FIT5212 - Assignment 1

Marks	Worth 50 marks, and 25% of all marks for the unit
Due Date	Due Week 7 – Lecture Date at 23:55pm
Extension	An extension could be granted under some circumstances. A special consideration application form must be submitted. Please refer to the university webpage on special consideration.
Lateness	For all assessment items handed in after the official due date, and without an agreed extension, a 10% penalty applies to the student's mark for each day after the due date (including weekends) for up to 10 days. Assessment items handed in after 10 days without special consideration will not be considered.
Authorship	This is an individual assessment. All work must be your own. All submissions will be placed through Turnitin. This makes plagiarism remarkably easy to identify for us.
Submission	Submission is 3 files: one PDF discussion report, and one Jupyter notebook with a PDF print of it. The three files must be submitted via Moodle. All files will go through Turnitin for plagiarism detection.
Programming language	Python in Jupyter

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). The fields are:

- *Title*: the full title
- *Abstract*: the full abstract
- *InformationTheory*: a "1" if it is classified as an Information Theory article, otherwise "0".
- *ComputerVision*: a "1" if it is classified as a Computer Vision article, otherwise "0".
- *ComputationalLinguistics*: a "1" if it is classified as a Computational Linguistics article, otherwise "0".

The three classes are *ComputationalLinguistics*, *InformationTheory* and *ComputerVision*. 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. Then repeat the same experiment using only the *Titles*. You should train different text classifiers using different configurations for the three binary prediction tasks. The variations we would like to consider are:

1. **Task:** 3 binary classification tasks
2. **Input:** use *Abstract*, and *Titles* alone (separate configurations)
3. **Algorithm**: use 2 different algorithms from tutorials, use the RNN and then choose one of the statistical classifiers (logistic regressions, SVM, etc.); you may use another classifier, but it should be readily available in Python
4. **Text preprocessing**: do 2 different versions of your choosing (for instance one version might be stemming, no stopwords, delete numbers, convert all letters to lowercase, etc.)
5. **Data size:** train on the first 1000 cases in the training set only, then train on all in the training set. For both cases, exclude the last 10% of your training data and use it as your validation/development set (not to be used during the training data).

So this makes 2 (i.e., abstract vs. title) by 3 (i.e., 3 binary classifiers) by 2 (i.e., 2 algorithms) by 2 (i.e., 2 preprocessing) by 2 (i.e., 2 training sizes) different configurations.

For each configuration test the algorithm on the test set (Note: when testing for the model trained on the Abstracts, you should use only the Abstracts of test set. Similarly for testing for the model trained on the Titles, you should use only the Titles of test set.) provided and report the following results in your notebook

- F1, precision, recall, accuracy
- 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? How the model trained on title compared with the one

trained on the abstracts? 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 exemplars 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.

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"

The notebook should:

- be run on either Google Colab or your own Jupyter Notebook
- have any special or unusual libraries indicated at the top of the file in commented out command form; they must be able to be installed from the standard Python repository,

    - e.g., "# !pip3 install gensim"
- assume the two datasets supplied exist in the current directory
- have been run successfully to completion prior to submission, so the results are all embedded in the notebook

The PDF file matching the notebook should print the last version of the notebook submitted.

All discussion and analysis must be written up in a single separate PDF file. This PDF report should have two discussion sections, "Part 1: Text Classification" and "Part 2: Topic Modelling", each being two pages long. It is expected these will refer to plots and tables in the separate notebook. The name of the file must be "report_012345678.pdf" where "012345678" is replaced by your own student ID. The pages should be A4 size with standard margins and 11 point font.

Therefore, three files are to be submitted, "code_012345678.ipynb", "code_012345678.pdf" and "report_012345678.pdf" where "012345678" is replaced by your own student ID.