DECISION TREES PART 1

Introduction

All theory issues were discussed during the lecture, so feel free to go back to the slides and notes from the lecture or [1].

There are many important questions that we need to ask ourselves every day for example: what should I eat? How can we deal with this problem? Which options do we have? How can we decide? Which criteria are crucial for us? Let's look at the graph below.

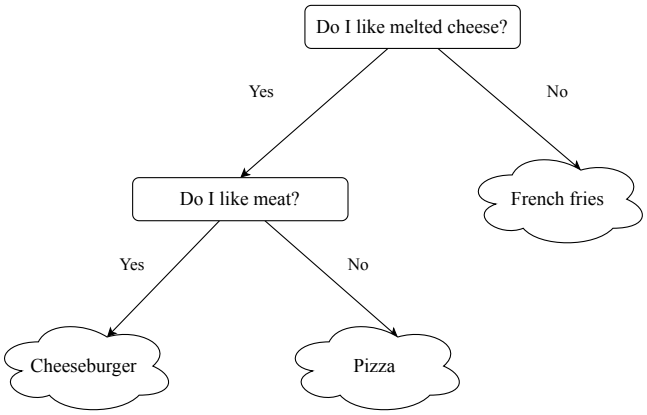


Figure 1: "What should I eat?"–graph.

Decision trees are used for discovering knowledge from given examples. They are constructed as graphs. They are constructed top down with the **root** at the top and proceeding down to **leaves**. [2] Each **node** represents the question about the feature (which we call **attribute**). Split comes from specific attribute values and it depends on the type of the data: binary, categorical, numeric. Leaves represent decision classes.

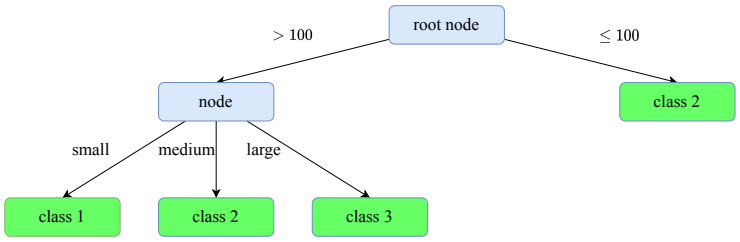


Figure 2: The names of each element in graph.

Question
Let's consider a simple decision tree which is given on Figure 3 and Table 1 which contains examples and attributes values for them.

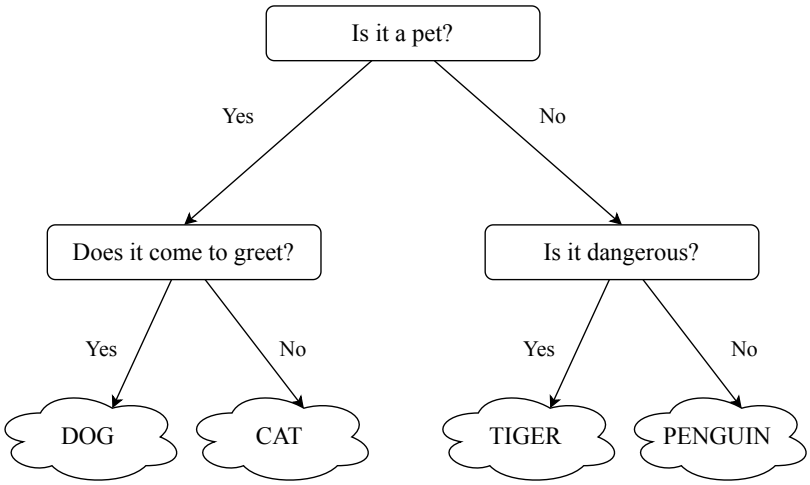


Figure 3: Animal example - decision tree.

Table 1: Animal example - data.

id	pet	greet	dangerous	size	decision
1	yes	yes	no	medium	DOG
2	yes	no	no	medium	CAT
3	no	no	yes	big	TIGER
4	no	no	no	medium	PENGUIN

How would you classify the following example:

id	pet	greet	dangerous	size	decision
5	yes	yes	no	medium	?

Decision trees induction

ID3

ID3 (Iterative Dichotomiser 3), proposed in the 1980s by J.R. Quinlan [2], is a basic algorithm for a tree induction. The tree is constructed from root to leaves, so we call it **top-down** construction.

