Statistical Methods in AI

Instructor : Prof Ravi Kiran Sarvadevabhatla

Deadline : 18 October 2023 11:55 P.M

Assignment - 3

General Instructions

- Your assignment must be implemented in Python.
- While you're allowed to use ChatGPT for assistance, you must explicitly declare in comments the prompts you used and indicate which parts of the code were generated with the help of ChatGPT.
- Plagiarism will only be taken into consideration for code that is not generated by ChatGPT. Any code generated with the assistance of ChatGPT should be considered as a resource, similar to using a textbook or online tutorial.
- The difficulty of your viva or assessment will be determined by the percentage of code in your assignment that is not attributed to ChatGPT. If during the viva if you are unable to explain any part of the code, that code will be considered as plagiarized.
- Clearly label and organize your code, including comments that explain the purpose of each section and key steps in your implementation.
- Properly document your code and include explanations for any non-trivial algorithms or techniques you employ.
- Ensure that your Jupyter Notebook is well-structured, with headings, subheadings, and explanations as necessary.
- Your assignment will be evaluated not only based on correctness but also on the quality of code, the clarity of explanations, and the extent to which you've understood and applied the concepts covered in the course.
- Make sure to test your code thoroughly before submission to avoid any runtime errors or unexpected behavior.
- The Deadline will not be extended.

- Moss will be run on all submissions along with checking against online resources.
- We are aware how easy it is to write code now in the presence of ChatGPT and Github Co-Pilot, but we strongly encourage you to write the code yourself.
- We are aware of the possibility of submitting the assignment late in github classrooms using various hacks. Note that we will have measures in place for that and anyone caught attempting to do the same would be give zero in the assignment.
- **SUBMISSION FORMAT : The submission should consist of a single file for Task 1,2 and 3**

  In this assignment, you are required to work with W&B library. Here are some resources on W&B logging and reporting :

  1. **Experiment management with Weights and Biases platform**
  2. **Reports by W&B**

  Datasets for Task 1 and 2 are uploaded on moodle, Link is given for Dataset for Task 3.

1 Multinomial Logistic Regression
[Marks : 40, Estimated Time : 2-3 days]

Implement a Multinomial logistic regression model from scratch using **numpy** and **pandas**. You have to train this model on Wine Quality Dataset to classify a wine's quality based on the values of its various contents.

1.1 Dataset Analysis and Preprocessing [5 marks]

1. Describe the dataset using mean, standard deviation, min, and max values for all attributes.
2. Draw a graph that shows the distribution of the various labels across the entire dataset. You are allowed to use standard libraries like Matplotlib.
3. Partition the dataset into train, validation, and test sets.
4. Normalise and standarize the data. Make sure to handle the missing or inconsistent data values if necessary.
5. Clearly identify and justify the selection of the two most significant features. (Hint: Utilize PCA from the previous assignment.)

1.2 Model Building from Scratch [20 marks]

1. Create a Multinomial Logistic Regression model from scratch and Use log loss i.e. cross entropy loss as loss function and Gradient descent as the optimization algorithm (write seperate methods for these).

2. Train the model, use sklearn classification report and print metrics on the validation set while training. Also report loss and accuracy on train set.

3. Using the two significant features you've identified in Task 1, plot the decision boundaries of your model. This will give a visual representation of how your model categorizes data points in the feature space.

1.3 Hyperparameter Tuning and Evaluation [15 marks]

1. Use your validation set and W&B logging to fine-tune the hyperparameters (learning rate , epochs) for optimal results.

2. Evaluate your model on test dataset and print sklearn classification report.

2 Multi Layer Perceptron Classification
[Marks : 60, Estimated Time : 4-5 days]

In this part, you are required to implement MLP classification from **scratch** using numpy, pandas and experiment with various activation functions and optimization techniques, evaluate the model's performance, visualize decision boundaries, and draw comparisons with the previously implemented multinomial logistic regression.

Use the same dataset as Task 1

2.1 Model Building from Scratch [20 marks]

Build an MLP classifier class with the following specifications: Do not use sklearn for this.

1. Create a class where you can modify and access the learning rate, activation function, optimisers, number of hidden layers and neurons.

2. Implement methods for forward propagation, backpropagation, and training.

3. Different activation functions introduce non-linearity to the model and affect the learning process. Implement the Sigmoid, Tanh, and ReLU activation functions and make them easily interchangeable within your MLP framework.

4. Optimization techniques dictate how the neural network updates its weights during training. Implement methods for the Stochastic Gradient Descent (SGD), Batch Gradient Descent, and Mini-Batch Gradient Descent algorithms from scratch, ensuring that they can be employed within your MLP architecture. Additionally, draw comparisons with the inbuilt ADAM optimizer in terms of performance.

