Assignment 2: Classification of Textual Data

COMP 551 Winter 2024, McGill University
Contact TAs: Alina Tan and Jonathan Colaco Carr

Released on February 5 midnight
Due on February 26 midnight

Please read this entire document before beginning the assignment.

Preamble

- This assignment is **due on February 26th at 11:59pm (EST, Montreal Time).**
- For late submission, 2^k percent will be deducted per k days of the delay.
- To use your 6-day quota as a team, submit your request by emailing `comp551.socs@mcgill.ca` with subject title "A2 extension request" and in the email body specify the number of days (max at 6 days) you need to submit your assignment. You can only submit the request ONCE. Once you request the days, the quota of every member on the team will be reduced by the days you requested even if you end up submitting your assignment prior to the extended deadline. Therefore, plan and use your quota wisely.
- This assignment is to be completed in groups of three. All members of a group will receive the same grade except when a group member is not responding or contributing to the assignment. If this is the case and there are major conflicts, please reach out to the contact TA or instructor for help and flag this in the submitted report. Please note that it is not expected that all team members will contribute equally. However every team member should make integral contributions to the assignment, be aware of the content of the submission and learn the full solution submitted.
- You will submit your assignment on MyCourses as a group. You must register your group on MyCourses and any group member can submit. See MyCourses or here for details.
- We recommend to use **Overleaf** for writing your report and **Google colab** for coding and running the experiments. The latter also gives access to the required computational resources. Both platforms enable remote collaborations.

- You should use Python for this and all assignments. You are free to use libraries with general utilities, such as matplotlib, numpy and scipy for Python, unless stated otherwise in the description of the task. In particular, in most cases you should implement the models and evaluation functions yourself, which means you should not use pre-existing implementations of the algorithms or functions as found in SciKit learn, and other packages. The description will specify this in a per case basis.

Synopsis

In this assignment, you will **implement logistic regression and multiclass regression** and evaluate these two algorithms against Decision Trees on two distinct textual datasets. The goal is to gain experience implementing these algorithms from scratch and to get hands-on experience evaluating their performances.

1 Task 1: Data preprocessing

Your first task is to turn the text data into tabular format with selected features as the words and the text documents as the training or test examples.

We will use two datasets in this project as described below.

1.1 IMDB Reviews

The IMDB Reviews data can be downloaded from here: `http://ai.stanford.edu/~amaas/data/sentiment/`.

To train your model, use only the reviews in the "train" folder. Report the performance of your model on the reviews in the "test" folder. Carefully read the README file to have a clear understanding of the data format. Briefly, `imdb.vocab` contains the vocabulary with one word per row. The row indices of the words are used to represent the feature indices that appear in the training and test documents in the "labeledBow.feat" files.

Task 1.1 The entire vocabulary size is 89526, which is also the total feature size. This is too big for training our custom logistic regression. As a preprocessing step, you will need to decide which features to use.

First, you may filter out words that appear in less than 1% of the documents and words that appear in more than 50% of the documents, which are the rare and "stopwords" respectively. Stopwords are the commonly used words that are not important to our tasks.

Second, you need to choose the top $D \in [100, 1000]$ features by their absolute regression coefficients with the *rating scores* (1-10) by using the **Simple Linear Regression** we covered in

Module 4.1. In other words, although eventually we will use logistic regression for binary classification on this data, we will perform linear regression (with the rating score as the target variable) in order to find important features. For this step, you must implement the Simple Linear Regression model from scratch (i.e. you cannot use the linear regression model from sklearn). Examine the top features with the most positive simple regression coefficients and the top features with most negative coefficients. Do they make sense for calling a movie good and bad, respectively?

1.2 20 news groups: a multi-class labelled textual dataset

The 20-news group dataset can be loaded directly using `sklearn.datasets.fetch_20newsgroups` (`https://scikit-learn.org/stable/modules/generated/sklearn.datasets.fetch_20newsgroups.html`).

Use the default train subset (subset='train', and **remove=(['headers', 'footers', 'quotes'**]) in sklearn.datasets) to train the multiclass prediction 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.

