Lab Class ML:III

By Wednesday, 2017-12-06 solutions for the following exercises have to be submitted: 1, 4, 5, 6.

Exercise 1 : Decision Trees

Construct a decision tree for each of the following boolean functions. Note: The target concept is the set of all models, i.e., set of interpretations (0/1 assignments to the boolean variables) that fulfill a formula.

(a) $A \wedge \neg B$

(b) $A \vee (B \wedge C)$

(c) A *XOR* B

(d) $(A \wedge B) \vee (C \wedge D)$

Exercise 2 : Decision Trees (Background)

(a) For the construction of a decision tree almost always a top-down greedy search in the hypothesis space is employed. Explain the term Greedy Search (synonymously: search with a greedy strategy). What are its advantages and what are its disadvantages? When is a greedy strategy useful? Which alternative strategies exist?

(b) The inductive bias of the Candidate Elimination algorithm is based on a different mechanism than the inductive bias of the ID3 algorithm. Explain this statement by analyzing the rationale of the inductive bias for each algorithm.

(c) What is the time complexity of the ID3 algorithm? Explain your answer.

Exercise 3 : Decision Trees (Overfitting)

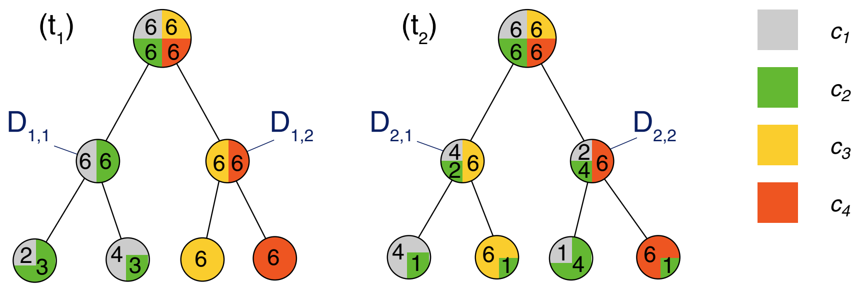
(a) What is overfitting?

(b) Why is the example set D partitioned in a test set and a training set? Is such a partitioning necessary to avoid overfitting?

(c) An approach to avoid overfitting is the use of so-called post-pruning algorithms: initially, an oversized decision tree is constructed, which then is generalized by means of pruning. Explain different pruning strategies such as reduced-error pruning and rule post pruning.

Exercise 4 : Impurity Functions

Let D be a set of examples over a feature space X and a set of classes $C = \{c_1, c_2, c_3, c_4\}$, with $|D| = 24$. Consider the following illustration of two possible decision trees, t_1 and t_2 – the colors represent the classes present in each document set associated with the nodes of the trees; the numbers denote how many examples of each class are present.



(a) First, consider only the first split that each of the two trees makes: compute $\Delta\iota(\{D_{1,1}, D_{1,2}\})$ and $\Delta\iota(\{D_{2,1}, D_{2,2}\})$ with the misclassification rate $\iota_{misclass}$ and the entropy criterion $\iota_{entropy}$ as splitting criterion. Interpret the results: which of $\{D_{1,1}, D_{1,2}\}$ or $\{D_{2,1}, D_{2,2}\}$ is the better first split?

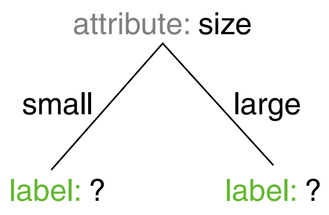
(b) Which of t_1, t_2 is the better decision tree, and why?

(c) Assuming the splits shown are the only possibilities, which of t_1 or t_2 would the ID3 algorithm construct, and why?

Exercise 5 : Cost functions

Consider the set of training examples describing mushrooms, and the simple one-level decision tree given below:

	Color	Size	Points	Eatability
1	red	small	yes	toxic
2	brown	small	no	eatable
3	brown	large	yes	eatable
4	green	small	no	eatable
5	red	large	no	eatable



(a) Determine the labels of all nodes using the cost function $\mathit{cost}(c' \mid c)$:

$$\mathit{cost}(c' \mid c) \begin{cases} = 1 & \text{if } c' \neq c, c \in C \\ = 0 & \text{otherwise} \end{cases}$$

(b) Devise a new cost function to ensure that none of the poisonous mushrooms in the training set are classified as eatable, and determine the labels of all nodes (for this exercise, the structure of the tree remains fixed).

(c) Using the formula given in the lecture slides, compute the misclassification costs of the tree for both cost functions.

Exercise 6 : $\boxed{\text{P}}$ Algorithm ID3 and Measuring Performance

Develop a basic Python implementation of the ID3 algorithm discussed in the lecture. Use the mushroom example data from the slides to develop and test your implementation. For convenience, represent the examples as a list of Python dictionaries mapping attributes to their values. Hence, the example dataset becomes:

```
mushrooms = \
[{'Color': 'red', 'Eatability': 'toxic', 'Points': 'yes', 'Size': 'small'},
 {'Color': 'brown', 'Eatability': 'eatable', 'Points': 'no', 'Size': 'small'},
 {'Color': 'brown', 'Eatability': 'eatable', 'Points': 'yes', 'Size': 'large'},
 {'Color': 'green', 'Eatability': 'eatable', 'Points': 'no', 'Size': 'small'},
 {'Color': 'red', 'Eatability': 'eatable', 'Points': 'no', 'Size': 'large'}]
```

(a) Implement a function `conditional_entropy(examples, attribute, target)` where `examples` is a list of dictionaries like above, and `attribute` and `target` are strings. The function should return $H(\texttt{target} \mid \texttt{attribute})$. Refer to the lecture notes for the computation of H, and remember that $\log_2(0)$ needs to be treated specially.

Verify a few test cases: for instance, `conditional_entropy(mushrooms, 'Color', 'Eatability')` should return `0.4`.

(b) Develop a class `ID3Node` which holds all the information related to one node in a decision tree, including: which attribute is tested by the node; the node label; the children in the decision tree, and which attribute value corresponds to which child. Your class should have at least the following two methods:

- `create_edge(self, attribute_value, child)`, where `child` is another `ID3Node`, and `attribute_value` is the corresponding value of the attribute tested by `self`.
- `classify(self, example)` where `example` is a single dictionary like those in the list above, and the return value is is one of the possible values of the target attribute.

Note: refer to the documentation if you are new to object-oriented programming with Python.

(c) Implement the function `id3(examples, attributes, target)` that builds a decision tree according to the algorithm from the lecture and returns the root node. The parameter `examples` is a list of dictionaries as above, `attributes` is a set of strings with the names of the predictors, and `target` is a string with the name of the response variable.

(d) Implement the function `misclassification_rate(tree, examples, target)`, where `tree` is the root node of a decision tree, and the other parameters defined as above. Verify that a tree trained using your `id3` function on all five mushroom examples achieves a misclassification rate of zero when tested on the same examples.

(e) Implement the function `cross_validate_id3(examples, target, k)` which splits the `examples` into `k` similarly-sized random subsets, and then trains and evaluates `k` decision trees using the k-fold cross-validation procedure. Your function should print out the misclassification rate for each fold, and report the average cross-validated misclassification rate at the end.

(f) Download the car evaluation data set from the UC Irvine Machine Learning Repository. Read the data set description, then convert the CSV data to our list-of-dictionaries format. Train and evaluate ID3 decision trees with 5-fold cross-validation, and compute the average misclassification rate.