Applied Artificial Intelligence:   
Report for Assignment 2

## Overview

We implemented a series of different classifiers in Visual C#.

If you run the application, on top of the window you can choose which classifiers shall be used. Beneath those options, there is a form to select two files that contain the training texts. The "Train" button trains the selected classifiers with the given data. The "Cross validate" button cross-validates the selected classifier on the given data, this is independent of the training. Underneath this form, there is the possibility to load a sample and have the classifier classify it. This, however, requires the classifiers to be trained beforehand.

## General Approach

Our general approach was that we create a set of classifiers that observe different features of the text. When classifying a sample, each classifier shall output a real number from the interval [-1, 1] whereas if a classifier outputs -1, it thinks that the sample belongs to the first text, and if it outputs 1, it thinks the sample belongs to the second text. We use the interval to be able to express uncertainties of the classifiers about a problem. This is important because then we combine the results of the classifiers to get a relatively reliable classification. Additionally, we weight the classifiers depending on the training data, to use primarily these features which are useful to characterize the texts.

## Classifiers

### Naive Bayes

At first, we investigated which classifiers for texts exist and found out, that the Naive Bayes classifier is useful for our purpose. It tries to find the best-fitting topic of a text based on word frequencies. If we assume, that the texts at hand have different topics, we can use Naive Bayes to find out the topic of a sample and then conclude the text it fits to. We implemented this together with Laplace smoothing to deal with words from the sample that not occur in the training texts.

We then tested this algorithm and found out, that it does not perform to well when the texts differ in size to much. We could tackle this problem by filtering out stop words and an arithmetic adaption to the Laplace smooting.

### Word Length and Sentence Length

We reasoned that the average word length and the average sentence length might characterize a text, so we implemented two classifiers for that. To enhance this approach a bit, we assumed that the word/sentence lengths follow a Normal Distribution. Having this assumption, we compute mean and variance of the lengths of the training sets. When classifying a sample, we then calculate the joint probabilities of all sentence/word lengths of the given sample for the Normal Distributions from the training data, to reason to which text the sample belongs.

### Exclamation Marks, Question Marks, and Apostrophes

The usage of questions, commands, and direct speech might be more frequent in a prosaic text than in a scientific text. So we count the frequency of occurrences of question marks (per sentence), exclamation marks (per sentence), and apostrophes (per word). This time, instead of employing the Normal Distribution, we simply collect the same features from the sample and compare them to the features collected from the training texts.

### Boosting

To combine the presented classifiers, we came up with our own Boosting classifier. It trains the elementary classifiers and then uses the single sections of the training texts as test samples to find out how good the classifiers perform. Then we assign weights to them based on the amount of errors that they make. When we classify a sample, we let at first the elementary classifiers classify it and then simply compute the weighted sum of their results to come up with an overall result.

# Validation

To validate our classifiers, we made use of the *leave-one-out cross-fold validation.* That means, we remove a section from the training data, train the classifiers with the remaining sections and then check if the removed section is classified correctly. We repeat this for every section in the training data.