CMPE 462 Machine Learning
Spring 2020 Project I
Due: April 23 by 11.59pm

Project Description

In this project, you are going to implement logistic regression from scratch. You are provided a subset of the famous handwritten digit dataset called MNIST. In the subset, you will find images of digit 1 and 5. Therefore, you will be solving a binary classification problem. The project includes feature extraction, model training, and evaluation steps.

Follow the steps in the provided notebook that would guide you through the solution step-by-step. Make sure that your notebook is self-contained with comment lines. You will also prepare a project report.

IMPORTANT NOTE: You are allowed to use third-party libraries such as `numpy` and `matplotlib`. However, logistic regression and gradient descent implementation will be from scratch. You are not allowed to import any function that accomplishes the task itself. For instance, you can use `numpy` arrays for matrix operations, but you cannot use `scikit-learn` to implement cross validation.

Task 1: Feature Extraction (35 Pts)

Load the training/test data and labels as numpy arrays (Hint: `np.load`). Train and test data are 1561×256 and 424×256 dimensional matrices, respectively. Each row in the aforementioned matrices corresponds to an image of a digit. The 256 pixels correspond to a 16×16 image. Label 1 is assigned to digit 1 and label -1 is assigned to digit 5.

1. (5 points) Display two of the digit images, one for digit 1 and one for digit 5. You can use the `imshow` function of `matplotlib` library. To be able to use the `imshow` function, you will first need to reshape 256 pixels to 16×16 matrix.

2. **Implementing Representation 1:** (10 points) Extract the symmetry and average intensity features discussed in the class (see logistic regression lecture notes). To compute the intensity features, compute the average pixel value of the image, and for the symmetry feature, compute the negative of the norm of the difference between the image and its y-axis symmetry. Search numpy's documentation to find suitable functions at each step. You should extract these two features for each image in the training and test sets. As a result, you should obtain a training data matrix of size 1561×2 and test data matrix of size 424×2.

   Throughout the notebook, we will refer the representation with these two features as **Representation 1**.

3. (5 points) Provide two scatter plots, one for training and one for test data. The plots should contain the average intensity values in the x-axis and symmetry values in the y-axis. Denote the data points of label 1 with a blue marker shaped o and the data points of label -1 with a red marker shaped x. (Hint: check out `plt.scatter` and

its `marker` and `color` parameters). Explicitly state the axis labels and figure title for both plots (Hint: `plt.xlabel, plt.ylabel, plt.title`).

4. **Implementing Representation 2:** (15 points) Come up with an alternative feature extraction approach. The features can again be 2-D, or higher dimensional. If you use any external resource, please cite the references. Explain the feature extraction procedure clearly. If it is an algorithm, provide the algorithm. If it is a function, provide the mathematical expressions.

   If your proposed features are 2-D or 3-D, provide the scatter plots similar to the previous step. We will refer this representation proposed by you as **Representation 2.**

Task 2: Logistic Regression (40 Pts)

1. (20 points) Implement the logistic regression classifier from scratch with gradient descent and train it using Representation 1 and Representation 2 as inputs. Concatenate 1 to your features for the intercept term. In this case, one data point will look like $[1, x_1, x_2]$, and the model vector will be $[w_0, w_1, w_2]$, where w_0 is the intercept parameter. You can refer to lecture notes (Logistic regression slides 29/30) to review the gradient descent learning algorithm and the logistic loss. To implement the gradient of the logistic loss with respect to w, first derive its expression by hand. Please include your derivation in your report.

   To prove that your implementation is converging, keep the loss values at each gradient descent iteration in a numpy array. After the training is finalized, plot the loss values with respect to iteration count (Hint: `plt.plot`). You should observe a decreasing loss as the number of iterations increases. Also, experiment with 5 different learning rates between 0 and 1 and plot the convergence curves for each learning rate in the same figure to observe the effect of the learning rate (step size) one convergence.

   To decide when to terminate the gradient descent iterations, check the absolute difference between the current loss value and the loss value of the previous step. If the difference is less than a small number, such as 10^{-5}, you can exit the loop.

2. (10 points) Implement logistic regression with ℓ_2 regularization, $||\mathbf{w}||_2^2$. Show that your implementation is working by visualizing the loss over iterations again. Visualization for a single learning rate and λ suffices for the task purposes.

3. (10 points) Implement a 5-fold cross validation procedure to find the optimal λ value for both Representation 1 and 2. Experiment with at least three different λ values between 0 and 1. Report the mean/std of cross validation accuracy of every representation/parameter combination as a table and clearly mark the best configuration.

Task 3: Evaluation (25 Pts)

1. (5 points) Train the logistic regression classifier on Representation 1 and 2. Report the training and test classification accuracies. Similarly, train the regularized logistic

regression classifier with the best λ you obtained by 5-fold cross validation. Report the training and test classification accuracies.

Note: You can calculate the classification accuracy as

$$\frac{\text{number of correctly classified samples}}{\text{total number of samples}} \times 100$$

2. (15 points) Visualize the decision boundary (the line that is given by $\mathbf{w}^T x = 0$) obtained from the logistic regression classifier learned without regularization. For this purpose, use only Representation 1. Provide two scatter plots for training and test data points together with the decision boundary shown on each of them.

3. (5 points) Comment on your work in your report. Include the answers for the following questions in your discussion.

   - Did regularization improve the generalization performance (did it help reducing the gap between training and test accuracies/errors)? Did you observe any difference between using Representation 1 and 2?
   - Which feature set does give the best results? Which one is more discriminative?
   - What would be your next step to improve test accuracy?

Submission Guidelines

- Download the provided dataset and the Jupyter Notebook. Follow the instructions in the Jupyter Notebook (.ipynb) file to complete the project.

- Prepare a comprehensive report in pdf format. Include your plots, tables, and comments for each task. Follow academic writing rules. Prefer a concise and clear language. You do not need to include code snippets in your report.

- Submit the completed .ipynb and .pdf files through the assignment *Project 1* on Moodle. Name your submission files with the student IDs of the group members (i.e. 2015400XXX_2015400YYY.ipynb). Please submit your pdf report through Turnitin assignment.

- The submitted .ipynb file should run from scratch without errors when accompanied with the provided data. Note that iPython kernel of Jupyter stores variables in the kernel over time and if your code depends on any such variable, it will not run on new sessions. Therefore, make sure that you can run your code from scratch (restart kernel & run all) before submitting the notebook. If your code does not run for any reason, you will be deducted 10 pts.

- Please fairly share the tasks among the group members. Write the student ID/IDs next to each task/code cell to specify who completed which task. If you do not specify any IDs, the instructor will assume each student put equal effort in that task.