Exercise for Machine Learning (SS 20)

Assignment 5: Decision Trees

Prof. Dr. Steffen Staab, steffen.staab@ipvs.uni-stuttgart.de
Alex Baier, alex.baier@ipvs.uni-stuttgart.de
Janik Hager, janik-manel.hager@ipvs.uni-stuttgart.de
Ramin Hedeshy, ramin.hedeshy@ipvs.uni-stuttgart.de
Analytic Computing, IPVS, University of Stuttgart

Submit your solution in Ilias as either PDF for theory assignments or Jupyter notebook for practical assignments.
Mention the names of all group members and their immatriculation numbers in the file.
Submission is possible until the following Monday, 01.06.2020, at 14:00.

1 Inductive Construction

Given the dataset in Table 1 construct a decision tree by hand using the top-down algorithm presented in the lecture. Your stop criteria are zero entropy or a depth of 2 (the root node is at depth 0, the first layer of inner nodes are at depth 1, ...).
Draw your final decision tree and provide your computation steps (information gain per relevant attribute for each split, class frequencies for each node, . . .).
Compute the error rate of your decision tree on the training data.
Solution:
Depth = 0:
Compute entropy for root node:

	$-$	$+$	H
root	125	115	0.9987

Compute information gain for each possible split of the root node:

F_1	$-$	$+$	H
0	70	50	0.9799
1	55	65	0.9950

$IG = 0.0113$

F_2	$-$	$+$	H
0	50	70	0.9799
1	75	45	0.9544

$IG = 0.0316$

Table 1: Dataset consists of 4 categorical features ($F_1 \in \{0,1\}$, $F_2 \in \{0,1\}$, $F_3 \in \{0,1,2\}$, $F_4 \in \{0,1\}$) and a binary classification target with labels $\{-,+\}$.

F_1	F_2	F_3	F_4	Instances $-$	Instances $+$
0	0	0	0	10	0
0	0	0	1	5	5
0	0	1	0	10	0
0	0	1	1	10	0
0	0	2	0	0	10
0	0	2	1	0	10
0	1	0	0	10	0
0	1	0	1	10	0
0	1	1	0	5	5
0	1	1	1	0	10
0	1	2	0	10	0
0	1	2	1	0	10
1	0	0	0	0	10
1	0	0	1	10	0
1	0	1	0	5	5
1	0	1	1	0	10
1	0	2	0	0	10
1	0	2	1	0	10
1	1	0	0	10	0
1	1	0	1	10	0
1	1	1	0	10	0
1	1	1	1	10	0
1	1	2	0	0	10
1	1	2	1	0	10

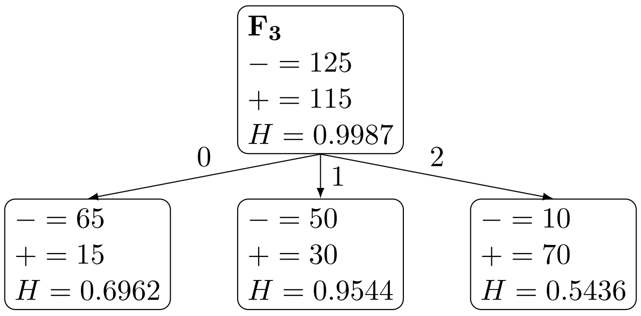
F_3	$-$	$+$	H
0	65	15	0.6962
1	50	30	0.9544
2	10	70	0.5436

$IG = 0.2674$

F_4	$-$	$+$	H
0	70	50	0.9799
1	55	65	0.9950

$IG = 0.0113$

Because F_3 yields the highest information gain, we perform the split accordingly and construct the following tree:



Depth=1:
For each child node in the new layer, we repeat the process of computing IGs and splitting. Because we have split on F_3, we do not need to consider for future splits.
Compute IG of all possible splits for child node $F_3 = 0$:

F_1	$-$	$+$	H
0	35	5	0.5436
1	30	10	0.8113

$IG = 0.0188$

F_2	$-$	$+$	H
0	25	15	0.9544
1	40	0	0.0000

$IG = 0.2190$

F_4	$-$	$+$	H
0	30	10	0.8113
1	35	5	0.5436

$IG = 0.0188$

We perform a split at F_2, as it achieves the highest IG.

