Lesson 22: Regularization

Data Science with Python – BSc Course

45 Minutes

Learning Objectives

The Problem: With many predictors, our model can memorize noise instead of learning patterns. How do we build models that generalize to unseen data?

After this lesson, you will be able to:

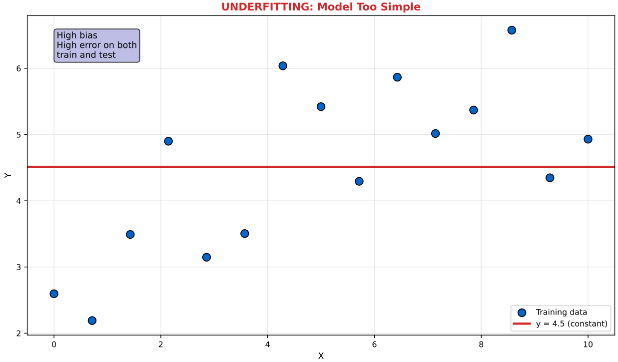
- Recognize overfitting and its causes
- Apply Ridge (L2) regularization to shrink coefficients
- Apply Lasso (L1) for automatic feature selection
- Tune the regularization strength with cross-validation

Finance Application: Building robust factor models with many correlated predictors

Underfitting: Model Too Simple

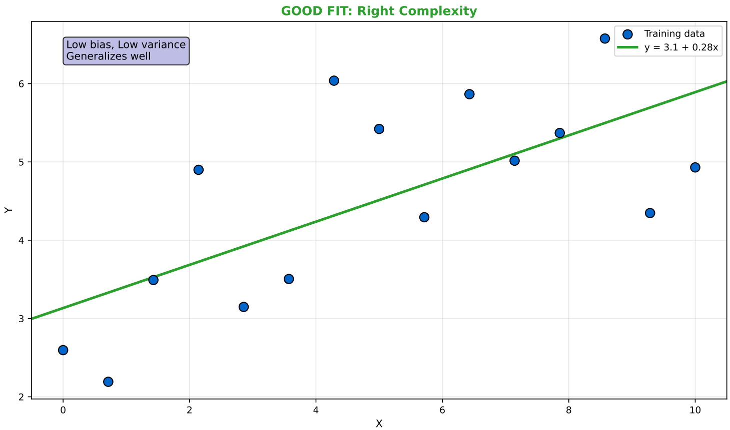
When Models Miss the Pattern

- Constant prediction ignores relationship in data
- High error on both training and test sets



Underfitting = high bias. The model is too simple to capture the true pattern.

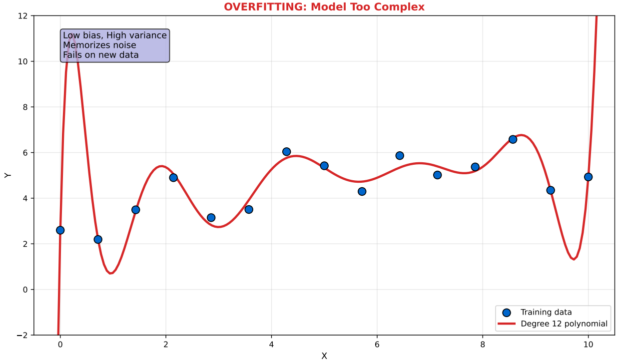
Good Fit: Right Complexity



The goal: A model complex enough to capture patterns, simple enough to generalize

