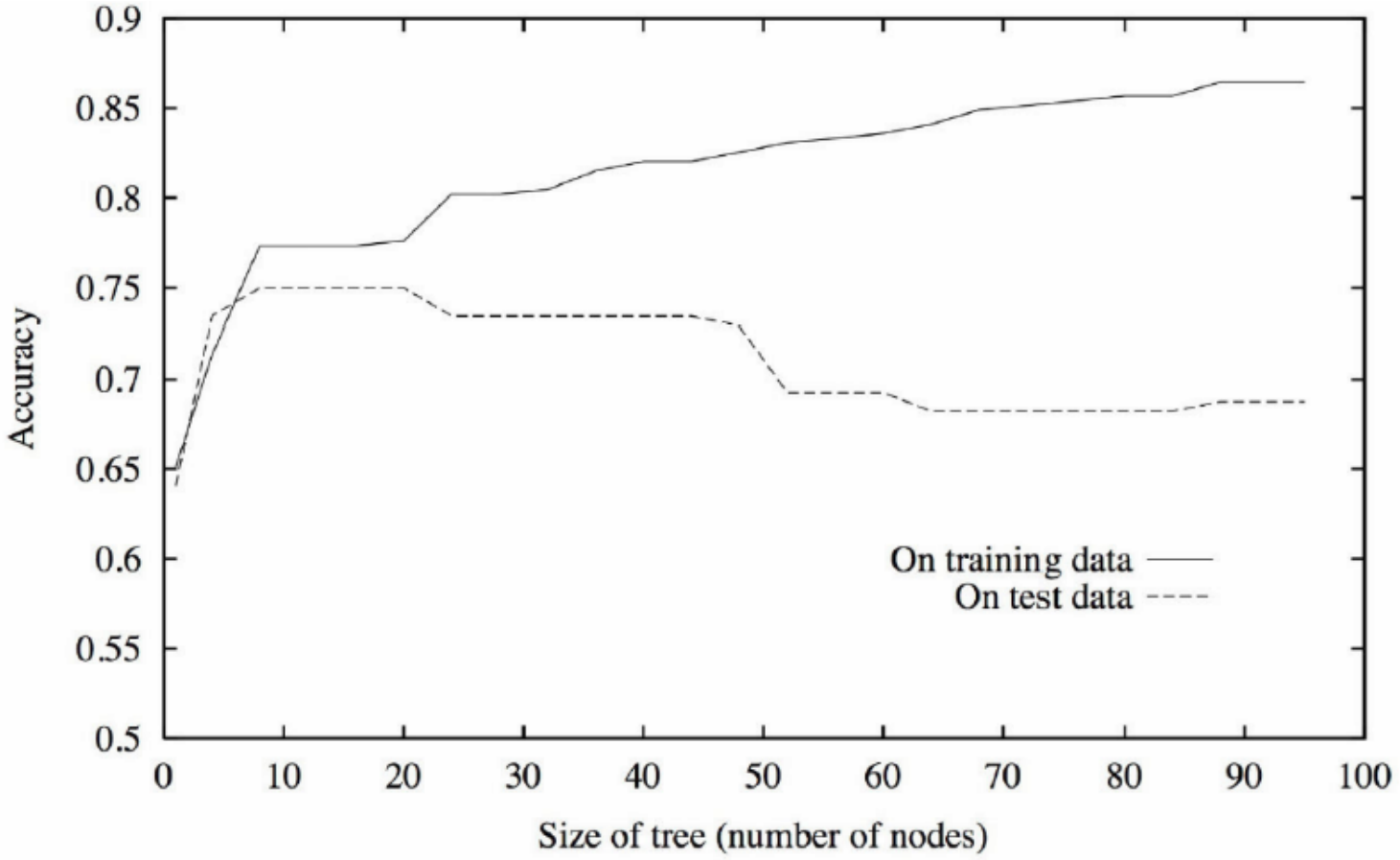
Example: Classification of D5 with solution c



$P(\texttt{Yes}) = 4/8 + 2/8 + 2/8 = 1 \qquad P(\texttt{No}) = 0$

DTs Learning: overfitting

Problem: Overfitting



(Partial) Solution: Pruning

Reduced Error Pruning

- Split `Tr` in `Tr'` and `Va` (validation set), `Tr = Tr' ∪ Va`
- Repeat until the performance gets worse:
  - For each (inner) node, evaluate the accuracy in `Va` obtained in the case when the subtree rooted on the node is pruned.
  - Carry out the pruning that corresponds to the best performance on the validation set.

In this case, "pruning" means replacing, in place of a subtree rooted in a node, a leaf node labeled with the most frequent label in the examples associated with that node.

Rule-Post Pruning

The basic idea is to turn the decision tree into a set of rules, and then do the rule pruning.

- A rule R_i is generated for each path $path(r, f_i)$ from the root $\mathtt{r}$ to the i-th leaf f_i. R_i will be of the form:

$$IF(Att_{i_1} = v_{i_1}) \cap (Att_{i_2} = v_{i_2}) \cap \cdots \cap (Att_{i_k} = v_{i_k}) THEN\ label_{f_i}$$

- An independent pruning is carried out for each rule R_i:
  - Performances obtained using individual R_i as a classifier are estimated
  - Preconditions (one or more) that lead to an increase in the performance estimate in the validation set are removed, using a greedy approach.
- The pruned R_i are sorted in descending order of performance; eventually, add a default rule that returns the most frequent class.

Classification with Post-Rule Pruning

Classification is carried out following the order established for the rules:

- The first rule whose precondition is satisfied by the instance is used to classify the instance;
- If no rules have preconditions satisfied, the default rule is used to classify the instance (the most frequent class of the training set is returned).

Post-Rule Pruning: Some considerations

Some considerations about Post-Rule Pruning:

- The performance estimate needed to carry out the pruning can be done either by using a validation set or by using a statistical test on training data;
- The transformation Tree $\rightarrow$ Rules allows us to generate rules where you can consider contexts for a node that do not necessarily contain the antecedent nodes (and in particular the root): we are actually changing the hypothesis space!
- Compared to trees, the rules are usually easier for a human to understand;
- Usually, Post-Rule Pruning is able to improve performance with respect to the tree on which it is applied and performs better than Reduced-Error Pruning.

Recap

Notions

- Inductive bias of decision trees
- Decision trees for real-valued attributes
- Decision trees for attributes with missing values
- The overfitting problem
- Decision trees and rules

Exercises

- Compare the version of decision trees used by the sklearn software with the theory seen in class.