Part 1: Text Classification

The content has been gathered from the popular academic website arXiv.org for articles tagged as computer science content (though some of these are in mathematics or physics categories). Training is 1990-2014 and testing is 2015 plus a bit of 2016. The fields are:

- *ID*: a unique alphanumeric ID.
- *URL*: a working URL if you prepend "http://".
- *Date*: the date in format YYYY-MM-DD.
- *Title*: the full title, though with non-ASCII characters modified and any "," deleted.
- *InfoTheory*: a "1" if it is classified as an Information Theorey article, otherwise "0".
- *CompVis*: a "1" if it is classified as a Computer Vision article, otherwise "0".
- *Math*: a "1" if it is classified as a Mathematics article as well, otherwise "0".
- *Abstract*: the full abstract, though with non-ASCII characters modified.

The *URL*, *Date* and *Title* are provided but not required in the assignment.

The three classes are *InfoTheory*, *CompVis* and *Math*. These can occur in any combination, so an article could be all three at once, two, one or none. Your job is to build text classifiers that predict each of these three classes individually using the *Abstract* field. The URL field is provided just in case you want to see the actual paper.

You should train different text classifiers using different configurations for the three Boolean prediction tasks. The variations we would like to consider are:

1. **Task:** 3 Boolean tasks
2. **Algorithm**: use 2 different algorithms from tutorials, use the RNN and then choose one of the statistical classifiers (SVN, logistic regressions, etc.); you may use another classifier but it should be readily available in Python, and it is not expected
3. **Text preprocessing**: do 2 different versions of your choosing (for instance one version might be snowball stemming, no stopwords, delete numbers, convert all letters to lowercase, etc.)
4. **Data size:** train on the first 1000 cases in the training set only, then train on all in the training set.

So this makes 3 by 2 by 2 by 2 different configurations. For each configuration test the algorithm on the test set provided and report the following results in your notebook

- F1, precision, recall
- precision-recall curve

being creative about how you assemble the different values and plot the curves. The discussion of these results should be in its own 2 page discussion section in the PDF report. How well did the two algorithms work, when and why? Which text pre-processing worked better, when and why? What insights do the various metrics and plots give you?

Part 2: Topic Modelling

The data used is the training data from Part 1. Your job is to perform appropriate text pre-processing and preparation and then design two different variations for running LDA using the `gensim.models.LdaModel()` function call and pre-processing steps such as given in the tutorial. Select appropriate choice of pre-processing and parameters to develop model outputs that are informative. Choices you might make in differentiating the two variations are:

- different pre-processing of text or vocabulary
- use of bi-grams or not
- different numbers of topics (e.g., K=10, K=40)

Now run these two on the first 1000 and the first 20,000 articles in the training data set. This means there are 2 by 2 different configurations for the LDA runs. Then make visualisations of some kind in the notebook. These should allow you to analyse and interpret the output of the topic models.

The actual discussion (analysis and interpretation) about the results should not appear in the notebook but be in the separate PDF discussion report. This is a 2 page discussion giving your analysis and findings that were presented in the notebook output. What sorts of topics do you see? Are all top topic words comprehensible sets of words? Perhaps find some articles that are exemplers and use them to illustrate key topics (but don't insert full articles in your report, not enough room, just extract a few lines or the title). Your analysis should serve three purposes:

1) to present what sorts of groupings there are about articles, and

2) to describe how the topic modelling presents this and any advantages or shortcomings of topic modelling for the role in 1), and

3) to explain how your two configurations and data set sizes (1000, 20000) compare.

This is a knowledge discovery task rather than a predictive task, so marks will be included for your ability to make novel findings from the topic models.

Initial Submission by the end of week 5

This initial submission is a formative assessment, meaning it is not marked and is only provided so you can obtain feedback on your plans for the final submission. This submission has no code. This submission is a 2 page PDF report outlining your experimental plans for the two parts, one page for each part. What configurations and pre-processing will you do? What sorts of results do you expect to see? How will you display and report the basic results? No code or analysis is needed at this stage.

Final Submission by the due date

All Python code must be included in a single Jupyter notebook that must be submitted. This should have clear headings "Part 1: Text Classification" then followed by "Part 2: Topic Modelling". It should have the students name and ID embedded in the first comment (in markdown). The name of the file should be "code_012345678.ipynb" where "012345678" is replaced by your own student ID. An example/skeleton notebook file "code_012345678.ipynb" with appropriate headings is included with the datasets. To complete the submission, use the export option on the notebook system and export to PDF. Save this as "code_012345678.pdf"