
Controlling Model Complexity for Generalization

David I. Inouye

| Better training fit can mean worse prediction on new data

We have learned how to minimize a training objective.

The ML goal is still performance on new examples.

Regularization limits how freely a model can fit the training data, trading some flexibility for potentially better generalization.

Recall KNN: a tiny neighborhood can follow sample-specific fluctuations.

Larger KNN neighborhoods and coefficient penalties both control flexibility

Mechanism	Weaker restriction	Stronger restriction
KNN neighborhood size k	Small k : follow nearby examples closely	Large k : average over more examples
Coefficient penalty strength λ	Small λ : allow larger coefficients	Large λ : discourage large coefficients

For a linear model, one common choice is **L2 regularization**:

$$J_{\lambda}(\boldsymbol{w}, b) = J_{\text{train}}(\boldsymbol{w}, b) + \frac{\lambda}{2} \|\boldsymbol{w}\|^2.$$

Different mechanisms, same purpose: control sensitivity to the training sample.

Too much restriction can hide useful signal. k and λ have no numerical equivalence.

| Both mechanisms can improve held-out prediction

Ames KNN using living area only — validation RMSE:

$k = 1$	$k = 100$	$k = 1758$ (all training sales)
\$75,300	\$57,700	\$80,200

Ames linear models using 226 encoded features — test RMSE:

No penalty: OLS	L2 penalty: validation-selected ridge
\$26,300	\$25,600

Compare **within each row**: these use different representations and evaluation splits.
The restriction can help; the strongest restriction need not be best.

| Choose the restriction with validation; **assess the choice on test data**

- 1. **Fit:** train each candidate using only training observations.
- 2. **Choose:** compare candidates on validation data.
- 3. **Assess:** evaluate the selected procedure on untouched test data.

Search positive λ values on a log scale: multiply, rather than add.

Search	Example λ values
Broad search: factors of 10	$10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 100$
Refine near a promising $\lambda = 1$	0.1, 0.3, 1, 3, 10

Common mistake: 0, 1, 2, 3, ... misses the entire range $0 < \lambda < 1$.

Include $\lambda = 0$ separately as the **no-penalty baseline**; zero has no logarithm.