Compute IG of all possible splits for child node $F_3 = 1$:

F_1	$-$	$+$	H
0	25	15	0.9544
1	25	15	0.9544

$IG = 0.0000$

F_2	$-$	$+$	H
0	25	15	0.9544
1	25	15	0.9544

$IG = 0.0000$

F_4	$-$	$+$	H
0	30	10	0.8113
1	20	20	1.0000

$IG = 0.0487$

We perform a split at F_4, as it achieves the highest IG.

Compute IG for all possible for child node $F_3 = 2$:

F_1	$-$	$+$	H
0	10	30	0.8113
1	0	40	0.0000

$IG = 0.1379$

F_2	$-$	$+$	H
0	0	40	0.0000
1	10	30	0.8113

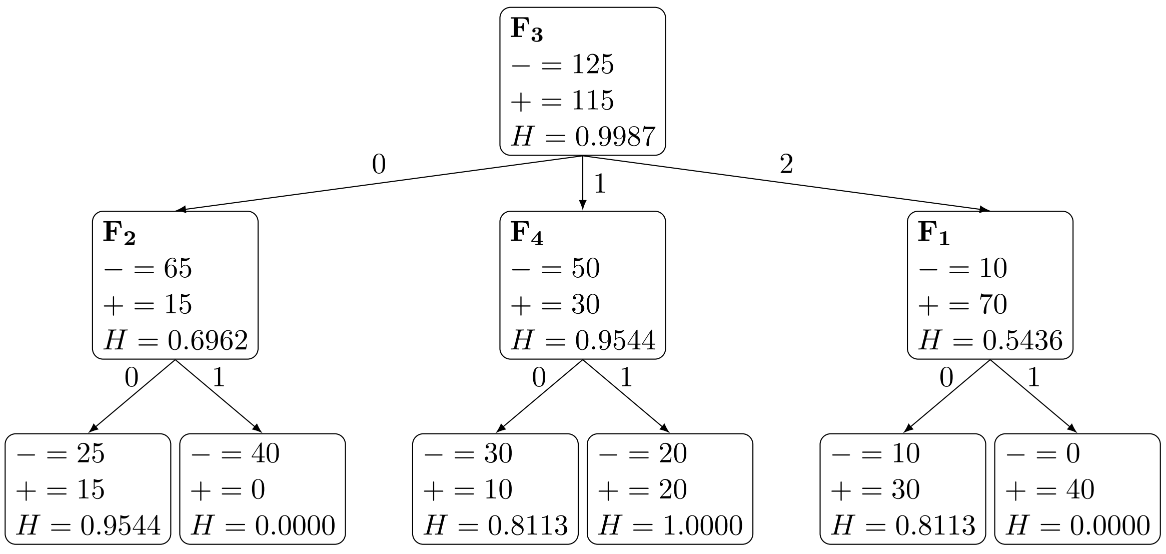
$IG = 0.1379$

F_4	$-$	$+$	H
0	10	30	0.8113
1	0	40	0.0000

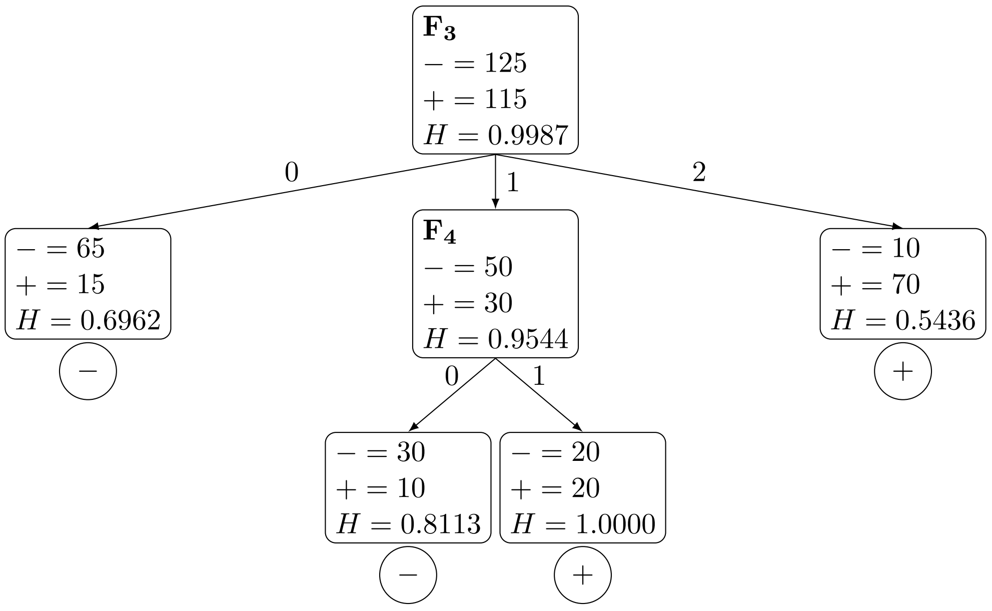
$IG = 0.1379$

All splits yield the same IG, we therefore arbitrarily choose to split at F_1.

We have determined splits for all nodes at depth 1 and can construct the following tree:



Depth = 2:
Our stopping criterion is a depth of 2, therefore our tree construction is finished. The classification decision of each leave node is determined by the majority of class labels. We can see that for the leaves of node F_2 and F_1 the same class is predicted, which allows us to prune them without any performance loss. The leaf $F_4 = 1$ has an equal number of negative and positive labels, accordingly we have to arbitrarily select one class label as classification decision. We choose the label $+$. This yields the following tree:



Error rate: Finally, we can compute the overall error rate on the training data:

$$E = \frac{15 + 10 + 20 + 10}{80 + 40 + 40 + 80} = \frac{55}{240} = 0.23$$

2 Minimal Error Pruning

Figure 1 shows a decision tree for classifiying mushrooms as either edible or poisonous. We used the same dataset as shown in the lecture and constructed it with scikit-learn with a maximum depth of 4. Because the implementation of scikit-learn uses a slightly different algorithm (CART, in the lecture: ID3), which allows continuous data, we have to apply one-hot encoding to each variable resulting in a binary decision tree.
Prune two decisions (inner nodes) of the decision tree based on the error rate. First, compute the overall error rate of the original tree. Second, consider the removal of each viable node by computing the error rate after removing it from the tree. Third, identify the node with the lowest error rate and prune it. Repeat this process to remove a second inner node.
Draw the decision tree after each pruning step. Provide your calculations for the pruning step.
Note: A node is viable for pruning, if all its children are leaf nodes. After pruning the chosen inner node turns into a leaf node. We define the overall error rate as (slide 44):

$$\frac{\text{number of misclassified samples in each leaf}}{\text{total number of samples}}$$

