Exploring Regularization Techniques for Small Data Sets
Ian Dardik, Heeru Ahuja

CS6140

1 Introduction

In this paper we focus on regularization techniques that improve linear regression modeling on small data sets. The classic Ordinary Least Squares (OLS) method becomes numerically unstable as the number of samples decrease, and has no solution when the number of parameters exceed the sample size. Statistical regularization can be applied to help stabilize the solution to a linear regeression problem and make modeling feasible for small data sets. In this paper we will explore and compare several options for building a regularized linear regression model in the context of a data set whose sample size is small compared to the number of parameters.

2 Methods

Statistical regularization helps stabilize our model and reduce the variance, which becomes increasingly important as the number of samples shrink. Unforunatley, regularization introduces bias into our model so we must choose an appropriate balance. The models we consider in this paper control the degree of regularization through one or two tuning parameters that are completely suppressed at zero, and influence our model more as they increase towards infinity. We can estimate the ideal tuning parameter values by using cross-validation to estimate the Mean Squared Error (MSE) for a small universe of tuning parameters, and select the parameters that minimize MSE. Each regularization technique that we consider is formulated as an optimization problem that minimizes an objective function J of the form:

$$\hat{\theta} = \arg\min_{\theta} J(\theta) = \arg\min_{\theta} \sum_{i=1}^{N} L(\theta) + \sum_{i=1}^{P} R(\theta, \lambda)$$

Where $L(\theta)$ is the loss function and $R(\theta, \lambda)$ is the regularization rule that depends on the tuning parameter λ, and sometimes an additional tuning parameter α. We often refer to the regularization rule as the *penalty*. We only consider squared loss in this paper, i.e. $L(\theta) = (Y_i - X_i'\theta)^2$, however we consider and discuss several options for choice of the regularization rule. The regularization techniques that we consider are:

- L1 penalty (LASSO)

- L2 penalty (Ridge)

- Elastic Net

- L2 with Custom Soft-Thresholding

- L$\frac{1}{2}$ penalty (not included in our Results section)

Many of the regularization techniques we discuss do not have a closed analytical solution, so we will consider two algorithms for numerically arriving at a solution: Batch Gradient Descent and Pathwise Coordinate Descent. The following sections detail how to use each regularization rule with each algorithm.

For the remainder of this paper we assume that all data X and responses Y are standardized.

2.1 Gradient Descent

Gradient Descent algorithms work by moving in the direction that the gradient of θ points, with distance scaled by a learning rate parameter α. The algorithm continuously updates it value for the parameter vector θ until convergence. Gradients point towards local optima, so we clearly need to use convex objective functions to arrive at a gobally optimal solution in Gradient Descent. Fortunately, computing the gradient of objective functions listed above is straightforward which makes Gradient Descent fairly simple to implement. Notice that Gradient Descent works by computing the gradient; it does not itself solve an optimization problem per se; since the algorithm is only dependent on the gradient of the objective function, we can parameterize it based on the gradient. Therefore we end up with a core algorithm the form: GRADIENT_DESCENT($\nabla\theta$)

2.2 Pathwise Coordinate Descent

While Gradient Descent operates on the entire parameter vector at once, Pathwise Coordinate Descent (PCD) concentrates on each parameter individually. Pathwise Coordinate Descent considers each parameter θ_j as its own optimization problem[1]:

$$\hat{\theta}_j = \arg\min_{\theta_j} J(\theta)$$

In each iteration of the PCD algorithm, we update each θ_j to be the solution to the individual optimization problem.

2.2.1 Parameterizing the PCD Algorithm

Pathwise Coordinate Descent is parameterized by the objective function J that it solves for each θ_j. However, for the universe of objective functions we consider in this paper, we can parameterize the PCD algorithm by just two nonnegative parameters we call A and B. For each objective function we consider below, we will show that the solution to optimizing each θ_j takes the form:

$$\theta_j = S\left(\sum_{i=1}^{N} X_{ij}(Y_i - \sum_{k \neq j}^{p} X_{ik}\theta_k), B\right)/A$$

Where we define[1] $S(t, B) = \text{sign}(t)(|t| - B)_+$, and $()_+$ is the soft-thresholding operator. It follows that each algorithm reduces to the following parameterized form $\text{PATHWISE\_CD}(A, B)$. In the following sections we will show how each of the objective functions we consider fit into this parameterization.

2.2.2 LASSO

LASSO uses the L1 norm for a penalty:

$$\hat{\theta}_j = \arg\min_{\theta_j} \frac{1}{2}\sum_{i=1}^{N}\left(Y_i - \sum_{k=1}^{p} x_{ik}\theta_k\right)^2 + \lambda\sum_{k=1}^{p}|\theta_k|$$

Where $\lambda \geq 0$ is a tuning parameter. Let J be the objective function we want to optimize above. We can solve for θ_j by finding the stationary points of J:

$$\begin{aligned}
\frac{\partial J}{\partial \theta_j} &= -\sum_{i=1}^{N} X_{ij}(Y_i - X_{ij}\theta_j - \sum_{k \neq j}^{p} X_{ik}\theta_k) + \lambda\,\text{sign}(\theta_j) \\
&= -\sum_{i=1}^{N} X_{ij}(Y_i - \sum_{k \neq j}^{p} X_{ik}\theta_k) + \sum_{i=1}^{N} X_{ij}^2\theta_j + \lambda\,\text{sign}(\theta_j) \\
&= -\sum_{i=1}^{N} X_{ij}(Y_i - \sum_{k \neq j}^{p} X_{ik}\theta_k) + (N-1)\theta_j + \lambda\,\text{sign}(\theta_j)
\end{aligned} \tag{1}$$

The last step follows from the fact that our data is standardized, and thus $\sum_{i=1}^{N} X_{ij}^2 = N - 1$.

