Ridge Regression

J.R. Powers-Luhn

***Abstract*—An examination is made of ridge regression for predicting a noisy data set. A demonstration of a technique for optimizing the regularization parameter is presented, as is a method that allows for multiple regularization parameters applied to each element of the input vector.**

I. Introduction

FITTING real-world data poses challenges beyond those that apply to modeling a well-understood, precise function. The use of actual data inevitably introduces noise into the training data set, resulting in the possibility of overfitting. This results in undesirable error when using a trained model to predict an unknown value. Ridge regression is a method to correct for this tendency. Ordinary regression seeks to minimize an error value, e.g. via the ordinary least squares method, which solves for $\vec{b}$ using equation 1.

$$\vec{b} = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\vec{y} \tag{1}$$

In the event of noisy data, however, the contribution of some parameters which are weakly correlated with the value to be predicted is likely to be overestimated and could potentially be inflated to fit noise. In order to correct for this, the cost function is modified in order to penalize ”overly complex” solutions. This new cost function is described in equation 2 [**?**].

$$C = \left|\mathbf{X}\vec{b} - \vec{y}\right|^2 + \alpha\left|\vec{b}\right|^2 \tag{2}$$

This solution for $\vec{b}$ is shown in equation 3.

$$\vec{b} = \left(\mathbf{X}^T\mathbf{X} + \mathbf{V}\vec{\alpha}^2\mathbf{V}^T\right)^{-1}\mathbf{X}^T\vec{y} \tag{3}$$

II. Methodology

A dataset of unknown provenance[1] was obtained for this analysis containing 5000 records, each consisting of 43 independent variables and one variable to be predicted (contained in the 35th column). The data were split into training (2000 records), validation (1500), and testing (1500) segments. The training data were scaled to a mean of zero with unit variance, with the parameters saved for later transformation of the testing and validation data.

TABLE I
Columns most strongly correlated with No. 35

Column	Correlation Factor
34	0.976479423106992
36	0.976444866562505
38	0.976295045635247
39	0.976378067244575

A. Optimizing α selection

In order to determine an optimal value for α, equation 3 was solved for several values in the range $[2.89, 164]$. These values were obtained by taking the singular value decomposition of the input matrix and selecting the minimum and maximum value from that range. Calculations were made using the optimized RidgeRegression object in the Python scikit-learn library[2]. Each solution (for each value of α evaluated) was used to calculate the mean squared error between the predicted values and the test data set.

A second approach was made to use the scikit-learn ridge regression cross-validation object to corroborate the results found above. 1000 candidate values of α were examined in the range 20-30.

B. Local α values

The same dataset used above was used to calculate a vector of 43 α values, one for each input variable. It was predicted that using local α values would allow for the rejection of weakly correlated or uncorrelated variables while not penalizing strongly correlated variables (see table I). A custom optimization function was derived to alter the values of $\vec{\alpha}$ to seek the best available mean squared value. Unfortunately, since this cost function had been entered manually it was not able to take advantage of the optimizations employed in the scikit-learn library. A cross validation minimizer was used with a seed value for each of the α_i of 40 (chosen arbitrarily).

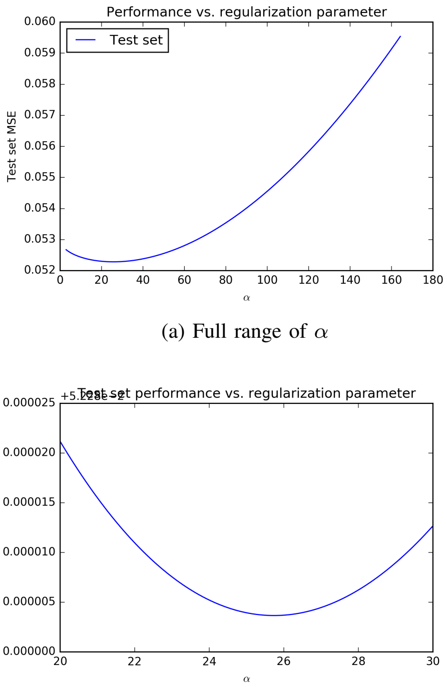
III. Results

A. Ordinary ridge regression

The mean squared error on the testing data set for each value of α can been seen in figures 1a and 1b. Visually, the minimum value appears to be just under 26. This did not align with the results of cross validation optimization, which found a minimum MSE for $\alpha \approx 20.86$. It is unclear why these values diverged, but it is theorized that this is due to differences in the magnitude of the noise in each data set. In order to check this, the minimum value for the validation set was also examined.

[1] The source of the dataset used is unknown, other than that it was provided for this assignment. A copy has been saved and is available upon request.

[2] http://scikit-learn.org/stable/modules/linear_model.html#ridge-regression



(a) Full range of α

(b) Zoom on local minimum

Fig. 1. Minimization of MSE by altering α value

An attempt was made to validate the selection of the optimal α parameter by plotting it against the magnitude of the coefficient vector, $\vec{b}$ (figure 2). While the "elbow" region of the curve was not sharply defined, it appeared to be approximately above 20, matching with the results of the cross validation calculation above.

B. Local α

A python function was written and the scipy optimize function was used to find the values for α that produced a minimum value for MSE on the test data set. The resultant vector is presented in table **??**. It was expected that the values corresponding to strongly correlated inputs would be low (to allow higher coefficients by reducing the second component of the cost function) while the other inputs would have high α values. This was born out by the code (see Appendix). The final value of the cost function (MSE) was 0.0511, slightly below the value for the cross-validation trained model (0.0523).

It was expected that the filter values resulting from the local α would also show that strongly correlated values were used while weakly correlated or uncorrelated values were heavily penalized (effectively unused). Unfortunately, at some point the code used to optimize this problem broke and data supporting or invalidating this conclusion were unavailable at the time of this report.

IV. Conclusion

Ridge regression was used to model a noisy dataset without overfitting the data. Various regularization parameters between the largest and smallest singular value decomposition were examined programmatically. While this appeared to be effective at converging on a good value for α, the values differed slightly for the training set instead of the test ad evaluation set. Further investigation will be necessary to determine the reason for this; it may have to do with the specific data involved in this data set, or it may relate to the cross validation calculator from scipy. Further investigation is warranted. An attempt was made to calculate local parameters for regression but this was unsuccessful–likely due to computer-related restrictions. Again, further work is warranted in this area.

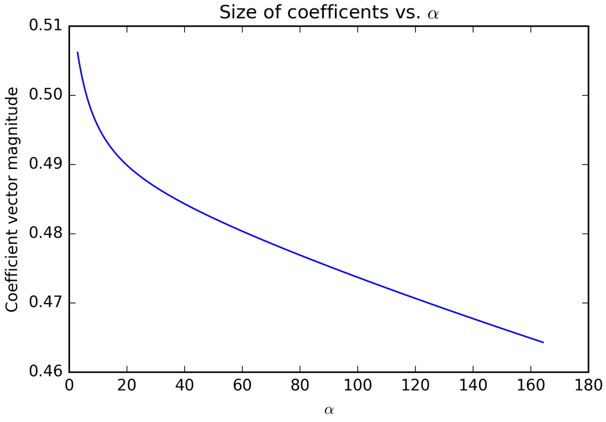


Fig. 2. Coefficient vector length vs regularization parameter

Appendix A
Python code

Calculations described in the methodology section were performed in a Jupyter notebook in Python. This code is attached to this document as an appendix.