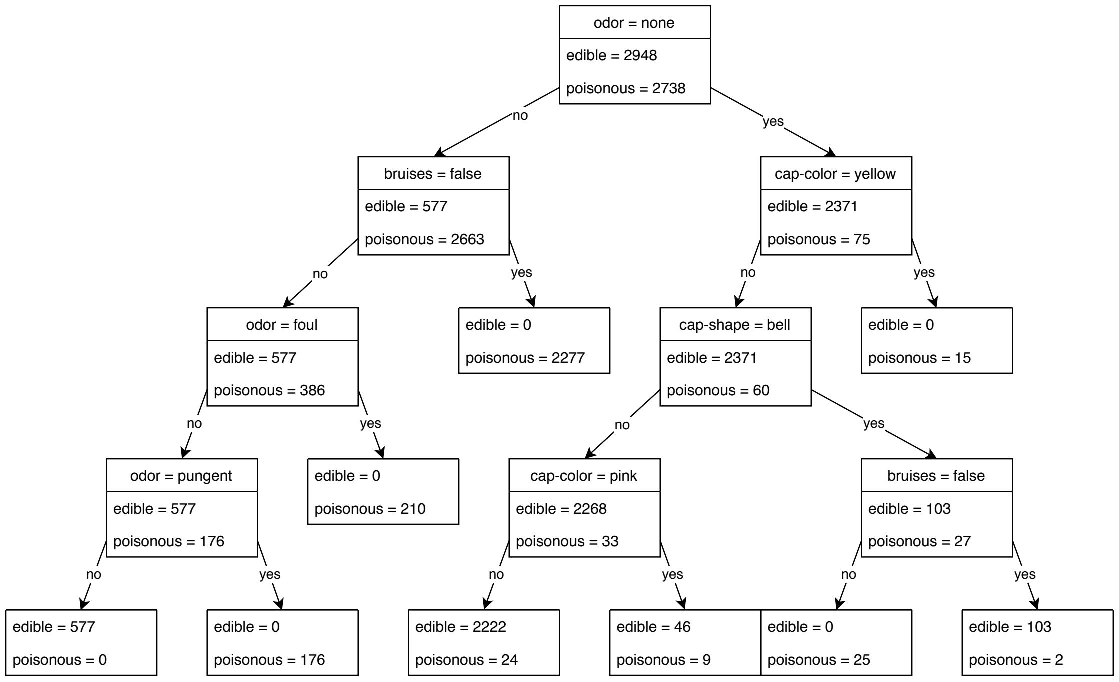


Figure 1: Decision tree of depth 4 classifies mushrooms as edible or poisonous. Inner nodes have a decision rule at the top, the left child is chosen if the rule does not apply and the right child is chosen if the rule applies. For each leaf and inner node, the class frequencies for the training data are summarized.

Solution:
The error rate for the original tree is:

$$E = \frac{24+9+2}{5686} = 0.0062$$

The error rate after removing "odor = pungent" is:

$$\frac{176+24+9+2}{5686} = 0.0371$$

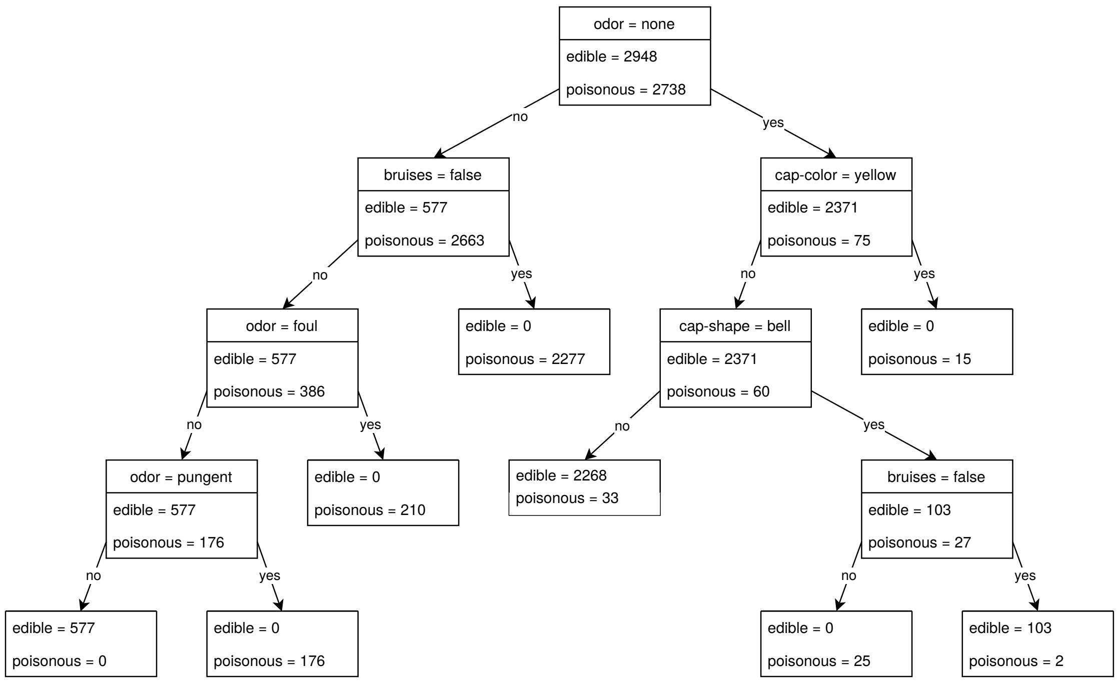
The error rate after removing "cap-color = pink" is:

