

CMPUT 267: Machine Learning I · Fall 2026

Bias, Variance, and Regularization

Xi Ye

2026-10-22

Assignment 5 Is Out

1. **Open:** Use the Assignment 5 student notebook in Google Colab.
2. **Copy:** Select **File** → **Save a copy in Drive** and work in your own copy.
3. **Learn:** If Colab is new to you, review the [Google Colab introduction](#).
4. **Complete:** Run the setup cells, follow the notebook instructions, and fill in only the solution blocks marked `### YOUR CODE HERE ###`. Keep the markers and provided code unchanged.
5. **Submit:** Restart the runtime and run all cells. Download the `.ipynb` file, then upload it to **Assignment 5 on Canvas**.

Recommended completion: October 29, 11:59 p.m. MT



Hard deadline: Nov 12, 11:59 p.m. MT

A complete submission receives full assignment credit. Test results provide diagnostic feedback.

Recall: Sources of Prediction Error

Last lecture, we separated the expected risk of a learned predictor into three terms.

$$\mathbb{E}_D[L(\hat{f}_D)] = \underbrace{\text{EE}}_{\text{Estimation error}} + \underbrace{\text{AE}}_{\text{Approximation error}} + \underbrace{L(f_{\text{Bayes}})}_{\text{Irreducible error}} .$$

- A richer class can reduce approximation error, but may increase estimation error.
- More training data can reduce estimation error; the other two terms stay fixed.



Can we keep a rich class while making the fitted predictor less sensitive to the training sample?

Recall: Estimation Error

Estimation error is the cost of learning from a finite sample instead of knowing the population.

$$\text{EE} = \underbrace{\mathbb{E}_D[L(\hat{f}_D)]}_{\text{Learned predictor's risk}} - \underbrace{L(f^*)}_{\text{Best risk within } \mathcal{F}}.$$

More Training Examples

Keep $\mathcal{F}$ fixed.

Usually decreases: empirical risk gives a more representative view of expected risk.

Richer Function Class

Keep the training-data size fixed.

Often increases: more candidates can fit accidental patterns in the sample.

Recall: Approximation Error

Approximation error is the cost of restricting predictors to a chosen function class.

$$\text{AE} = \underbrace{L(f^*)}_{\text{Best risk within } \mathcal{F}} - \underbrace{L(f_{\text{Bayes}})}_{\text{Best risk over all predictors}} .$$

More Training Examples

Keep $\mathcal{F}$ fixed.

Unchanged: more data does not change which functions the class can represent.

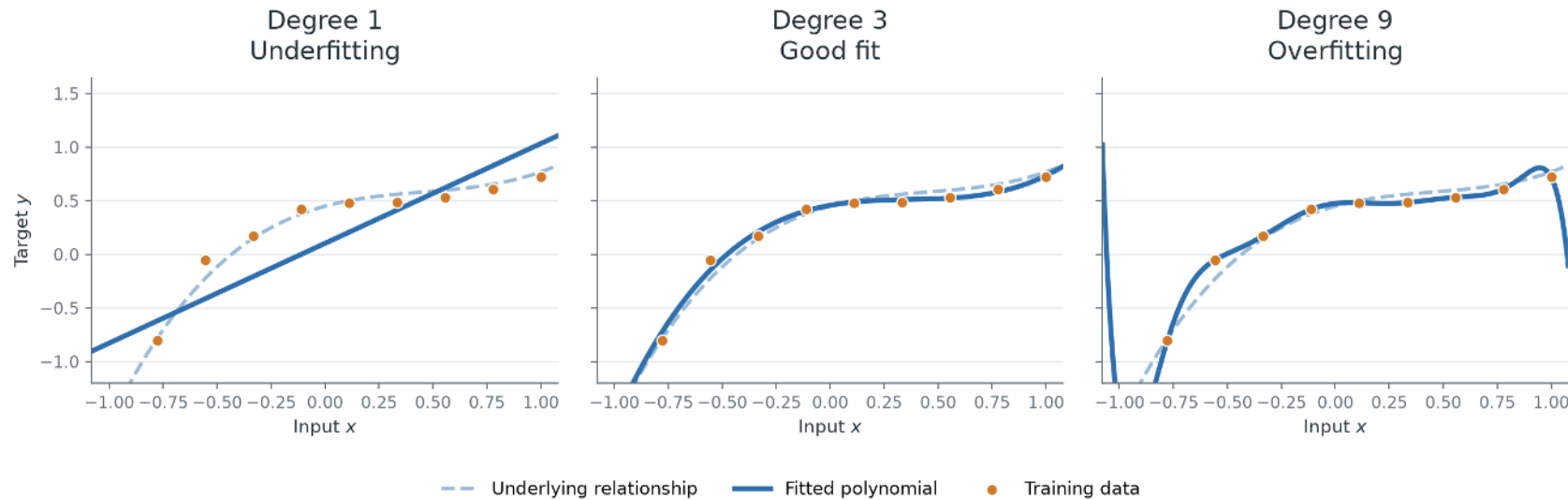
Richer Function Class

Keep the training-data size fixed.

Cannot increase: the best predictor can use the new candidates or keep the old one.

Choosing Polynomial Degree

Choosing polynomial degree changes model flexibility in **discrete steps**.



❓ What if we want flexibility somewhere between degree 3 and degree 4?

We will introduce **regularization** for finer control over the fitted model.

Before Regularization: Bias and Variance

To understand how regularization affects prediction error, we first need the **bias-variance tradeoff**.

Imagine training the same learner on different datasets of the same size.

Bias

Train predictors on different datasets and average their predictions at the same input.

How far is this average from the **Bayes prediction**?

Variance

How much do individual predictions change across training datasets?

Bias, Variance, and Irreducible Error

Another decomposition: **bias, variance, and irreducible error**.

(This requires squared loss: $\ell(\hat{y}, y) = (\hat{y} - y)^2$.)

$$\begin{aligned} \mathbb{E}_D[L(\hat{f}_D)] = & \underbrace{\mathbb{E}_{\mathbf{X}}[(\bar{f}(\mathbf{X}) - f_{\text{Bayes}}(\mathbf{X}))^2]}_{\text{Bias}} \\ & + \underbrace{\mathbb{E}_{\mathbf{X}}[\text{Var}_D(\hat{f}_D(\mathbf{X}))]}_{\text{Variance}} + \underbrace{L(f_{\text{Bayes}})}_{\text{Irreducible error}} . \end{aligned}$$

Here, $\bar{f}$ averages predictions across training datasets.

