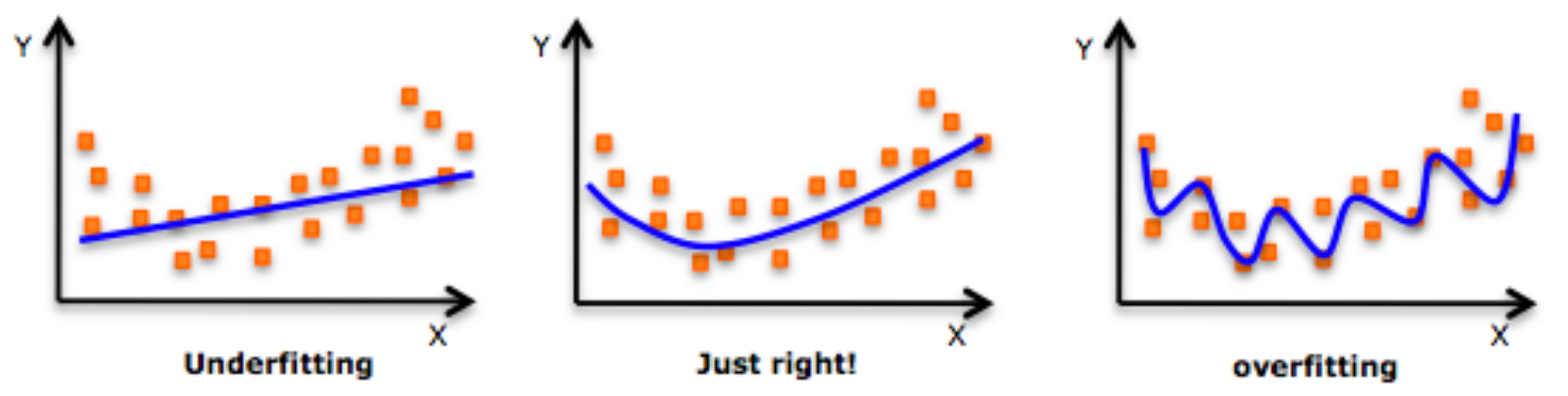
# Regularization

Reduce Overfitting By Punishing Complexity



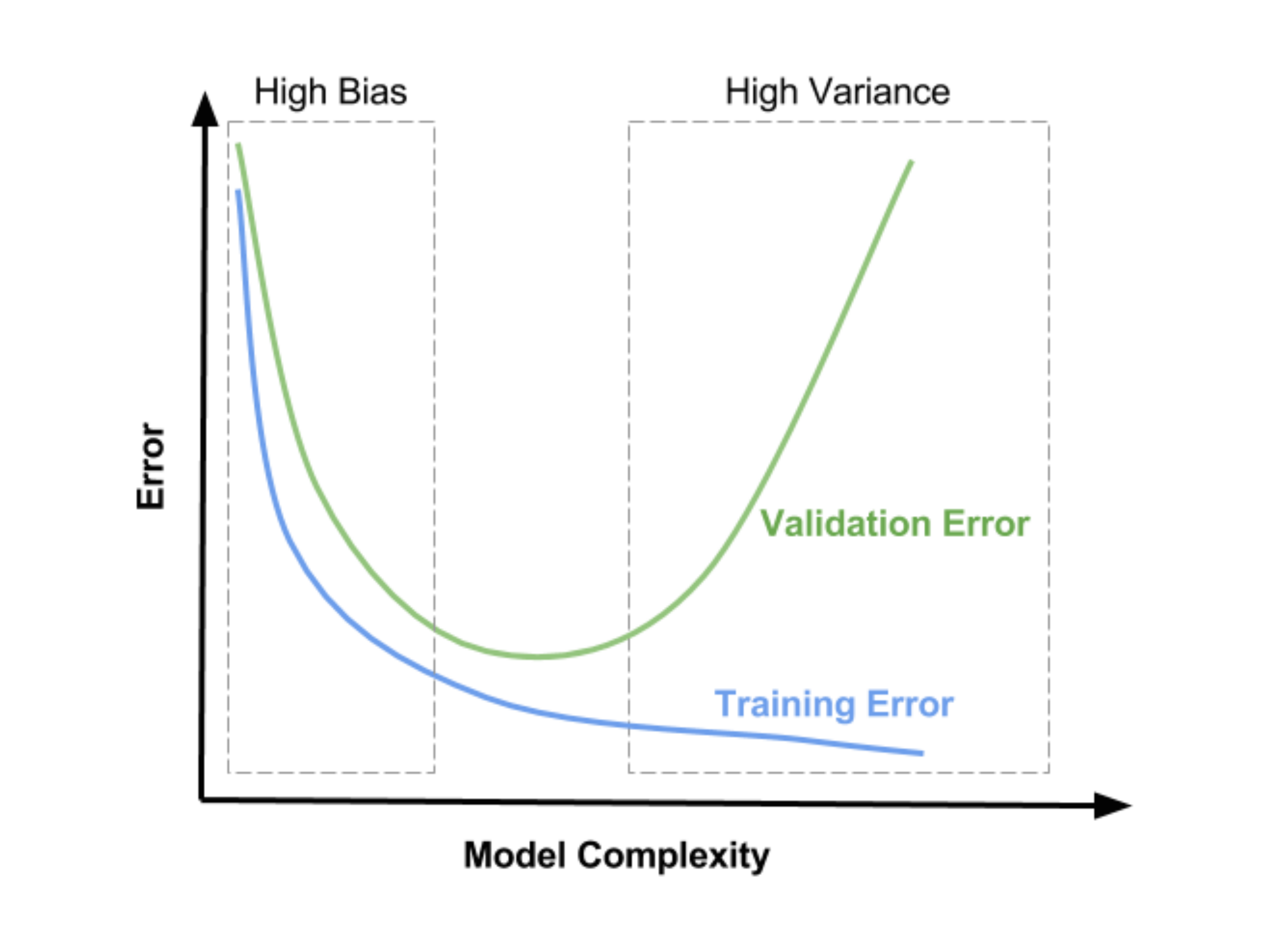
# Roadmap

1. The basics: prereqs and review
2. Motivation: what does regularization solve?
3. Methods in detail
