Ridge Regression

Noted by Md. Inzamamul Lohani

Software Engineering, IIT, DU

November 22, 2026

Contents

1 The problem: why do we need Ridge Regression? 4
1.1 Start from what you already know . . . 4
1.2 The hidden issue: fitting the observed data is not the whole goal . . . 4
1.3 The central question . . . 4
1.4 What Ridge is not . . . 4

2 Multicollinearity: when predictors are highly correlated 5
2.1 What does correlation between predictors mean? . . . 5
2.2 Why does this create a problem for OLS? . . . 5
2.3 The matrix explanation . . . 5
2.4 A simulation of coefficient instability . . . 6

3 Ridge intuition: how does Ridge address the problem? 6
3.1 The idea before the formula . . . 6
3.2 What is shrinkage? . . . 6
3.3 Why can smaller coefficients help? . . . 7
3.4 A key distinction . . . 7

4 Mathematical foundation: the Ridge cost function 7
4.1 Build the objective from the OLS objective . . . 7
4.2 Understand each part . . . 7
4.3 Why is it called an L_2 penalty? . . . 7
4.4 The role of lambda . . . 7
4.5 Why squares? . . . 8

5 Derivation: deriving the Ridge Regression estimator 8
5.1 Step 1: write the objective . . . 8
5.2 Step 2: expand the residual term . . . 8
5.3 Step 3: combine the quadratic terms . . . 8
5.4 Step 4: differentiate . . . 8
5.5 Step 5: set the gradient equal to zero . . . 9
5.6 Step 6: solve for the coefficient vector . . . 9
5.7 Why does adding $\lambda \mathbf{I}_p$ help? . . . 9
5.8 A proof of positive definiteness . . . 9
5.9 Dimensions check . . . 10
5.10 A tiny numerical example . . . 10

6 Lambda: understanding the regularization parameter 10

6.1 Lambda controls the compromise . . . 10
6.2 What happens at different values? . . . 10
6.3 Coefficient paths . . . 11
6.4 Does larger lambda always mean a better model? . . . 11

7 Geometrical intuition: Ridge using geometry 11
7.1 The constrained version . . . 11
7.2 Understand the contour picture . . . 12
7.3 Why does Ridge shrink coefficients? . . . 12
7.4 A note about the intercept . . . 12

8 Bias–variance trade-off: why does Ridge introduce bias? 12
8.1 First, understand bias . . . 12
8.2 Why would we accept bias? . . . 13
8.3 Prediction error decomposition . . . 13
8.4 The key lesson . . . 13

9 Standardization: why must we scale features? 13
9.1 The problem with different units . . . 14
9.2 Standardization formula . . . 14
9.3 Why do we center the response? . . . 14
9.4 A critical rule: avoid data leakage . . . 15

10 Practical implementation: Ridge Regression in Python 15
10.1 Install the libraries . . . 15
10.2 Create an illustrative dataset . . . 15
10.3 Why use a pipeline? . . . 15
10.4 Evaluate predictions . . . 16
10.5 Understand the metrics . . . 16
10.6 Compare Ridge with OLS . . . 16

11 Model evaluation: choosing the best lambda using cross-validation 17
11.1 Why do we need to choose lambda? . . . 17
11.2 K-fold cross-validation . . . 17
11.3 Cross-validation score . . . 17
11.4 Use RidgeCV with a pipeline . . . 17
11.5 Final evaluation on the test set . . . 18

Final review: the complete Ridge story 18

Self-test 19

1 The problem: why do we need Ridge Regression?

1.1 Start from what you already know

In multiple linear regression, we describe the response using p predictors:

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} + \varepsilon_i.$$

The coefficients describe how the predicted response changes when a predictor changes, holding the other predictors fixed.

In matrix notation,

$$\boxed{\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}},$$

where $\mathbf{X}$ contains the observed predictor values, $\boldsymbol{\beta}$ contains the unknown coefficients, and $\boldsymbol{\varepsilon}$ contains the errors.

For the moment, suppose the intercept has been handled separately or the variables have been centered. OLS chooses coefficients by minimizing

$$\boxed{\mathrm{RSS}(\boldsymbol{\beta}) = \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2.}$$

The familiar OLS estimator, when $\mathbf{X}'\mathbf{X}$ is invertible, is

$$\boxed{\widehat{\boldsymbol{\beta}}_{\mathrm{OLS}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}.}$$

1.2 The hidden issue: fitting the observed data is not the whole goal

Imagine two models. Model A fits the training observations extremely closely, but its coefficients change dramatically when a few observations change. Model B has slightly more training error, but its coefficients and predictions are much more stable.

Which model would you prefer for predicting new observations? Often, Model B.

OLS focuses on minimizing training RSS. It does not explicitly discourage large or unstable coefficient estimates. When predictors contain overlapping information, this can become a serious problem.

1.3 The central question

Ridge Regression asks:

> Can we keep the model reasonably close to the observed data while discouraging unnecessarily large coefficient values?

This is the central idea of **regularization**. Ridge is a regularized linear regression method. It adds a penalty to the OLS objective.

1.4 What Ridge is not

Ridge does not change the basic idea of a linear prediction. It does not automatically discover a nonlinear relationship. It changes how the coefficients are estimated by adding a preference for smaller coefficient magnitudes.

Remark 1.1. A larger coefficient is not automatically bad. Ridge is useful when the stability gained by shrinking coefficients is worth the small amount of additional bias that shrinkage may introduce.

2 Multicollinearity: when predictors are highly correlated

2.1 What does correlation between predictors mean?

Suppose a dataset contains two predictors, x_1 and x_2. If knowing x_1 already tells us much of the value of x_2, the predictors are strongly correlated. They carry similar information.

