ROB313 Assignment 2: Exploring Generalized Linear Methods

Andrew Jairam

1006941972

andrew.jairam@mail.utoronto.ca

Introduction and Objectives

In the first assignment, it was found that the k-NN regression method is quite weak, and as such, it is desired to discover different regression methods that are stronger in general. This assignment aims to uncover a class of such better methods: generalized linear models. The purpose of this assignment is to understand how to implement a few different generalized linear models involving kernelized approaches, and observe their performance. In particular, regression using gaussian radial basis functions and a greedy implementation using a dictionary of basis functions will be designed to understand considerations and practices behind these methods. Additionally, derivations for the weight vector using Tikhonov regularization and for a treatment of the kernel as the 'feature' will be done to explore different interpretations of the least squares problem presented in lecture.

Using Tikhonov Regularization to Derive Weights

In this section, the closed form expression for the weights of the generalized linear model $\hat{f} = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(\boldsymbol{x})$, which is equivalent to solving the optimization problem:

$$\widehat{\mathbf{w}} = arg\min_{\boldsymbol{w}} \left(\sum_{i=1}^{N} (y^{(i)} - w_0 - \sum_{j=1}^{M-1} w_j \, \phi_j(\boldsymbol{x}^{(i)}))^2 + \sum_{i=1}^{M} \sum_{j=1}^{M} \Gamma_{ij} w_{i-1} w_{j-1} \right)$$

This equation can be rewritten in vectorized form to simplify calculations as the following:

$$\widehat{\boldsymbol{w}} = arg \min_{\boldsymbol{w}} [\,(\boldsymbol{y} - \Phi\boldsymbol{w})^T(\boldsymbol{y} - \Phi\boldsymbol{w}) + \boldsymbol{w^T\Gamma w}]$$

$$= arg \min_{\boldsymbol{w}} [\,\boldsymbol{y^Ty} - \boldsymbol{2y^T}\Phi\boldsymbol{w} + \boldsymbol{w^T}\Phi^{\boldsymbol{T}}\Phi\boldsymbol{w} + \boldsymbol{w^T\Gamma w}]$$

Defining the loss function $L = [\boldsymbol{y^Ty} - \boldsymbol{2y^T}\Phi\boldsymbol{w} + \boldsymbol{w^T}\Phi^{\boldsymbol{T}}\Phi\boldsymbol{w} + \boldsymbol{w^T\Gamma w}]$, we can solve the vectorized equation by taking the derivative of L and setting it to zero:

$$\frac{dL}{d\boldsymbol{w}} = -2\Phi^{\boldsymbol{T}}\boldsymbol{y} + \boldsymbol{2}\Phi^{\boldsymbol{T}}\Phi\boldsymbol{w} + \boldsymbol{2\Gamma w} = 0$$

$$\boldsymbol{2}\Phi^{\boldsymbol{T}}\Phi\boldsymbol{w} + \boldsymbol{2\Gamma w} = 2\Phi^{\boldsymbol{T}}\boldsymbol{y}$$

$$(\boldsymbol{2}\Phi^{\boldsymbol{T}}\Phi + \boldsymbol{2\Gamma})\boldsymbol{w} = 2\Phi^{\boldsymbol{T}}\boldsymbol{y}$$

This yields the result for the optimal $\boldsymbol{w}$:

$$\boldsymbol{w} = (\Phi^{\boldsymbol{T}}\Phi + \ \boldsymbol{\Gamma})^{-1}\Phi^{\boldsymbol{T}}\boldsymbol{y}$$

Deriving a Different Way to Compute the Dual Variable Weights

The derivation for the computational strategy to estimate $\boldsymbol{\alpha}$ is presented here. As instructed, the expression to minimize takes the form:

$$\widehat{\boldsymbol{\alpha}} = arg\ \min_{\boldsymbol{\alpha}} \left(\sum_{i=1}^{N} (y^{(i)} - \sum_{j=1}^{N} \alpha_j\, k(\boldsymbol{x}, \boldsymbol{x}^{(j)}))^2 + \lambda \sum_{i=1}^{N} {\alpha_i}^2 \right)$$