Set $\frac{\partial J}{\partial \theta_j} = 0$ and, for convenience, let $Q = \sum_{i=1}^{N} X_{ij}(Y_i - \sum_{k \neq j}^{p} X_{ik}\theta_k)$ and and we get:

$$(N-1)\theta_j + \lambda\,\text{sign}(\theta_j) = Q$$

To solve for θ_j notice that when $\theta_j \geq 0$, $\theta_j = \frac{Q-\lambda}{N-1}$ so $Q \geq \lambda$. However when $\theta_j \leq 0$, $\theta_j = \frac{Q+\lambda}{N-1}$ so $Q \leq -\lambda$. Notice the following two properties:

      (1) $|Q| \geq \lambda$
      (2) $\theta_j \geq 0$ iff $Q \geq 0$ and $\theta_j \leq 0$ iff $Q \leq 0$

Therefore, we can write the solution for θ_j as:

$$\theta_j = \frac{\text{sign}(Q)(|Q| - \lambda)_+}{N - 1}$$

Where the soft-thresholding operator guarantees property (1), while multiplying by $\text{sign}(Q)$ makes use of (and guarantees) property (2). Recall that $S(t, B) = \text{sign}(t)(|t| - B)_+$, then our final solution is:

$$\theta_j = S\left(\sum_{i=1}^{N} X_{ij}(Y_i - \sum_{k \neq j}^{p} X_{ik}\theta_k), \lambda\right) / (N - 1)$$

Notice this yields a solution nearly identical to Elements of Statistical Learning[1]; the difference is that we divide by $N - 1$ and we believe this is likely a typo in the book.

Notice that the soft-thresholding parameter is λ; this explains why larger λ values causes LASSO to drive parameter values to zero. It is now clear that LASSO performs subset selection due to the choice of regularization penalty *and* algorithm.

Finally, it is clear that the parameterization for LASSO is PATHWISE_CD$(N - 1, \lambda)$.

2.2.3 Elastic Net

The Elastic Net uses the tuning parameter $\alpha \in [0, 1]$ to compromise between Ridge and Lasso penalization. Ridge penalizes the L2 norm, so Elastic Net solves the following optimization problem:

$$\hat{\theta}_j = \arg\min_{\theta_j} \frac{1}{2} \sum_{i=1}^{N} \left(Y_i - \sum_{k=1}^{p} x_{ik}\theta_k\right)^2 + \lambda \sum_{k=1}^{p} \left(\alpha\theta_k^2 + (1 - \alpha)|\theta_k|\right)$$

This math is very similar to LASSO above so we have left the details to section 5.2. Our final solution is:

$$\theta_j = S\left(\sum_{i=1}^{N} X_{ij}(Y_i - \sum_{k \neq j}^{p} X_{ik}\theta_k), B\right) / A$$

Where $A = (N - 1) + 2\lambda\alpha$, and $B = \lambda(1 - \alpha)$. This equation is already in our desired form, and hence the parameterization is PATHWISE_CD$(N - 1 + 2\lambda\alpha, \lambda(1 - \alpha))$.

2.2.4 Ridge

Ridge Regularization has a closed form analytical solution so it does not require an algorithm. Nevertheless, we will discuss Ridge in the context of PCD to realize why the L2 penalty does not perform subset selection while the L1 does. We can simply use our solution for the Elastic Net with $\alpha = 1$ for the Ridge solution. This implies that $A = (N - 1) + 2\lambda$ and $B = 0$, yielding:

$$\theta_j = S\left(\sum_{i=1}^{N} X_{ij}(Y_i - \sum_{k \neq j}^{p} X_{ik}\theta_k), 0\right) / (N - 1 + 2\lambda)$$

With parameterization PATHWISE_CD$(N - 1 + 2\lambda, 0)$. Notice that the soft-thresholding parameter is zero; this implies that *no* soft-thresholding will be applied for Ridge, regardless of the parameter λ. This explains why Ridge tends to shrink parameter values but doesn't drive them towards zero.