Task 1.2 For the sake of this assignment, it is ok to work with a partial dataset with only 5 categories out of the 20 available in the dataset. You may choose your favourite 5 categories. One tip is that choosing 5 distinct categories (e.g., comp.graphics, misc.forsale, rec.sport.baseball, sci.med, talk.politics.guns) may be easy to debug your code because they are easy to distinguish by the corresponding key words.

Similar to task 1.1, you can filter out rare words, stopwords, and words that are not relevant to any of the 5 class labels. Since we are dealing with discrete class labels, we will use something different from simple regression to select features externally. For example, you may use Mutual Information (MI) (`https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mutual_info_score.html`) to select the top $M \in [10, 100]$ feature words per class and take the union of all top feature words to train your multiclass model.

You may choose other ways to select feature words as long as you report what you did in your report in the end. One thing to keep in mind is that our custom multiclass regression may be slow and without regularization you may want to keep the number of features fairly low. For instance, with 100 feature words per class, we can still have up to 500 features in total for 5 categories.

2 Task 2: Implement Logistic and Multiclass classifiers

You should follow the equations that are presented in the lecture slides in Module 4.2 and 4.3, and you must implement the models from scratch (i.e., you cannot use scikit-learn or any other pre-existing implementations of these methods). However, you are free to implement these models based on the code provided in the Colab notebook as you see fit.

In particular, your two main tasks are to:

1. Implement and evaluate Logistic Regression on the IMDB data
2. Implementing and evaluate the Multiclass Regression on the 5-class prediction from the 20-news-group data

You are free to implement these models in any way you want, but you must use Python and you must implement the models from scratch (i.e., you cannot use SciKit Learn or similar libraries). Using the numpy package, however, is allowed and encouraged. Regarding the implementation, we recommend the following approach:

- Implement the Logistic Regression and Multiclass Regression models as Python classes. You should use the constructor for the class to initialize the model parameters as attributes, as well as to define other important properties of the model.
- Your model class for each algorithm should have (at least) two functions:
  - Define a `fit` function, which takes the training data (i.e., $\mathbf{X}$ and $\mathbf{y}$)—as well as other hyperparameters (e.g., the learning rate and/or number of gradient descent iterations)—as input. This function should train your model by modifying the model parameters.
  - Define a `predict` function, which takes a set of input points (i.e., $\mathbf{X}$) as input and outputs predictions (i.e., $\hat{\mathbf{y}}$) for these points.
- For evaluating binary classification, you should use Receiver Operating Characteristic (ROC) curve and area under the ROC curve (AUROC) to evaluate the model classification as we covered in class. For evaluating multi-class prediction, you should compute classification accuracy.
- As a comparison, use **Decision Trees** from sklearn for both binary and multi-label classification tasks.

Task 2.1 Check gradient computed by your implementations using small perturbation, monitor the cross-entropy as a function of iteration, and report your findings on both datasets.

Task 3: Run experiments

The goal of this project is to have you explore linear classification and examine feature importance by the linear coefficients. *Evaluate the performance using AUROC for binary classification and accuracy for multi-class classification.* You are welcome to perform any experiments and analyses you see fit, **but at a minimum you must complete the following experiments in the order stated below**:

1. Report the top 10 features with the most positive coefficients and the top 10 features with the most negative coefficients on the IMDB data using simple linear regression on the movie rating scores.
2. Implement from scratch and conduct
   (a) **Binary classification** on the **IMDb Reviews**
   (b) **Multi-class classification** on the **20 news group dataset**
3. On the same plot, draw ROC curves and report the AUROC values of logistic regression and Decision Trees on the IMDB data binary classification task
4. Report the multiclass classification accuracy of multiclass linear regression and Decision Trees on the 5 chosen classes from the 20-news-group data
5. Further, with a plot, compare the accuracy of the two models as a function of the size of dataset (by controlling the training size). For example, you can randomly select 20%, 40%, 60% and 80% of the available training data and train your model on this subset and evaluate the trained model on the held-out test set.