Bias: average prediction versus Bayes prediction.

Variance: fitted predictions versus their average.

Irreducible error: label randomness that remains at a fixed input.

The Average Predictor

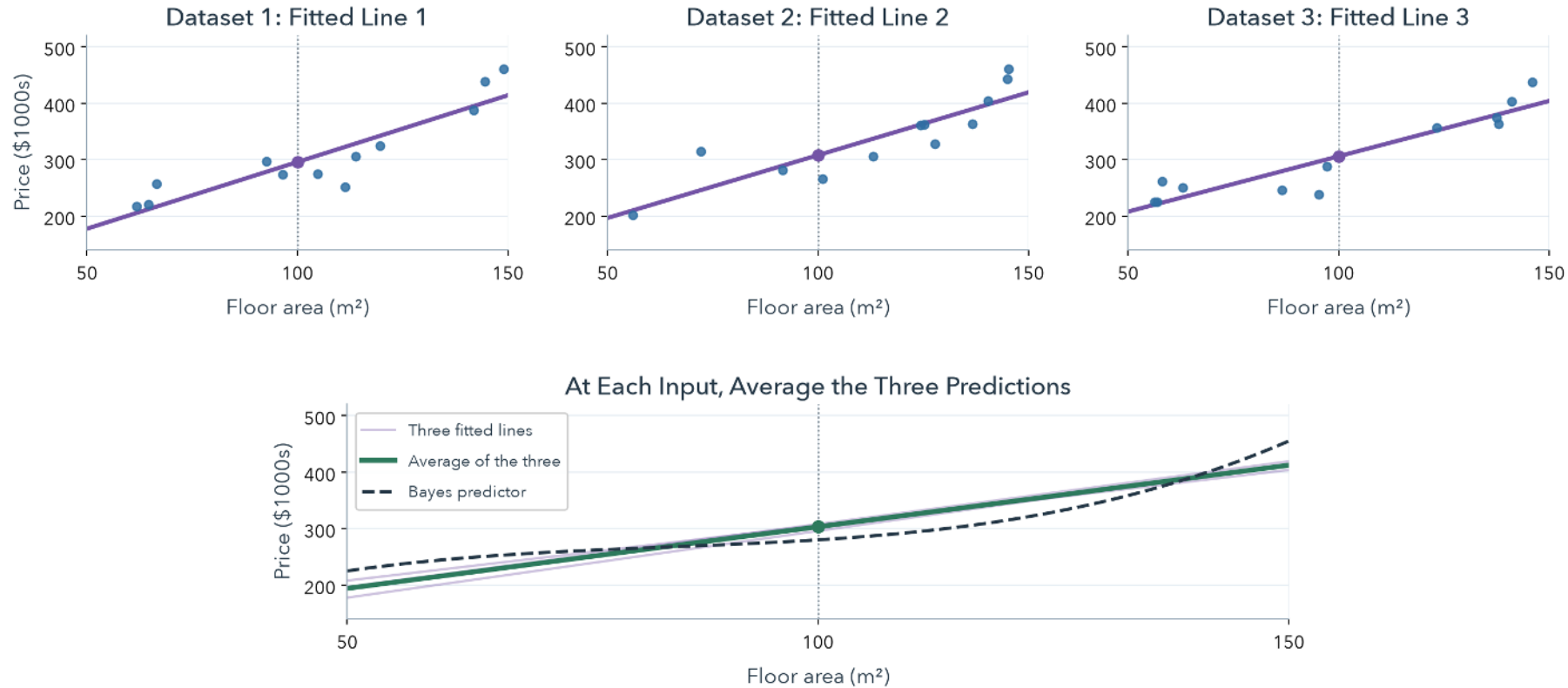
To describe sensitivity to the sample, imagine repeating training on different datasets of the same size.

The **average predictor** averages their predictions at each input.

$$\bar{f}(\mathbf{x}) = \mathbb{E}_D[\hat{f}_D(\mathbf{x})].$$

Example: The Average Predictor

Fit a line to each of **three training datasets** (12 houses per dataset).



? Does the average predictor match the Bayes predictor?

Bias

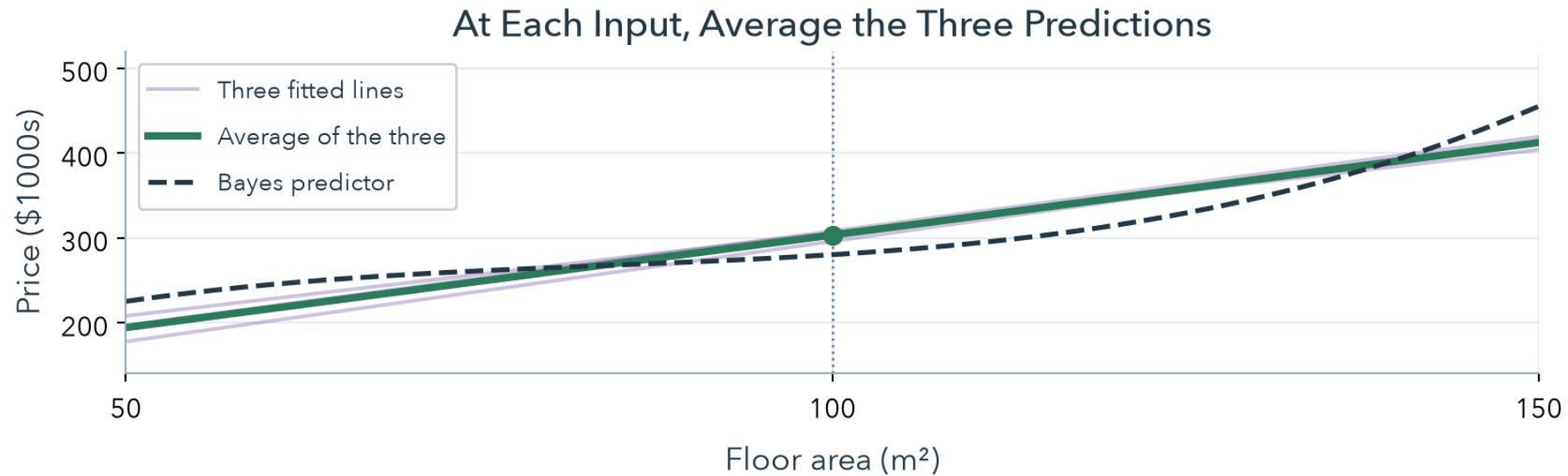
$$\text{Bias} = \mathbb{E}_{\mathbf{X}} \left[\left(\bar{f}(\mathbf{X}) - f_{\text{Bayes}}(\mathbf{X}) \right)^2 \right], \quad \bar{f}(\mathbf{x}) = \mathbb{E}_D[\hat{f}_D(\mathbf{x})].$$

