MiniProject 2: Multi-Class Logistic Regression and Gradient Descent

COMP 551, Fall 2020, McGill University

Please read this entire document before beginning the assignment.

Preamble

- This mini-project is **due on November 18th at 11:59pm ET.** 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 be completed in groups of three. All members of a group will receive 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. You must register your group on MyCourses and any group member can submit. See MyCourses for details.
- You are free to use libraries with general utilities, such as matplotlib, numpy and scipy for Python. **However, you should implement everything (e.g., 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, etc.** The only exception to this rule is made clear below.

Background

In this miniproject you will implement multi-class logistic regression and compare it against one of several other classification algorithms that we have covered in the class so far, namely K-NN, Naive Bayes or Decision Tree. *For these algorithms you are allowed to use Scikit-Learn implementation* (or any other package). The goal is to gain experience implementing an algorithm from scratch, become familiar with Gradient-based optimization and to a minor extent get hands-on experience comparing the performance of different models.

Datasets

Since the focus of this mini-project is not on data-preprocessing, we will use datasets that are easily accessible from Scikit-Learn. However, you still need a basic understanding of datasets used in your experiments, so that you can provide a meaningful analysis of your results.

- **One dataset directly available in Scikit-Learn**: Digists dataset (see `https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_digits.html`)
- **One dataset from OpenML**: you can access OpenML datasets (`https://www.openml.org`) directly from Scikit-Learn; see `https://scikit-learn.org/stable/datasets/index.html#openml`. Choose at least one additional dataset (where the task is classification) from that repository for your analysis.

If there are missing values in your selected dataset, it is up to you to perform required preprocessing steps. For categorical features use one-hot encoding of the features.

Softmax Regression

For implementation you should follow the equations that are presented in the lecture slides, and you must implement the models from scratch (i.e., you cannot use Scikit Learn or any other pre-existing implementations). In the code template released for the gradient descent lecture, we have showed how to implement a *model* (the class ”Linear-Regression”) and an *optimizer* (the class ”GradientDescent”). In this project, you are required to follow this template respectively to implement the softmax regression *model* as well as gradient descent with momentum *optimizer* then train your model on the selected dataset and report accordingly. In particular, your main tasks are:

1. Implement multi-class logistic regression model. Following the **LinearRegression** class of the code template released for the gradient descent lecture, implement this class such that the **fit** function receives an **optimizer** as an input.
2. Follow the template of **GradientDescent** class, implement mini-batch optimization using gradient descent with momentum. An instance of this optimizer is passed to the logistic regression class to fit the data.

You will need to adjust these methods to also receive a validation set, so that you can track the training and validation accuracy in each step of the optimization. You may add extra arguments to the functions or extra functions to the classes, should you deem necessary.

Required Analysis

Your analysis should have the following components:

Hyper-parameters of the optimization procedure. Analyze the quality of the solution found (in terms of the validation accuracy) and the run-time for different batch-sizes, learning rates, and momentum parameters. Since there are many combinations of values for these parameters, first try to find a good combination (e.g., using simple grid search), and then see how the performance and run-time of your algorithm changes as you vary one of these hyper-parameters. *Implement and use 5-fold cross validation to estimate performance, in terms of accuracy, in all of the experiments with respect to batch sizes, learning rates, and momentum parameters, and report the best hyper-parameters you found.* You should track and analyze the validation and training accuracy (and optionally the cost) to better understand and analyze the effect of different hyper-parameters. Each fold in cross-validation will produce one of these training and validation curves. Plot one of these training and validation curves for several representative choices of hyper-parameters in your report.

Termination condition. For a given training and validation set, track the training and validation accuracy in each iteration. A good strategy for termination is to end the optimization if the validation error has not been decreasing in the past T iterations (e.g. $T = 20$). In this case your optimizer should return the model with the *best validation accuracy*, not the model at the last step of gradient descent.

Comparison against another classifier For the second classifier you can use off-the-shelf implementation. Again use (your own implementation of) 5-fold cross validation for hyper-parameter tuning and reporting the accuracy of both classifiers on two datasets.

Deliverables

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

1. **code.zip**: Your data processing, classification and evaluation code (as some combination of .py and .ipynb files).

2. **writeup.pdf**: Your (max 3-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, 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)

Introduction (5+ sentences) Summarize the project task, the two datasest, and your most important findings. This should be similar to the abstract but more detailed.

Datasets (3+ sentences) Very briefly describe the datasets and how you processed them. Any data-normalization or special handling of the missing data should be described here. Present the exploratory analysis you have done to understand the data, e.g. class distribution. This analysis may be more relevant for your second (OpenML) dataset.

Results (10+ sentences, possibly with figures or tables) Describe the results of all the experiments mentioned above as well as any other interesting analysis you performed. If your implementation used a special setup (e.g., you did 10-fold cross validation, and your stopping criteria was more involved) please explain here. At a minimum you must report:

1. A discussion of how the multi-class logistic regression performance (e.g., convergence speed) depends on the parameters of optimization. (Note: a figure would be an ideal way to report these results).
2. A discussion of the training and validation curve for different optimization hyper-parameters
3. A comparison of the accuracy of your second classifier and logistic regression on both datasets.

Discussion and Conclusion (4+ sentences) Summarize the key takeaways from the project and possibly some improvements that you would consider if you had more time or resources.

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

Evaluation

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

- Completeness (25 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 you observe the correct trends in the experiments (e.g., in comparing learning rates)?
  - Does your code structure follow the template?
- Writing and code 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)?
  - Is your code clearly written and structured?

- Originality / creativity / more extensive analysis (10 points)
  - Did you go beyond the bare minimum requirements for the experiments? For example, you could investigate other optimization techniques mentioned in the class, or investigate regularization.
  - Did you provide a more extensive and conclusive analysis, for example by considering a range of different classifiers and datasets of different characteristics (e.g., size and number of features)? Were you able to draw more convincing or new conclusions based on these 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.**