COMP 4900B Introduction to Machine Learning
Homework 2

Winter 2020
School of Computer Science, Carleton University

This mini-project is to be completed in groups of two. You will submit your assignment on CuLearn as a group. You must register your group on CuLearn.

Introduction

This assignment will give you experience with working with text data, sentiment analysis, and machine learning with scikit-learn. You will implement methods for classifying movie reviews to either positive or negative. Download the training and test data from the link provided on the course homepage in CuLearn. Each line in `train.csv` includes two fields: text of a review, and its sentiment. Use `train.csv` for training your models.

Hint 1: you may use `CountVectorizer()` with `binary=False` to obtain a binary vector representation for a text.

Hint 2: when debugging your code, you might start off by using only a quarter of train samples and a quarter test samples. When you are done with code debugging, apply your code to the whole train and test data.

Your tasks

- Part 1: using the data provided in `train.csv`
    1. Implement Bernoulli Naive Bayes from scratch. Since your implementation may not be optimized for speed, you can set the `max_features=5000` in `CountVectorizer()`. This will consider only the top 5000 words i.e. `m=5000`.
    2. You need to do additional experiments using at least one additional classifiers from the SciKit learn package (excluding Bernoulli Naive Bayes). For example Logistic Regression, Decision Tree, etc.
    3. Implement a model validation (e.g. using k-fold cross-validation) and use it to report, in your write-up, the performance of the above mentioned classification models (i.e., your Naive Bayes model and the SciKit learn model(s)). You can use KFold from SciKit learn.
- Part 2: run your best model (obtained in Part 1) on the data provided in `test.csv` and submit the result on Kaggle competition. (See below for more details)
- Prepare and submit your report (max 5 pages) along with your codes (details below).

Kaggle competition

`test.csv` contains the test samples. Each row in `test.csv` contains review id and the text of the review. You need to make a prediction for every test sample and submit a `.csv` file in Kaggle where each line contains review id and its predicted label which is either positive or negative. `ExampleSubmissionRandom` is an example of the kind of csv file you should submit in Kaggle. (Note: in the example file, the sentiment written in front of each id is just a random value and you should replace them with your predicted values). You are limited to two submissions per day. You must register in the Kaggle competition using your `carleton` email address. If you already have a Kaggle account under a different email address, do not delete your account. You only need to change your email in your Kaggle profile. **You must form a team in the competition page on Kaggle, the name of your team must be the same as the name of your group on CuLearn (e.g. Team Name: Group 2-10).** Except where explicitly noted, you are free to use any Python library for this project. You are not allowed to use any training data other than what is provided for the competition.

Report

We are flexible on how you report your results, but you must adhere to the following structure:

Abstract (100-250 words): provide a summary of the project task and highlight your most important findings.

Introduction (at least one paragraph): Summarize the project task, the dataset and important findings. This is similar to the abstract but you should provide more details.

Datasets (at least one paragraph): Briefly describe the dataset. Also, describe the preprocessing steps for preparing the feature vectors. Note: You do not need to explicitly verify that the data satisfies the i.i.d. assumption (or any of the other formal assumptions for linear classification).

Proposed approach: Briefly describe the features you designed and the methods you have implemented or used for this project. No need to provide detailed derivations and proofs but you should provide some (a few sentences) background, description and motivation for each model. You should properly cite and acknowledge previous works/publications that you use or build upon. Discuss any decision about training/validation splits, algorithm selection, regularization, hyper-parameters, etc.

Results: Summarize your results using tables and/or figures. Discuss the results for each model (for example accuracy, runtime, etc.). Since you do not have the labels for the test set, your results should be based on your validation set(s). Report your test set leaderboard accuracy too.

Discussion and conclusion: Discuss and summarize the key takeaways from the project and possible directions for future investigation.

Statement of contributions: (1-3 sentences) State the breakdown of the workload across the team members.

You are expected to discuss your findings with scientific rigour.

The report is limited to 5 pages (single-spaced, minimum font size of 11 and 1 inch minimum margin each side). Imagine you are writing a paper for a conference. We highly recommend to use LaTeX for preparing your report.

Appendix To facilitate the grading process, attach the codes for your implementation to the end of your report. This does **not** count towards the page limit of the report. You are still required to include your implementation of Bernoulli Naive Bayes along with the rest of your codes in `codes.zip`.

Deliverables

- `report.pdf`: Your report as a single pdf file.
- `code.zip`: You codes (e.g. .py, .ipynb, etc.) which must work with Python 3.6 in Colab. Include a `readme` file and provide instruction for TA on how to replicate your results on Colab. All the results including your leaderboard submission must be reproducible in Colab using the submitted `code.zip`. Points will be deducted if we have a hard time reading or understanding the structure of your code. Do not include report.pdf in the code.zip file.

Evaluation

This is an open-ended project. Feel free to go beyond the minimal requirements. The evaluation has two parts each worth 50 points out of 100.

Performance (50 points): This is based on the performance of your best model on the held-out test set on the Kaggle competition. Your grade will be computed based on a linear interpolation between three points: the 2nd top group, a TA baseline and a random baseline. The random baseline is the score needed to get more than 0% on the competition. The TA baseline is the score needed to get 75% on the competition. In other words, if your score is between the random and TA baseline, your grade is a linear interpolation between 0% and 75% on the competition; likewise, if your score is between the TA baseline and the 2nd best group, your grade will be between 75% and 100% on the competition. In addition to the above criteria, the top two groups all receive 100%. Additionally, the top group will receive 10% points as bonus.

Quality of your report and proposed methodology (50 points): Your report should be both thorough and concise. It will be judged based on its scientific quality including but not limited to: Does the report include all the required experiments? Is the report technically sound? How thorough/rigorous are your experiments? Is the report well-organized and coherent? Is the report clear and free of grammatical errors and typos? Does the report contain sufficient and appropriate references and related work?

All members of a group will receive the same mark.

Final remarks

You are expected to display initiative, creativity, scientific rigour and critical thinking skills. You don't need to restrict yourself to the requirements listed above - feel free to go beyond, and explore further.

You can discuss methods and technical issues with members of other teams, but you cannot share any code or data with other teams. Any team found to cheat (e.g. use external information, use resources without proper references) on either the code, predictions or written report will receive a score of 0 for all components of the project.