Pattern and Speech Recognition Tutorial
Exercise 7

Exercise 1 (7 points)

Decision trees are a simple way to classify continuous and discrete data. In this exercise you will generate a given decision tree manually and have a first look at decision tree construction. The input is a set of samples $\mathcal{D} = \{(x_i, \omega_i)\}$ where x_i are the attributes and ω_i is the class label of sample i.

A decision tree takes some x_i and asks a sequence of questions about it, each question depends on the previous answer. The output is either a probability distribution over class labels or the most likely class label at a leaf. We restrict ourselves to binary decision trees and questions about single attributes.

1. Load the data (*voting.tab*) from `https://goo.gl/ZHWJ71`.

2. Implement the decision tree given from figure 1. Your function should take a sample $d_i \in \mathcal{D}$ and return the path of the tree from root to leaf with the answers corresponding to each node.

3. Apply the tree to the data set. Adapt the code from (2) such that for each node the number of samples of each class label is stored for all samples reaching this node. Plot a histogram for the two class labels for each node. Report the class with the highest probability for every node. What does the histogram tell you about the quality of the questions?

4. Use the information from (3) to compute the misclassification rate of every node. The misclassification rate is the proportion of misclassified samples. Assume that you always predict the most likely class at every node.

5. The histogram of each node induces a probability distribution over the class labels. The entropy (slide 23) of a probability distribution is a measure of its uncertainty. For two classes ω_1, ω_2 the entropy is defined as

$$H(\omega_1, \omega_2) = -(P(\omega_1) \cdot \log_2 P(\omega_1) + P(\omega_2) \cdot \log_2 P(\omega_2))$$

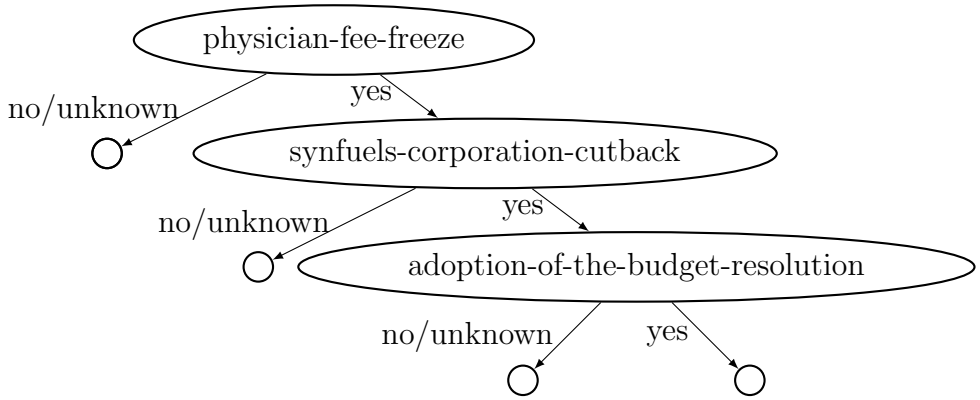

Figure 1: Decision Tree

Compute the entropy for the distribution of every node.

6. Now, we are interested in the reduction in entropy for every (non–leaf) node. Therefore, subtract from the entropy of every node the weighted average of the entropy of its child nodes. Weight every child nodes entropy with the probability to go to this child in the next step. The result is called a nodes (expected) *information gain*. Use the equation from slide 26.

7. When constructing a decision tree, one is interested in minimizing the (expected) number of tests necessary to identify a class. However, finding an optimal decision tree is computationally infeasible (NP–complete). The solution is to use a greedy approach where one chooses a node and a question which give the most information for the classification. The information gain is a good measure for this (explain why). Which question would you ask next to maximize the information gain? Assume you want to split the last "yes"–node.

8. How would you design an algorithm to construct a complete decision tree based on some arbitrary data $\{(x_i, \omega_i)\}$. You do not(!) have to implement it, just **explain** your idea. Assume that the data in x_i can be categorical or continuous.

Exercise 2 (3 points)

Consider the data from *voting.tab* again.

1. What are the prior probabilities of the two classes?

2. What is $P(x[\text{"education-spending"}] = \text{no} \mid \omega_k)$ for $\omega_k \in \{\text{republican, democrat}\}$?

3. Which class has the highest probability, given that $x[\text{"education-spending"}] = \text{no}$, i.e. determine $\overline{\omega} = \underset{\omega_k}{\text{argmax}}\ P(\omega_k \mid x[\text{"education-spending"}] = \text{no})$.

4. Construct a loss function λ such that the conditional risk of predicting republican and democrat is equal, given that $x[\text{"education-spending"}] = \text{yes}$. Show that your loss function fulfills this property.

Submission architecture

You have to generate a **single ZIP file** respecting the following architecture:

```
tutorial1_<matriculation_nb1>_<matriculation_nb2>_<matriculation_nb3>
|
+--- source
|      |
|      +------- file 1
|      +------- file 2
|      +------- ...
+--- rapport.pdf
+--- README.txt
```

where

- **source** contains the source code of your project,
- **rapport.pdf** is the report where you present your solution with **the explanations (!)** and the plots,
- **README** which contains group member informations (name, matriculation numbers and emails) and a **clear** explanation about how to compile and run your source code

The ZIP filename has to be :

`tutorial7_<matriculation_nb1>_<matriculation_nb2>_<matriculation_nb3>.zip`

You have to choose between the following languages **python** or **matlab**. Other languages won't be accepted.

Some hints

We advice you to follow the following guidelines in order to avoid problems :

- Avoid building complex systems. The exercises are simple enough.
- Do not include any executables in your submission, as this will cause the e-mail server to reject it.

Grading

Send your assignment to the tutor who is responsible of your group:

- Gerrit Gromann `gerritgr@gmail.com`
- Sbastien Le Maguer `slemaguer@coli.uni-saarland.de`
- Kata Naszdi `b.naszadi@gmail.com`

The email subject should start with `[PSR TUTORIAL 7]`