Regularization in Regression: Ridge, Lasso, and Beyond

George Michailidis

gmichail@ucla.edu

STAT 102B

The Problem of Overfitting

- The goal of building predictive statistical and machine learning models (linear/logistic regression, MLP) is to generalize from training data to yet unseen data
- Overfitting: Model performs well on training data, but poorly on test data
- Causes of overfitting:
  - Too many features relative to observations
  - Excessive model complexity
  - Noise in the training data
- Need for regularization: Finding the right balance between model fit and model complexity

Remarks on Overfitting and the Use of a Validation Data Set

Recall that a portion of the data was reserved as a validation set when training an MLP, to select a robust MLP that performs well on the test data set

In statistics, cross-validation is often used to select a robust model

Cross-Validation vs Proper Validation Set

- Cross-Validation (CV) mechanics:
  - A resampling technique used to estimate model performance
  - Commonly used form: k-fold cross-validation
  - Splits data into k subsets, trains on $k-1$ and validates on the remaining one; repeats k times
  - Performance is averaged over all k folds

- Proper Validation Set mechanics:
  - A fixed portion of the data (e.g., 20%) set aside for tuning hyperparameters or selecting models
  - Not used during training
  - Helps prevent overfitting to the test set during model selection.

Cross-Validation or Validation Set? When to Use Which Strategy

Use Cross-Validation When:

- Data are limited and need to be used efficiently
- Comparing multiple models or tuning hyperparameters
- Need a robust estimate of generalization error
- Performing model selection or evaluation

Use a Proper Validation Set When:

- Ample data available to split into train/val/test
- Tuning hyperparameters or performing early stopping
- Building a production pipeline
- Want to avoid data leakage and ensure test set purity

What is Regularization?

- A technique to prevent overfitting by adding a regularization/penalty term to the loss function
- General form:

$$\min_{\theta} \underbrace{L(\theta; X, y)}_{\text{Loss function}} + \underbrace{\lambda R(\theta)}_{\text{Regularization term}}$$

  where X contains the data of the predictors, y the data for the response (numerical in a regression, categorical in a classification model), and θ is the model parameter we optimize using the algorithms covered in class thus far
- Benefits:
  - Prevents model from fitting noise
  - Improves generalization
  - Can provide feature selection
  - Deals with multicollinearity
- The tuning parameter λ controls the regularization strength
- A good value of λ can be selected through a validation set

Hence, the objective function becomes

$$\min_{\theta} f(\theta) = L(\theta; X, y) + \lambda R(\theta)$$

Widely used regularization/penalty terms

Let $\theta \in \mathbb{R}^p$

1. $\|\theta\|_2^2 = \theta^\top \theta = \sum_{j=1}^p \theta_j^2$
   this is the regularization term that leads to ridge regression
2. $\|\theta\|_1 = \sum_{j=1}^p |\theta_j|$
   this is the regularization term that leads to lasso (Least Absolute Shrinkage and Selection Operator) regression
3. $\|\theta\|_2 = \sqrt{\theta^\top \theta}$ this is the regularization term that leads to group lasso regression

Ridge Regression: Overview

- Objective function:

$$\min_{\beta} \frac{1}{2n} \|X\beta - y\|_2^2 + \lambda \|\beta\|_2^2$$

- Shrinks all coefficients toward zero proportionally
- Particularly useful when dealing with multicollinearity

Ridge Regression: Mathematical Formulation

$$\hat{\beta}_{ridge} = \arg\min_{\beta} \left\{ \frac{1}{2n} \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{p} x_{ij} \beta_j \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\}$$

$$= \arg\min_{\beta} \left\{ \frac{1}{2n} \|X\beta - y\|_2^2 + \lambda \|\beta\|_2^2 \right\}$$

- $\lambda \geq 0$ is the regularization parameter
- As $\lambda \to 0$: Ridge solution approaches the least squares solution
- As $\lambda \to \infty$: The values of the regression coefficients become close to zero

The Objective Function and its Gradient

$$L(\beta) = \frac{1}{2n}\left[(y - X\beta)^\top (y - X\beta) + \lambda\beta^\top \beta\right]$$

$$= \frac{1}{2n}\left[y^\top y - 2\beta^\top X^\top y + \beta^\top X^\top X\beta + \lambda\beta^\top \beta\right]$$

$$= \frac{1}{2n}\left[y^\top y - 2\beta^\top X^\top y + \beta^\top (X^\top X + \lambda I)\beta\right]$$

Gradient of the Objective Function

$$\nabla_\beta L(\beta) = \frac{1}{n}\left[-X^\top y + (X^\top X + \lambda I)\beta\right]$$