At one input $\mathbf{x}$, $b(\mathbf{x}) = \bar{f}(\mathbf{x}) - f_{\text{Bayes}}(\mathbf{x})$ measures how far the average prediction deviates from the Bayes prediction.

Why There Is Bias

We restrict the learner to a **function class** $\mathcal{F}$. For every training dataset, the fitted predictor must come from that class.

If f_{Bayes} lies outside $\mathcal{F}$, no fitted predictor can match it at every input.



For instance, if every fitted predictor is a line, and their **average is also a line**. It cannot match the curved Bayes predictor, so a systematic difference remains.

Bias, Variance, and Irreducible Error

Another decomposition: **bias, variance, and irreducible error**.

(This requires squared loss: $\ell(\hat{y}, y) = (\hat{y} - y)^2$.)

$$\begin{aligned}\mathbb{E}_D[L(\hat{f}_D)] &= \underbrace{\mathbb{E}_{\mathbf{X}}[(\bar{f}(\mathbf{X}) - f_{\text{Bayes}}(\mathbf{X}))^2]}_{\text{Bias}} \\ &\quad + \underbrace{\mathbb{E}_{\mathbf{X}}[\text{Var}_D(\hat{f}_D(\mathbf{X}))]}_{\text{Variance}} + \underbrace{L(f_{\text{Bayes}})}_{\text{Irreducible error}} .\end{aligned}$$

Variance

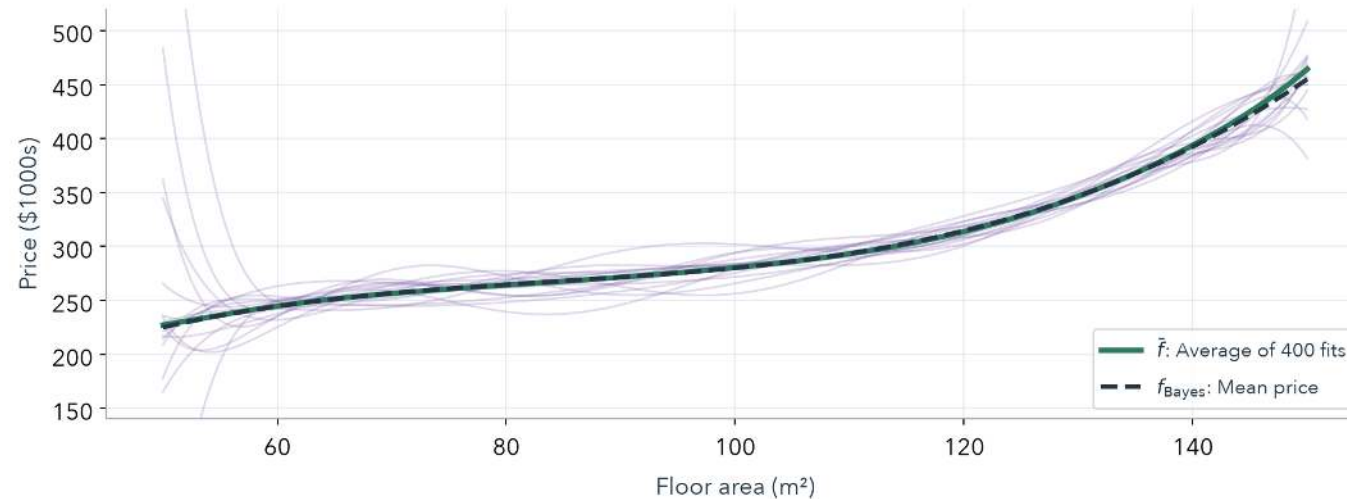
$$\text{Variance} = \mathbb{E}_{\mathbf{x}} \left[\mathbb{E}_D \left[(\hat{f}_D(\mathbf{x}) - \bar{f}(\mathbf{x}))^2 \right] \right], \quad \bar{f}(\mathbf{x}) = \mathbb{E}_D[\hat{f}_D(\mathbf{x})].$$

At one input $\mathbf{x}$, $\hat{f}_D(\mathbf{x}) - \bar{f}(\mathbf{x})$ measures how far a fitted prediction deviates from the average prediction across training datasets.

Averaging this squared deviation over datasets gives the pointwise variance $v(\mathbf{x})$.

Why There Is Variance

Different training datasets contain different examples and label fluctuations. Keep the class and fitting procedure fixed. Changing the dataset can change predictions.



At the same input, fitted predictions vary around their **average**. This variation contributes to variance.

Connecting the Two Decompositions

Let us assume that $\bar{f} = f^*$: the **average learned predictor** equals the **best predictor in the function class**.

$$\underbrace{\bar{f}(\mathbf{x}) = \mathbb{E}_D[\hat{f}_D(\mathbf{x})]}_{\text{Average over training datasets}} \quad \underbrace{f^* \in \arg \min_{f \in \mathcal{F}} L(f)}_{\text{Best in class}}.$$

Under this assumption and **squared loss**:

The **variance of fitted predictions, averaged over inputs**, accounts for estimation error.

The **squared bias, averaged over inputs**, accounts for approximation error.

Optional: Variance and Estimation Error

Assume $\bar{f} = f^*$ and squared loss. Write $\delta_D(\mathbf{x}) = \hat{f}_D(\mathbf{x}) - f^*(\mathbf{x})$.