4. Theory: why does it work?

# The Basics

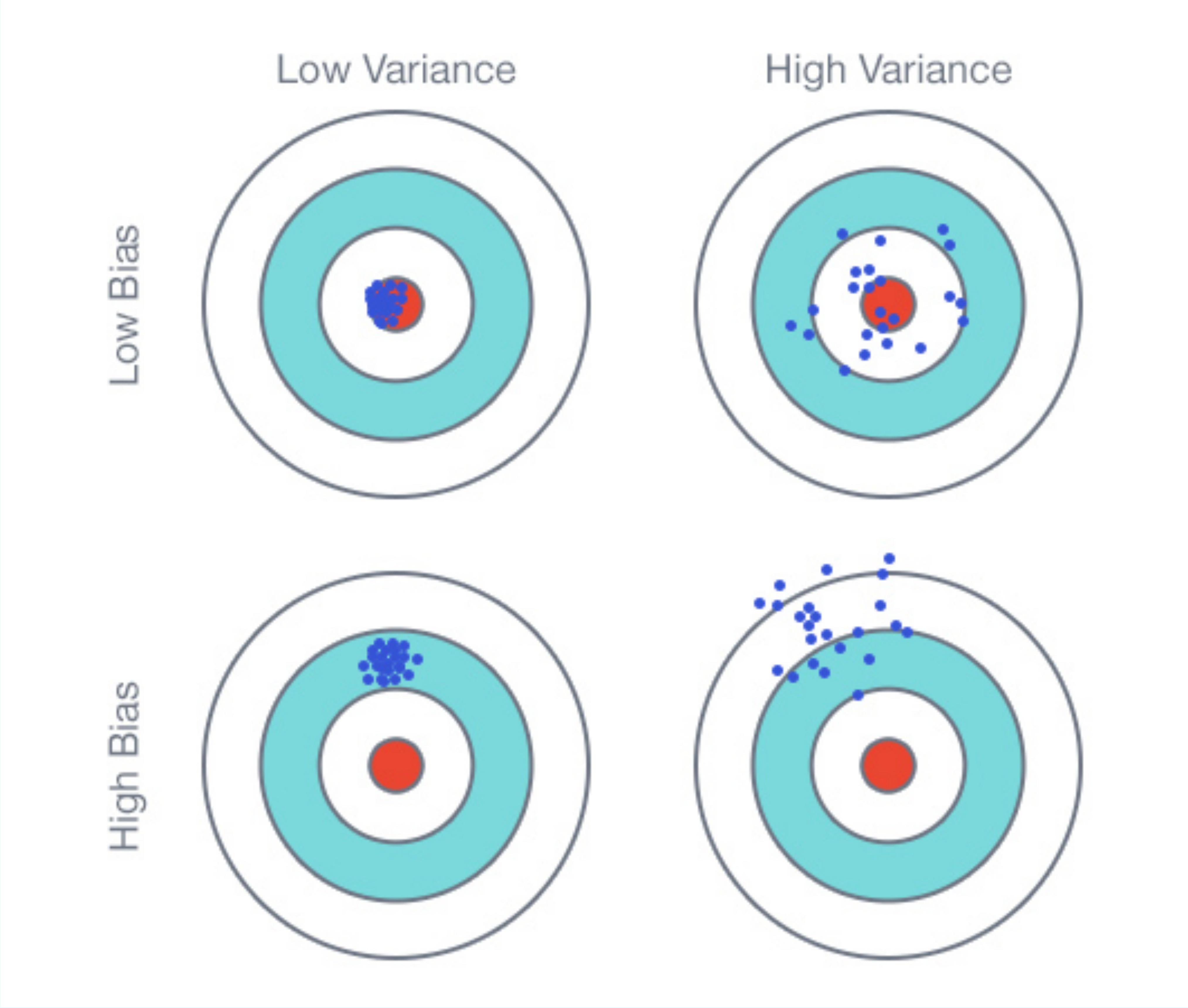
- **Cost functions**: minimize to fit model
- **Bias-variance tradeoff**: dangers of complexity in generalization error
- **Linear Regression**, including polynomial