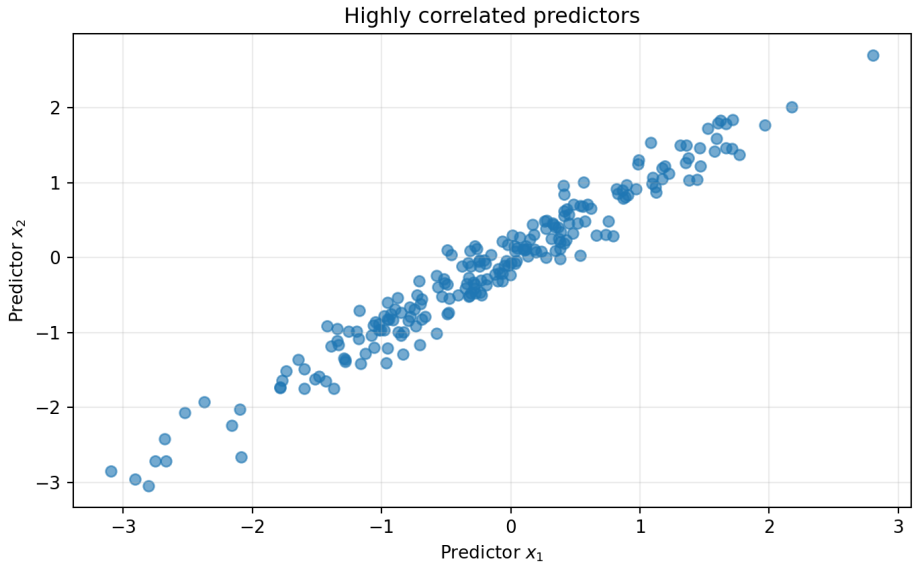

Figure 1: Illustration of two strongly correlated predictors. The points lie close to a straight-line pattern, indicating that the predictors carry overlapping information.

2.2 Why does this create a problem for OLS?

Consider the model

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \varepsilon.$$

If x_1 and x_2 move together, it can be difficult to determine which predictor deserves credit for a change in y.

For example, suppose study hours and time spent using a learning app are strongly related. If both are included, the data may not contain enough independent variation to estimate their separate effects precisely.

The model may still predict reasonably well, but the individual coefficients can become unstable. A small change in the sample may produce a large change in $\widehat{\beta}_1$ and $\widehat{\beta}_2$.

2.3 The matrix explanation

The OLS estimator contains $(\mathbf{X}'\mathbf{X})^{-1}$. The matrix $\mathbf{X}'\mathbf{X}$ summarizes relationships among the columns of $\mathbf{X}$. Strong dependence among columns can make this matrix close to singular.

For a symmetric matrix, eigenvalues describe the amount of information in different directions. If an eigenvalue is very small, inversion divides by a very small number. Small disturbances can then be magnified.

$$\mathbf{X}'\mathbf{X} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}', \qquad \mathbf{\Lambda} = \operatorname{diag}(d_1, \ldots, d_p).$$

The inverse contains $1/d_j$. If d_j is tiny, the corresponding direction can be highly sensitive.

2.4 A simulation of coefficient instability

The following illustration repeatedly samples similar data and estimates the same two coefficients. OLS estimates spread widely because the predictors are highly correlated. Ridge estimates are more concentrated. The exact numerical pattern depends on the simulated data and penalty.

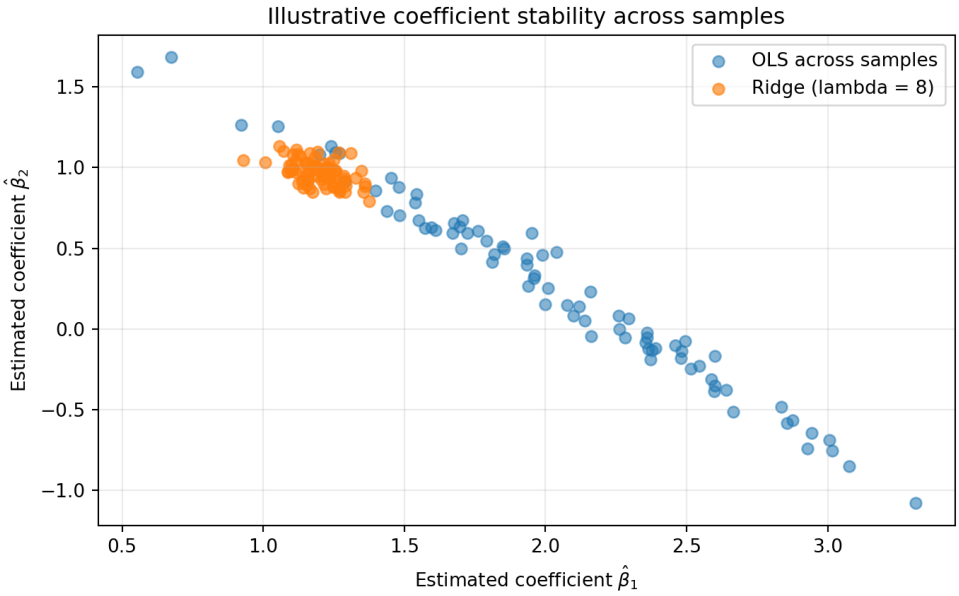

Figure 2: Illustrative coefficient estimates across repeated samples. Ridge shrinkage can reduce the variation of estimates when predictors are highly correlated.

Remark 2.1. Multicollinearity is not the only reason to use Ridge. Ridge can also help when the number of predictors is large relative to the number of observations, or when prediction stability is important.

3 Ridge intuition: how does Ridge address the problem?

3.1 The idea before the formula

OLS says: choose coefficients that make residuals as small as possible.