$$\frac{33+2}{5686} = 0.0062$$

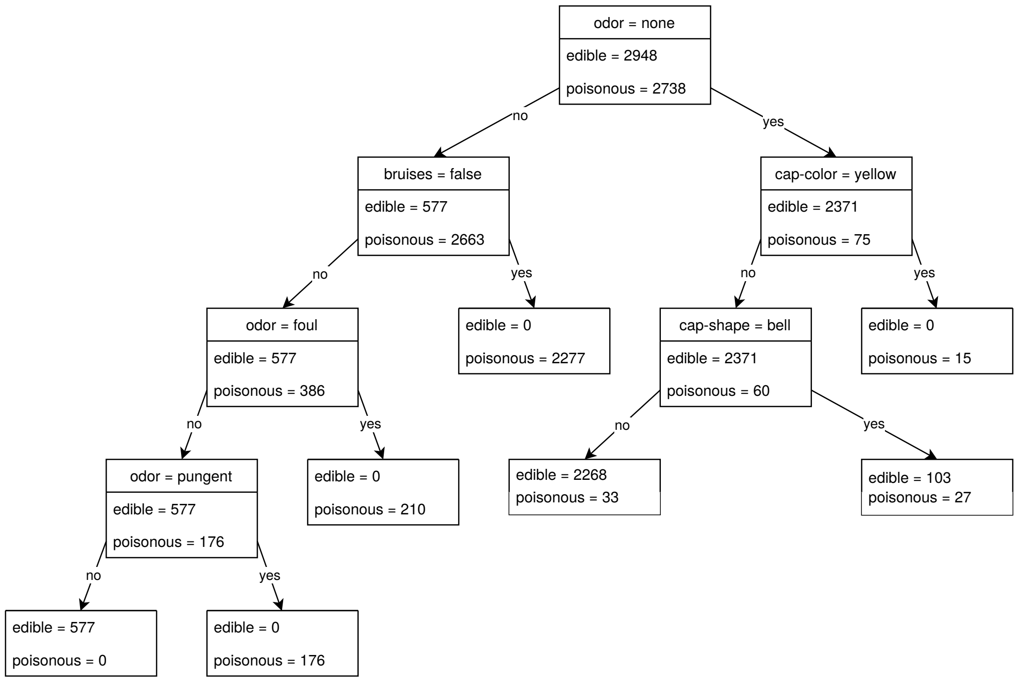
The error rate after removing "bruises = false" is:

$$\frac{24+9+27}{5686} = 0.0106$$

Our first pruning operation removes the node "cap-color = pink", since the error rate does not chance due to its removal. This results in the following tree:



For the second pruning operation, we can reuse our previous computations. We prune the node "bruises = false", as it leads to the lowest increase in error rate. Accordingly, our final pruned tree follows:



3 Regression with Decision Trees and kNN

For all students other than B.Sc. Data Science.

Decision trees and k-nearest-neighbors are not limited to classification but can also be used for regression. Research and explain the following two questions in sufficient detail:

1. How does the construction of regression trees differ to classification trees? How is a prediction computed in a regression tree?

2. How can kNN be used for regression?

Please cite your sources.