Solve the gradient equation: $\nabla_\beta L(\beta) = 0$

$$-X^\top y + (X^\top X + \lambda I)\hat{\beta}_{\text{ridge}} = 0$$

$$(X^\top X + \lambda I)\hat{\beta}_{\text{ridge}} = X^\top y$$

Ridge Regression: Closed-Form Solution

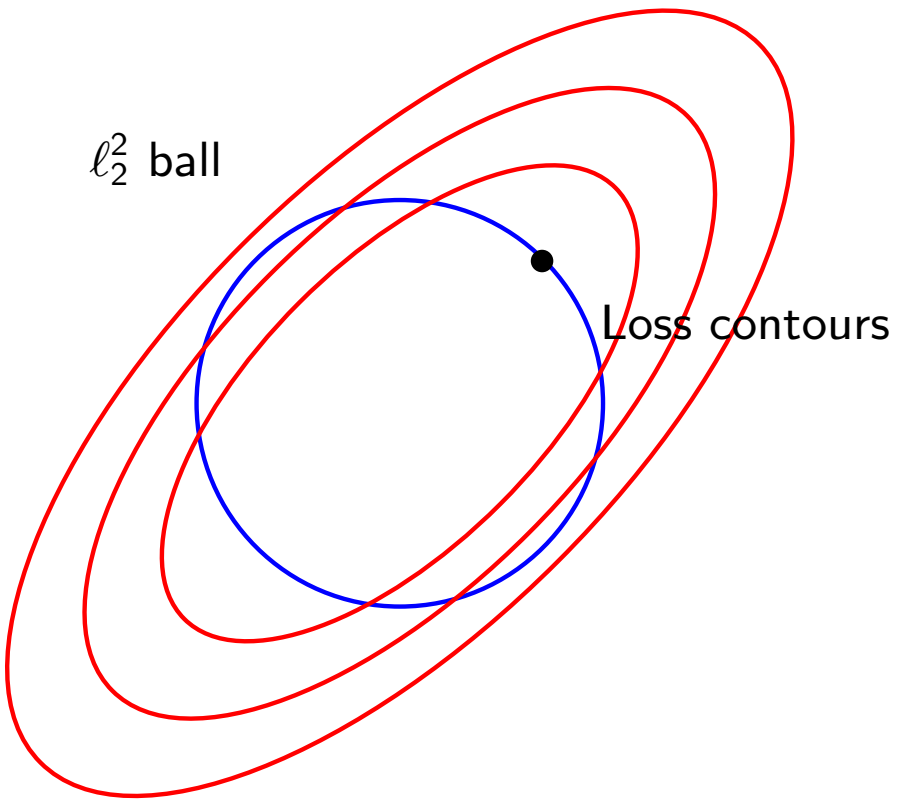
- Ridge regression has a convenient closed-form solution:

$$\hat{\beta}_{ridge} = (X^T X + \lambda I)^{-1} X^T y$$

- Computational advantages:
  - Adding λI ensures matrix invertibility (even with collinearity)
  - Addresses ill-conditioned problems
  - Stabilizes numerical computations

Geometric Interpretation of Ridge Regression

- The regularization term defines a circle/sphere/hypersphere centered at $\vec{0}$
- Loss function contours are elliptical (their shape depends on how correlated the variables are)
- Solution occurs where the ellipse touches the circle
- Smooth constraint region means coefficients are rarely exactly zero



Digression: Ridge Regression and Effective Degrees of Freedom

- Ridge introduces the concept of "effective degrees of freedom" (df)
- For OLS: df = p (number of parameters)
- For Ridge: df = trace$(X(X^TX + \lambda I)^{-1}X^T)$
- Can be expressed in terms of singular values:

$$\text{df}(\lambda) = \sum_{j=1}^{p} \frac{d_j^2}{d_j^2 + \lambda}$$

where d_j are singular values of X

- As λ increases, effective df decreases
- Provides a measure of model complexity

Gradient Descent for Ridge Regression - I

Objective Function:

$$L(\beta) = \frac{1}{2n}\|y - X\beta\|_2^2 + \lambda\frac{1}{2}\|\beta\|_2^2$$

Remark:
Note that in the new objective function, the regularization term is scaled by $\frac{1}{2}$ instead of $\frac{1}{2n}$

This would allow us to use small positive values of λ, irrespective of the number of observations in the data set

Gradient:

$$\nabla_\beta L(\beta) = -\frac{1}{n}\left[X^\top(y - X\beta)\right] + \lambda\beta$$

Gradient Descent Update:

$$\beta_{k+1} = \beta_k - \eta\left[-\frac{1}{n}\left(X^\top(y - X\beta_k)\right) + \lambda\beta_k\right]$$

Gradient Descent for Ridge Regression - II

Remarks:

- η is the step size and can be selected based on the strategies previously presented
- The $\lambda \geq 0$ parameter is fixed and its best value is selected based on the validation set
- GD stops when one of the standard stopping criteria is met; e.g., $\|\beta_{k+1} - \beta_k\|_2$

Illustration - I

Consider a data set comprising $n = 500$ observations and $p = 200$ predictors

There is significant multicollinearity in the data;
one metric used to reflect that is the condition number, defined as

$$\text{CN} = \frac{\lambda_{\max}\left(\frac{X^\top X}{n}\right)}{\lambda_{\min}\left(\frac{X^\top X}{n}\right)}$$

For this data set, CI$\approx$ 3000

The results based on backtracking line search and different values of the tuning parameter λ are shown next

