Assignment 2 (12.5 pts)

Due March 2, 2023, 23:59 EST

Q1) 2pts Derive a closed form expression for the weights of the generalized linear model, $\widehat{f}(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(\mathbf{x})$, using a least-squares loss and general Tikhonov regularization. The optimization problem to be solved for the weights can be written as

$$\underset{\mathbf{w} \in \mathbb{R}^M}{\arg\min} \left(\sum_{i=1}^{N} \left(y^{(i)} - w_0 - \sum_{j=1}^{M-1} w_j \phi_j(\mathbf{x}^{(i)}) \right)^2 + \sum_{i=1}^{M} \sum_{j=1}^{M} \Gamma_{ij} w_{i-1} w_{j-1} \right),$$

where $\mathbf{\Gamma} \in \mathbb{R}^{M \times M}$ is a symmetric positive semi-definite matrix whose ijth entry is given by Γ_{ij}.

Q2) 2pts Considering the GLM

$$\widehat{f}(\mathbf{x}, \boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i k(\mathbf{x}, \mathbf{x}^{(i)}),$$

derive a computational strategy to estimate $\boldsymbol{\alpha} = \{\alpha_1, \alpha_2, \dots, \alpha_N\}^T \in \mathbb{R}^N$ by minimizing the objective function $\sum_{i=1}^{N} \left(y^{(i)} - \widehat{f}(\mathbf{x}^{(i)}, \boldsymbol{\alpha}) \right)^2 + \lambda \sum_{i=1}^{N} \alpha_i^2$. Compare this expression for the weights to those we derived in class using the dual representation. Are they different or the same? Explain why.

Q3) 4pts Construct a radial basis function (RBF) model that minimizes the least-squares loss function. Use a Gaussian kernel and consider the grid of shape parameter values $\theta = \{0.05, 0.1, 0.5, 1, 2\}$, consider the grid of regularization parameters $\{0.001, 0.01, 0.1, 1\}$, and construct the model using Cholesky factorization. Select the hyperparameters across the grid of possible values by evaluating on the validation set. Construct the model on the datasets `rosenbrock` (with `n_train=1000, d=2`), and `mauna_loa`. Use both the training and validation sets to predict on the test set, and format your results in a table (present test RMSE for regression datasets).

Q4) 4.5pts Implement a greedy regression algorithm using a dictionary of basis functions. Design the dictionary of basis functions by observing the structure of the one-dimensional `mauna_loa` training dataset. The dictionary should contain at least 200 basis functions. Justify your design choices[1]. Use the orthogonal matching pursuit metric to select a new basis function at each iteration. Use the minimum description length (MDL) defined below as a stopping criterion for your greedy algorithm

$$\frac{N}{2} \log(\ell_2 - \text{loss}) + \frac{k}{2} \log N,$$

[1] Note that you shouldn't need to consider each basis function individually. It is likely that the basis functions you design will have free parameters in which case you could include multiple basis functions with different values of these free parameters in your dictionary.

where ℓ_2-loss is simply the least-squares training error and k is the iteration number (or number of terms in the greedy model). The MDL metric can be considered to be a surrogate of the generalization error – in other words, this metric will decrease as the model complexity (k) grows and then increase as overfitting starts to occur.

Apply your algorithm to the `mauna_loa` dataset. Use both the training and validation sets to predict on the test set, plot the prediction relative to the test data, and present the test RMSE. Comment on the performance of your model. Also, report and comment on the sparsity of your model.

Submission guidelines: Submit an **electronic copy** of your report (**maximum 10 pages** in at least 10pt font) in **pdf** format and **documented** Python scripts. You should include a file named "README" outlining how the scripts should be run. Upload both your report in pdf format and a single tar or zip file containing your code and README to Quercus. You are expected to verify the integrity of your tar/zip file before uploading. Do not include (or modify) the supplied *.npz data files or the data_utils.py module in your submission. The report must contain

- Objectives of the assignment
- A brief description of the structure of your code, and strategies employed
- Relevant figures, tables, and discussion

Do not use scikit-learn for this assignment, the intention is that you implement the simple algorithms required from scratch. Also, for reproducibility, always set a seed for any random number generator used in your code. For example, you can set the seed in numpy using `numpy.random.seed`.