Learning from Data – Week 3
Assignment 3: Support Vector Machines and K-Means

General remarks

This assignment is meant to get your acquainted with Support Vector Machines as well as the K-Means clustering algorithm. You will also have to explore feature contribution quite a bit further. Additionally, you're encouraged to experiment a little further with report writing, by making your own choices (how do you want to report on the algorithms/approaches? All in one method section? Or do you want this to be a point in determining the structure of your paper. Experiment!)

For the practical parts, you are asked to train and test a few models using SVM, and produce what you think the best settings are for the data we're using. Also, you will have to perform some clustering experiments using KMeans.

What you have to hand in:

- code with your SVM model (see Exercise 3.1). You have to assume that we will run it like this:
  `LFDassignment3_SVM_yourname.py <trainset> <testset>`
  where trainset and testset have the same format. You are given the trainset, but we're holding out the test set. Please, make sure that your script is taking arguments.
- code with your clustering models (see Exercise 3.2), please call it `LFDassignment3_KM_yourname.py`.
- optional: code with your extra clustering assignment (see Exercise 3.4), please call it `LFDassignment3_KMextra_yourname.py`.
- research report (based on the template) that describes what you've done and also includes answers/discussion to all questions in this document. Please, make sure you submit a `pdf` file.

Deadline: 7 December 2016, 11:00pm.

Data

For this assignment, we will be using the review data we used for Assignment 1 and Assignment 2, and which you find on Nestor in this assignment's materials. We will be using both the sentiment as well as the topic labels.

Additionally, and optionally (see Exercise 3.4), you will use yet another dataset which contains sets of documents written by different authors, all on the same topic. This dataset is a subset of the Reuters Corpus Volume 1 (RCV1, `http://www.daviddlewis.com/resources/testcollections/rcv1/`), and was created by Zhi Liu (National Engineering Research Center for E-Learning, Hubei Wuhan, China). The top 50 authors (with respect to total size of articles) were selected, and for each of them 50 articles were picked that have at least one subtopic of the class CCAT(corporate/industrial). This is a way to minimise the topic factor in distinguishing among the texts. The training corpus consists of 2,500 texts (50 per author); the test corpus also includes 2,500 texts (50 per author), non-overlapping with the training texts.

Exercise 3.1 – Support Vector Machines

You will be working on the binary sentiment classification with the dataset from Assignment 2, which is uploaded on Nestor again for this assignment. We are again withholding the test set, and we will evaluate your model on it.

3.1.1 Default settings

Run a support vector machine with a linear kernel (`cls = svm.SVC(kernel='linear', C=1.0)`) on the binary sentiment classification. Use default settings and report results using either cross-validation or a portion of the dataset as development set. Just make sure you document what you have done.

3.1.2 Setting C

Try to change the value of the C parameter, and see if you can get better results (either with x-validation or on a development set, just use the same settings as above). Please, in the report include details of what you observe by changing C, and also explain what the C parameter is used for.

3.1.3 Using a non-linear kernel

Do the same as above, but use a radial basis function (rbf) kernel. You will also have to specify a *gamma* parameter, in addition to C (e.g. `cls = svm.SVC(kernel='rbf', gamma=0.7, C=1.0)`) You can check information on the *gamma* (and C) parameter here: `scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html`. Experiment with changing these values, and report what you observe. It has been claimed that rbf kernels are not great for text classification, and are normally outperformed by linear kernels — do you observe the same?

3.1.4 Best SVM model

Based on your observations and results on x-validated data or on your development set, make a final decision on your kernel and parameters, and report it in the document. Send in your best SVM model, like you did for Assignment 2, calling it:
`LFDassignment3_SVM_yourname.py`. You are expected to experiment with adding features, too. Possibly n-grams might help? Or part-of-speech information? You have to build larger, more informed systems, so this is a good place to start experimenting. We will run it on held-out data, and send back the results to you. Please, make sure that your script takes arguments – it will be run like this:
`LFDassignment3_SVM_yourname.py <trainset> <testset>`. Note that if you also work on Exercise 3.3, you can include the clustering-based features directly in this script, without having to submit an additional one.