Illustration - II

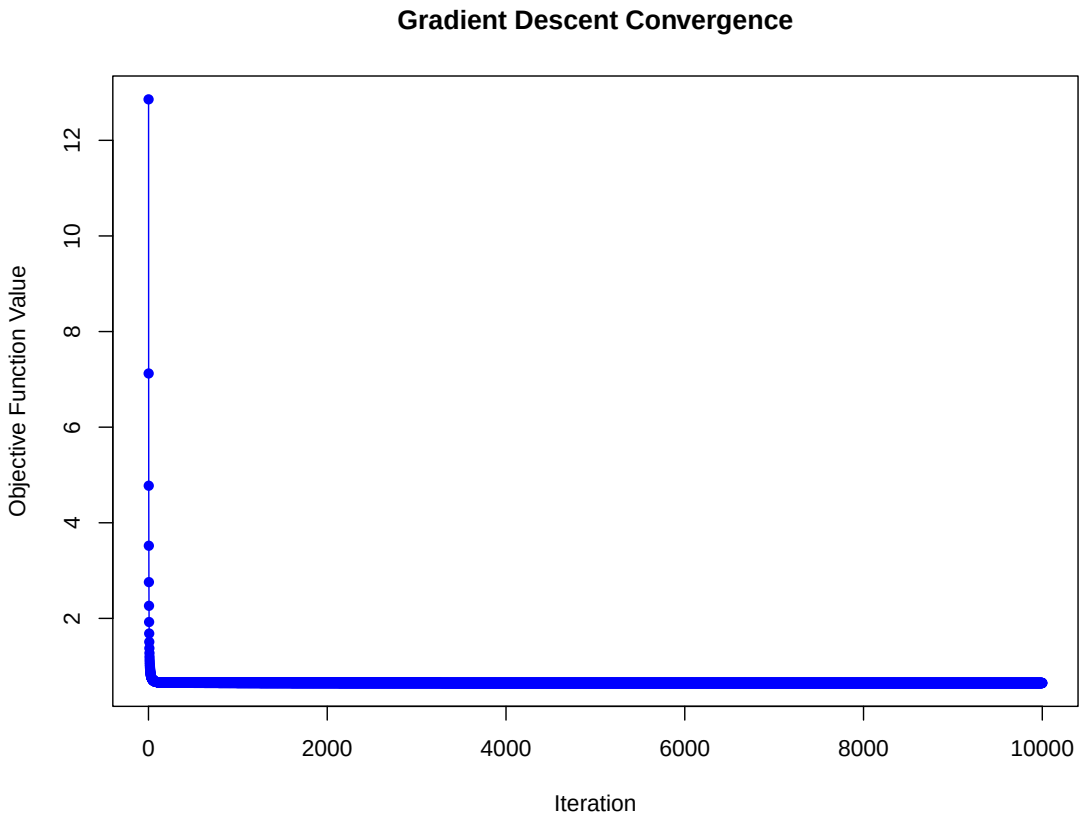


Figure 1: $\lambda = 0$, $\|\beta_{\text{true}} - \beta_{\text{ridge}}\|_2 = 0.42$

Illustration - III

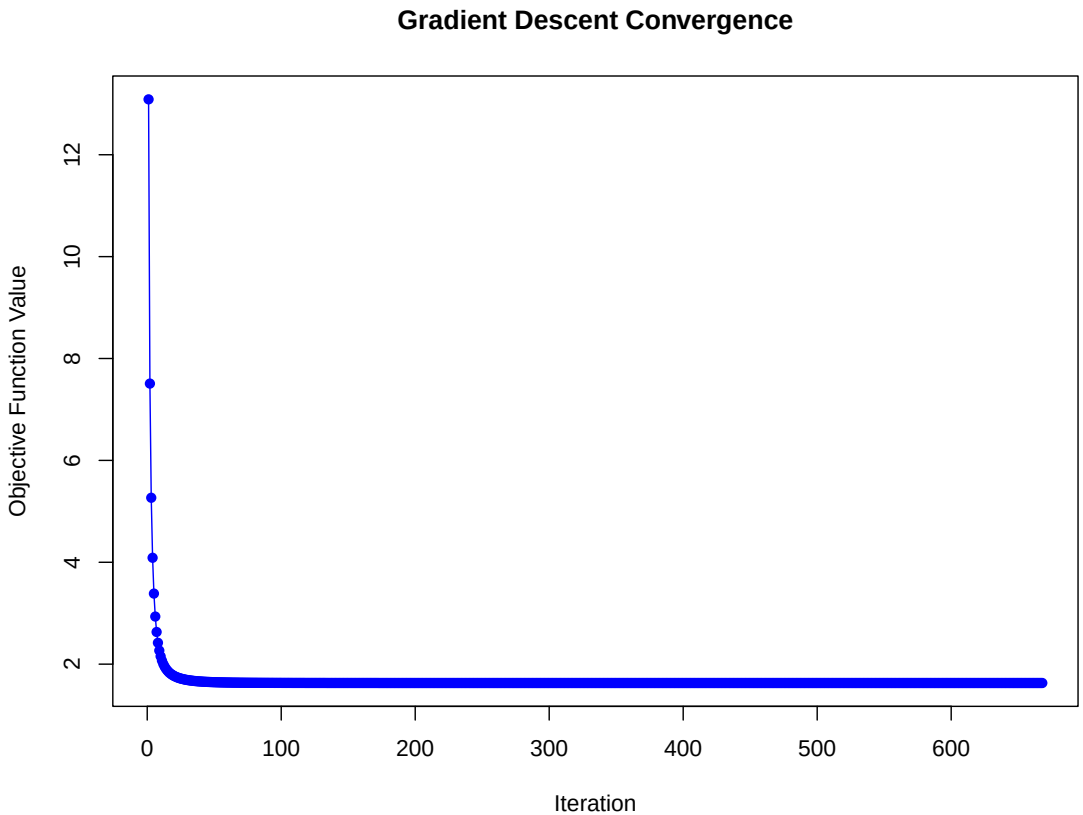


Figure 2: $\lambda = 0.8$, $\|\beta_{\text{true}} - \beta_{\text{ridge}}\|_2 = 0.28$