2.2 Model Training & Hyperparameter Tuning using W&B [10 marks]

Effective tuning can vastly improve a model's performance. Integrate Weights & Biases (W&B) to log and track your model's metrics. Using W&B and your validation set, experiment with hyperparameters such as learning rate, epochs, hidden layer neurons, activation functions, and optimization techniques. You have to use W&B for loss tracking during training and to log effects of different activation functions and optimizers on model performance.

1. Log your scores - loss and accuracy on validation set and train set using W&B.
2. Report metrics: accuracy, f-1 score, precision, and recall. You are allowed to use sklearn metrics for this part.
3. You have to report the scores(ordered) for all the combinations of :
    - Activation functions : sigmoid, tanh and ReLU (implemented from scratch)
    - Optimizers : SGD, batch gradient descent, and mini-batch gradient descent (implemented from scratch) and ADAM(use in-built ADAM optimiser).
4. Tune your model on various hyperparameters, such as learning rate, epochs, and hidden layer neurons.
    - Plot the trend of accuracy scores with change in these hyperparameters.
    - Report the parameters for the best model that you get (for the various values you trained the model on).
    - Report the scores mentioned in Point 2 for all values of hyperparameters in a table.

2.3 Evaluating Model [10 marks]

1. Test and print the classification report on the test set. (use sklearn)
2. Clear visualization of the MLP's decision boundaries for the top two features.

3. Provide an analytical comparison with the decision boundaries and results of the logistic regression model.

2.4 Multi-Label Classification [20 marks]

For this part, you will be training and testing your model on Multilabel dataset: "advertisement.csv" as provided in Assignment 1.

1. Modify your model accordingly to classify multilabel data.
2. (a) Log your scores - loss and accuracy on validation set and train set using W&B.
   (b) Report metrics: accuracy, f-1 score, precision, and recall.
   (c) You have to report the scores(ordered) for all the combinations of :
      - Activation functions : sigmoid, tanh and ReLU (implemented from scratch)
      - Optimizers : SGD, batch gradient descent and mini-batch gradient descent (implemented from scratch) and ADAM(use in-built ADAM optimiser).
   (d) Tune your model on various hyperparameters, such as learning rate, epochs, and hidden layer neurons.
      - Plot the trend of accuracy scores with change in these hyperparameters.
      - Report the parameters for the best model that you get (for the various values you trained the model on).
      - Report the scores mentioned in Point b for all values of hyperparameters in a table.
3. Evaluate your model on the test set and report accuracy, f1 score, precision, and recall.

3 Multilayer Perceptron Regression
[Marks : 50, Estimated Time : 3-4 days]

In this task, you will implement a Multi-layer Perceptron (MLP) for regression from scratch, and integrate Weights & Biases (W&B) for tracking and tuning. Using the **Boston Housing dataset**, your have to predict housing prices while following standard machine learning practices.

3.1 Data Preprocessing [5 marks]

1. Describe the dataset using mean, standard deviation, min, and max values for all attributes.

2. Draw a graph that shows the distribution of the various labels across the entire dataset. You are allowed to use standard libraries like Matplotlib.

3. Partition the dataset into train, validation, and test sets.

4. Normalise and standarize the data. Make sure to handle the missing or inconsistent data values if necessary.

3.2 MLP Regression Implementation from Scratch [20 marks]

In this part, you are required to implement MLP regression from scratch using numpy, pandas and experiment with various activation functions and optimization techniques, and evaluate the model's performance.

1. Create a class where you can modify and access the learning rate, activation function, optimisers, number of hidden layers and neurons.

2. Implement methods for forward propagation, backpropagation, and training.

3. Implement the Sigmoid, Tanh, and ReLU activation functions and make them easily interchangeable within your MLP framework.

4. Implement methods for the Stochastic Gradient Descent (SGD), Batch Gradient Descent, and Mini-Batch Gradient Descent algorithms from scratch, ensuring that they can be employed within your MLP architecture. Additionally, draw comparisons with the inbuilt ADAM optimizer in terms of performance and convergence speed.

3.3 Model Training & Hyperparameter Tuning using W&B [20 marks]

1. Log your scores - loss (Mean Squared Error) on the validation set using WB.

2. Report metrics: MSE, RMSE, R-squared.

3. You have to report the scores(ordered) for all the combinations of :

   - Activation functions : sigmoid, tanh and ReLU (implemented from scratch)
   - Optimizers : SGD, batch gradient descent and mini-batch gradient descent (implemented from scratch) and ADAM(use in-built ADAM optimiser).

4. Tune your model on various hyperparameters, such as learning rate, epochs, and hidden layer neurons.

- Report the parameters for the best model that you get (for the various values you trained the model on).
- Report the scores mentioned in Point 2 for all values of hyperparameters in a table.

3.4 Evaluating Model [5 marks]

1. Test your model on the test set and report loss score (MSE, RMSE, R-squared).