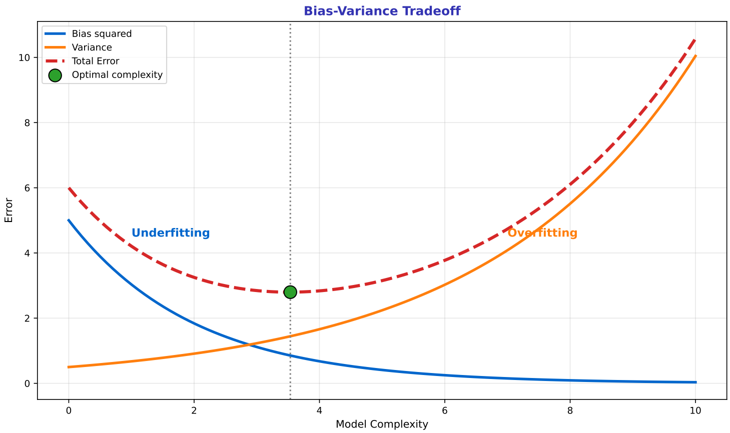
When Models Memorize Noise

- Perfect fit on training data, poor on new data
- High-degree polynomials chase every point



Overfitting = high variance. Model changes drastically with different training samples.

Bias-Variance Tradeoff

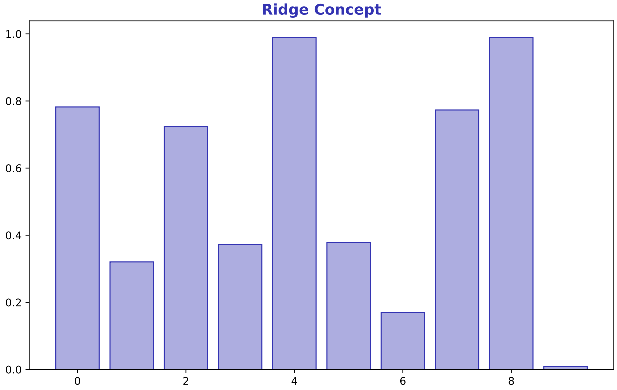


Regularization helps find the optimal complexity – not too simple, not too complex

Ridge Regression (L2)

Shrink All Coefficients Toward Zero

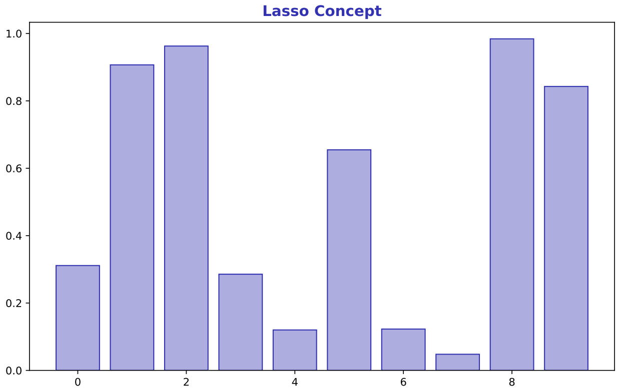
- Add penalty: Loss $= \sum(y - \hat{y})^2 + \lambda \sum \beta_j^2$
- Large λ = stronger shrinkage, simpler model



Ridge keeps all features but reduces their influence – good for multicollinearity

Some Coefficients Go to Exactly Zero

- Add penalty: Loss $= \sum (y - \hat{y})^2 + \lambda \sum |\beta_j|$
- L1 penalty creates sparse solutions (automatic feature selection)

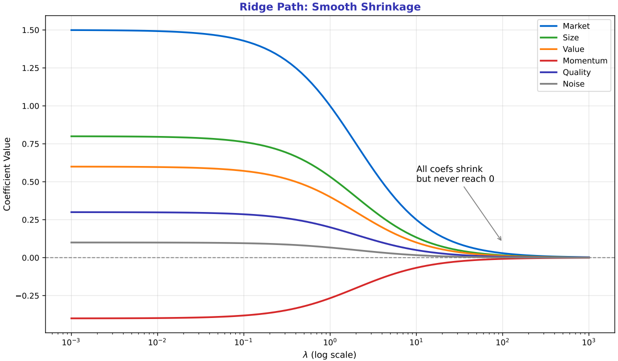


Lasso eliminates irrelevant features – use when you suspect many predictors are useless