Illustration - IV

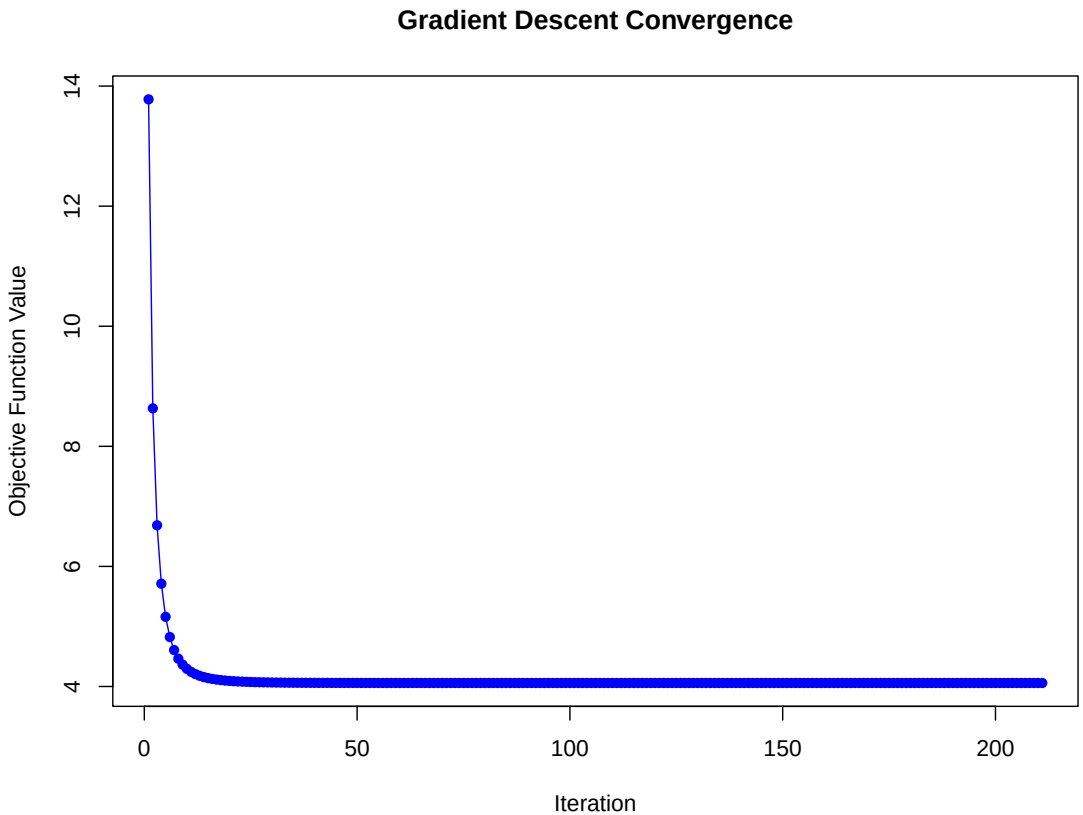


Figure 3: $\lambda = 2.0$, $\|\beta_{\text{true}} - \beta_{\text{ridge}}\|_2 = 0.38$

Illustration - V

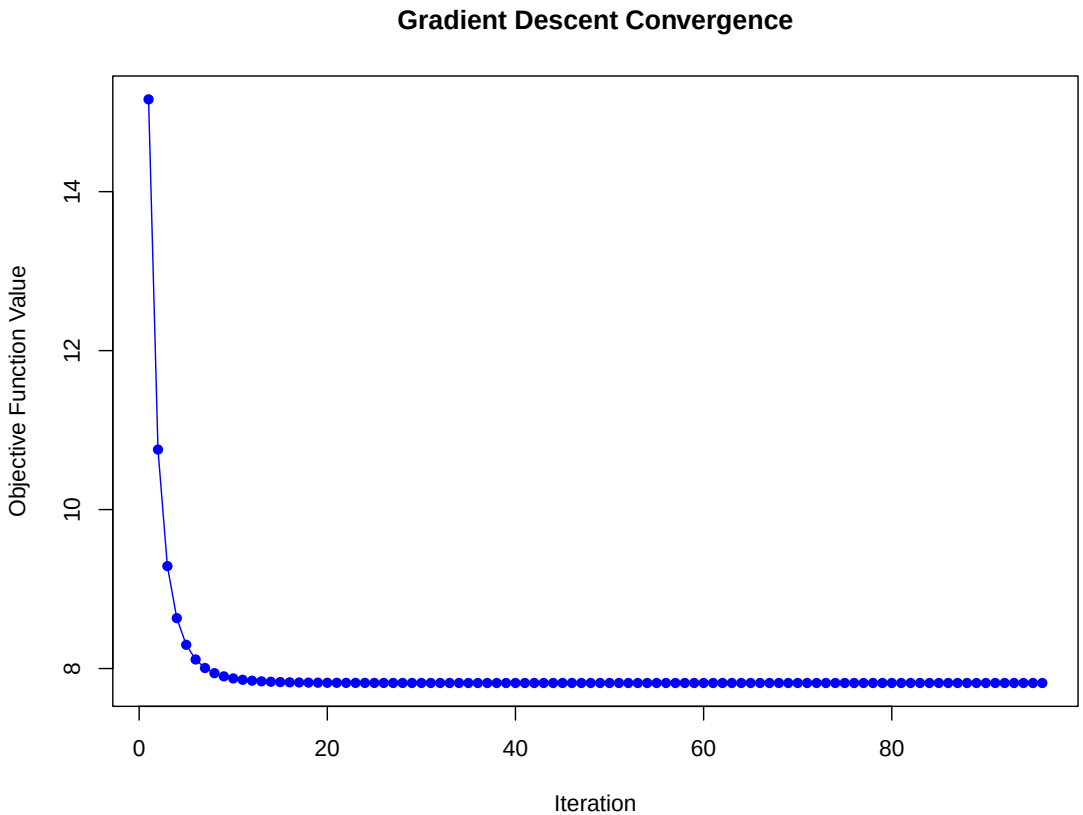


Figure 4: $\lambda = 2.0$, $\|\beta_{\text{true}} - \beta_{\text{ridge}}\|_2 = 0.53$

Illustration - VI

It can be seen that as λ increases, more shrinkage (values get closer to 0) is applied to the regression coefficients

Statistically, the ridge estimates become more biased, but their variability reduces

This is an instantiation of the bias-variance tradeoff in the context of regression

Bias-Variance Tradeoff: Balancing Fit and Model Complexity

- Goal: Minimize expected prediction error:

$$\mathbb{E}\left[(y - \hat{f}(x))^2\right] = [\text{Bias}(\hat{f}(x))]^2 + \text{Var}(\hat{f}(x)) + \sigma^2$$

- Bias: It corresponds to the expected difference between the predicted value $\hat{y} = \hat{f}(x)$ and the actual value y for new values of the predictor x
- Variance: It corresponds to the variance of the predicted values
- Tradeoff:
  - Low bias $\rightarrow$ typically means high variance (overfitting)
  - Low variance $\rightarrow$ typically means high bias (underfitting)
- Solution: Select "model complexity" that balances bias and variance (e.g., via validation set)