Ridge says: choose coefficients that make residuals small, but also prefer coefficients that are not unnecessarily large.

There are therefore two competing goals:

1. **Good fit:** predictions should remain close to observed responses.
2. **Controlled coefficients:** the coefficient vector should not become excessively large.

3.2 What is shrinkage?

Shrinkage means pulling coefficient estimates toward zero. Ridge usually does not force coefficients to become exactly zero. Instead, it reduces their magnitudes.

For example, an OLS estimate of 4.0 might become a Ridge estimate of 3.2. An OLS estimate of -2.0 might become -1.5. These numbers are only illustrative; actual shrinkage depends on the data and λ.

3.3 Why can smaller coefficients help?

When predictors are correlated, the model may use large positive and negative coefficients to compensate for one another. Such a solution can fit the training data but react strongly to small data changes.

Ridge discourages these large coefficient combinations. The result is often a more stable model.

3.4 A key distinction

Ridge does not say "make every coefficient as close to zero as possible" without qualification. If that were the only goal, the all-zero vector would win. Ridge balances coefficient size against fit. The penalty is only one part of the objective.

4 Mathematical foundation: the Ridge cost function

4.1 Build the objective from the OLS objective

The OLS objective is

$$\mathrm{RSS}(\boldsymbol{\beta}) = \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 .$$

Ridge adds a penalty on squared coefficient magnitudes:

$$\boxed{J(\boldsymbol{\beta}) = \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_2^2 .}$$

Because

$$\|\boldsymbol{\beta}\|_2^2 = \boldsymbol{\beta}'\boldsymbol{\beta} = \beta_1^2 + \beta_2^2 + \cdots + \beta_p^2,$$

we can also write

$$\boxed{J(\boldsymbol{\beta}) = \mathrm{RSS}(\boldsymbol{\beta}) + \lambda \sum_{j=1}^{p} \beta_j^2.}$$

4.2 Understand each part

- $\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2$ is the residual sum of squares. It measures the mismatch between observed and fitted responses.
- $\sum_j \beta_j^2$ measures the squared size of the coefficient vector.
- λ controls how strongly coefficient size is penalized.

4.3 Why is it called an L_2 penalty?

The Euclidean length of a vector is

$$\|\boldsymbol{\beta}\|_2 = \sqrt{\beta_1^2 + \cdots + \beta_p^2}.$$

Ridge penalizes its square, $\|\boldsymbol{\beta}\|_2^2$. This is called an L_2 regularization penalty.

4.4 The role of lambda

The parameter λ is nonnegative:

$$\lambda \geq 0.$$

When $\lambda = 0$, the penalty disappears and the objective is OLS. When λ is positive, larger coefficient vectors receive a larger penalty. As λ increases, the optimization gives more importance to keeping coefficients small.

Penalty	Meaning
$\lambda = 0$	OLS objective
Small $\lambda > 0$	Mild shrinkage
Moderate λ	Stronger control of coefficient size
Very large λ	Penalized coefficients approach zero

4.5 Why squares?

Squaring makes the penalty smooth and differentiable. Large coefficients are penalized more heavily than small ones. It also leads to a convex quadratic optimization problem with a convenient closed-form solution.

5 Derivation: deriving the Ridge Regression estimator

This chapter derives the estimator carefully. The purpose is to understand where every term comes from.

5.1 Step 1: write the objective

Assume the intercept is not included in the penalized vector for now:

$$J(\boldsymbol{\beta}) = (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})'(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) + \lambda\boldsymbol{\beta}'\boldsymbol{\beta}.$$

5.2 Step 2: expand the residual term

Use the identity $(A - B)'(A - B) = A'A - A'B - B'A + B'B$:

$$J(\boldsymbol{\beta}) = \mathbf{y}'\mathbf{y} - \mathbf{y}'\mathbf{X}\boldsymbol{\beta} - \boldsymbol{\beta}'\mathbf{X}'\mathbf{y} + \boldsymbol{\beta}'\mathbf{X}'\mathbf{X}\boldsymbol{\beta} + \lambda\boldsymbol{\beta}'\boldsymbol{\beta}.$$

The two middle expressions are scalars and are transposes of one another. A scalar equals its transpose, so

$$\mathbf{y}'\mathbf{X}\boldsymbol{\beta} = \boldsymbol{\beta}'\mathbf{X}'\mathbf{y}.$$

Therefore,

$$J(\boldsymbol{\beta}) = \mathbf{y}'\mathbf{y} - 2\boldsymbol{\beta}'\mathbf{X}'\mathbf{y} + \boldsymbol{\beta}'\mathbf{X}'\mathbf{X}\boldsymbol{\beta} + \lambda\boldsymbol{\beta}'\boldsymbol{\beta}.$$

5.3 Step 3: combine the quadratic terms

Since

$$\boldsymbol{\beta}'\mathbf{X}'\mathbf{X}\boldsymbol{\beta} + \lambda\boldsymbol{\beta}'\boldsymbol{\beta} = \boldsymbol{\beta}'(\mathbf{X}'\mathbf{X} + \lambda\mathbf{I}_p)\boldsymbol{\beta},$$

we obtain

$$J(\boldsymbol{\beta}) = \mathbf{y}'\mathbf{y} - 2\boldsymbol{\beta}'\mathbf{X}'\mathbf{y} + \boldsymbol{\beta}'(\mathbf{X}'\mathbf{X} + \lambda\mathbf{I}_p)\boldsymbol{\beta}.$$

5.4 Step 4: differentiate

We use two matrix-calculus identities:

$$\frac{\partial(\mathbf{c}'\boldsymbol{\beta})}{\partial\boldsymbol{\beta}} = \mathbf{c}, \qquad \frac{\partial(\boldsymbol{\beta}'\mathbf{A}\boldsymbol{\beta})}{\partial\boldsymbol{\beta}} = (\mathbf{A} + \mathbf{A}')\boldsymbol{\beta}.$$