2.2.5 Custom Shrinkage Methods

We have found above that LASSO uses λ exclusively for linear shrinkage and soft-thresholding, while Ridge uses λ exclusively for inverse shrinkage. The Elastic Net implicitly combines both shrinkage techniques using two tuning parameters; here we propose three "custom" ideas for combining the shrinkage types using a single tuning parameter. Our hope is that using a single tuning parameter makes it easier to find the optimal bias-variance tradeoff using cross-validation. These custom ideas, inspired by Nearest Shrunken Centroids, apply soft-thresholding to the Ridge solution in the form: PATHWISE_CD($N - 1 + 2\lambda, STP(\lambda)$), where STP is the soft-thresholding penalty function. The three custom ideas we will explore are:

1. Ridge with Soft-Thresholding: PATHWISE_CD($N - 1 + 2\lambda, \lambda$)

2. Ridge with Quadratic Soft-Thresholding: PATHWISE_CD($N - 1 + 2\lambda, \lambda^2$)

3. Ridge with Exponential Soft-Thresholding: PATHWISE_CD($N - 1 + 2\lambda, e^\lambda - 1$)

2.2.6 L0.5 Norm

For the penalty term in the optimization problem, we choose q = 0.5 i.e., 0<q<1

$$\hat{\theta}_j = \text{argmin}_\theta \left(\frac{1}{2} \sum_{i=1}^{N} [Y_i - \sum_{k=1}^{p} X_{i_k} \theta_k]^2 + \lambda \sum_{k=1}^{p} \sqrt{|\theta_k|} \right)$$

The math is very similar to the one for LASSO as shown above(Section 2.2.2). So, we have left the details to the appendix.

$$\theta_j = \frac{sign(T).(|T| - \frac{\lambda}{2\sqrt{\theta_j}})}{N - 1}$$

Where, A = N-1 ; B = $\frac{\lambda}{2\sqrt{\theta_j}}$

As seen here, when 0<q<1, the solution is in terms of itself. The solution here is non-trivial or non-convex. This solution is in our desired form of Pathwise Coordinate Descent. We have left the implementation as a future scope of the project.

2.3 Experimental Design

In order to compare each regularization technique, we generated two data sets on which we trained and tested our suite of algorithms. Our models include 27 individual parameters, where $X_0...X_{25}$ are continuous and X_{26} is a categorical variable representing a month of the year. The first model includes four orthogonal design variables:
$$m_{ortho}(X) = 2X_1 + 3X_7^2 + 4X_8 + 2X_{37} + \epsilon_{ortho}$$
Where $\epsilon_{ortho} \sim \mathcal{N}(0, 4)$. Our second model includes two design variables that are each correlated with two parameters:
$$m_{corr}(X) = 2X_1^3 I_{Dec}(X_{26}) + 3X_{23}X_{24} + \epsilon_{corr}$$
Where $\epsilon_{corr} \sim \mathcal{N}(0, 1)$, and the indicator I_{Dec} is one if the given argument is "Dec" and zero otherwise. For both data sets we generated four sets of samples:

1. "$n < p$": a training data set where the number of samples is smaller than the number of parameters. We chose 20 samples.

2. "$n = p$": a training data set where the number of samples is approximately equal to the number of parameters We chose 40 samples even though we have 38 variables in our design matrix. We use 5-fold cross-valdiation in which we require the number of samples to be divisble by 5; hence the choice of 40 samples.

3. "$n > p$": a training data set where the number of samples exceeds the number of parameters. We chose 80 samples; we purposely chose a number of samples that exceeds the number of parameters, but is still relatively small.

4. "Test": a testing data set with a large number of samples for estimating the Mean Squared Error (MSE) of the fitted model. We chose 10,000 samples.

For each data set, we trained each algorithm using a design matrix that is linear in each of the 26 continuous variables, and included 12 extra indicator variables for the categorical month variable. We chose optimal tuning parameters for each algorithm using 5-fold cross-validation and then computed the test MSE on the "Test" data set. Our results are included in the following section.

3 Results

For each algorithm, we have included the Test MSE, parameter subset selection, and the number of correctly and incorrectly selected parameters. In addition, we have included three columns comparing each algorithm to the sklearn library results, where we subtract our implementation value from the library value. We have the following remarks on our results:

- Our LASSO implementation and sklearn have similar performance based on Test MSE. The margin is small, but our implementation outperforms the library on the orthogonal data set in the $n < p$ and $n > p$ case.

- Our algorithms generally outperform the library in terms of subset selection. In particular, Ridge with Exponential Soft-Thresholding either beats or ties the library in subset selection for every case, and does so by a decent margin in the orthogonal case.

- Both our algorithms and the library performed poorly on the correlated data set.

- LASSO with Batch Gradient Descent performed very poorly on subset selection, which is completely expected by the choice of algorithm.
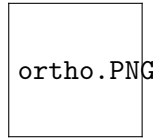
Figure 1: Orthogonal Dataset Results
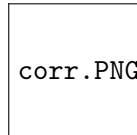
Figure 2: Correlated Dataset Results

4 Discussion

While the main focus of this paper is on comparing regularization techniques, it is worth noting that our data suggests that some form of regularization is important to stabilize linear regression. In section 5.4, the training error graphs in figures 4, 8, 12, 13, 16 and 26 show how the MSE can spike as regularization approaches zero, while the training error graphs in figures 3, 10, 24 and 25 show how the MSE can become highly volitile as regularization approaches zero.

Our results do not show a clear choice of algorithm that minimizes prediction error on our data sets. However, Ridge with Exponential Soft-Thresholding does stand out as the best performing algorithm in terms of subset selection. Unfortunately our experimental data is narrow in scope–using just two data sets, one of which saw poor performances across all algorithms–thus more testing is necessary before drawing any conclusions about Ridge with Exponential Soft-Thresholding. There may be merit in continuing to experiment with this technique: the training error graphs in figures 21, 22, and 23 of section 5.4 suggest that we may not have considered the optimal tuning parameter. The algorithms can take a long time to converge as regularization

decreases, making it tough to test all of our desired options of tuning parameters.

