FIT5212 TP5 in 2021: Assignment 1

Marks	**Worth 50 marks**, and **30% of all marks for the unit**
Due date	Sunday September 19th, 11:55PM Melbourne time
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. The maximum extension the CE can give is 2 days.
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 (25 marks)

You will train some text classification on a set of 25,000 news articles from ‘BuzzFeed’, ‘Upworthy’, ‘ViralNova’, ‘Thatscoop’, ‘Scoopwhoop’ , ‘ViralStories’. ‘WikiNews’, ’New York Times’, ‘The Guardian’, and ‘The Hindu’. Your task is to use the article's headlines to determine whether or not a headline is a clickbait. The fields are:

- ID: a unique alphanumeric ID.
- text: the full article headline with utf-8 encoding
- class: a "1" if it is classified as a clickbait article, otherwise "0".

Your job is to build a text classifier that predicts the class of each headline using the text field. You should train different text classifiers using different configurations for this Boolean prediction task. The variations we would like to consider are:

1. **Task:** one Boolean task
2. **Algorithm:** use 3 different algorithms from tutorials, for one of which you must use the RNN and then you can choose two of the statistical classifiers (SVN, logistic regressions, RF, 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 different configurations. For each configuration, tune the model’s hyperparameters using the 10-fold cross-validation technique and test the algorithm on the test set provided and report the following results in your notebook

- F1, precision, recall
- precision-recall curve (only for the six full model (i.e., full train set))

being creative about how you assemble the different values and plot the curves. The discussion of these results should be in its own discussion section in the PDF report with minimum of 2 pages and 7 page maximum excluding the references. How well did the three algorithms work, when and why? Which text pre-processing worked better, when and why? What insights do the various metrics and plots give you? What was the effect of the train set size? Did it matter or not? Why?

Bonus: Kaggle Competition (0 mark)

We provide a Kaggle competition for this task in which you can test the performance of your best classifier and compare it with your peers. To submit your prediction, you need to use the **kaggle_testset** data set provided in Moodle and submit your predictions here in Kaggle. More information can be found on the competition page here. Please note that the maximum daily submission is set to 2, so use them wisely!

Please note that we only use this in-class competition as a place to compare your models with each other. Your performance in the competition has no effect on your mark and the final results solely rely on your Moodle submissions.

Part 2: Topic Modelling (25 mark)

The data used is the training data from Part 1. Your job is to perform appropriate text preprocessing 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. For the pre-processing step, you can use the same pre-processing steps as you used in Part 1.

Select appropriate choice of pre-processing and parameters to develop model outputs that are informative. Choices you should make in differentiating the four variations are:

- two different preprocessing of text or vocabulary,
- and two different numbers of topics (e.g., K=5, K=15).

Now run these four configurations full-size articles in the training data set (**remember that this is a clustering task hence the 'class' column should be excluded from your data**). This means there are 4 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 to 7 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. 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 four configurations 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.

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" 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 2 to 7 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 or similar.

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.

Good luck!