Then $\mathbb{E}_D[\delta_D(\mathbf{x})] = \bar{f}(\mathbf{x}) - f^*(\mathbf{x}) = 0$ at every input.

$$\begin{aligned}\text{EE} &= \mathbb{E}_D[L(\hat{f}_D)] - L(f^*) \\ &= \mathbb{E}_{D, \mathbf{X}, Y}[(\delta_D(\mathbf{X}) + f^*(\mathbf{X}) - Y)^2 - (f^*(\mathbf{X}) - Y)^2] \\ &= \mathbb{E}_{\mathbf{X}}[\mathbb{E}_D[\delta_D(\mathbf{X})^2]] + 2\mathbb{E}_{\mathbf{X}, Y}\left[(f^*(\mathbf{X}) - Y)\underbrace{\mathbb{E}_D[\delta_D(\mathbf{X})]}_0\right] \\ &= \mathbb{E}_{\mathbf{X}}[\mathbb{E}_D[(\hat{f}_D(\mathbf{X}) - \bar{f}(\mathbf{X}))^2]] \\ &= \mathbb{E}_{\mathbf{X}}[\text{Var}_D(\hat{f}_D(\mathbf{X}))].\end{aligned}$$

The cross term is zero: D is independent of the fresh example $(\mathbf{X}, Y)$.

Optional: Bias and Approximation Error

Assume $\bar{f} = f^*$ and squared loss. Then $b(\mathbf{x}) = \bar{f}(\mathbf{x}) - f_{\text{Bayes}}(\mathbf{x}) = f^*(\mathbf{x}) - f_{\text{Bayes}}(\mathbf{x})$.

Write $\varepsilon = Y - f_{\text{Bayes}}(\mathbf{X})$. Since $f_{\text{Bayes}}(\mathbf{X}) = \mathbb{E}[Y \mid \mathbf{X}]$, we have $\mathbb{E}[\varepsilon \mid \mathbf{X}] = 0$.

$$\begin{aligned} \text{AE} &= L(f^*) - L(f_{\text{Bayes}}) \\ &= \mathbb{E}_{\mathbf{X}, Y} [(f^*(\mathbf{X}) - Y)^2 - (f_{\text{Bayes}}(\mathbf{X}) - Y)^2] \\ &= \mathbb{E}_{\mathbf{X}, Y} [(b(\mathbf{X}) - \varepsilon)^2 - \varepsilon^2] \\ &= \mathbb{E}_{\mathbf{X}} [b(\mathbf{X})^2] - 2\mathbb{E}_{\mathbf{X}} \left[b(\mathbf{X}) \underbrace{\mathbb{E}[\varepsilon \mid \mathbf{X}]}_0 \right] \\ &= \mathbb{E}_{\mathbf{X}} [(\bar{f}(\mathbf{X}) - f_{\text{Bayes}}(\mathbf{X}))^2]. \end{aligned}$$

The cross term vanishes because label noise has conditional mean zero at every input.

Bias, Variance, and Function-Class Size

Keep the **training-data size fixed** and enlarge $\mathcal{F}$. Assume $\bar{f} = f^*$ for each class.

Bias Contribution

Squared bias averaged over inputs; corresponds to AE.

Decreases or stays the same: a richer class can represent the population pattern better.

Variance Contribution

Prediction variance averaged over inputs; corresponds to EE.

Typically increases: more candidate functions can fit sample-specific fluctuations.

Beyond the Equality Assumption

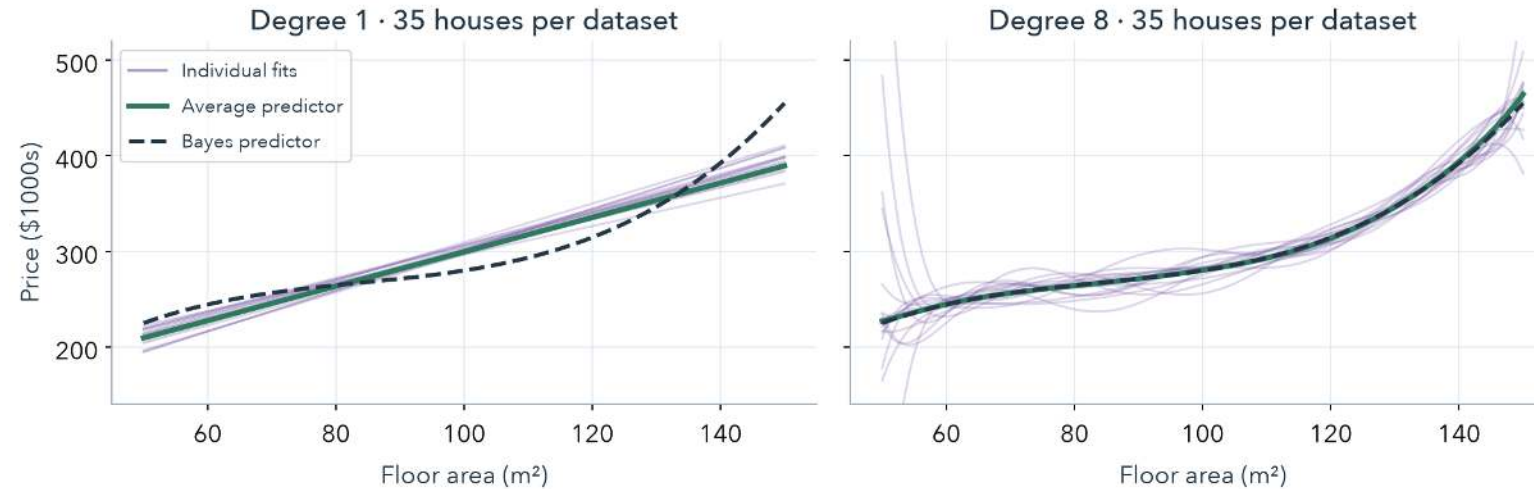
Even when $\bar{f} \neq f^*$, a richer function class often shows the same **bias-variance tradeoff**. Keep the training-data size fixed.