More generally, it is not clear that any of the custom shrinkage methods we proposed are worthwhile. However it is worth noting that squashing linear/soft-thresholding and inverse shrinkage into a single tuning parameter has a clear advantage: it is easier to implement. While Elastic Net also combines linear/soft-thresholding and inverse shrinkage into a single regularization technique, we found that it was slightly easier to implement cross-validation for a single tuning parameter and *far* easier to graph the results for a single tuning parameter. We experimented with 3D graps to no avail; we ended up plotting Elastic Net as a function of α values, choosing the best λ value at each value of α (see section 5.4 figures 27-32).

We have also found that choosing tuning parameters via cross-validation alone is not necessarily easy. Our graphs clearly show that training error is not a convex function of tuning parameters, however our analysis would have benefited greatly from a more robust method of choosing our tuning parameters.

Finally, it is worth noting that implementing the algorithms–particularly PCD–was not trivial. Understanding PCD requires understanding the optimization problem that each regularization technique uses; this was not trivial at first but led to many key insights, including the custom shrinkage methods from section 2.2.5.

References

[1] Trevor Hastie, R. T. (2009). The Elements of Statistical Learning: Data Mining, Inference and Prediction. Springer, New York, NY.

[2] Robert Tibshirani (1996). Journal of the Royal Statistical Society. Regression Shrinkage and Selection via the Lasso.

5   Appendix

5.1   Github Repository

`https://github.com/iandardik/CS6140_project`

- Algorithms, testing, and charts: pathwise_cd_final.ipynb

- Data set generation: dataset_generation.ipynb

- Testing with sklearn library implementation: Lasso with PCD.ipynb, Elastic Net with PCD.ipynb

- Data sets (orthogonal and correlated): data/

- Background math: notes/

5.2   Elastic Net Derivation

The Elastic Net uses the tuning parameter $\alpha \in [0, 1]$ to compromise between Ridge and Lasso penalization. It solves the following optimization problem:

$$\hat{\theta}_j = \arg\min_{\theta_j} \frac{1}{2} \sum_{i=1}^{N} \left(Y_i - \sum_{k=1}^{p} x_{ik}\theta_k \right)^2 + \lambda \sum_{k=1}^{p} \left(\alpha\theta_k^2 + (1-\alpha)|\theta_k| \right)$$

Let J be our objective function. We can solve for θ_j by finding the stationary points of J:

$$\frac{\partial J}{\partial \theta_j} = - \sum_{i=1}^{N} X_{ij}(Y_i - \sum_{k \neq j}^{p} X_{ik}\theta_k) + (N-1)\theta_j + 2\lambda\alpha\theta_j + \lambda(1-\alpha)\operatorname{sign}(\theta_j) \tag{2}$$

Set $\frac{\partial J}{\partial \theta_j} = 0$ and, for convenience, let $Q = \sum_{i=1}^{N} X_{ij}(Y_i - \sum_{k \neq j}^{p} X_{ik}\theta_k)$, $A = (N-1) + 2\lambda\alpha$, and $B = \lambda(1-\alpha)$ and and we get:

$$A\theta_j + B \, \text{sign}(\theta_j) = Q$$

Notice that since $A \geq 0$ and $B \geq 0$, we can use the same two case logic from the LASSO section, yielding the solution:

$$\theta_j = \frac{\text{sign}(Q)(|Q| - B)_+}{A}$$

Our final solution is:

$$\theta_j = S\left(\sum_{i=1}^{N} X_{ij}(Y_i - \sum_{k \neq j}^{p} X_{ik}\theta_k), B\right)/A$$

5.3 $L^{\frac{1}{2}}$ Norm Derivation

For the penalty term in the optimization problem, we choose q = 0.5 i.e., 0<q<1

$$\hat{\theta}_j = argmin_\theta(\frac{1}{2}\sum_{i=1}^{N}[Y_i - \sum_{k=1}^{p} X_{i_k}\theta_k]^2 + \lambda\sum_{k=1}^{p}\sqrt{|\theta_k|]}) \tag{3}$$

We consider the tuning parameter $\lambda \geq 0$. Now, we consider the above optimization problem as 'O' and take its derivative with respect to θ_j to find the stationary points.

$\frac{\partial O}{\partial \theta_j} = -\sum_{i=1}^{N} x_{ij}(Y_i - x_{ij}\theta_j - \sum_{k \neq j}^{p}x_{ij}\theta_j) + \frac{\lambda sign(\theta_j)}{2\sqrt{|\theta_j|}}$

$\frac{\partial O}{\partial \theta_j} = \sum_{i=1}^{N} x_{ij}(Y_i - \sum_{k \neq j}^{p}x_{ij}\theta_j) - \sum_{i+1}^{N}x_{ij}^2\theta_j + \frac{\lambda sign(\theta_j)}{2\sqrt{|\theta_j|}}$

Now, since our data is standardised,

$$\sum_{i+1}^{N} x_{ij}^2 = N - 1 \tag{4}$$

Setting the derivative to 0.

