# MiniProject 2: IMDB Sentiment Analysis

COMP 551, McGill University
Professor: William L. Hamilton

February 5, 2019

**Please read this entire document before beginning the assignment.**

## Preamble

- This mini-project is **due on Feb. 22nd at 11:59pm.** Late work will be automatically subject to a 20% penalty and can be submitted up to 5 days after the deadline. No submissions will accepted after this 5 day period.
- This mini-project is to completed in groups of three. All members of a group will recieve the same grade. It is not expected that all team members will contribute equally to all components. However every team member should make integral contributions to the project.
- You will submit your assignment on MyCourses as a group, and you will also submit to a Kaggle competition. **You must register in the Kaggle competition using the email that you are associated with on MyCourses (i.e., @mail.mcgill.ca for McGill students). You can register for the competition at:** `https://www.kaggle.com/t/b95c2a432a9445d6a01a7a95d51d1dd5`. As with MiniProject 1, you must register your group on MyCourses and any group member can submit. **You must also form teams on Kaggle and you must use your MyCourses group name as your team name on Kaggle. All Kaggle submissions must be associated with a valid team registered on MyCourses.**
- Except where explicitly noted in this specification, you are free to use any Python library or utility for this project. All your code must be compatible with Python 3.

## Background

In this mini-project you will develop models to predict the sentiment of IMBD reviews. IMDB is a popular website and database of movie information and reviews (`https://www.imdb.com/`). The goal is to classify IMBD reviews as positive or negative based on the language they contain. You will be competing with other groups to achieve the best accuracy in a competition. **However, your performance on the competition is only one aspect of your grade. We also ask that you implement a minimum set of models and report on their performance in a write-up.**

# Tasks

You are welcome to try any model you like on this task, and you are free to use any libraries you like to extract features. However, **you must meet the following requirements:**

- You must implement a Bernoulli Naive Bayes model from scratch (i.e., without using any external libraries such as SciKit learn). You are free to use any text preprocessing that you like with this model. **Hint 1:** you many want to use Laplace smoothing with your Bernoulli Naive Bayes model. **Hint 2:** you are allowed to choose the vocabulary for your model (i.e, which words you include vs. ignore), but you should provide justification for the vocabulary you use.
- You must run experiments using at least two different classifiers from the SciKit learn package (which are not Bernoulli Naive Bayes). Possible options are:
  - Logistic regression (`https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html`)
  - Decision trees (`https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html`)
  - Support vector machines [to be introduced in Lecture 10 on Feb. 12th] (`https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html`)
- You must try at least two different feature extraction pipelines for processing the text data (e.g., using binary occurrences vs. tf-idf weighting). You should experiment with these features using one of the SciKit learn models from requirement 2 above, and note that you only need to experiment with these different features for one model (i.e., don't need to try to use different features with the Bernoulli Naive Bayes model).
- You must develop a model validation pipeline (e.g., using k-fold cross validation or a held-out validation set) and report on the performance of the above mentioned model variants.

# Deliverables

You must submit two separate files to MyCourses (**using the exact filenames and file types outlined below**):

1. **code.zip**: A collection of .py, .ipynb, and other supporting code files, which must work with Python version 3. You must include your implementation of Bernoulli Naive Bayes and **it must be possible for the TAs to reproduce all the results in your report and your Kaggle leaderboard submissions using your submitted code.** Please submit a README detailing the packages you used and providing instructions to replicate your results.
2. **writeup.pdf**: Your (max 5-page) project write-up as a pdf (details below).

## Project write-up

Your team must submit a project write-up that is a maximum of five pages (single-spaced, 10pt font or larger; extra pages for references/bibliographical content and appendices can be used). We highly recommend that students use LaTeX to complete their write-ups and use the bibtex feature for citations. **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 sentiment classification problem.

**Dataset and setup (3+ sentences)** Very briefly describe the dataset and any basic data pre-processing methods that are common to all your approaches (e.g., tokenizing). **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 (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). You should report your leaderboard test set accuracy in this section, but most of your results should be on your validation set (or from cross validation).

**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 (10 points)
  - Did you submit all the materials?
  - Did you perform all the required tasks?
  - Did you follow the guidelines for the project writeup?
  - Are your results reproducable?
- Performance (30 points)

  - The performance of your models will be evaluated on the Kaggle competition. Your grade will be computed based on your performance on a **held-out test set**. The grade computation is a linear interpolation between the performance of a random baseline and the top-performing group in the class; **however, Prof. Hamilton's benchmark performance represents a ceiling on this competition: any groups beating his benchmark are guaranteed full-points and the grading will not interpolate based on scores above his benchmark.**
  - Thus, if we let $X$ denote your accuracy on the held-out test set, $B$ denote the accuracy of the random baseline, $T$ denote the accuracy of the top-performing group, and $H$ denote Prof. Hamilton's benchmark accuracy, then the scoring function is given by

$$\text{points} = \frac{50 * (X - B)}{\min(H, T) - B}$$

  - In addition to the above, the top-3 performing groups will receive a bonus of 5 points.

- Quality of write-up and proposed methododology (60 points)
  - Is your proposed methodology technically sound?
  - Does your report clearly explain your models, experimental set-up, results, figures (e.g., don't forget axis labels and captions on figures, don't forget to explain figures in the text).
  - Is your repory well-organized and coherent?
  - 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)?
  - Does your report include adequate citations?

# Final remarks

You are expected to display initiative, creativity, scientific rigour, critical thinking, and good communication 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.

**Rules specific to the Kaggle competition:**

- Don't cheat! You must submit code that can reproduce the numbers of your leaderboard solution.
- The classification challenge is based on a public dataset. You must not attempt to cheat by searching for information about the test set. Submissions with suspicious accuracies and/or predictions will be flagged and your group will receive a 0 if you used external information about the test set at any point.
- Do not make more than one team for your group (e.g., to make more submissions). You will receive a grade of 0 for intentionally creating new groups with the purpose of making more Kaggle submissions.