Change	Bias contribution	Variance contribution
Richer class; fixed data size	Usually decreases	Often increases

These are trends in the **squared-bias and variance contributions** to expected risk. Without the assumption, their individual correspondence with AE and EE need not hold.

Check: Underfitting and Overfitting

Compare repeated fits using the same training-data size.



Underfitting

High bias or high variance?

High bias, low variance: similar fits, but their average misses the pattern.

Overfitting

High bias or high variance?

Low bias, high variance: fits vary around a good average.

Controlling the Bias-Variance Tradeoff

We have seen the **bias-variance tradeoff**: a richer function class can reduce bias, but may increase variance.

Choosing the function class—for example, the polynomial degree—is one way to control this tradeoff.

But degree changes in **discrete steps**: degree 3, degree 4, and so on.



Recall our earlier question: what if we want finer control, somewhere between degree 3 and degree 4? How can we do that?

Regularization

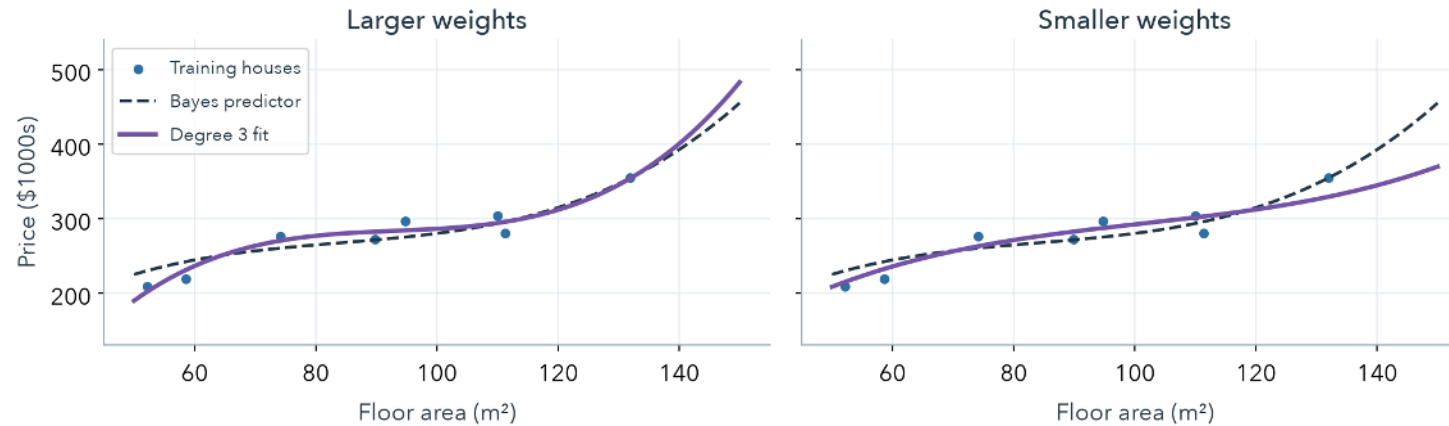
Regularization changes the training objective to discourage certain predictors, such as those with large coefficients.

$$\text{Training objective} = \text{Data-fitting loss} + \text{Penalty.}$$

 Keep a rich polynomial model, but control how strongly it can respond to the training sample.

How Weights Change a Polynomial

Same degree **3**



$$f(z) = 286.2 + 23.9z + 50.2z^2 + 122.7z^3$$

$$\|\mathbf{w}_{1:3}\|_2 = \sqrt{w_1^2 + w_2^2 + w_3^2} \approx 134.7$$

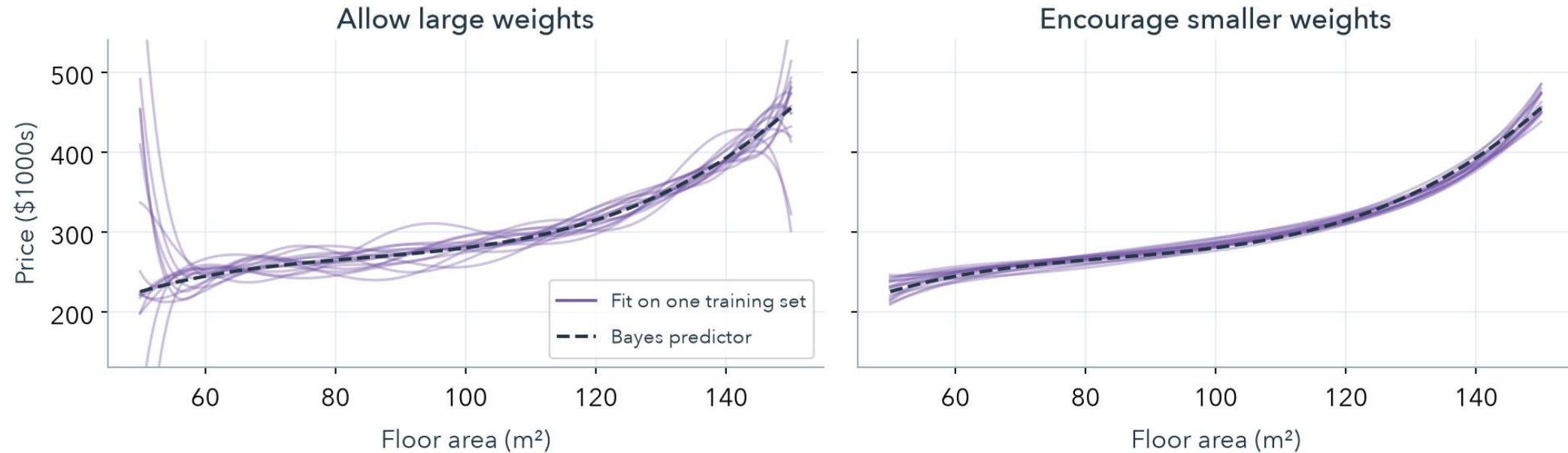
$$f(z) = 292.1 + 45.6z - 3.1z^2 + 35.0z^3$$

$$\|\mathbf{w}_{1:3}\|_2 = \sqrt{w_1^2 + w_2^2 + w_3^2} \approx 57.6$$

The larger quadratic and cubic weights create more pronounced bends. Smaller weights reduce these contributions while keeping the same degree.

