Text classification

In this exercise you will implement a text classifier based on the Perceptron algorithm. You will use the version of Perceptron adapted for multiway classification. Let $C$ be the set of classes between which we wish to distinguish, let $N$ be the number of training samples, and let $d$ be the dimension of our samples.

Classification algorithm

A multiway perceptron keeps a set of $C$ weight vectors $w_1 \ldots w_C$. The vector $w_c$ represents the centroid of class $c$. Classification works as follows: Given a sample $x$,

- compute the scalar product $\langle x, w_c \rangle = \sum_{i=1}^{d} x_i w_{ci}$ between $x$ and each of the class vectors $w_c$,
- order these scores, and assign the highest scoring class to $x$.

Training algorithm

Let $x^{(n)}$ be the n-th training sample, and $c^{(n)}$ its true class. Training the weights $w_c$ works as follows:

1: Input $I$: number of iterations
2: for $c = 1..C$ do
3: Initialize $w_c := 0$
4: for $i = 1..I$ do
5: for $n = 1...N$ do
6: $\hat{c}^{(n)} := \text{argmax}_c \langle w_c, x^{(n)} \rangle$
7: if $\hat{c}^{(n)} \neq c^{(n)}$ then
8: $w_{c^{(n)}} := w_{c^{(n)}} + x^{(n)}$
9: $w_{\hat{c}^{(n)}} := w_{\hat{c}^{(n)}} - x^{(n)}$
10: return $w$

The perceptron classifies each training sample according to the current class weights. If its classification is erroneous, it updates the weight vectors of the incorrectly guessed label and the correct label. This is repeated $I$ times.

Note: When training, you should order the training data randomly before processing!

1 Data and features

You can download the full dataset from [the web page][1]

1. Implement the classification and training algorithms.

2. Train your perceptron on the training portion of the data set. Vary the following settings:

- **Features**: Use the top 100 (binary word presence/absence) features according to your feature ranking from the previous exercise, i.e. the top features for $IG$ and for $\chi^2$ separately. Alternatively, you can use the list of features on the web page[2]

---

- **Number of iterations in training**: Use 1, 5, and 10 iterations for training.

3. Report the classification accuracy on the test portion of the dataset for the different settings.

4. Plot the length of the weight vectors as a function of the iteration number. Do the weight vectors stabilize? If so, after how many iterations?

5. What influence does the order of the training samples have?

Your submission should include a brief report on your solution, as well as the source code you wrote to solve the exercise. If possible, avoid using nonstandard libraries not included in the distribution of the programming language you used. Make sure to include brief instruction on how to run your code.

Send your solutions to [arif.khan@lsv.uni-saarland.de](mailto:arif.khan@lsv.uni-saarland.de) (if you are attending the Wednesday tutorial) or [andrea.fischer@lsv.uni-saarland.de](mailto:andrea.fischer@lsv.uni-saarland.de) (if you are attending the Tuesday tutorial) by **Friday, 14 June**.

**Important**: Please use PDF as a document format. If you need to compress files, use ZIP or GZIP.