$$\sum_{i=1}^{N} x_{ij}(Y_i - x_{ij}\theta_j) = (N-1)\theta_j + \frac{\lambda sign(\theta_j)}{2\sqrt{|\theta_j|}} \tag{5}$$

$$T = \sum_{i=1}^{N} x_{ij}(Y_i - x_{ij}\theta_j) \tag{6}$$

As described before for LASSO, we can divide this into a 2-case argument and parameterize it in terms of A and B for implementing PCD.
The 2-case argument is:

1. T $\geq \frac{\lambda}{2\sqrt{|\theta_j|}}$

2. $\theta_j \geq 0$ if T ≥ 0 else $\theta_j < 0$ if T < 0

Finally, we can write our solution as:

$$\theta_j = \frac{sign(T)(|T| - \frac{\lambda}{2\sqrt{|\theta_j|}})}{N-1} \tag{7}$$

Where,

A = N-1 ; B = $\frac{\lambda}{2\sqrt{\theta_j}}$

As seen here, when 0<q<1, the solution is in terms of itself. The solution here is non-trivial or non-convex. This solution is in our desired form of Pathwise Coordinate Descent. We have left the implementation as a future scope of the project.
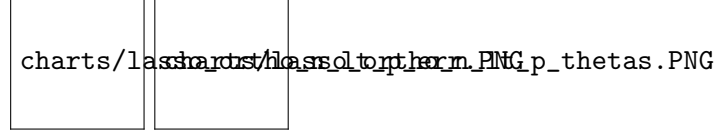
5.4 Cross-Validation Training Results

charts/lasso_rst/lasso_l_toptherm.PNG_p_thetas.PNG

Figure 3: LASSO, $n < p$ orthogonal data set

charts/lasso_rst/lasso_ce_optherm.PNG_p_thetas.PNG
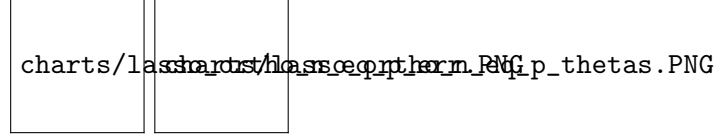
Figure 4: LASSO, $n = p$ orthogonal data set

charts/lasso_rst/lasso_cg_toptherm.PNG_p_thetas.PNG
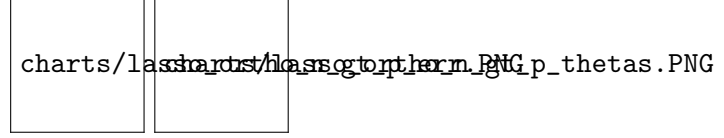
Figure 5: LASSO, $n > p$ orthogonal data set

charts/lasso_cor/lasso_t_corerm.PNG_p_thetas.PNG
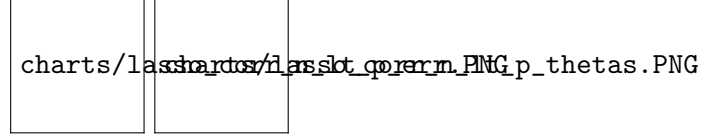
Figure 6: LASSO, $n < p$ correlated data set

charts/lasso_cor/lasso_q_corerm.PNG_p_thetas.PNG
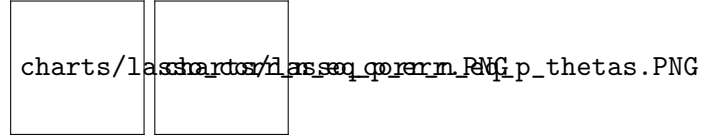
Figure 7: LASSO, $n = p$ correlated data set

charts/lasso_cor/lasso_t_corerm.PNG_p_thetas.PNG
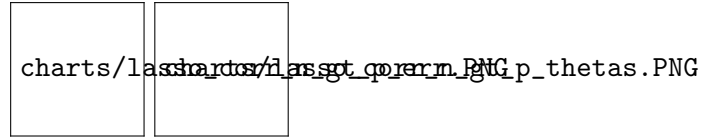
Figure 8: LASSO, $n > p$ correlated data set

charts/ridge_rst/charts/ridge_rst_l_toptherm.PNG_p_thetas.PNG
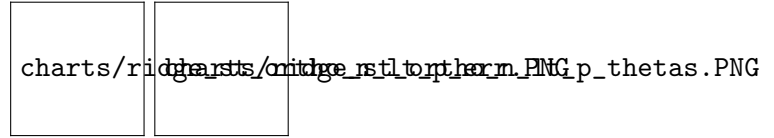
Figure 9: Ridge with Soft-Thresholding, $n < p$ orthogonal data set

charts/ridge_rst/charts/ridge_rst_e_optherm.PNG_p_thetas.PNG
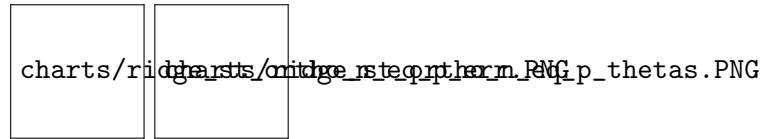
Figure 10: Ridge with Soft-Thresholding, $n = p$ orthogonal data set

charts/ridge_rsts/ridge_nstg_orthonorm.PNG charts/ridge_rsts/ridge_nstg_orthonorm_p_thetas.PNG
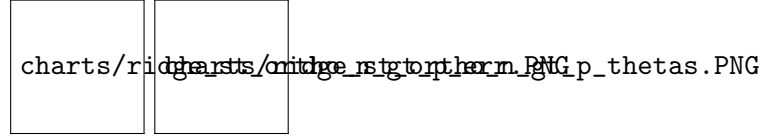
Figure 11: Ridge with Soft-Thresholding, $n > p$ orthogonal data set