Smaller Weights and Variance

Refit degree 9 on **different training sets**



Following sample-specific fluctuations can make the fitted curve change substantially across datasets: **high variance**. Encouraging smaller weights makes these fits more stable.

Penalizing the Weights

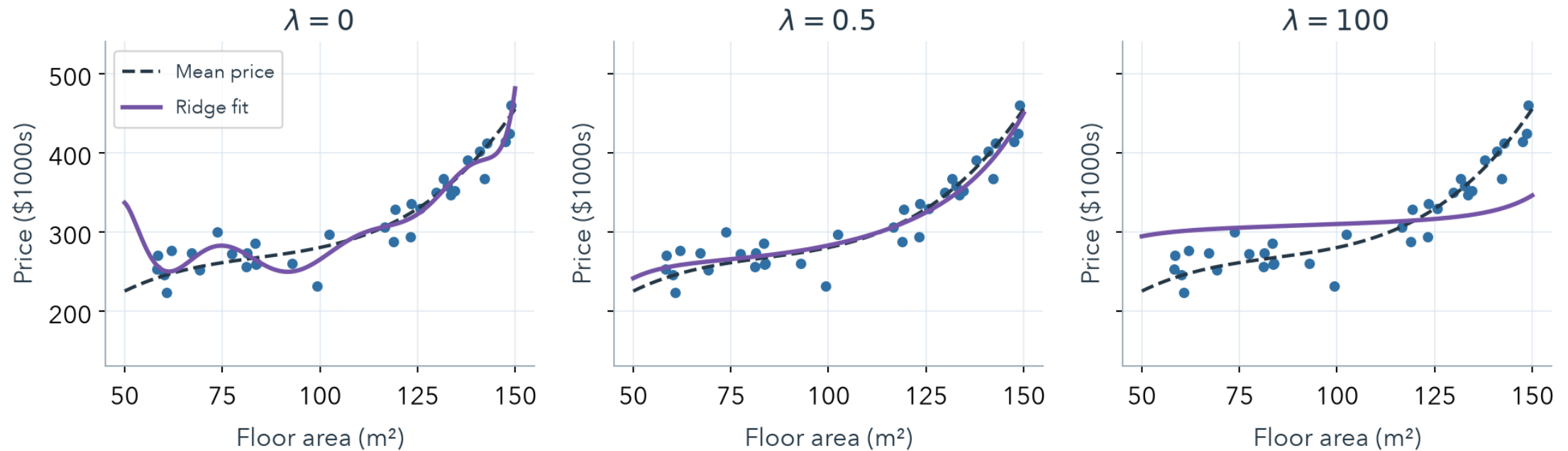
For squared loss, penalize weights $w_1, \dots, w_d$, leaving the intercept w_0 unpenalized.

$$\hat{L}_\lambda(\mathbf{w}) = \underbrace{\frac{1}{n} \sum_{i=1}^n (\mathbf{x}_i^\top \mathbf{w} - y_i)^2}_{\text{Training MSE}} + \underbrace{\frac{\lambda}{n} \sum_{j=1}^d w_j^2}_{\text{Weight penalty}} .$$

The nonnegative hyperparameter λ controls penalty strength. This squared-loss, squared-weight method is **ridge regression**.

Example: Regularized Polynomial Fits

All three fits use degree 9 and the same training houses. The penalty grows from left to right.

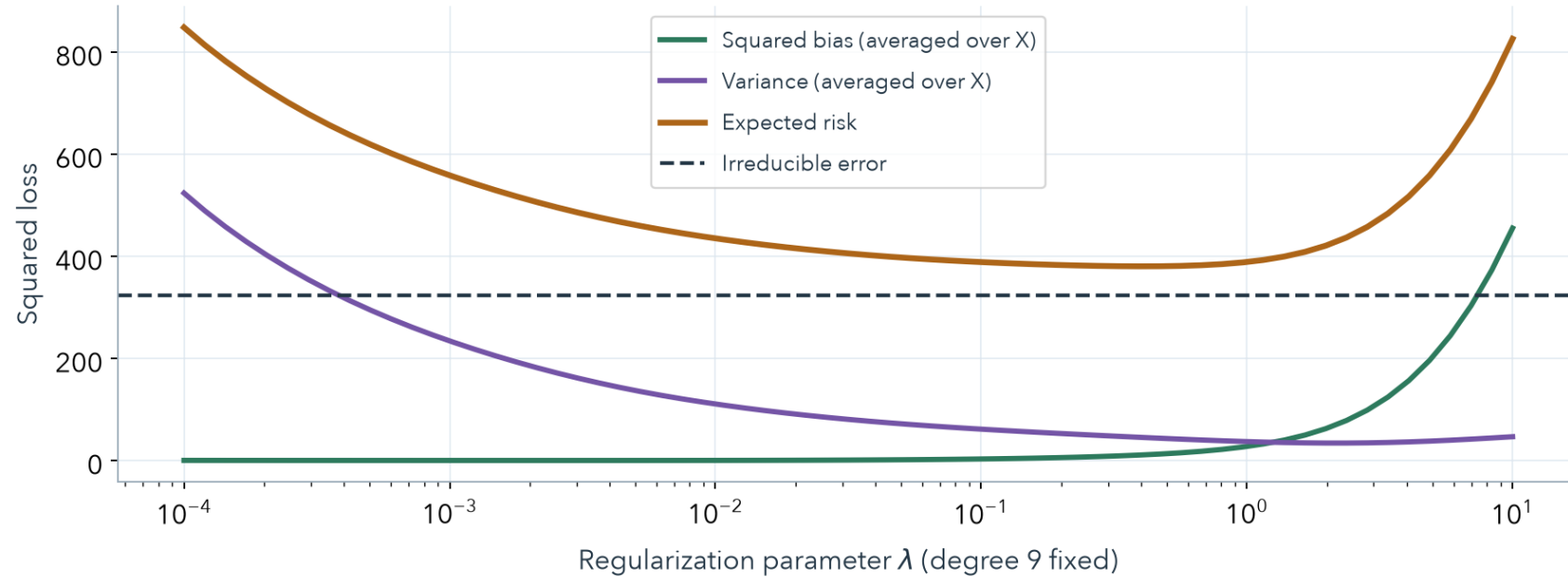


Why Regularization Can Help

Stronger regularization typically **decreases variance**, at the cost of **increasing bias**.