Lasso Regression: Overview

- LASSO = Least Absolute Shrinkage and Selection Operator
- Uses ℓ_1 regularization: $\sum_{j=1}^{p} |\beta_j|$
- Key property: Produces sparse solutions (many regression coefficients are shrunk to exactly zero)
- Performs simultaneous variable selection and regularization
- Particularly useful for high-dimensional data when $p > n$

Lasso Regression: Mathematical Formulation

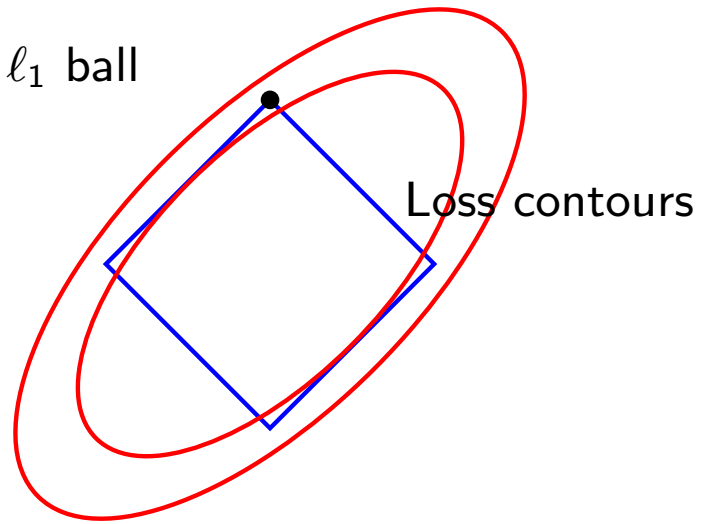
$$\hat{\beta}_{lasso} = \arg\min_{\beta} \left\{ \frac{1}{2n} \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{p} x_{ij} \beta_j \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\}$$

$$= \arg\min_{\beta} \left\{ \frac{1}{2n} \|y - X\beta\|_2^2 + \lambda \|\beta\|_1 \right\}$$

- $\lambda \geq 0$ controls the strength of regularization
- As $\lambda \to 0$: Lasso solution approaches the least squares solution
- As $\lambda \to \infty$: All coefficients become exactly zero
- No closed-form solution due to non-differentiability of the ℓ_1 norm

Geometric Interpretation of Lasso

- The ℓ_1 constraint defines a diamond / hyper-diamond
- Loss function contours are elliptical
- Solution often occurs at a corner of the diamond
- Corners correspond to sparse solutions (some $\beta_j = 0$)
- Explains why Lasso performs variable selection



Lasso: Why ℓ_1 regularization produces sparse regressions coefficients

- Lasso applies a constant penalty to any non-zero coefficient
- Small coefficients incur almost the same penalty as large ones
- Often more efficient to set small coefficients to exactly zero
- Contrast with Ridge: Penalty is proportional to coefficient magnitude
- In high dimensions (very large number of predictors p relative to the sample size n):
  - Ridge: Many small non-zero coefficients
  - Lasso: Few non-zero coefficients, many zero ones
- Lasso performs feature selection - valuable for interpretation

Challenges in Lasso Optimization

- The Lasso objective function combines:
  - Smooth, differentiable term: $f(\beta) = \frac{1}{2n}\|X\beta - y\|_2^2$
  - Non-smooth, non-differentiable term: $g(\beta) = \lambda\|\beta\|_1$
- No closed-form solution exists (unlike Ridge regression)
- Standard gradient descent fails due to non-differentiability;
  since the gradient of $g(\beta)$ is not defined everywhere, GD can not be operationalized
- A special variant of the standard gradient descent algorithm:
  proximal gradient

Introduction to Proximal Gradient Methods

Back to general notation

- Designed for optimization problems of the form:

$$\min_x F(x) = f(x) + g(x), \quad x \in \mathbb{R}^n$$

where:

- f is differentiable and assumed to possess a global minimum (not necessarily unique)
- g is NOT differetiable, but possesses a global minimum

- Key idea: Split the problem into differentiable and non-differentiable parts
- Lasso regression is an instance of this optimization problem:
  - $f(\beta) = \frac{1}{2n}\|X\beta - y\|_2^2$ (differentiable)
  - $g(\beta) = \lambda\|\beta\|_1$ (non-differentiable)

Digression: the Proximal Operator

- For a function g and parameter $t > 0$, the proximal operator is:

$$\text{prox}_{t,g}(z) = \arg\min_x \left\{ g(x) + \frac{1}{2t}\|x - z\|_2^2 \right\}$$

- Intuitively: Find a point that:
  - Minimizes g
  - Stays close to the input point z
  - Parameter t balances these objectives
- For selected functions g, the proximal operator has a closed-form solution

The Proximal Gradient Descent Algorithm - I

$$x_{k+1} = \text{prox}_{\eta_k, g}\left(x_k - \eta_k \nabla f(x_k)\right)$$

- η_k: step size
- $\text{prox}_{\eta_k, g}(z) := \arg\min_x \left\{ \frac{1}{2\eta_k} \|x - z\|^2 + g(x) \right\}$.
- Generalizes gradient descent to handle non-differentiable regularization terms

The Proximal Gradient Descent Algorithm - II

Essentially, the proximal gradient algorithm can be decomposed into the following two steps:

1. Regular GD update based on the differentiable f function only, at current value x_k:
$$y_k = x_k - \eta_k \nabla f(x_k)$$
2. Proximal based update incorporating the non-differentiable function g
$$x_{k+1} = \text{prox}_{\eta_k, g}(y_k)$$

Note that the generic parameter t in the definition of the proximal operator, corresponds to the step size in the implementation of the proximal gradient algorithm