Note: The above experiments are the minimum requirements that you must complete; however, this project is open-ended. For this part, you might try different learning rates, investigate different stopping criteria for the gradient descent, try linear regression for predicting ratings in the IMDB data, try different text embedding methods as alternatives to bag of words.

You are also welcome and encouraged to try any other model covered in the class (including regularized regression such as Ridge and LASSO), and you are free to implement them yourself or use any Python library that has their implementation (e.g. scikit-learn).

You are encouraged to explore more classes (greater than 5). In scenarios involving a higher number of classes, it may be challenging to distinguish closely related or similar classes. Rather than focusing solely on the most probable class, consider examining the top k (e.g. k =3) predicted classes. A correct prediction is determined by whether the true label is within the top k predicted labels. The scoring mechanism involves assigning a score of 1 if the correct label is among the top k predictions and 0 otherwise.

Of course, you do not need to do all of these things, but look at them as suggestions and try to demonstrate curiosity, creativity, rigour, and an understanding of the course material in how you run your chosen experiments and how you report on them in your write-up.

Deliverables

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

1. `assignment2_group-k.ipynb`: Your data processing, classification and evaluation code should be all in one single Jupyter Notebook. Your notebook should reproduce all the results in your reports. The TAs may run your notebook to confirm your reported findings.
2. `assignment2_group-k.pdf`: Your (max 8-page) assignment write-up as a pdf (details below).

where k is your group number.

Project write-up

Your team must submit a project write-up that is a maximum of **8 pages** (single-spaced, 11pt font or larger; minimum 0.5 inch margins, an extra page for references/bibliographical content can be used). We highly recommend that students use LaTeX to complete their write-ups. You have some flexibility in how you report your results, but you must adhere to the following structure and minimum requirements:

Abstract (100-250 words) Summarize the project task and your most important findings. For example, include sentences like "In this project we investigated the performance of linear classification models on two benchmark datasets", "We found that the logistic/multiclass regression approach achieved worse/better accuracy than Decision Trees and was significantly faster/slower to train."

Introduction (5+ sentences) Summarize the project task, the two datasests, and your most important findings. This should be similar to the abstract but more detailed. You should include background information and citations to relevant work (e.g., other papers analyzing these datasets).

Datasets (5+ sentences) Very briefly describe the datasets and how you processed them. Describe the new features you come up with in detail. Present the exploratory analysis you have done to understand the data, e.g. class distribution.

Results (7+ sentences corresponding to 7 figures) Describe the results of all the experiments mentioned in Task 3 (at a minimum) as well as any other interesting results you find. At a minimum you must have these 7 plots:

1. A horizontal bar plot showing the top 20 features (10 most positive and 10 most negative) from the Simple linear regression on the IMDB data with the coefficients as the x-axis and the feature names (i.e., words) as the y-axis
2. Convergence plot on how the logistic and multiclass regression converge given a reasonably chosen learning rate.
3. A single plot containing two ROC curves of logistic regression and sklearn-DT (Decision Trees) on the IMDB test data.
4. A bar plot that shows the AUROC of logistic regression and DT on the test data (y-axis) as a function of the 20%, 40%, 60%, 80%, and 100% training data (x-axis)
5. A bar plot that shows the classification accuracies of multiclass regression and DT on the test data (y-axis) as a function of the 20%, 40%, 60%, 80%, and 100% training data (x-axis)
6. A horizontal bar plot showing the top 20 features (10 most positive and 10 most negative) from the logistic regression on the IMDB data with the coefficient as the x-axis and the feature names (i.e., words) as the y-axis
7. A heatmap showing the top 5 most positive features as rows for each class as columns in the multi-class classification on 4 the chosen classes from the 20-news group datasets. Therefore, your heatmap should be a 20-by-4 dimension.

Discussion and Conclusion (5+ sentences) Summarize the key takeaways from the project (Do the top features make sense?) and possibly directions for future investigation.

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

Evaluation

The assignment 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 correctly?

  - Are your reported accuracies close to the reference solutions?
  - Do your selected top features actually improve performance than randomly chosen features on the IMDB data?
  - Do you observe the correct trends in the experiments (e.g., comparing training size)?
- 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?
  - **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.

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.**