Ridge Coefficient Path

Smooth Shrinkage Toward Zero

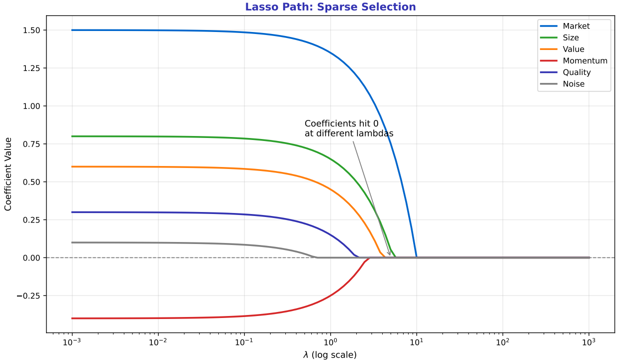
- All coefficients shrink as λ increases
- Coefficients approach but never reach zero



Ridge keeps all features in the model – good when all predictors may be relevant

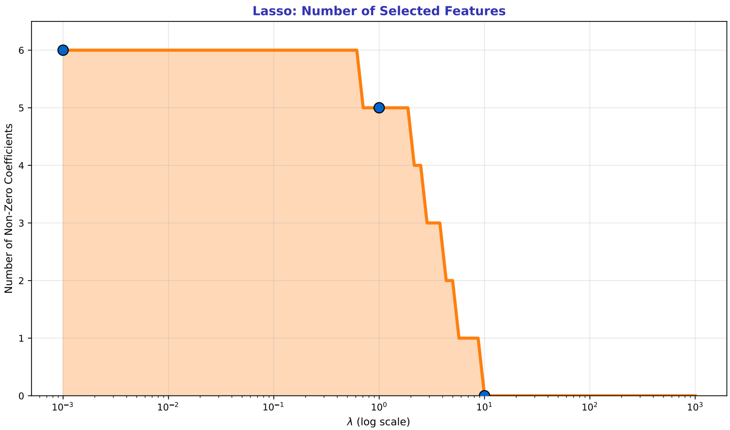
Sparse Feature Selection

- Coefficients hit exactly zero at different λ values
- Weaker predictors are eliminated first



Lasso automatically selects the most important features

Lasso Feature Count

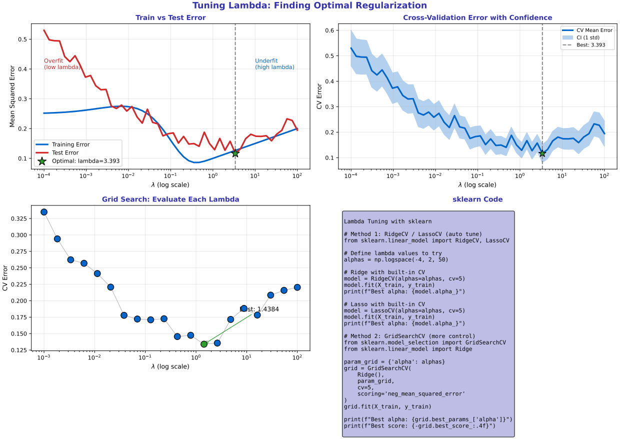


As λ increases, fewer features remain – use CV to find optimal sparsity

Tuning Lambda

Finding the Right Penalty Strength

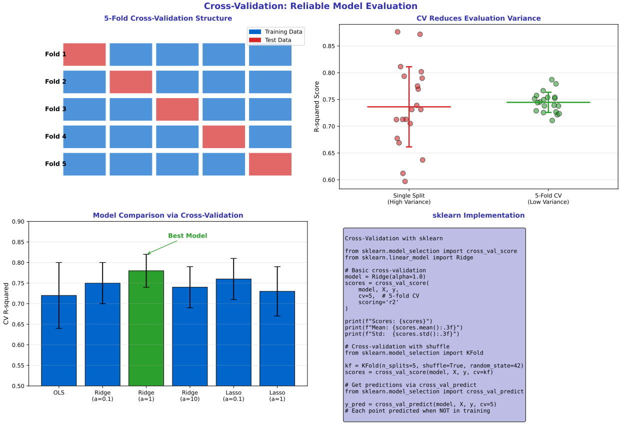
- Too small: overfitting (model too complex)
- Too large: underfitting (model too simple)