This can be rewritten in vectorized form to make calculations simpler, taking the following form:

$$\widehat{\boldsymbol{\alpha}} = arg\ \min_{\boldsymbol{\alpha}} [\ (\boldsymbol{y} - \boldsymbol{K\alpha})^T(\boldsymbol{y} - \boldsymbol{K\alpha}) + \lambda \boldsymbol{\alpha^T\alpha}]$$

$$= arg\ \min_{\boldsymbol{\alpha}} [\ \boldsymbol{y^Ty} - \boldsymbol{2y^TK\alpha} + \boldsymbol{\alpha^TK^TK\alpha} + \boldsymbol{\lambda\alpha^T\alpha}]$$

Defining the loss function as $L = \boldsymbol{y^Ty} - \boldsymbol{2y^TK\alpha} + \boldsymbol{\alpha^TK^TK\alpha} + \boldsymbol{\lambda\alpha^T\alpha}$, we can solve for the weights similarly as before:

$$\frac{dL}{d\boldsymbol{\alpha}} == -2\boldsymbol{K^Ty} + \boldsymbol{2K^TK\alpha} + \boldsymbol{2\lambda I\alpha} = 0$$

$$(\boldsymbol{2K^TK} + \ \boldsymbol{2\lambda I})\boldsymbol{\alpha} = 2\boldsymbol{K^Ty}$$

$$\boldsymbol{\alpha} = (\boldsymbol{K^TK} + \ \lambda \boldsymbol{I})^{-1}\boldsymbol{K^Ty}$$

But since $\boldsymbol{K} = \Phi^{\boldsymbol{T}}\Phi$ is symmetric, we can simplify the expression by dropping the transposes to get:

$$\boldsymbol{\alpha} = (\boldsymbol{KK} + \ \lambda \boldsymbol{I})^{-1}\boldsymbol{Ky}$$

Comparing to $\boldsymbol{\alpha} = (\boldsymbol{K} + \ \lambda \boldsymbol{I})^{-1}\boldsymbol{y}$ as derived from lecture, one can see that the two expressions are not the same. This is because the $\boldsymbol{\alpha}$ we solve for here is the weight of each kernel evaluation, whereas the $\boldsymbol{\alpha}$ derived in lecture came from the regularized weights of each feature instead. Essentially, in our derivation here, the kernel evaluation is treated as the 'feature' and the weights $\boldsymbol{\alpha}$ are found using a regularized least squares problem framing, whereas in lecture, the actual features (i.e, training points) are used instead.

To see this clearer, one could take the loss function given in lecture that solves for the weights $L(\boldsymbol{w}) = ||\boldsymbol{y} - \Phi\boldsymbol{w}||_2^{\ 2} = \ (\boldsymbol{y} - \Phi\boldsymbol{w})^T(\boldsymbol{y} - \Phi\boldsymbol{w}) + \lambda \boldsymbol{w^Tw}$ and compare it to the loss function derived here: $L(\boldsymbol{\alpha}) = (\boldsymbol{y} - \boldsymbol{K\alpha})^T(\boldsymbol{y} - \boldsymbol{K\alpha}) + \lambda \boldsymbol{\alpha^T\alpha}$**.** We can see clearly now that the kernel matrix Φ is replaced by the gram matrix **K**: so clearly, the 'feature' is treated as the kernel expressions.

RBF Regression

The performance of an exponential radial basis function will now be examined using the Mauna Loa and Rosenbrock datasets. Such a kernel has the general form $k(r) = \exp\left(-\frac{r^2}{\theta}\right)$, where θ is the shape parameter: a hyperparameter to be chosen. Additionally, the regularization parameter λ was also identified as a hyperparameter to be chosen using validation data. To choose the best hyperparameters out of the given set, RBF regression was used to compute the **α** coefficients corresponding to each set of hyperparameters using the validation set, where Cholesky Factorization was used to perform this computation. To determine the best performing hyperparameters, validation RMSE was computed between the predictions on the validation set and the actual data, and the hyperparameters exhibiting the least loss across the validation set were chosen. The results of the optimal hyperparameters across the test data is shown in Figure 1 below, which presents the test RMSE and validation RMSE.