charts/ridge_rsts/ridge_slt_corr.PNG charts/ridge_rsts/ridge_slt_corr_p_thetas.PNG
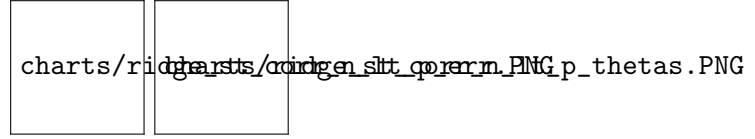
Figure 12: Ridge with Soft-Thresholding, $n < p$ correlated data set

charts/ridge_rsts/ridge_seq_corr.PNG charts/ridge_rsts/ridge_seq_corr_p_thetas.PNG
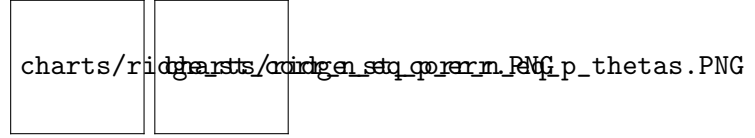
Figure 13: Ridge with Soft-Thresholding, $n = p$ correlated data set

charts/ridge_rsts/ridge_sgt_corr.PNG charts/ridge_rsts/ridge_sgt_corr_p_thetas.PNG
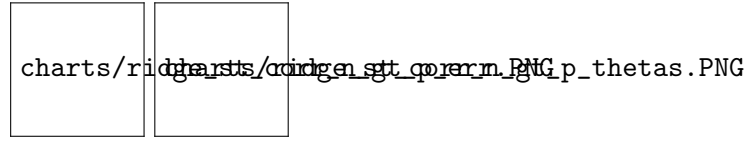
Figure 14: Ridge with Soft-Thresholding, $n > p$ correlated data set

charts/ridge_rqsts/ridge_ql_orthonorm.PNG charts/ridge_rqsts/ridge_ql_orthonorm_p_thetas.PNG
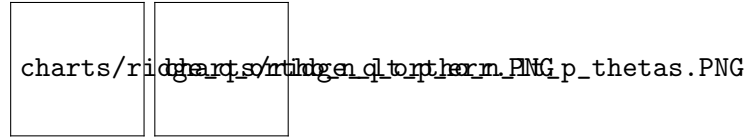
Figure 15: Ridge with Quadratic Soft-Thresholding, $n < p$ orthogonal data set

charts/ridge_rqsts/ridge_qeq_orthonorm.PNG charts/ridge_rqsts/ridge_qeq_orthonorm_p_thetas.PNG
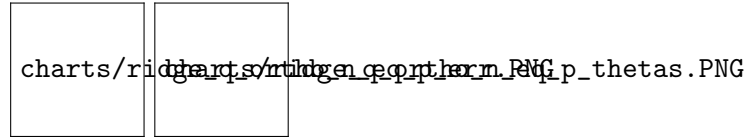
Figure 16: Ridge with Quadratic Soft-Thresholding, $n = p$ orthogonal data set

charts/ridge_rqsts/ridge_qgt_orthonorm.PNG charts/ridge_rqsts/ridge_qgt_orthonorm_p_thetas.PNG
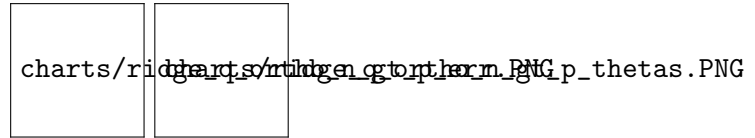
Figure 17: Ridge with Quadratic Soft-Thresholding, $n > p$ orthogonal data set

charts/ridge_rqsts/ridge_lqt_corr.PNG charts/ridge_rqsts/ridge_lqt_corr_p_thetas.PNG
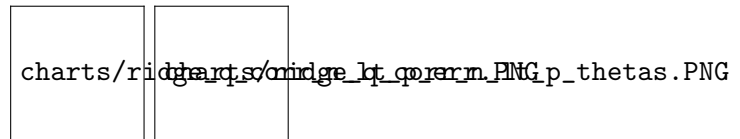
Figure 18: Ridge with Quadratic Soft-Thresholding, $n < p$ correlated data set

charts/ridge_rqsts/ridge_eq_corr.PNG charts/ridge_rqsts/ridge_eq_corr_p_thetas.PNG
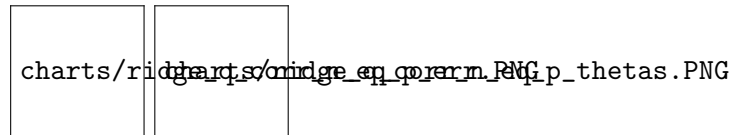
Figure 19: Ridge with Quadratic Soft-Thresholding, $n = p$ correlated data set

