Machine Learning 2013: Project 3 - Text Classification Report

fregli@student.ethz.ch
ganzm@student.ethz.ch
sandrofe@student.ethz.ch

December 19, 2013

Experimental Protocol

The project consists of two distinct steps. Preprocessing and transforming the input data and as a second part learning and predicting the results.

Extract the raw data set and place the csv files *training.csv*, *testing.csv*, *vaildation.csv* into a subfolder of the matlab source directory called *data*. Relative to the source folder one should finde the following path *./data/testing.cvs*.

Perform Matlab preprocessing by running the files *first.m* and *learn.m* in sequential order. This generates the files *X-val.csv*, *X-test.csv*, *X.csv* which will be used in the next step.

Perform the learning and prediction step by executing the python script *run.py* .

1 Tools

As tools we used Matlab for the preprocessing. Some of the team members had to install the string toolbox to have it running. The machine learning part of the project was performed with Python and scikit-learn.

2 Algorithm

2.1 Preprocesing

Create Bag of words with frequencies In a first step all words occurring in all city names are put into a bag of words and ordered according to their frequency in which they appear.

Fix Typos In this step we try to fix typos. Starting from the word which occurs the least we try to assign it to another word which is observed more often. Two words are assigned to each other when

they are close to each other. As a proximity measure the Levenshtein distance with a linear threshold is used. A linear threshold means that longer words are allowed to have more typos than shorter and can still be considered to be close to each other.

Having assigned low frequency words to others the bag of words with frequencies is calculated again. The new bag of words can now be seen as having eliminated typos.

Create Stem list By the time we arrive at this step the number of features has already drastically decreased (about 1000 distinct words are left). We now try to find words which have the same stem (like *working*,*worker*). Each word is now compared with each other by counting the number of subsequent characters which are equal (starting from the left of the word). Exceeds the number of equal characters divided by the length of the word a certain threshold both words are assumed to have the same stem (*working* and *worker* become *work*). Having words replaced and merged with their stem words the bag of word is generated again.

Boolean matrix Having the final version of the bag of words we now create a boolean matrix from the training, test and validation set. It may look like this.

yrjhnjcnfy	erwtonm	uhl	blub	...	city code	country code
1	1	0	0		129771	196
0	0	0	1		819100	458
...						

We then reduce the number of features further to about 200 by removing the word with a low total of occurrences.

2.2 Learn and Predict

The resulting data of the preprocessing step is trained and estimated using the python one versus all linear SVM algorithm.

3 Features

Feature preprocessing is described in 2.1

4 Parameters

The several parameters used in the preprocessing step were determined by looking at the result and by try and error.

Parameters required in the learning phase were obtained using crossvalidation.

5 Lessons Learned

Do not use Matlab for string processing. It will work out but it is not convenient at all.