The regularization strength λ gives us a **knob to tune the bias-variance tradeoff**, while keeping the polynomial degree fixed.

Regularization and Bias-Variance



Here, moderate regularization lowers expected risk by reducing variance enough to offset the increase in squared bias.

Interpreting the Regularization Parameter

Choice	Effect on fitting
$\lambda = 0$	Ordinary least squares in the chosen class
Moderate λ	Trade some training fit for smaller weights
Very large λ	Non-intercept weights approach zero

 With an unpenalized intercept, the large- λ limit predicts the mean training label.

Check: Does Stronger Regularization Always Help?

Keep the class, data size, inputs, and population fixed.

❓ Does Stronger Regularization Always Help?

No. it could lead to underfitting

(In Practice) Feature Scale Matters

Floor area (m²) and **number of rooms** have very different scales.

Feature	Example value x_j	Example weight w_j	Contribution $w_j x_j$	Penalty term w_j^2
Floor area (m ²)	100	1	100	1
Rooms	4	25	100	625

Both features contribute 100, but their weights differ greatly. The penalty is **dominated by the room weight**.

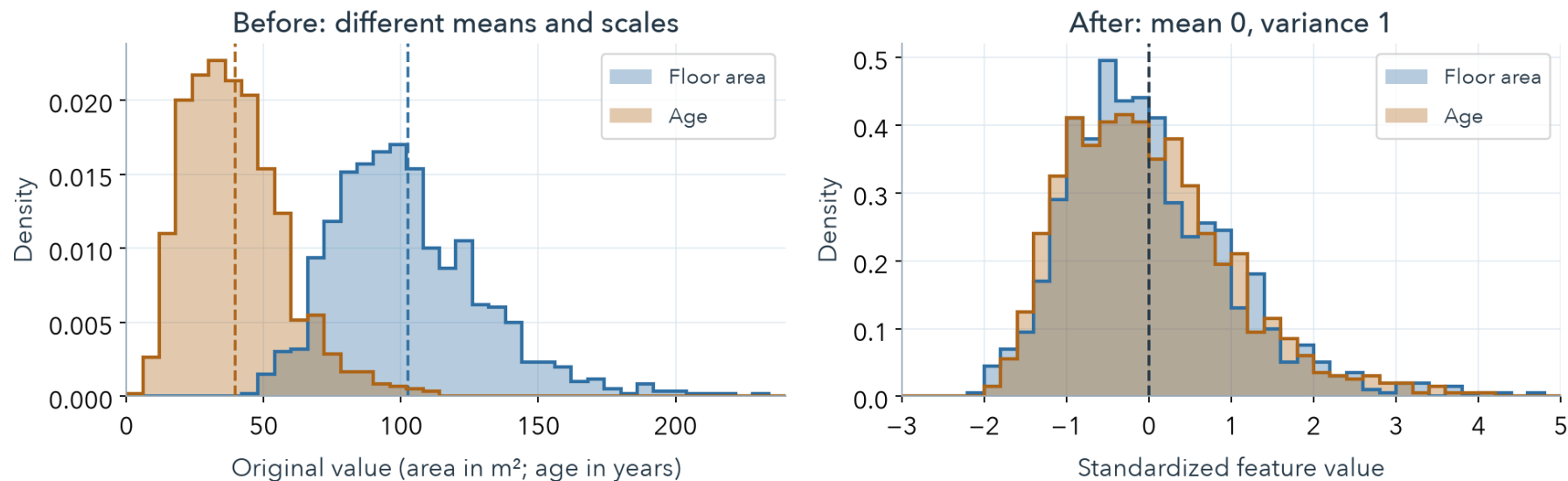


In practice, **adjust feature scales before regularization** to make the penalty comparable across features.

Normalizing the Features

We typically **standardize** each feature to have mean 0 and variance 1:

$$\tilde{x}_j = \frac{x_j - \mu_j}{s_j}, \quad \mu_j = \text{training mean}, \quad s_j = \text{training standard deviation}.$$



Use the **training** means and standard deviations to transform validation and test data too.

Optimization with Squared Loss and Regularization

Recall: Gradient Descent

We minimize the training objective by repeatedly following its negative gradient:

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \nabla \hat{L}_\lambda(\mathbf{w}^{(t)}).$$

Recall the augmented loss:

$$\hat{L}_\lambda(\mathbf{w}) = \underbrace{\frac{1}{n} \sum_{i=1}^n (\mathbf{x}_i^\top \mathbf{w} - y_i)^2}_{\text{Training MSE: gradient already known}} + \underbrace{\frac{\lambda}{n} \sum_{j=1}^d w_j^2}_{\text{Weight penalty}}.$$

We only need to **add the gradient of the penalty**.

Adding the Penalty Gradient

For an individual weight w_j , with $j \geq 1$:

$$\frac{\partial \hat{L}_\lambda}{\partial w_j} = \underbrace{\frac{2}{n} \sum_{i=1}^n (\mathbf{x}_i^\top \mathbf{w} - y_i) x_{ij}}_{\text{Training-MSE gradient from before}} + \underbrace{\frac{2\lambda}{n} w_j}_{\text{New penalty gradient}} .$$

Since $\frac{\partial}{\partial w_j} w_j^2 = 2w_j$, the extra term is proportional to the current weight.

The intercept w_0 has no penalty, so its gradient is unchanged.

How Regularization Changes the Update

Write $g_j = \frac{\partial \hat{L}}{\partial w_j}(\mathbf{w}^{(t)})$ for the training-MSE gradient.