The matrix $\mathbf{X}'\mathbf{X} + \lambda\mathbf{I}_p$ is symmetric. Hence

$$\frac{\partial J}{\partial \boldsymbol{\beta}} = -2\mathbf{X}'\mathbf{y} + 2(\mathbf{X}'\mathbf{X} + \lambda\mathbf{I}_p)\boldsymbol{\beta}.$$

5.5 Step 5: set the gradient equal to zero

At the minimum, the gradient vanishes:

$$-2\mathbf{X}'\mathbf{y} + 2(\mathbf{X}'\mathbf{X} + \lambda\mathbf{I}_p)\boldsymbol{\beta} = \mathbf{0}.$$

Divide by two:

$$-\mathbf{X}'\mathbf{y} + (\mathbf{X}'\mathbf{X} + \lambda\mathbf{I}_p)\boldsymbol{\beta} = \mathbf{0}.$$

Rearrange:

$$(\mathbf{X}'\mathbf{X} + \lambda\mathbf{I}_p)\boldsymbol{\beta} = \mathbf{X}'\mathbf{y}.$$

5.6 Step 6: solve for the coefficient vector

For $\lambda > 0$, the matrix $\mathbf{X}'\mathbf{X} + \lambda\mathbf{I}_p$ is invertible. Therefore,

$$\boxed{\widehat{\boldsymbol{\beta}}_{\text{Ridge}} = (\mathbf{X}'\mathbf{X} + \lambda\mathbf{I}_p)^{-1}\mathbf{X}'\mathbf{y}.}$$

This is the Ridge Regression estimator.

5.7 Why does adding $\lambda\mathbf{I}_p$ help?

The identity matrix is

$$\mathbf{I}_p = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}.$$

Thus, $\lambda\mathbf{I}_p$ adds λ to every diagonal entry of $\mathbf{X}'\mathbf{X}$.

More importantly, if the eigenvalues of $\mathbf{X}'\mathbf{X}$ are $d_1, \ldots, d_p$, the eigenvalues of $\mathbf{X}'\mathbf{X} + \lambda\mathbf{I}_p$ are

$$d_1 + \lambda, \ldots, d_p + \lambda.$$

Since $d_j \geq 0$ and $\lambda > 0$, every eigenvalue is positive. The matrix is positive definite and therefore invertible.

5.8 A proof of positive definiteness

For any nonzero vector $\mathbf{v}$,

$$\begin{aligned} \mathbf{v}'(\mathbf{X}'\mathbf{X} + \lambda\mathbf{I}_p)\mathbf{v} &= \mathbf{v}'\mathbf{X}'\mathbf{X}\mathbf{v} + \lambda\mathbf{v}'\mathbf{v} \\ &= (\mathbf{X}\mathbf{v})'(\mathbf{X}\mathbf{v}) + \lambda \|\mathbf{v}\|_2^2 \\ &= \|\mathbf{X}\mathbf{v}\|_2^2 + \lambda \|\mathbf{v}\|_2^2 > 0. \end{aligned}$$

This proves positive definiteness.

5.9 Dimensions check

If $\mathbf{X}$ has dimension $n \times p$, then

$$\mathbf{X}'\mathbf{X} : (p \times n)(n \times p) = p \times p,$$

and

$$\mathbf{X}'\mathbf{y} : (p \times n)(n \times 1) = p \times 1.$$

The inverse matrix is $p \times p$, so the final result is $p \times 1$, which is exactly the dimension of $\boldsymbol{\beta}$.

5.10 A tiny numerical example

Suppose

$$\mathbf{X} = \begin{bmatrix} 1 & 1 \\ 2 & 2 \\ 3 & 3 \end{bmatrix}, \qquad \mathbf{y} = \begin{bmatrix} 2 \\ 4 \\ 6 \end{bmatrix}.$$

Then

$$\mathbf{X}'\mathbf{X} = \begin{bmatrix} 14 & 14 \\ 14 & 14 \end{bmatrix}, \qquad \mathbf{X}'\mathbf{y} = \begin{bmatrix} 28 \\ 28 \end{bmatrix}.$$

The columns of $\mathbf{X}$ are identical, so $\mathbf{X}'\mathbf{X}$ is singular. OLS cannot uniquely decide how to split the total slope between the two identical predictors.

For $\lambda > 0$,

$$\mathbf{X}'\mathbf{X} + \lambda \mathbf{I}_2 = \begin{bmatrix} 14 + \lambda & 14 \\ 14 & 14 + \lambda \end{bmatrix}.$$

Its determinant is

$$(14 + \lambda)^2 - 14^2 = \lambda(28 + \lambda) > 0.$$

So Ridge can solve the system. By symmetry, the two coefficients are equal:

$$\widehat{\beta}_1 = \widehat{\beta}_2 = \frac{28}{28 + 2\lambda}.$$

At $\lambda = 1$, each coefficient is $28/30 \approx 0.933$. Their sum is approximately 1.867, close to the total slope of 2 that reproduces the response.

Remark 5.1. This example is intentionally extreme: the predictors are exactly identical. It makes the matrix-stabilizing role of Ridge especially clear.

6 Lambda: understanding the regularization parameter

6.1 Lambda controls the compromise

The Ridge objective is

$$J(\boldsymbol{\beta}) = \mathrm{RSS}(\boldsymbol{\beta}) + \lambda \|\boldsymbol{\beta}\|_2^2.$$

Think of λ as a volume control for the penalty term. A larger value makes large coefficients more expensive.

