Lab assignment 2: Naive Bayes classifier

- Task 1: Data preprocessing
- Task 2: Build a naive Bayes classifier

Describe everything in a report.

Task 1: Data preprocessing

Here you have to download data and prepare them for being used with a Matlab program.

Download the Weather data set and the Weather data description.

Remark: For use with Matlab it is convenient to convert all attribute values (called "levels" when they are nominal) into integers>=1. So they can be used to index matrices. You can do so by either using a plain text editor, or reading the data into a spreadsheet table and then saving it back (remember to "save as" a plain text file or at most a .csv file, not a spreadsheet file).

Task 2: Build a naive Bayes classifier

Here you have to create a program (a Matlab function for instance) that takes the following parameters:

1. a set of data, as a *n* x *d+1* matrix, to be used as the training set
2. another set of data, as a *m* x *c* matrix, to be used as the test set

The program should:

1. Check that the number *c* of columns of the second matrix is at least the number *d* of columns of the first matrix - 1
2. Check that no entry in any of the two data sets is <1
3. Train a Naive Bayes classifier on the training set (first input argument), using its last column as the target
4. Classify the test set according to the inferred rule, and return the classification obtained
5. If the test set has a column number d+1, use this as a target, compute and return the error rate obtained (number of errors / *m*)

Write a script that, **without the user's intervention**:

1. loads the weather data set (already converted in numeric form)
2. splits it into a training set with 10 randomly chosen patterns and a test set with the remaining 4 patterns
3. calls the naive Bayes classifier and reads the results
4. prints the results: classification for each pattern of the test set, corresponding target if present, error rate if computed.

Remarks: The classifier should be programmed in such a way to be suitable for the following situations:

- other data sets of different cardinalities (*n*, *m*) and dimensionality (*d*)
- test sets not including a target
- test sets including attribute values that **were not in the training set**.

In the last case the program should issue an error and discard the corresponding pattern, because if you use that value to index a matrix it will likely go out of bounds (e.g., if you computed probabilities for values 1, 2 and 3 and you get value 4 in the test).

Use the slides to implement the classifier. Note that there are some attributes that are non-binary.