MiniProject 2: Classification of Textual Data

COMP 551 (001/002), Winter 2020, McGill University

General information

- **Due on March 12th at 11:59pm**. Can be submitted late until March 17th at 11:59pm with a 20% penalty.
- To be completed in groups of three, all members of a group will receive the same grade. You can have the same groups as the first miniproject but feel free to reorganize if needed.
- To be submitted through MyCourses as a group. You need to re-register your group on MyCourses and any group member can submit the deliverables, which are the following two files:
  1. **code.zip**: Your data processing, classification and evaluation code (.py and .ipynb files).
  2. **writeup.pdf**: Your (max 5-page) project write-up as a pdf (details below).
- Except where explicitly noted, you are free to use any Python library or utility for this project.
- Main TA: Yanlin Zhang, yanlin.zhang2@mail.mcgill.ca

Problem definition

In this mini-project, we will develop models to classify textual data, input is text documents, and output is categorical variable (class labels).

Datasets

Use the following datasets in your experiments.

- 20 news group dataset. Use the default train subset (subset='train', and remove=(['headers', 'footers', 'quotes']) in sklearn.datasets) to train the models and report the final performance on the test subset. Note: you need to start with the text data and convert text to feature vectors. Please refer to `https://scikit-learn.org/stable/tutorial/text_analytics/working_with_text_data.html` for a tutorial on the steps needed for this.
- IMDB Reviews: `http://ai.stanford.edu/~amaas/data/sentiment/` Here, you need to use only reviews in the train folder for training and report the performance from the test folder. You need to work with the text documents to build your own features and ignore the pre-formatted feature files.

Models

Apply and compare the performance of following models:

- Logistic regression: sklearn.linear_model.LogisticRegression
- Decision trees: sklearn.tree.DecisionTreeClassifier

- Support vector machines: sklearn.svm.LinearSVC
- Ada boost: sklearn.ensemble.AdaBoostClassifier
- Random forest: sklearn.ensemble.RandomForestClassifier

You are welcome and encouraged to try any other model covered in the class, and you are free to implement them yourself or use any Python library that has their implementation, e.g. the above links from the SciKit learn package. You need to still understand what is the exact model being used. You are also free to use any Python libraries you like to extract features and preprocess the data, and to tune the hyper-parameters.

Validation

Develop a model validation pipeline (e.g., using k-fold cross validation or a held-out validation set) and study the effect of different hyperparamters or design choices. In a single table, compare and report the performance of the above mentioned models (with their best hyperparameters), and mark the winner for each dataset and overall.

Write-up instructions

Project write-up is a five pages PDF document (single-spaced, 10pt font or larger; extra pages allowed for references and appendices). Using LaTeX is recommended, and overleaf is suggested for easy collaboration. *You are free to structure the report how you see fit*; below are general guidelines and recommendations, *but this is only a suggested structure and you may deviate from it as you see fit.*

- Abstract (100-250 words) Summarize the project task and your most important findings.
- Introduction (5+ sentences) Summarize the project task, the dataset, and your most important findings. This should be similar to the abstract but more detailed.
- Related work (4+ sentences) Summarize previous literature related to the multi-class classification problem and text classification.
- Dataset and setup (3+ sentences) Very briefly describe the dataset and explain how you extracted features and other data pre-processing methods that are common to all your approaches (e.g., tokenizing).
- Proposed approach (7+ sentences) Briefly describe the different models you implemented/compared and the features you designed, providing citations as necessary. *If you use or build upon an existing model based on previously published work, it is essential that you properly cite and acknowledge this previous work.* Discuss algorithm selection and implementation. Include any decisions about training/validation split, regularization strategies, any optimization tricks, setting hyper-parameters, etc. It is not necessary to provide detailed derivations for the models you use, but you should provide at least few sentences of background (and motivation) for each model.
- Results (7+ sentences, possibly with figures or tables) Provide results on the different models you implemented (e.g., accuracy on the validation set, runtimes). .
- Discussion and Conclusion (3+ sentences) Summarize the key takeaways from the project and possibly directions for future investigation.
- Statement of Contributions (1-3 sentences) State the breakdown of the workload.

Evaluation

The mini-project is out of 100 points, and the evaluation breakdown is as follows:

- Completeness (20 points)

  - Did you submit all the materials?
  - Did you run all the required experiments?
  - Did you follow the guidelines for the project write-up?
- Correctness (40 points)
  - Are your models implemented and applied correctly?
  - Are your reported accuracies close to the reference solutions?
  - Do your proposed features actually improve performance, or do you adequately demonstrate that it was not possible to improve performance?
  - Do you observe the correct trends in the experiments?
- Writing quality (25 points)
  - Is your report clear and free of grammatical errors and typos?
  - Did you go beyond the bare minimum requirements for the write-up (e.g., by including a discussion of related work in the introduction)?
  - Do you effectively present numerical results (e.g., via tables or figures)?
- Originality / creativity (15 points)
  - Did you go beyond the bare minimum requirements for the experiments? For example, you could investigate different hyperparameters, different feature extractions, combining different models, etc. to achieve better performance. *Note:* Simply adding in a random new experiment will not guarantee a high grade on this section! You should be thoughtful and organized in your report.
- Best performance bonus (15 points): the top performing group(s if there is ties) will receive a bonus of 15 points, conditioned that they are better than the TA's baseline. This is based on the performance on the test set provided with the data, which is not supposed to be used during the training or tuning of your models. Any choice of hyper-parameters should be explained to make sure information in the test set is not being used.
- Bad practice penalty (-30 points): if you have information leak from test set to the training procedure, you will be penalized by 30 points. Make sure to not touch the test set, until reporting final results.