Soft-Thresholding: Proximal Operator for ℓ_1 Norm

- For $g(x) = \lambda\|x\|_1$, the proximal operator is the soft-thresholding function:

$$\text{prox}_{t,\lambda\|\cdot\|_1}(z)_i = S_{t,\lambda}(z_i) = \begin{cases} z_i - t\lambda & \text{if } z_i > t\lambda \\ 0 & \text{if } |z_i| \leq t\lambda \\ z_i + t\lambda & \text{if } z_i < -t\lambda \end{cases}$$

- or more compactly:

$$S_{t,\lambda}(z) = \text{sign}(z) \cdot \max(|z| - t\lambda, 0)$$

- Key property: Automatically produces sparse solutions by setting small values to zero

Proximal Gradient Descent Algorithm for Lasso Regression

- **Initialize:** $\beta_0 \in \mathbb{R}^p$, step size $\eta > 0$
- **for** $k = 0, 1, 2, \ldots$ until stopping criterion is satisfied **do**
  - Gradient step: $\tilde{\beta}_k = \beta_k - \eta_k \nabla f(\beta_k)$,
    where $\nabla f(\beta) = -\frac{1}{n}\left[X^\top (y - X\beta)\right]$ (the gradient of the SSE(β) function)
  - Proximal step: $\beta_{k+1} = \text{prox}_{\eta_k, \lambda\|\cdot\|_1}(\tilde{\beta}_k)$ {Apply soft-thresholding}
- **end for**
- **Return:** $\hat{\beta}_{\text{lasso}}$

Essentially alternates between:

- A standard gradient descent step on the differentiable component of the objective function
- A proximal operation to handle the non-differentiable component of the objective function

Illustration - I

Consider a data set comprising $n = 400$ observations and $p = 500$ predictors

Further, only 5% of the regression coefficients of the predictors are non-zero

The least squares solution does NOT exist, since $X^\top X$ is not invertible

The results based on backtracking line search and different values of the tuning parameter λ are shown next