charts/ridge_rqsts/ridge_gt_corr.PNG charts/ridge_rqsts/ridge_gt_corr_p_thetas.PNG
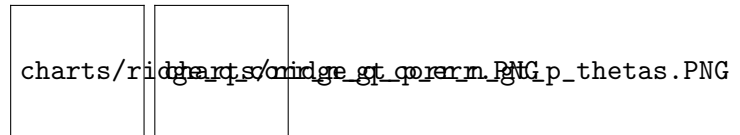
Figure 20: Ridge with Quadratic Soft-Thresholding, $n > p$ correlated data set

charts/ridge_exp/ridge_expl_to_orthonorm.PNG charts/ridge_exp/ridge_expl_to_orthonorm_p_thetas.PNG
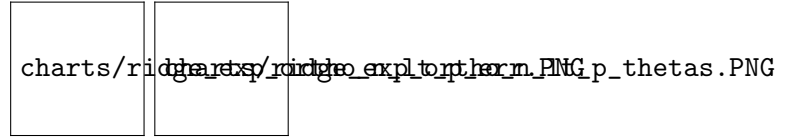
Figure 21: Ridge with Exponential Soft-Thresholding, $n < p$ orthogonal data set

charts/ridge_exp/ridge_expeq_orthonorm.PNG charts/ridge_exp/ridge_expeq_orthonorm_p_thetas.PNG
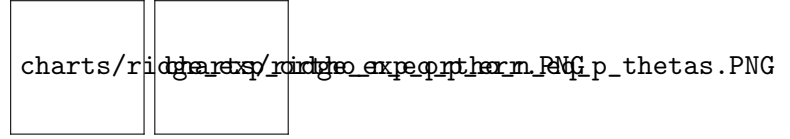
Figure 22: Ridge with Exponential Soft-Thresholding, $n = p$ orthogonal data set

charts/ridge_exp/ridge_expg_orthonorm.PNG charts/ridge_exp/ridge_expg_orthonorm_p_thetas.PNG
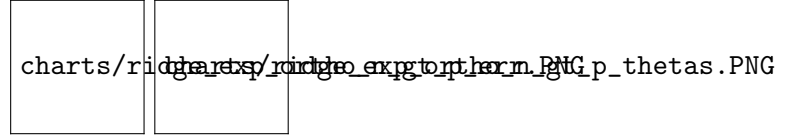
Figure 23: Ridge with Exponential Soft-Thresholding, $n > p$ orthogonal data set

charts/ridge_exp/ridge_rexpl_corr_m.PNG charts/ridge_exp/ridge_rexpl_corr_m_p_thetas.PNG
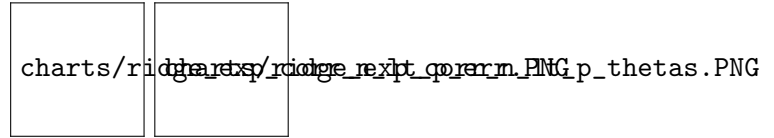
Figure 24: Ridge with Exponential Soft-Thresholding, $n < p$ correlated data set

charts/ridge_exp/ridge_rexpq_corr_m.PNG charts/ridge_exp/ridge_rexpq_corr_m_p_thetas.PNG
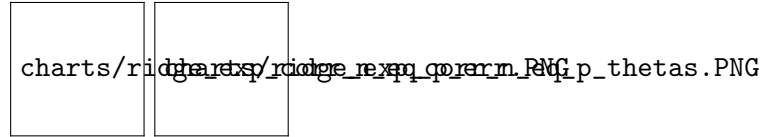
Figure 25: Ridge with Exponential Soft-Thresholding, $n = p$ correlated data set

charts/ridge_exp/ridge_rexpg_corr_m.PNG charts/ridge_exp/ridge_rexpg_corr_m_p_thetas.PNG
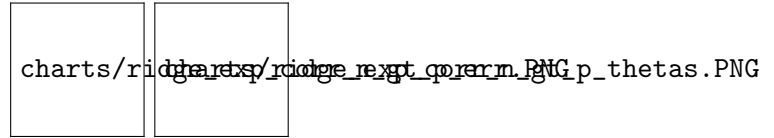
Figure 26: Ridge with Exponential Soft-Thresholding, $n > p$ correlated data set

charts/en_charts/enl_to_orthonorm.PNG charts/en_charts/enl_to_orthonorm_p_thetas.PNG
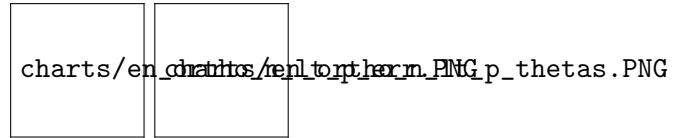
Figure 27: Elastic Net, $n < p$ orthogonal data set

charts/en_charts/eneq_orthonorm.PNG charts/en_charts/eneq_orthonorm_p_thetas.PNG
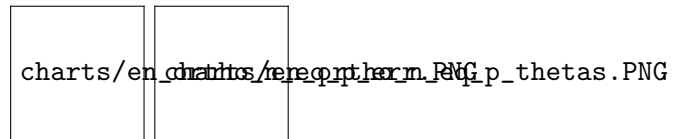
Figure 28: Elastic Net, $n = p$ orthogonal data set

charts/en_charts/eng_to_orthonorm.PNG charts/en_charts/eng_to_orthonorm_p_thetas.PNG
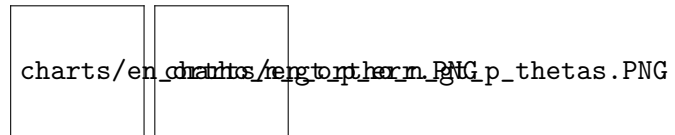
Figure 29: Elastic Net, $n > p$ orthogonal data set