Exercise 3.2 – K-Means

3.2.1 Six way classification

Run K-Means on the dataset we used for Assignment 1, which is uploaded again on Nestor for this assignment. We will be using the six-way topic information for this portion of the exercise, so that you can set K to 6: `km = KMeans(n_clusters=6, n_init=1, verbose=1)`.
Note that `n_init` sets the number of times that the algorithm is run on different centroids as seeds. In this example it only does it once, while default is 10. You can try change this, and observe what happens.

To evaluate the clustering you will use two measures: Rand Index (`adjusted_rand_score`), and something similar to F-measure. Scikit-learn implements an entropy-based measure that is is designed to be analogous to F-measure, in that it is defined as the weighted harmonic mean of two values, homogeneity (h, the precision analogue) and completeness (c, the recall analogue). This measure is called V-measure, and can be calculated as $v = 2*(homogeneity * completeness)/(homogeneity + completeness)$.
In scikit-learn: `v_measure_score(Ytest,Yguess)`, or, to obtain all three:
`homogeneity_completeness_v_measure(Ytest,Yguess)`. As in F-measure, you can assign more or less weight to precision/homogeneity ($\beta < 1$) or recall/completeness ($\beta > 1$). However, this isn't implemented in scikit-learn and you will just use the standard settings. You can import all measures from `sklearn.metrics.cluster`.

The cluster assignments are stored in `km.labels_`, and you can retrieve them from there so that you can compare the obtained scores against the gold labels. Please, comment on the scores you obtain.

3.2.2 Binary classification and feature selection

For this portion of the exercise, we will use again the dataset from Assignment 1, also used in the previous exercise on clustering. From the whole dataset select reviews belonging to just two classes — you can choose any two classes (e.g. `music` and `health`, or `dvd` and `books` or whatever you like; you can even experiment with different pairs, but you aren't required to do it). The distribution of sentiment across classes is evenly distributed, so you can be guaranteed that whichever two classes you're picking, you are also preserving approximately the same number of positive and negative reviews.

Run K-Means and see what happens. Do you get better scores when the gold labels are the sentiment labels or the topic labels?

In class we mentioned that you can 'steer' clustering by choosing what features should be used. What you have to do is play with features in such a way that keeping $K = 2$, you can generate two different clustering models. One should score well when compared to the topic classification labels, and the other one should score well when compared to the sentiment classification labels. Please, in the report explain what you have done, what you have observed, and the results you obtain.

Exercise 3.3 – Extra I

You don't have to do this portion of this exercise, but you can if you wish. In Exercise 3.1.4 you are asked to produce your best SVM model, playing with parameter tuning, but also including additional features. Do you think that the information you get from clustering might be a helpful feature? You can try experiment with including it in your model. Please note that if you also work on this exercise, you can include the clustering-based features directly in your best SVM model script, without having to submit an additional one.

Exercise 3.4 – Extra II

You don't have to do this portion of this exercise, but you can if you wish. For this extra, optional, assignment you will run a K-means clustering algorithm on the additional dataset (`C50.zip`). Given that you know that you have 50 different authors, you can set $K = 50$. What you are asked to do is to run the clustering using a bag-of-words approach as well as (or in combination with) word and character n-grams. You are asked to report on what you observe in terms of performance. What you learn from this experiment is to work with a larger number of clusters, and, mostly, with a different set of features, such as character n-grams. Please note that you are given both a training and a testset because you can use this data also in a supervised classification setting. You can tune your clustering on the trainset so as to determine which features work best, and then run it on the testset.

If you run this experiment, you can include its description however you want. You can either split the whole experimental part in Experiment 1 (Data, Method, Results) and Experiment 2 (Data, Method, Results), or include everything in one Data section, or divide up the experiments according to the approach used, namely SVM vs Clustering. Use some creativity, and see what structure fits best with what you did and what you found.