Illustration - II
$\lambda = 0.15$

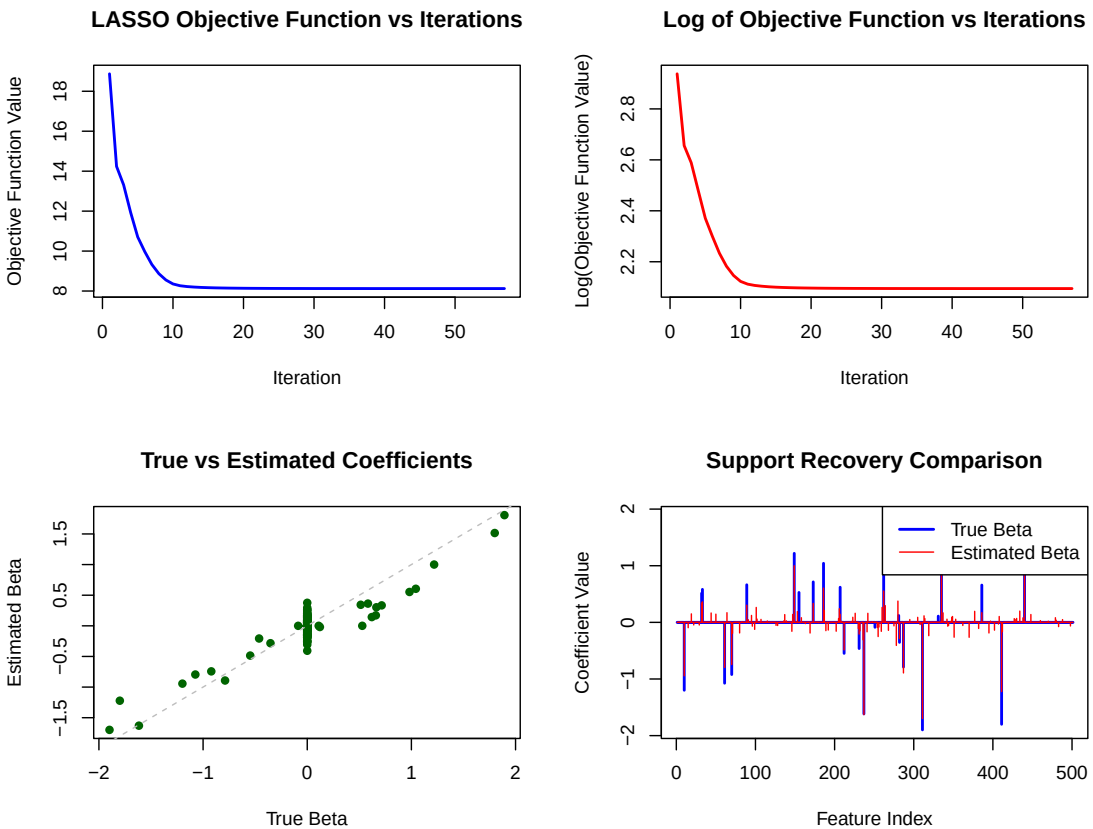


Figure 5: TP=23, FP=104, TN=371, FN=2

Illustration - III
$\lambda = 0.35$

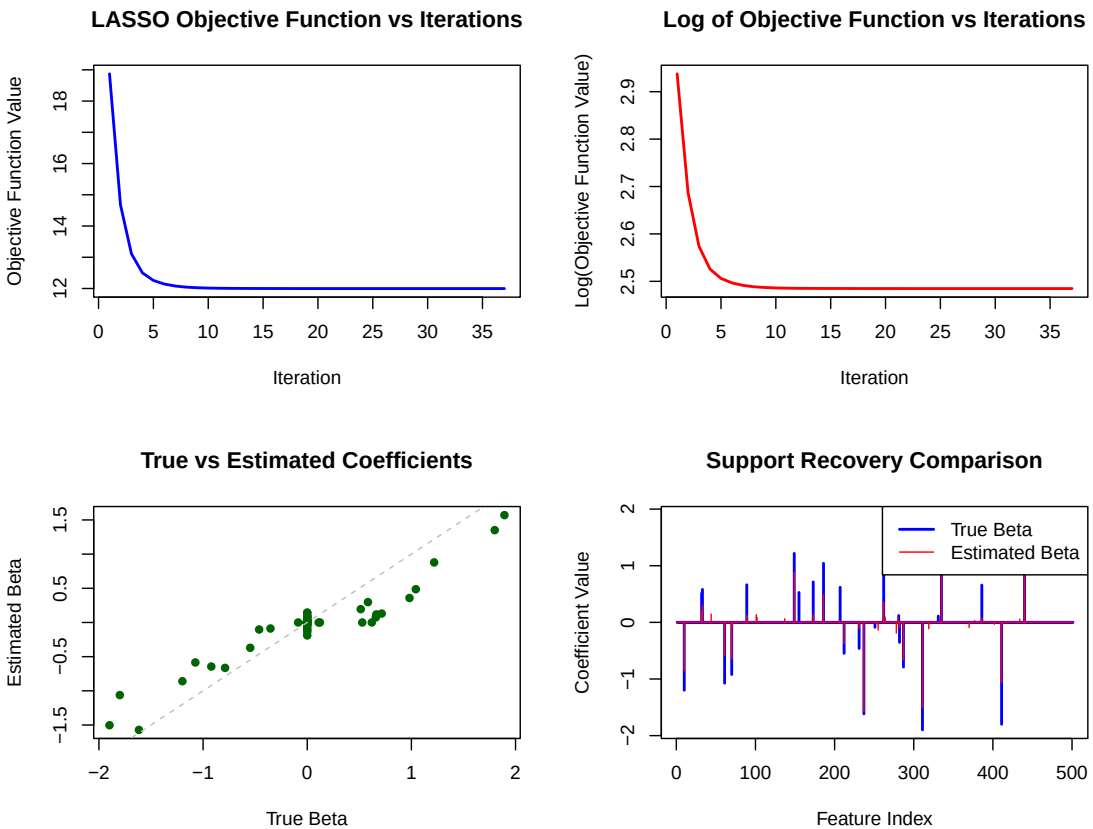


Figure 6: TP=19, FP=1, TN=474, FN=6

Illustration - IV
$\lambda = 0.50$

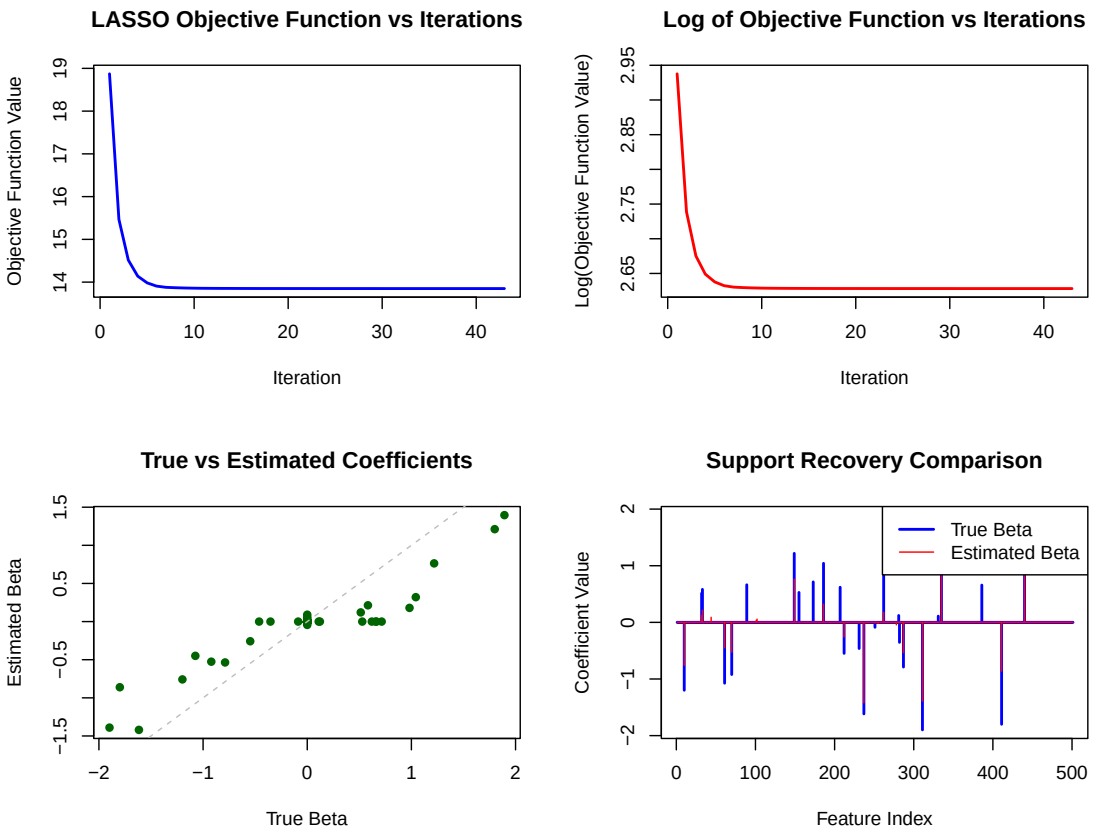


Figure 7: TP=15, FP=6, TN=469, FN=10

Illustration - V
$\lambda = 1$

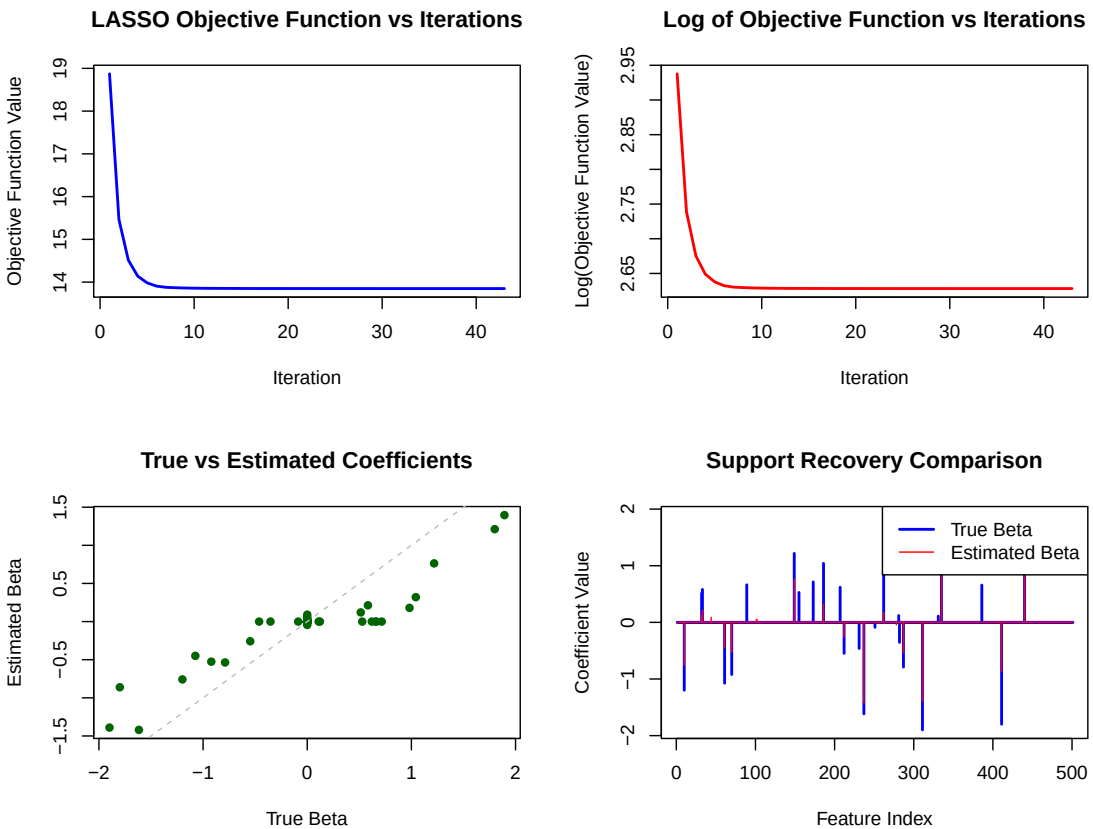


Figure 8: TP=10, FP=0, TN=475, FN=15

The Proximal Operator for the ℓ_2 norm

For $g(x) = \lambda \|x\|_2$

$$\text{prox}_{t,\lambda\|\cdot\|_2}(z) = \begin{cases} \left(1 - \frac{t\lambda}{\|z\|_2}\right) z & \text{if } \|z\|_2 > t\lambda \\ 0 & \text{if } \|z\|_2 \leq t\lambda \end{cases}$$

Key Properties:

- Group shrinkage: Entire vector z is either shrunk proportionally or set to zero
- Unlike ℓ_1 proximal operator, which operates element-wise
- Preserves the direction of z when non-zero
- Similar to "vector soft-thresholding"

Group lasso Regression: Problem Formulation

Group Lasso Objective Function:

$$\min_{\beta} \underbrace{\frac{1}{2n}\|y - X\beta\|_2^2}_{\text{loss function}} + \underbrace{\lambda \sum_{g=1}^{G} \|\beta_g\|_2}_{\text{group penalty}}$$

where:

- $\beta = (\beta_1, \beta_2, \ldots, \beta_G)$ with $\beta_g \in \mathbb{R}^{p_g}$ for $g = 1, \ldots, G$
- $X = [X_1, X_2, \ldots, X_G]$ with $X_g \in \mathbb{R}^{n \times p_g}$
- $\sum_{g=1}^{G} p_g = p$ (total number of predictors)

Key Idea:

