CS 419M: Programming Assignment 1

March 2022

This assignment is entirely programming-based. The task is hosted on Kaggle here. **Kaggle registration instructions:**

1. Go to the Kaggle competition link given above.
2. Create a new login using your roll number. It is important that you use your roll number and nothing else.
3. Please contact the TAs if you need any help setting this up.

1 Introduction

You might have seen *"Wine Connoisseurs"* in TV shows and movies that act all snobby and talk about how judging wine quality is an intricate task that requires you to consider various *subjective* factors like intensity, finish and flavours into consideration. You, taking a more scientific approach and as a budding ML enthusiast, posit that you can create an objective mathematical model that can rate wine quality using simple Machine Learning techniques.

In this programming assignment, your task is to find the best linear regression model that most accurately predicts wine quality for a set of test wines using only *physicochemical* features of wine as input data.

The dataset containing train and test data is available to you at the Kaggle link mentioned above. You can generate your own validation data for hyperparameter tuning by suitably splitting the given labeled training data. Note that you should train the model that you will use for your final predictions on your Train+Val data.

Within `code.py`, you will find a function `closed_soln(phi, y)` that computes the closed-form solution for an unregularized least squares linear regression model. You will also see placeholders for functions with self-explanatory names (e.g. `compute_RMSE(phi, w, y)`, `generate_output(phi_test, w)`) that need to be filled in. The function `main()` has been expanded to show the sequence of steps we will run to evaluate tasks 1 to 3. Please use **Python 3.7** to run your code and minimize version discrepancies during grading.

Note: Model Predictions must be **integers** within range **[3, 9]**.

Submission deadline of this assignment is 11:59 PM, 31st March, '22 on Moodle.

2 Tasks

1. Answer the following preliminary questions in `submission/report.pdf`: [**3 points**]

   - Since our target variable can take only a small number of discrete integer values, should we treat this as a classification problem instead? Why or why not? [**1 points**]
   - Considering this task as a regression problem, our go-to evaluation metric of choice is RMSE. Propose another metric for evaluating the wine-quality predictions that is more interpretable. Comment on the interpretation that this metric will offer. [**2 points**]

2. Implement the gradient descent algorithm to estimate the model parameters for an unregularized least squares linear regression model. Add your implementation to the `gradient_descent(phi, y)` function. Compute the weights using gradient descent and find predicted wine quality for the development set that you created. Report the root mean squared error you obtain on these samples in `submission/report.pdf`. Set the learning rate in `gradient_descent(phi, y)` to the value we should use during grading. (Your weight vector should start from an all-zeros vector or a random initialization.) **Tip:** You will find it useful to apply some form of feature normalization on your inputs before optimizing your loss function. [**10 points**]

   Report the following details in `submission/report.pdf`:

   - What is the absolute difference between the following two calls, `compute_RMSE(phi, w1, y)` and `compute_RMSE(phi, w2, y)` on the dev set, where `w1` and `w2` were obtained using the closed-form solution and gradient descent respectively? [**2 points**]
   - What stopping criterion did you use to determine the convergence of gradient descent? [**1 point**]
   - Instead of batch gradient descent, implement stochastic gradient descent in `sgd(phi, y)`. Let `w1` and `w2` be the respective weight vectors. Report the absolute difference between the two calls `compute_RMSE(phi_train, w1, y)` and `compute_RMSE(phi_train, w2, y)` on your dev set. (Tune and set the learning rate such that this difference is minimized.) [**5 points**]

3. Within `pnorm(phi, y, p)`, optimize a p-norm regularized least squares objective function (where p is 2 or 4) using either gradient descent or stochastic gradient descent. Your regularization term now becomes $\lambda(||w||_p)^p$ where p-norm of a d-dimensional w is defined as $||w||_p = (\sum_{i=1}^{d} |w_i|^p)^{1/p}$ and λ is a non-negative value. This term is added to the standard least squares objective to form the p-norm regularized objective. Report root mean squared error on all the development set samples when p=2 and p=4 in submission/report.pdf. Set λ in `pnorm(phi, y, p)` to the value we should use during grading. [**5 points**]

4. Create subsets of the training data containing the first 10%, 25%, 50%, 75%, and 100% instances in your training data. Create a plot where the X-axis shows the size of subset used for training and the Y-axis is the RMSE on the development data. You

can create a new function within `code.py` and use existing libraries (like matplotlib) to create this plot. Add it to `submission/report.pdf`. [**3 points**]

5. Of the 12 features, which are the two most and the two least useful ones? Describe how you identified these features in `submission/report.pdf`. [**4 points**]

6. The objective for this last part is to build the best possible regression model and you are also free to use open-source library implementations. Submit your predictions on Kaggle to check your test performance and compete on the leaderboard. (Make sure that your submission file follows the same order of the samples given in test.csv.)

   Use this as an opportunity to explore the various regression techniques out there and to learn which methods are suited to what kind of tasks. Some links to get you started: [1] [2] [3]

   Top-scoring performers on the"Private Leaderboard" (with a suitable threshold determined after the submission date) will be awarded bonus credit. Describe your selected approach for this part in `submission/report.pdf`. Also submit your code for this part within `code.py`. (You can define any new functions for this last part and run this within `main()`.) [**5 points**]

3 Submission

The directory submission will contain 2 files, `code.py` and `report.pdf`. The directory should be compressed for submission on Moodle using the command:

```
tar -czvf submission.tgz submission
```