EE769 Introduction to Machine Learning

Programming Assignment 1: Gradient descent and regularization for linear regression

Data: TempTrain.csv, TempTest.csv

Strict instructions (otherwise, your assignment will not be graded):

- Up to two people can team up, but only one should submit, and both should understand the entire code
- Submit a .ipynb file with all the intermediate outputs, graphs, and comments to demonstrate your understanding. The name of the file should be YourRollNo_YourFriendsRollNo_1.ipynb
- Submit a YourRollNo_YourFriendsRollNo_1.py file closely mimics your own .ipynb file that can be called from the command line using python, that only uses argv, numpy and pandas that can be called with a folder name to read the training file from the folder, train a model after validation, write weights of the model in a weights.csv (single column, no header, bias in the end) in the same folder, read test file from the same folder, output predictions on the test file in another predictions.csv (single column, no header).
- Submit weight.csv and prediction.csv.
- Combine all files into YourRollNo_YourFriendsRollNo_1.zip file to submit on Moodle.
- In the .ipynb and .py the following should be present:

1. Write the following functions for linear regression with the following input-output specifications:
    a. Normalization:
        i. Input 1: a data matrix **X**
        ii. Output 1: Vector of means
        iii. Output 2: Vector of standard deviations
        iv. Output 3: Normalized data matrix **X**
    b. Prediction:
        i. Input 1: a data matrix **X**
        ii. Input 2: a weight vector (with bias included) **w**
        iii. Output 1: Prediction **y = [X 1] w**, where **1** is a vector of all 1's.
    c. Loss for penalized linear regression:
        i. Input 1: error **e**
        ii. Input 2: weights **w**
        iii. Input 3: Hyperparameter $\lambda \geq 0$
        iv. Output 1: Mean square error + λ times L2 norm of **w** (except for the bias term)
        v. Output 2: Normalized root mean square error normalized by variance of the target **t**
    d. Gradient:
        i. Input 1: a data matrix **X**
        ii. Input 2: weight vector **w**
        iii. Input 3: Hyperparameter $\lambda \geq 0$
        iv. Hint: You can call any of the previous functions within this function, if it helps
        v. Output: Gradient of loss with respect to the weights
    e. Gradient descent optimizer for linear regression:
        i. Input 1: a data matrix **X**
        ii. Input 2: weight vector **w**
        iii. Input 3: Hyperparameter $\lambda \geq 0$
        iv. Input 4: Maximum number of iterations t
        v. Input 5: Smallest change in NRMSE ε
        vi. Input 6: learning rate η
        vii. Hint 1: You can call any of the previous functions within this function, if it helps

viii. Hint 2: Run a gradient descent loop that exits if max-iter is exceeded or the change in NRMSE < ε, and plot NRMSE vs. iter
ix. Output 1: Optimized weights
x. Output 2: Error vector **e**

f. Main:
i. Read training data from file (.ipynb can use any read method, but .py should use sys.argv to ask for the folder (not the file))
ii. Split independent variables (first 17 columns) and dependent variable (Next_Tmax)
iii. Normalize
iv. Split into training and validation
v. Run a loop over 1/λ
vi. For each value of 1/λ, randomly initialize the weights, run gradient descent, and plot NRMSE
vii. Experiment with different values of η and ε to see the impact on number of iterations and final error (only one value is needed for .py file)
viii. Read test data from the file
ix. Normalize using the training averages and standard deviations.
x. Compute NRMSE, RMSE, and MAE on the test file using the best model (lowest error, and not the lowest loss), plot y vs. t, and print the weights (plotting not needed for .py file, but file writing functions not needed for .ipynb file)

2. Repeat the exercise for MSE loss, but L1 penalty on the weights. (Not needed in .py file)
3. Repeat the exercise with the pseudo-inverse method using numpy.linalg. (Not needed in .py file)
4. Comment on the results, including test errors (NRMSE, RMSE, and MAE, not needed in .py file).