Dataset	***Optimal Parameters***	***Validation RMSE***	***Test RMSE***
Mauna Loa	*θ = 1.0, λ = 0.001*	0.17543551317670739	0.13944932164766788
Rosenbrock	*θ = 2.0, λ = 0.001*	0.19323958697382382	*0.8139518491879253*

Figure 1: Results of the RBF Regression Analysis

To briefly comment on results, it appears that in general, larger θ and smaller λ are ideal. Since the optimal choices of λ are close to zero, the effects of regularization are small for these two datasets. However, it can be noted that a higher shape parameter was optimal for the Rosenbrock dataset. It can be speculated that this is the case because the dataset is less well-conditioned: a plot of the dataset shows that the data has an erratic trend. As such, a wider Gaussian would better fit the dataset, and a larger shape parameter is chosen.

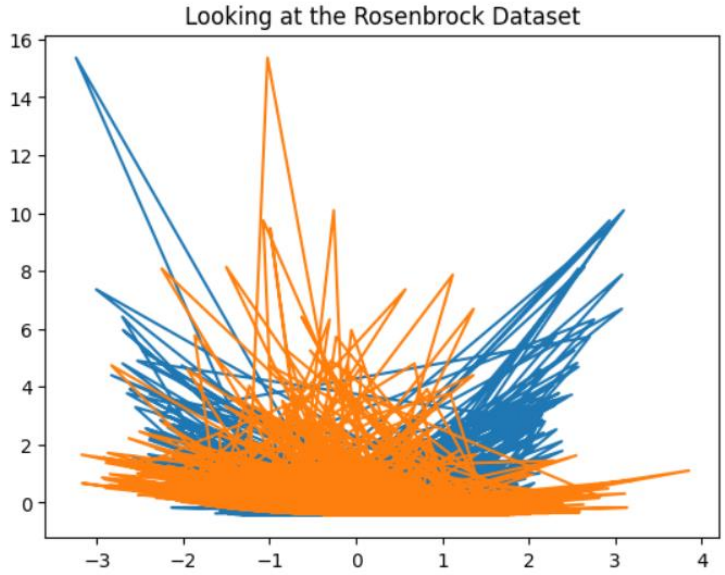

Figure 2: Observing the Shape of the Rosenbrock Dataset

Greedy Regression

To study the greedy regression algorithm, a 200 basis function dictionary was designed for the Mauna Loa dataset. To do this, the Mauna Loa training dataset was plotted to observe its shape as shown in Figure 3 below.