Without regularization: $w_j^{(t+1)} = w_j^{(t)} - \eta g_j.$

With regularization: $w_j^{(t+1)} = w_j^{(t)} - \eta g_j - \frac{2\eta\lambda}{n} w_j^{(t)}.$

The extra term **pulls the weight toward zero**: subtract from a positive weight, add to a negative weight.

$$w_j^{(t+1)} = \underbrace{\left(1 - \frac{2\eta\lambda}{n}\right)}_{\text{Shrinkage factor}} w_j^{(t)} - \eta g_j.$$

Selecting the Regularization Strength

Fit a model for each candidate λ , then compare its **validation MSE**.

$$\text{Training objective} = \text{Training MSE} + \frac{\lambda}{n} \sum_{j=1}^d w_j^2,$$

$$\text{Validation score} = \frac{1}{n_{\text{val}}} \sum_{i=1}^{n_{\text{val}}} (\hat{f}_{\lambda}(\mathbf{x}_i) - y_i)^2.$$

i Do not add the weight penalty on validation data. We care about prediction MSE on new examples. The penalty guides training.

Example: Choosing λ

Which λ gives the lowest validation MSE?

λ	Training objective (MSE + penalty)	Training MSE	Validation MSE (no penalty)
0	319.2	319.2	415.1
0.01	331.5	326.3	398.9
0.1	362.6	334.1	364.4
1	543.0	373.6	345.3
10	1549.0	755.8	567.1

Choose $\lambda = 1$: training MSE rises from **319.2 to 373.6**, but validation MSE falls from **415.1 to 345.3**.

Accepting a worse fit to the training data can improve predictions on new data.

Summary: Regularization

- **Training objective:** add a weight penalty to the training MSE.
- **Bias-variance tradeoff:** stronger regularization typically reduces variance and increases bias, with the function class fixed.
- **Optimization:** add the penalty gradient; its contribution pulls non-intercept weights toward zero.
- **Feature scale:** standardize features using training-set statistics.
- **Model selection:** choose λ using validation MSE **without the penalty**.

Chapter Review: Evaluating and Improving Predictors

Review: Training Loss Is Optimistic

We choose the predictor to fit the training data, so its training loss is typically **optimistic about performance on new data**.

A low training loss can reflect both the population pattern and sample-specific noise.

 Use held-out data to select models and assess how well they generalize.

Review: Training, Validation, and Test Data

All available data

Training	Validation	Test
Fit coefficients	Choose degree or λ	Evaluate final predictor



Keep the test set untouched during model selection. For squared loss, compare **validation and test MSE without the weight penalty**.

Two Ways to Decompose Expected Risk

$$\mathbb{E}_D[L(\hat{f}_D)] = \underbrace{\mathbb{E}_D[L(\hat{f}_D)] - L(f^*)}_{\text{Estimation error}} + \underbrace{L(f^*) - L(f_{\text{Bayes}})}_{\text{Approximation error}} + \underbrace{L(f_{\text{Bayes}})}_{\text{Irreducible error}} .$$

$$\mathbb{E}_D[L(\hat{f}_D)] = \underbrace{\mathbb{E}_{\mathbf{X}}[(\bar{f}(\mathbf{X}) - f_{\text{Bayes}}(\mathbf{X}))^2]}_{\text{Bias contribution}} + \underbrace{\mathbb{E}_{\mathbf{X}}[\text{Var}_D(\hat{f}_D(\mathbf{X}))]}_{\text{Variance}} + \underbrace{L(f_{\text{Bayes}})}_{\text{Irreducible error}} .$$

The second view requires squared loss. When $\bar{f} = f^*$, squared bias averaged over inputs equals AE, and variance averaged over inputs equals EE.

How the Different Errors Change

The usual trends in our polynomial-regression setting:

Change	Estimation error	Approximation error	Variance	Bias contribution
More training data n	Usually decreases	Unchanged	Usually decreases	Often similar
Higher degree p , no penalty	Often increases	Decreases until $f_{\text{Bayes}} \in \mathcal{F}_p$; then zero	Often increases	Usually decreases
Stronger penalty λ , fixed degree	Depends on tradeoff	Unchanged	Usually decreases	Usually increases

More data: increasing n typically reduces EE and variance. AE is unchanged; in the classical picture, bias is unchanged too.

How the Different Errors Change

The usual trends in our polynomial-regression setting:

Change	Estimation error	Approximation error	Variance	Bias contribution
More training data n	Usually decreases	Unchanged	Usually decreases	Often similar
Higher degree p , no penalty	Often increases	Decreases until $f_{\text{Bayes}} \in \mathcal{F}_p$; then zero	Often increases	Usually decreases
Stronger penalty λ , fixed degree	Depends on tradeoff	Unchanged	Usually decreases	Usually increases

Richer class: with $\lambda = 0$, increasing degree p typically increases EE and variance while decreasing AE and bias.

How the Different Errors Change

The usual trends in our polynomial-regression setting:

Change	Estimation error	Approximation error	Variance	Bias contribution
More training data n	Usually decreases	Unchanged	Usually decreases	Often similar
Higher degree p , no penalty	Often increases	Decreases until $f_{\text{Bayes}} \in \mathcal{F}_p$; then zero	Often increases	Usually decreases
Stronger penalty λ , fixed degree	Depends on tradeoff	Unchanged	Usually decreases	Usually increases

Zero approximation error: if $f_{\text{Bayes}} \in \mathcal{F}_p$, then $L(f^*) = L(f_{\text{Bayes}})$ and AE is zero.

How the Different Errors Change

The usual trends in our polynomial-regression setting:

Change	Estimation error	Approximation error	Variance	Bias contribution
More training data n	Usually decreases	Unchanged	Usually decreases	Often similar
Higher degree p , no penalty	Often increases	Decreases until $f_{\text{Bayes}} \in \mathcal{F}_p$; then zero	Often increases	Usually decreases
Stronger penalty λ , fixed degree	Depends on tradeoff	Unchanged	Usually decreases	Usually increases

Regularization leaves AE unchanged: changing λ does not change the function class $\mathcal{F}_p$.

How the Different Errors Change

The usual trends in our polynomial-regression setting:

Change	Estimation error	Approximation error	Variance	Bias contribution
More training data n	Usually decreases	Unchanged	Usually decreases	Often similar
Higher degree p , no penalty	Often increases	Decreases until $f_{\text{Bayes}} \in \mathcal{F}_p$; then zero	Often increases	Usually decreases
Stronger penalty λ , fixed degree	Depends on tradeoff	Unchanged	Usually decreases	Usually increases

Effect on EE is unclear: decreasing λ may bring the fit closer to f^* , but also makes it vary more across datasets. We will not analyze this further.