Main scheme of the tree construction:

- select *"best"* attribute $\boldsymbol{a}$,
- for each value of the attribute $\boldsymbol{a}$ create a branch and descendant node,
- partition all training examples and assign them to the proper node, basing on the value on the attribute $\boldsymbol{a}$,
- if all examples that were assigned to one node have the same class, stop the procedure of creation and create the leaf node with the proper class; otherwise, continue with the steps above recursively.

Question
But how we can choose the best attribute? Which features say that this attribute is "good"? What do we want from such an attribute?

As our main concern is to have leaves with examples that have the same value on the chosen attribute and are assigned to the same class, we want from the attributes to make the partition in the described way. Such a characteristic is called **purity**. We want each node to be as "clean" as it is possible.

To make it possible to calculate which attribute should be chosen, we will use **entropy**. Entropy is the amount of information in random variable. It is based on the probability distribution of a random variable and is a good heuristic for the purity of the attribute.

$$entropy(S) = -\sum_{i=1}^{n} P(x_i) log P(x_i)$$

Questions

1. Having the full deck of cards, probability of choosing red and black one is the same and equals 0.5. Calculate the entropy. What do we know from the given value of entropy?
2. Let's consider that we have just a few cards. The probability p of choosing the red one equals 0.3. Calculate the entropy. What does it change?
3. What is the entropy if we have only red cards?

Having the entropy value, we can say something about one random variable. But what we can say if we have two random variables, what is the influence on our knowledge if we know one of the random variables? The measure that will help us is called **conditional entropy** and intuitively gives us for each subtree a weighted sum of the information content.

$$entropy(S|A) = \sum_{j=1}^{m} \frac{|S_j|}{|S|} entropy(S_j)$$

If we want to have the information about possible reduction on entropy after making a split on one attribute, we can calculate **information gain**. Attribute with higher information gain can better classify train data.

$$gain(S, A) = entropy(S) - entropy(S|A)$$

Example
In this task the training set of examples from [4] is considered. Robert wants to buy a new car. Below you can see some of the cars which are evaluated on following attributes: buying price, number of doors and safety. There is only simple decision: is it right for Robert or not.

Attributes:

- *buying_price*: low, med, high, vhigh;
- *doors*: 3, 4, 5more;
- *safety*: low, med, high.

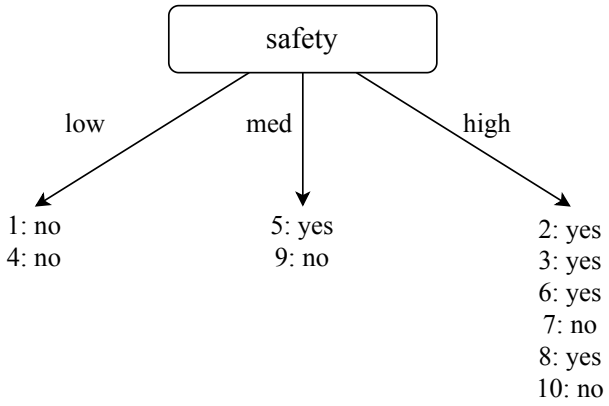
Class values: yes, no.

	buying_price	doors	safety	decision
1	high	4	low	no
2	high	5more	high	yes
3	low	5more	high	yes
4	low	5more	low	no
5	low	4	med	yes
6	low	4	high	yes
7	med	4	high	no
8	med	3	high	yes
9	vhigh	3	med	no
10	vhigh	5more	high	no

Let's start from calculation of entropy for the whole set of examples:

$$entropy(S) = -\frac{1}{2} log_2 \frac{1}{2} - \frac{1}{2} log_2 \frac{1}{2} = 1$$

Then we will consider each attribute, starting from *safety*.



***safety* = low**: $entropy(S_{1s}) = -\frac{0}{2} * log_2\frac{0}{2} - \frac{2}{2} * log_2\frac{2}{2} = 0$

***safety* = med**: $entropy(S_{2s}) = -\frac{1}{2} * log_2\frac{1}{2} - \frac{1}{2} * log_2\frac{1}{2} = 1$

***safety* = high**: $entropy(S_{3s}) = -\frac{4}{6} * log_2\frac{4}{6} - \frac{2}{6} * log_2\frac{2}{6} = 0.918$

Then we can calculate conditional entropy:

$$entropy(S|safety) = \frac{2}{10} * 0 + \frac{2}{10} * 1 + \frac{6}{10} * 0.918 = 0.7508$$

Finally, we can calculate information gain:

$$gain(S, safety) = 1 - 0.7508 = 0.2492$$

.