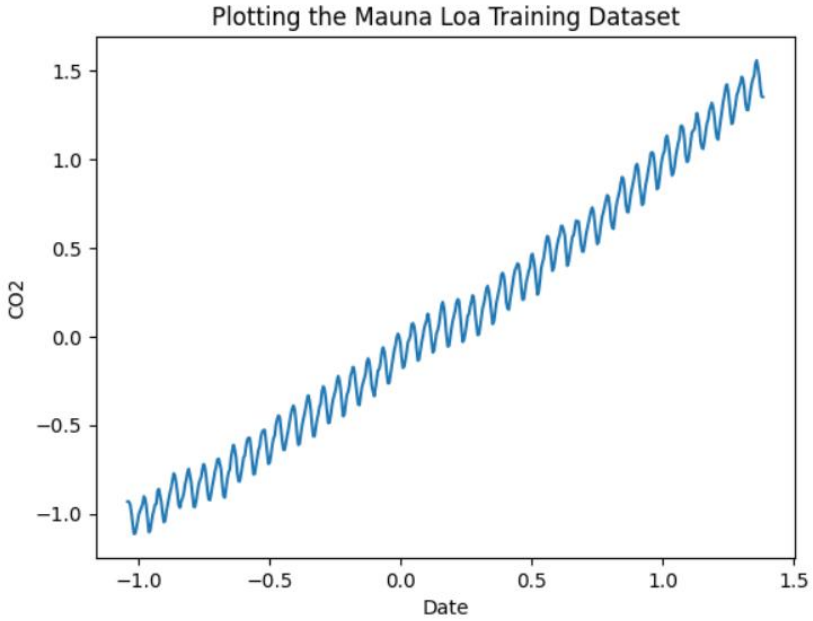

Figure 3: Observing the Shape of the Mauna Loa Dataset

The data here appears as an upward trending sinusoid, which suggests a basis function like the form $f(x) = w_1 sinax + w_2 x + w_3$. Thus, the basis dictionary was constructed using polynomial terms and sinusoidal terms with varying coefficient a. Specifically, four polynomial terms with coefficients from 0 to 3 were used, where the extra polynomial terms were introduced to test if the hypothesized basis function is correct. Higher order polynomials were not chosen because they would likely cause overfitting. The remaining 196 basis functions were split up between $\sin \pi ax$ and $\cos \pi ax$ terms equally, where the π coefficient was necessary to compress the 'waves' more: without this π on the coefficient, the model was not fitting properly.

To choose basis functions from this dictionary, the orthogonal matching pursuit metric was used as described below:

$$J(\phi_i) = \frac{(\Phi(:,i)^T \boldsymbol{r}^{(k)})^2}{\Phi(:,i)^T \Phi(:,i)}$$

Where $\Phi(:,i)$ is the i^{th}column of the Φ basis matrix, and $\boldsymbol{r^{(k)}} = \boldsymbol{y^{(k)}} - \widehat{\boldsymbol{y^{(k)}}}$. In timestep i, a function is chosen from the dictionary that reduces the training error the most using the orthogonal matching pursuit metric, and the weights for the i basis functions are

calculated as a vector using SVD on the Φ matrix. The stopping criterion used in this implementation was the MDL criterion given in the instructions, which can be calculated using:

$$MDL = \frac{N}{2}\log\left(l_2 - loss\right) + \frac{k}{2}\log N$$

When the MDL for step i exceeded the MDL for step $i-1$, the program was stopped as this would indicate that the algorithm is beginning to overfit.

With this method, the following 14 basis prediction function was generated:

$$\begin{aligned}\hat{y} = {} & -0.15\sin 70\pi x + 0.01\cos 70\pi x \\ & - 0.10\sin 35\pi x \\ & + 0.01\cos 35\pi x + 0.004\cos 10\pi x \\ & + 0.005\sin 5\pi x + 0.01\sin 4\pi x \\ & + 0.008\cos 4\pi x \\ & + 0.01\cos 2\pi x - 0.007\sin \pi x + 0.03\cos \pi x + 0.15x^2 + 0.99x - 0.02\end{aligned}$$

This resembles the expected prediction function: it can be see that the linear x term dominates the prediction function. A test RMSE of 0.0456 was yielded using this prediction, which is an indication of good accuracy, especially compared to the high RMSE error found when implementing k-NN regression. Figure 4 shows the predictions plotted on the actual data, and visually, it appears that the prediction function is very slightly overfitting the actual data, but the effect of this is quite small. However, this overfitting could be disastrous to a dataset if it is less well-shaped than the Mauna Loa dataset is. It can be observed that the prediction function performs better on 'linear' parts where the slope is not changing much, which is an interesting conclusion.

The sparsity of the model is quite good due to the selected14 basis function approximation. The model is indeed sparse because only certain select functions are chosen from the dictionary that yield the best results in fitting: so the basis functions not chosen effectively have weight zero. This leads to numerous benefits of sparsity: namely, overfitting prevention, computing memory in the fact that not every function has to be weighted, and computing storage for the same reason. Of course, overfitting is a present issue in this analysis, but the effects are quite small likely due to sparsity.