6.2 What happens at different values?

- At $\lambda = 0$, Ridge is OLS, assuming a unique OLS solution.
- At a small positive value, coefficients are usually close to OLS.
- At a larger value, coefficient magnitudes are reduced more strongly.
- As $\lambda \to \infty$, penalized slope coefficients approach zero.

6.3 Coefficient paths

The following graph shows how estimated coefficients can change as lambda moves across a range. It is generated from an illustrative dataset.

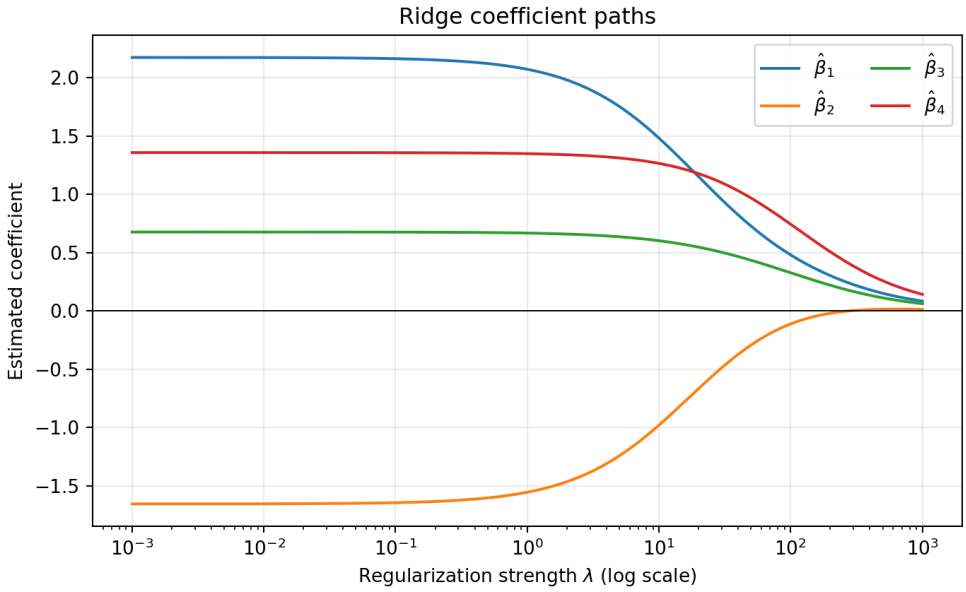

Figure 3: Ridge coefficient paths. Coefficients generally move smoothly toward zero as the penalty becomes stronger.

6.4 Does larger lambda always mean a better model?

No. A very large penalty can shrink useful coefficients too much. The model may underfit. A very small penalty may fail to control instability. The goal is to find a value that performs well on data not used to fit the model.

Remark 6.1. Do not choose lambda by minimizing training RSS. Training RSS generally favors no penalty. Use validation data or cross-validation to estimate performance on unseen observations.

7 Geometrical intuition: Ridge using geometry

7.1 The constrained version

Ridge can be written in two equivalent ways for suitable choices of the penalty parameter and constraint size.

Penalized form:

$$\min_{\boldsymbol{\beta}} \mathrm{RSS}(\boldsymbol{\beta}) + \lambda \|\boldsymbol{\beta}\|_2^2 .$$

Constrained form:

$$\boxed{\min_{\boldsymbol{\beta}} \mathrm{RSS}(\boldsymbol{\beta}) \quad \text{subject to} \quad \|\boldsymbol{\beta}\|_2^2 \leq t.}$$

The constrained form says: find the best fit, but keep the coefficient vector inside a circle (in two dimensions) or a sphere (in higher dimensions).

7.2 Understand the contour picture

An RSS contour connects coefficient combinations with the same RSS. For a quadratic least-squares objective, these contours are ellipses. The OLS estimate is at the center of the lowest reachable contour.

The Ridge constraint is an L_2 circle centered at the origin. Ridge chooses the point where the smallest possible RSS contour touches the allowed region.

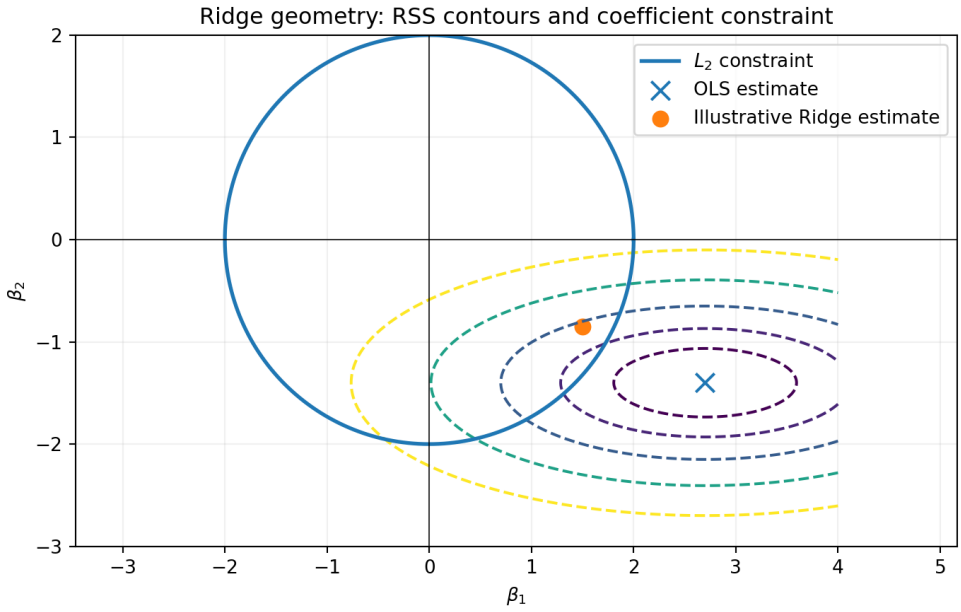

Figure 4: Schematic geometric interpretation of Ridge. Dashed curves are RSS contours; the circle is an L_2 constraint. The points are illustrative.

