Assignment 2 (17 pts)

Due March 9, 2020

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

$$\underset{\mathbf{w} \in \mathbb{R}^M}{\arg\min} \left(\sum_{i=1}^{N} \left(y^{(i)} - w_0 - \sum_{j=1}^{M-1} w_j \phi_j(\mathbf{x}^{(i)}) \right)^2 + \sum_{i=0}^{M} \sum_{j=0}^{M} \Gamma_{ij} w_i w_j \right),$$

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) 3pts Implement a generalized linear model (GLM) that minimizes the least-squares loss function (using the SVD). By observing the structure of the one-dimensional `mauna_loa` training dataset, design a compact set of basis functions to construct a GLM for this dataset. Justify your design choices. Additionally, choose a regularization parameter λ, and use the validation set to verify your model. Finally, 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.

Q3) 3pts Considering the same basis functions used in Q2, construct a kernelized GLM from the dual perspective. Select a positive regularization parameter, and construct the model using the Cholesky factorization. Comment on the computational cost and memory requirements of the dual approach versus the primal approach. Also, visualize your designed kernel by plotting $k(0, z)$, and by plotting $k(1, z+1)$ where $z \in [-0.1, 0.1]$. Is your kernel translation-invariant (stationary)? Make your argument based on your plots of the kernel.

Q4) 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 `iris`, `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, and test accuracy for classification datasets).

Q5) 5pts Implement a greedy regression algorithm using a dictionary of Gaussian kernels centered at the training points. 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

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

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 `rosenbrock` dataset (with `n_train=200,d=2`) for three different settings of the shape parameter $\{0.01, 0.1, 1.0\}$. Report the test error and sparsity of your model for these different cases.

Submission guidelines: Submit an **electronic copy** of your report in **pdf** format, and **documented** python scripts. You should include a file named "README" outlining how the scripts should be run. Upload a single tar or zip file containing all files 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`