charts/en_charts/enl_corr_m.PNG charts/en_charts/enl_corr_m_p_thetas.PNG
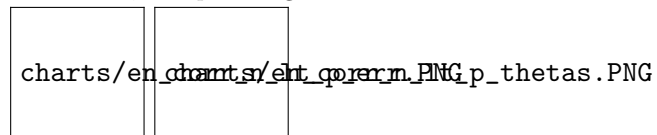
Figure 30: Elastic Net, $n < p$ correlated data set

charts/en_charts/eq_corr_m.PNGp_thetas.PNG
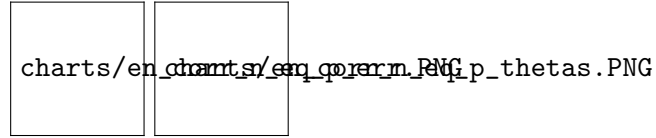
Figure 31: Elastic Net, $n = p$ correlated data set

charts/en_charts/egt_corr_m.PNGp_thetas.PNG
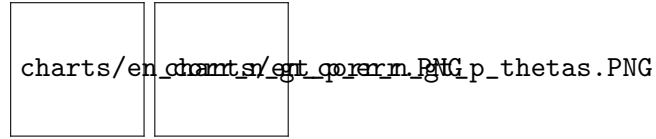
Figure 32: Elastic Net, $n > p$ correlated data set

charts/bgd_charts/bgd_lt_orthor_m.PNGp_thetas.PNG
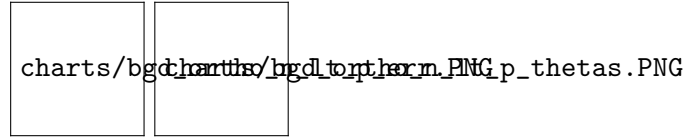
Figure 33: LASSO with Batch Gradient Descent, $n < p$ orthogonal data set

charts/bgd_charts/bgd_eq_orthor_m.PNGp_thetas.PNG
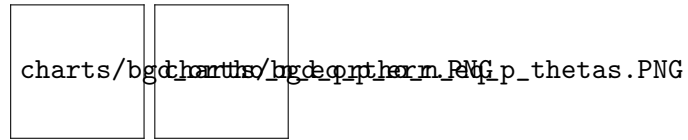
Figure 34: LASSO with Batch Gradient Descent, $n = p$ orthogonal data set

charts/bgd_charts/bgd_gt_orthor_m.PNGp_thetas.PNG
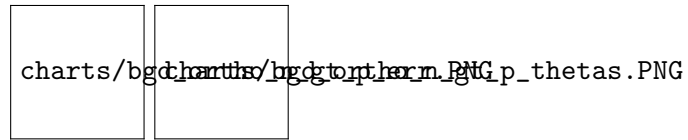
Figure 35: LASSO with Batch Gradient Descent, $n > p$ orthogonal data set

charts/bgd_charts/bgd_lt_corr_m.PNGp_thetas.PNG
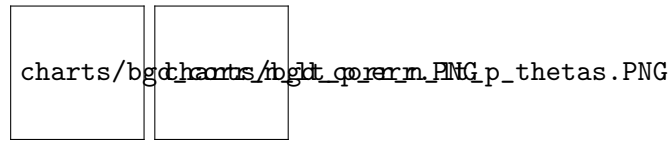
Figure 36: LASSO with Batch Gradient Descent, $n < p$ correlated data set

charts/bgd_charts/bgd_eq_corr_m.PNGp_thetas.PNG
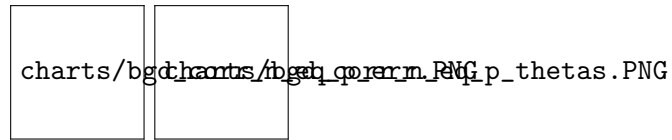
Figure 37: LASSO with Batch Gradient Descent, $n = p$ correlated data set

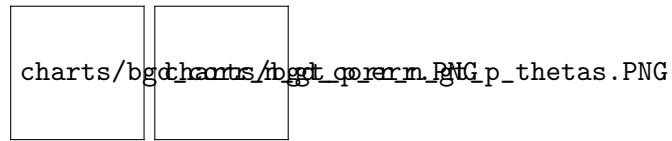charts/bgd_charts/bgd_gt_corr_m.PNGp_thetas.PNG

Figure 38: LASSO with Batch Gradient Descent, $n > p$ correlated data set

6 Statement of Contribution

Ian Dardik

- Theory for LASSO, Elastic Net, Ridge (including all custom methods)

- Implementation of BGD (from homework), PCD, and k-fold cross validation.

- Generating and compiling results, charts, and tables

- Worked on data generation and generated our data sets, also worked on properly using sklearn LASSO

Heeru Ahuja

- Theory for $L\frac{1}{2}$ regularization

- Foundation for implementing data generation and using sklearn LASSO and Elastic Net

- Implementation of cyclic coordinate descent with LASSO-exempted due to redundancy

- Compilation of documentation for Project Proposal,Report and Presentation