Task

Now is your turn to calculate entropy for each subtree considering *buying_price* attribute, conditional entropy and information gain for this attribute.

Task

Having the values from above example and task and results for *doors* attribute as follows, on which attribute will you make split?

$$entropy(S|doors) = 1$$

However the measure *information gain* is not without flaws.

Question
Can you find any potential problems with *information gain*? In which situations it will not work in the way we want?

The problem can occur for attributes with a big number of different values (or extremely with the unique value on the attribute for each example). Information gain prefers attributes that make "clearer" split on examples. Such a situation can be problematic for classifying test data and potentially leads to **overfitting** to the train data on which we built our decision tree. To deal with such a problem there comes the measure **gain ratio** which takes into account not only the purity of the node but also the distribution of examples into branches. The main idea is to normalize information gain by the **intrinsic information** which takes into account the size of each branch. Importance of attribute decreases as intrinsic information gets larger.

$$IntrinsicInfo(S, A) = -\sum_{i=1}^{n} \frac{|S_i|}{|S|} log_2 \frac{|S_i|}{|S|}$$

$$GainRatio(S, A) = \frac{Gain(S, A)}{IntrinsicInfo(S, A)}$$

Example Going back to our example, let's calculate gain ratio for each attribute on which we could make a split, starting from *safety*.

$$IntrinsicInfo(S, safety) = -\frac{2}{10} log_2 \frac{2}{10} - \frac{2}{10} log_2 \frac{2}{10} - \frac{6}{10} log_2 \frac{6}{10} = 1.371$$

$$GainRatio(S, safety) = \frac{0.2492}{1.371} = 0.1818$$

Task
Calculate intrinsic info and gain ratio for the other attributes. Are the best attributes from information gain and gain ratio the same? Why there is possibility that they are different?

Question
Can you see potential problem for gain ratio?

Handling continuous-valued attributes [3]

Question
It cannot be so easy that we have only categorical values on each attribute. But how can we deal with continuous values on attributes?

One of the approaches is to define a metric on which we could establish thresholds between elements and then determine which threshold(s) is/are the best for our problem. It is our choice and of course the type of problem that will affect the number of thresholds that we finally choose. What needs to be remembered is the fact that thresholds should be made on the boundary of two classes.

Example

- The first example is pretty easy: there is clear pass from one class to another so we want to put the boundary between values 26 and 34. Usually, we use mean value of this two as a threshold, this time it is 30.

id	1	2	3	4	5	6	7	8
age	18	21	24	26	34	41	46	52
class	Y	Y	Y	Y	N	N	N	N

- The second example is a bit more complicated. We still have two classes to which we want to assign our examples but there is no one clear pass between classes as it is above. The potential candidates still can be determined as mean of the values on which there is change for example 21 and 24 (with mean equals to 22,5). Having determined all possible candidates for being a threshold, we can calculate information gain for each possibility and then choose the one with better score.

id	1	2	3	4	5	6	7	8
age	18	21	24	26	34	41	46	52
class	Y	Y	N	N	N	Y	Y	Y

- The third example shows classification to more than two classes. You need to remember than you can put more than one boundary on the value, especially when you have more than two classes.

id	1	2	3	4	5	6	7	8
age	18	21	24	26	34	41	46	52
class	1	1	2	2	2	3	3	3

References

[1] KRAWIEC, Krzysztof; STEFANOWSKI, Jerzy. Uczenie maszynowe i sieci neuronowe. Wydaw. Politechniki Poznańskiej, 2003.

[2] QUINLAN, J.. Ross . Induction of decision trees. Machine learning, 1986, 1.1: 81-106.

[3] FAYYAD, Usama M.; IRANI, Keki B. On the handling in decision tree of continuous-valued attributes generation. Machine Learning, 1992, 8: 87-102.

[4] DUA, D.; GRAFF, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.