KIV/UIR-E - Project Specification

Document Classification

The main goal of this project consists in design and implementation of a program to classify text documents into classes according to their content, such as weather, sports, politics, etc. The program must full-fill the following conditions:

- Use the corpus of English documents available at http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html. For training / testing split division use the field LEWISSPLIT="TRAIN" / LEWISSPLIT="TEST" from *.sgm files. Consider only the first class of the document, i.e. from the topic list $< D\, money - fx < /D >< D > dlr < /D >< D > yen < /D >< D > dmk < /D >$ use only "money" tag.
- Another option is to use the Czech corpus available at http://ctdc.kiv.zcu.cz/). Consider also only the first class of the document, e.g. document 05857_**zdr**_ptr_eur.txt belongs to class "**zdr**" - health service.). This corpus is detailed at http://ctdc.kiv.zcu.cz/lrec18.pdf.
- Implement at least three different (parametrization) algorithms for document representation (to create feature vectors).
- Implement at least three different classification algorithms (supervised learning)
  - Minimum Distance Classifier
  - Naive Bayes classifier
  - Your choice
- Functionality as follows:
  - execution with parameters: training_set, testing_set, parameterization_algorithm, classification_algorithm, name_of_model

    The program trains the classifier on the given training set with the specified parameterization and classification algorithms. Then, it evaluates the classification accuracy and saves the trained model for the later use (e.g., with a GUI) into a file.

  - execution with one parameter = model_name

    The program is executed with a simple GUI and the previously saved classification model. The program allows to classify documents written from the keyboard (or copied from the clipboard).
- Evaluate the quality of the classifier on the given data, use the accuracy metric. Test all combination representation / classification algorithm (6 results).
- There are no programming language constraints.

Bonuses:

- Use some existing classification tool (e.g. Weka or scikit-learn) and compare your results with the results of this classifier.
- Realize unsupervised learning (e.g. algorithm k-means) and the results compare with supervised classification.