To improve results, more accurate basis functions could potentially be chosen: such as choosing decimal values for the a coefficients on the sine terms. However, the low test RMSE exhibited in this implementation indicates that the model fits quite well, and micro-improvements are not likely to affect the fitting results much.

To conclude, note that the performance is likely good because a set of basis functions was able to be identified that modelled the dataset shape well enough. If such a set is not able to be found, the greedy regression algorithm would not be a viable strategy to perform a regression analysis.

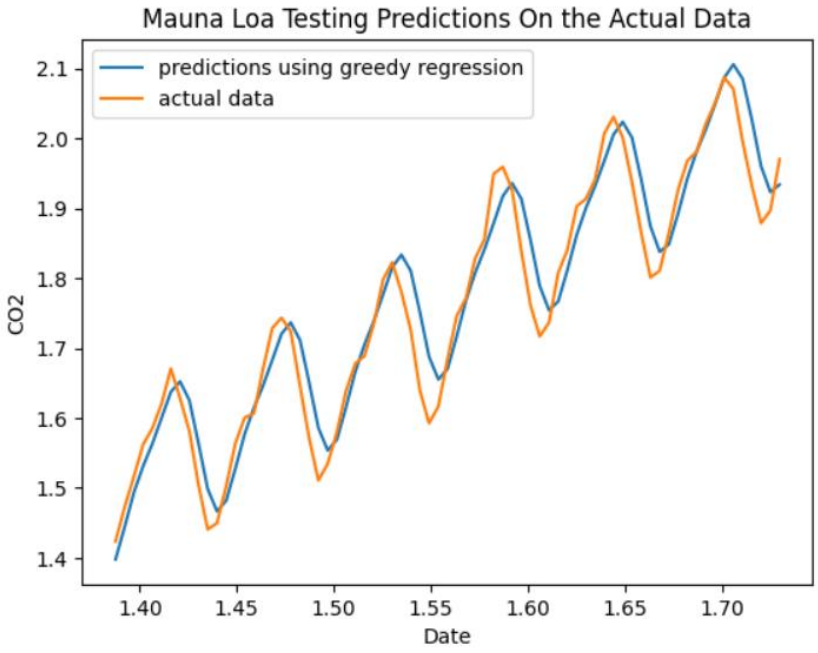

Figure 4: Resulting Plot of the Implemented Greedy Regression Algorithm

Conclusion

Using Tikhonov Regularization, the weight vector was found to be $\boldsymbol{w} = (\Phi^{\boldsymbol{T}}\Phi + \boldsymbol{\Gamma})^{-1}\Phi^{\boldsymbol{T}}\boldsymbol{y}$, derived from the least squares problem. Additionally, treating the kernelized functions as 'features' and solving for the weights $\boldsymbol{\alpha}$, a different expression was derived in comparison from the $\boldsymbol{\alpha}$ in lecture, which was derived starting with the actual features instead of the kernelized approximations.

The analysis of RBF regression yielded the conclusion that larger shape parameters θ are needed when the trend of the data is more erratic, but outliers are 'shaved off' by the regularization parameter λ. Analysis of the Rosenbrock and Mauna Loa datasets indicated choices of λ close to zero: meaning that not a lot of regularization is required for both datasets.

Finally, the greedy algorithm yielded a 14 basis sparse model with a small test RMSE. These results indicate that this greedy algorithm is the best model of regression studied in the course thus far. However, the algorithm seems to be prone to overfitting: hence why it is called a 'greedy' algorithm. For more varying datasets, it is likely that the greedy regression algorithm would not perform as well as it did for the Mauna Loa data set. Additionally, an observation that was recorded is that the greedy algorithm performs very well on linear segments, and not so well when the rate of change is varying. As such, the greedy algorithm is best used when we can find a good set of basis functions.