7.3 Why does Ridge shrink coefficients?

The constraint limits the total squared coefficient size:

$$\beta_1^2 + \beta_2^2 \leq t.$$

If the OLS point lies outside the allowed circle, Ridge cannot choose it. It must select a point closer to the origin. This is the geometric meaning of shrinkage.

7.4 A note about the intercept

The usual intercept is not penalized. The circle applies to the slope coefficients, not to the intercept. This is one reason centering predictors and the response is convenient.

8 Bias–variance trade-off: why does Ridge introduce bias?

8.1 First, understand bias

An estimator is unbiased if its expected value equals the true parameter:

$$\mathrm{E}[\widehat{\beta}_j] = \beta_j.$$

Ridge deliberately shrinks coefficients. Therefore, in general,

$$\mathrm{E}[\widehat{\boldsymbol{\beta}}_{\mathrm{Ridge}}] \neq \boldsymbol{\beta}.$$

This is why Ridge introduces bias.

8.2 Why would we accept bias?

Because the OLS estimator can have high variance. Variance measures how much an estimate would change across repeated samples from the same process.

When predictors are highly correlated, OLS estimates can vary considerably from one sample to another. Ridge often reduces this variation by shrinking coefficients.

8.3 Prediction error decomposition

At a fixed input value, a standard squared prediction error decomposition is

$$\begin{aligned}\mathrm{E}[(Y_0 - \widehat{f}(\mathbf{x}_0))^2] = \; & \mathrm{Var}(\widehat{f}(\mathbf{x}_0)) \\ & + \mathrm{Bias}(\widehat{f}(\mathbf{x}_0))^2 + \sigma^2.\end{aligned}$$

Here σ^2 is irreducible noise: variation in the response that the predictors and model cannot explain.

Ridge may increase squared bias but reduce variance. If the variance reduction is larger than the bias increase, total prediction error decreases.

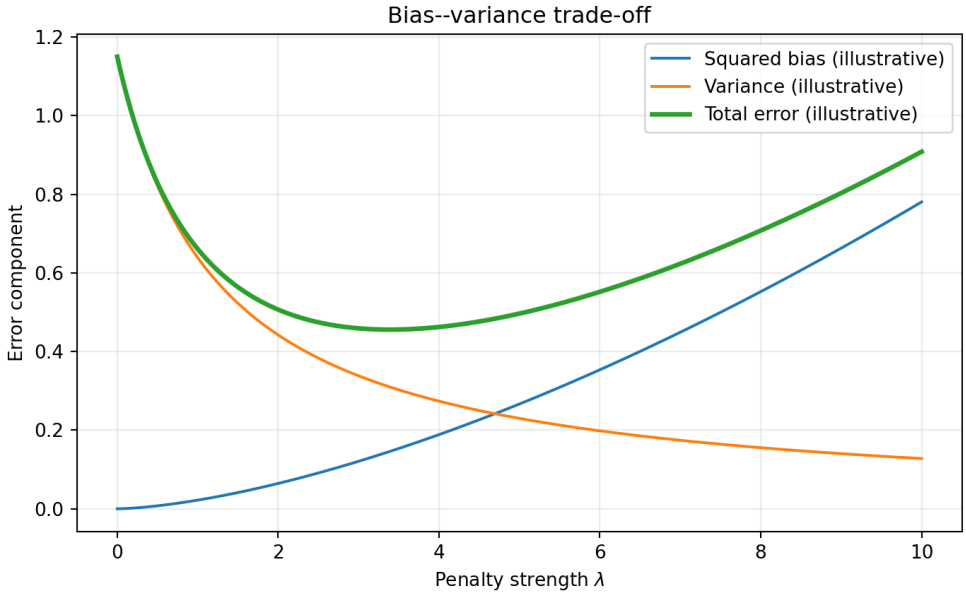

Figure 5: Conceptual bias–variance curves. These curves are illustrative; the exact shapes depend on the dataset and model.

8.4 The key lesson

The goal is not to make bias zero at all costs. The goal is good prediction on new data. A slightly biased but stable estimator can outperform an unbiased estimator with high variance.

9 Standardization: why must we scale features?

9.1 The problem with different units

Suppose one predictor is measured in years, with values around 20, while another is measured in annual income, with values around 50,000. The numerical sizes of their coefficients can be very different simply because the units are different.

Ridge penalizes coefficient magnitudes:

$$\lambda(\beta_1^2 + \cdots + \beta_p^2).$$

If the predictor units change, the coefficients change too. Without scaling, the penalty can treat predictors unfairly.

9.2 Standardization formula

For predictor j, calculate its training-set mean and standard deviation:

$$\bar{x}_j = \frac{1}{n}\sum_{i=1}^{n} x_{ij}, \qquad s_j = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_{ij} - \bar{x}_j)^2}.$$

Then transform each observation:

$$\boxed{z_{ij} = \frac{x_{ij} - \bar{x}_j}{s_j}.}$$

After standardization, each feature has mean approximately zero and sample standard deviation one.