- Variables are partitioned into predefined groups (due to prior knowledge)
- Either all variables in a group are selected or none are
- Enforces structured sparsity based on domain knowledge

Proximal Gradient for Group Lasso

Proximal Gradient Update:

- GD update:

$$\tilde{\beta}_k = \beta_k - \eta_k \nabla f(\beta_k),$$

  where $\nabla f(\beta_k) = -\frac{1}{n}\left[X^\top (y - X\beta_k)\right]$

- Group-wise Proximal Operator:

$$\beta_{k+1} = \text{prox}_{\eta_k, \lambda \sum_{g=1}^{G} \|\cdot\|_2}\left(\tilde{\beta}_{g,k}\right)$$

for each group $g = 1, 2, \cdots, G$, where $\tilde{\beta}_{g,k} \in \mathbb{R}^{p_g}$ is the subvector of $\tilde{\beta}_k$ corresponding to group g

Nesterov Momentum Proximal Gradient Descent

Key idea: Add Nesterov momentum term to the standard proximal gradient algorithm

- Algorithm:

$$\begin{aligned} y_k &= x_k + \xi_k(x_k - x_{k-1}) \\ \tilde{y}_k &= y_k - \eta_k \nabla f(y_k) \\ x_{k+1} &= \text{prox}_{\eta_k, g}(\tilde{y}_k) \end{aligned}$$

- Significantly faster convergence with minimal added complexity

Selection of the Step Size

- Backtracking line search is an efficient method
- For very large dimensionality problems (e.g., regression problems with thousands of predictors) AdaGrad can be also effective

What about Proximal Stochastic Gradient Descent

For regression problems with a very large number of observations and regularization, SGD can be useful

Algorithm:
Simply apply proximal gradient descent to a mini-batch of the data