Use cross-validation to find the lambda that minimizes test error

sklearn Makes It Easy

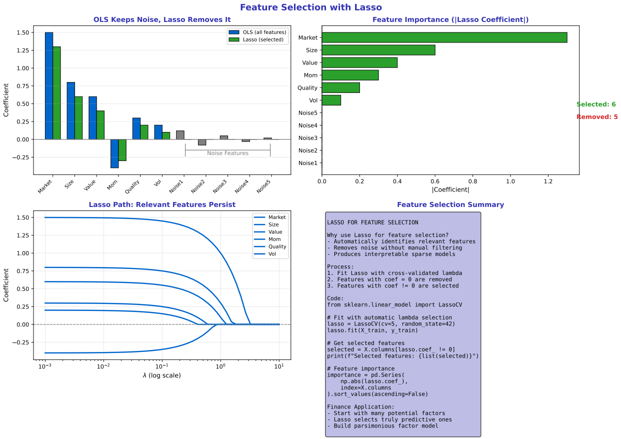
- `RidgeCV` and `LassoCV` automatically search lambda values
- K-fold CV: split data K ways, train on K-1, test on 1, average



Rule: Use `RidgeCV(alphas=[0.1, 1, 10, 100])` to search logarithmically

Which Predictors Actually Matter?

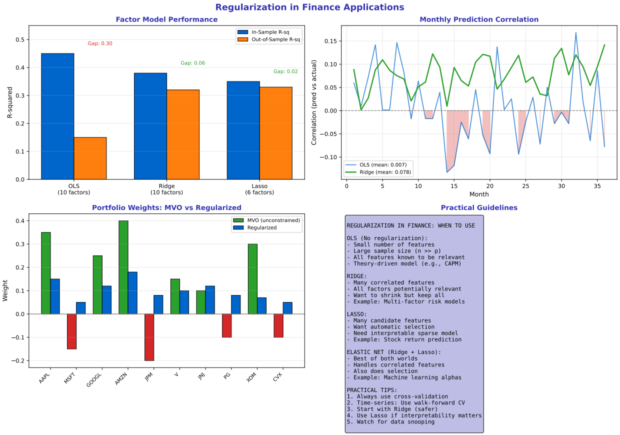
- Non-zero Lasso coefficients = selected features
- Zero coefficients = features eliminated by the model



Finance insight: Lasso often keeps 3-5 factors from a candidate set of 20+

Building Robust Return Predictions

- Many candidate factors are correlated (value, quality, momentum...)
- Regularization prevents unstable, extreme factor weights



Industry practice: Regularized regression for combining alpha signals

Hands-On Exercise (25 min)

Task: Compare Ridge vs Lasso on Multi-Factor Data

1. Create synthetic data with 20 features (only 5 are truly predictive)
2. Fit OLS, Ridge, and Lasso models
3. Compare test set R^2 for each model
4. Plot Lasso coefficients – which features were selected?

Deliverable: Bar chart of coefficients comparing OLS vs Ridge vs Lasso.

Extension: Use LassoCV to find optimal lambda and report selected features

Lesson Summary

Problem Solved: Regularization prevents overfitting when we have many predictors or limited data.

Key Takeaways:

- Ridge (L2) shrinks all coefficients – handles multicollinearity
- Lasso (L1) sets some coefficients to zero – automatic feature selection
- Use cross-validation (RidgeCV, LassoCV) to tune λ

Next Lesson: Regression Metrics (L23) – how do we measure model quality?

Memory: Ridge = Ridge keeps all features. Lasso = Lasso Loses features (L for Lose).