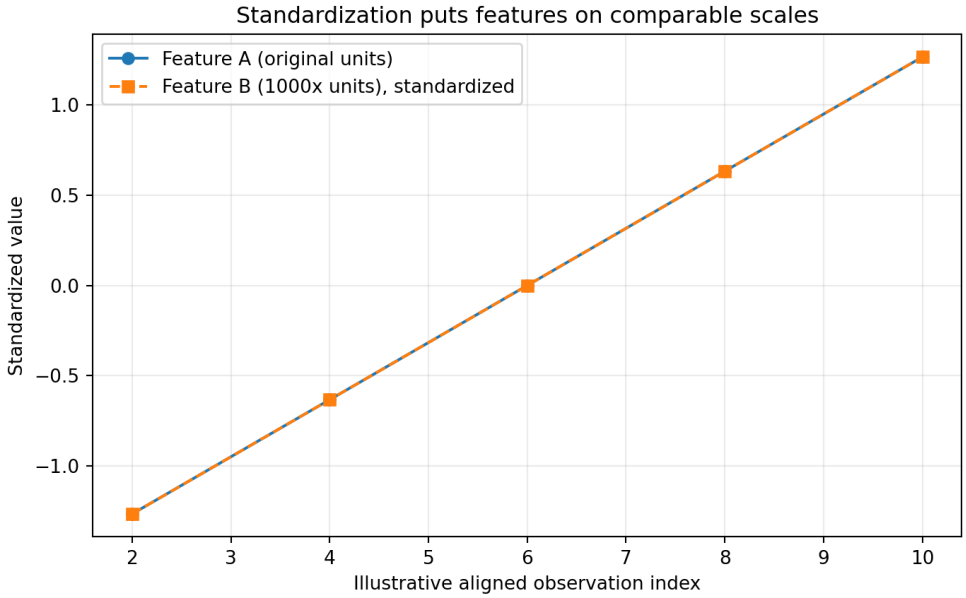

Figure 6: Standardization makes feature scales comparable. Ridge is sensitive to coefficient scale, so this preprocessing is usually important.

9.3 Why do we center the response?

Centering the response makes the intercept easier to handle. In the standard Ridge setup, the intercept is not penalized. After fitting centered data, recover it using

$$\boxed{\hat{\beta}_0 = \bar{y} - \sum_{j=1}^{p} \hat{\beta}_j \bar{x}_j.}$$

9.4 A critical rule: avoid data leakage

Calculate means and standard deviations using only the training portion of the data. Apply those same values to validation and test data. If you use the full dataset to calculate scaling statistics before splitting, information from the held-out data leaks into the training process.

A machine-learning pipeline handles this correctly when used inside cross-validation.

10 Practical implementation: Ridge Regression in Python

10.1 Install the libraries

The example uses NumPy, pandas, Matplotlib, and scikit-learn:

```
python -m pip install numpy pandas matplotlib scikit-learn
```

10.2 Create an illustrative dataset

We use a synthetic regression dataset so the example is reproducible and does not depend on downloading a file.

```
import numpy as np
from sklearn.datasets import make_regression
from sklearn.model_selection import train_test_split

X, y = make_regression(
    n_samples=500,
    n_features=8,
    n_informative=6,
    noise=18,
    random_state=42
)

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)
```

10.3 Why use a pipeline?

A pipeline ensures that standardization is learned from the training data inside each fold, then the Ridge model is fitted to the transformed data. This helps prevent data leakage.

```
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import Ridge

model = make_pipeline(
    StandardScaler(),
    Ridge(alpha=1.0)
)

model.fit(X_train, y_train)
predictions = model.predict(X_test)
```

In scikit-learn, the parameter is named `alpha`; it plays the role of the Ridge penalty strength λ in our mathematical notation.

10.4 Evaluate predictions

```
from sklearn.metrics import (
    mean_squared_error,
    mean_absolute_error,
    r2_score
)

mse = mean_squared_error(y_test, predictions)
rmse = np.sqrt(mse)
mae = mean_absolute_error(y_test, predictions)
r2 = r2_score(y_test, predictions)

print("MSE:", mse)
print("RMSE:", rmse)
print("MAE:", mae)
print("R-squared:", r2)
```

10.5 Understand the metrics

- **MSE** averages squared prediction errors. Large errors receive a particularly strong penalty.
- **RMSE** is the square root of MSE and is expressed in the same units as the response.
- **MAE** averages absolute prediction errors and is less sensitive to large individual errors than MSE.
- R^2 compares the model with a baseline that predicts the training response mean. It can be negative on test data.

10.6 Compare Ridge with OLS

```
from sklearn.linear_model import LinearRegression
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler

ols = make_pipeline(
    StandardScaler(),
    LinearRegression()
)

ridge = make_pipeline(
    StandardScaler(),
    Ridge(alpha=1.0)
)

ols.fit(X_train, y_train)
ridge.fit(X_train, y_train)

ols_pred = ols.predict(X_test)
ridge_pred = ridge.predict(X_test)

print("OLS RMSE:",
      mean_squared_error(y_test, ols_pred) ** 0.5)

print("Ridge RMSE:",
      mean_squared_error(y_test, ridge_pred) ** 0.5)
```

Do not assume Ridge must win. The test result depends on the data-generating process and the penalty value.

11 Model evaluation: choosing the best lambda using cross-validation

11.1 Why do we need to choose lambda?

Lambda controls the balance between fitting the data and shrinking coefficients. A value that is too small may leave instability; a value that is too large may underfit.

We need a principled way to compare candidate values. Cross-validation estimates how well each candidate generalizes to data not used in its fit.

11.2 K-fold cross-validation

The basic procedure is:

1. Divide the training data into K approximately equal folds.
2. Choose one fold as validation data and fit on the other $K - 1$ folds.
3. Evaluate predictions on the held-out fold.
4. Repeat until each fold has been used as validation data.
5. Average the validation errors for each candidate lambda.
6. Select the candidate with the best average validation score.

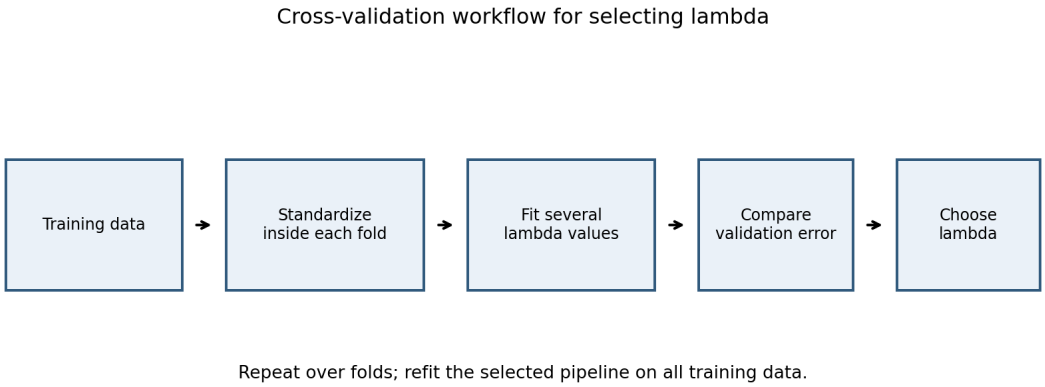

Figure 7: A simplified workflow for selecting Ridge's penalty using cross-validation.

11.3 Cross-validation score

For squared-error loss, a conceptual K-fold score is

$$\mathrm{CV}(\lambda) = \frac{1}{K} \sum_{k=1}^{K} \left[\frac{1}{n_k} \sum_{i \in \mathcal{V}_k} (y_i - \widehat{y}_i^{(-k)}(\lambda))^2 \right].$$

Here $\mathcal{V}_k$ is the validation fold and $\widehat{y}_i^{(-k)}(\lambda)$ is the prediction from a model trained without that fold.

11.4 Use RidgeCV with a pipeline

```python
import numpy as np
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import RidgeCV

alphas = np.logspace(-4, 4, 100)

ridge_cv = make_pipeline(
    StandardScaler(),
    RidgeCV(
        alphas=alphas,
        cv=5,
        scoring="neg_mean_squared_error"
    )
)

ridge_cv.fit(X_train, y_train)

ridge_step = ridge_cv.named_steps["ridgecv"]
print("Selected alpha:", ridge_step.alpha_)
```

The pipeline places the scaler inside the cross-validation process, so each fold learns its own scaling parameters from its training portion. The test set remains untouched during selection.

11.5 Final evaluation on the test set

After selecting lambda using training data and cross-validation, evaluate the final fitted pipeline on the test set:

```python
from sklearn.metrics import mean_squared_error, r2_score

final_predictions = ridge_cv.predict(X_test)

test_rmse = np.sqrt(
    mean_squared_error(y_test, final_predictions)
)
test_r2 = r2_score(y_test, final_predictions)

print("Test RMSE:", test_rmse)
print("Test R-squared:", test_r2)
```

Remark 11.1. The test set should be used for final evaluation, not repeated tuning. If you repeatedly inspect test performance and adjust the model, the test set gradually becomes part of the model-selection process.

Final review: the complete Ridge story

The learning journey can be summarized in eleven connected steps:

1. OLS minimizes residual sum of squares.
2. Correlated predictors can make OLS coefficients unstable.
3. Ridge adds a preference for smaller coefficient magnitudes.
4. Its cost function combines RSS and an L_2 penalty.
5. Differentiation gives the modified normal equations.
6. Lambda controls the strength of shrinkage.

7. Geometry interprets Ridge as an RSS minimization under a coefficient constraint.
8. Ridge trades some bias for a possible reduction in variance.
9. Standardization makes the penalty fair across predictor scales.
10. Python pipelines implement scaling and Ridge together.
11. Cross-validation selects lambda; a separate test set evaluates final performance.

The central estimator is

$$\boxed{\widehat{\beta}_{\text{Ridge}} = (\mathbf{X}'\mathbf{X} + \lambda \mathbf{I}_p)^{-1}\mathbf{X}'\mathbf{y}}$$

when the intercept has been handled separately and all entries of the coefficient vector are penalized.

For a standard model with an unpenalized intercept, center the predictors and response or use a penalty matrix with zero in the intercept position.

Self-test

Try these questions without looking at the earlier chapters.

1. What does OLS minimize?
2. What is multicollinearity?
3. Why can multicollinearity make coefficient estimates unstable?
4. What does the Ridge penalty measure?
5. Why is $\lambda \mathbf{I}_p$ added to $\mathbf{X}'\mathbf{X}$?
6. What happens when $\lambda = 0$?
7. What happens to penalized coefficients as λ becomes very large?
8. Why does Ridge introduce bias?
9. How can reducing variance improve prediction?
10. Why should features be standardized?
11. Why must scaling be learned inside each cross-validation training fold?
12. Why should the test set be kept separate until final evaluation?

Short answer guide

1. Residual sum of squares.
2. Strong linear dependence among predictors.
3. The data struggle to distinguish their separate effects.
4. The squared Euclidean magnitude of the coefficient vector.
5. It shifts eigenvalues upward and makes the matrix positive definite for positive lambda.
6. Ridge reduces to OLS when the OLS solution is uniquely defined.
7. They approach zero.

8. Shrinkage changes the expected coefficient estimates.
9. The reduction in variance can be larger than the increase in squared bias.
10. The penalty otherwise depends on arbitrary measurement units.
11. To prevent information leakage from validation observations.
12. To obtain an honest estimate of generalization performance.

Where to go after Ridge

After mastering these eleven topics, the next natural subjects are:

- the Bayesian interpretation of Ridge Regression;
- singular value decomposition and Ridge shrinkage factors;
- the Ridge hat matrix and effective degrees of freedom;
- Lasso and Elastic Net;
- Principal Component Regression.