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2,$$

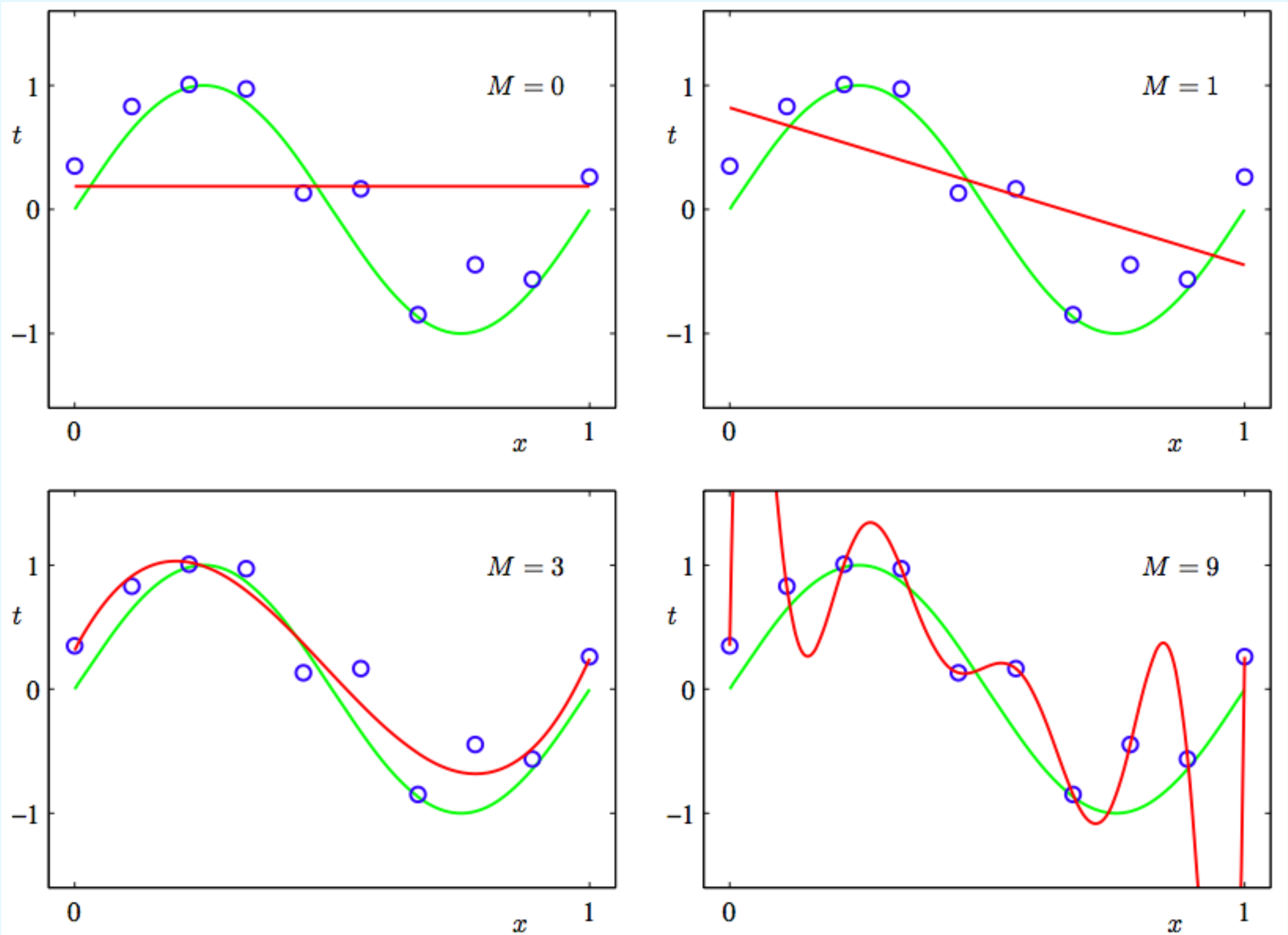



$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \cdots + \beta_n x^n + \varepsilon.$$

# Bias/Variance Visualized



# Linear regression: we can control fit with polynomial degree…



# But what if we want more granular tuning?

- <u>Example</u>: degree 1 model may be overfit, but constant model is underfit

- Also want: adjust complexity without fundamentally changing model

- **Solution**: regularization. Include complexity penalty directly in the cost function

**new cost function**
M(w): model error
R(w): complexity cost
lambda: adjustable weight of complexity cost

$$M(\mathbf{w}) + \lambda R(\mathbf{w})$$

# The Linear Regression Setting

- <u>Ridge regression</u>: penalty term = sum of squared coefficients

**Fit model by minimizing:**

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}\beta_j^2 = \mathrm{RSS} + \lambda\sum_{j=1}^{p}\beta_j^2,$$

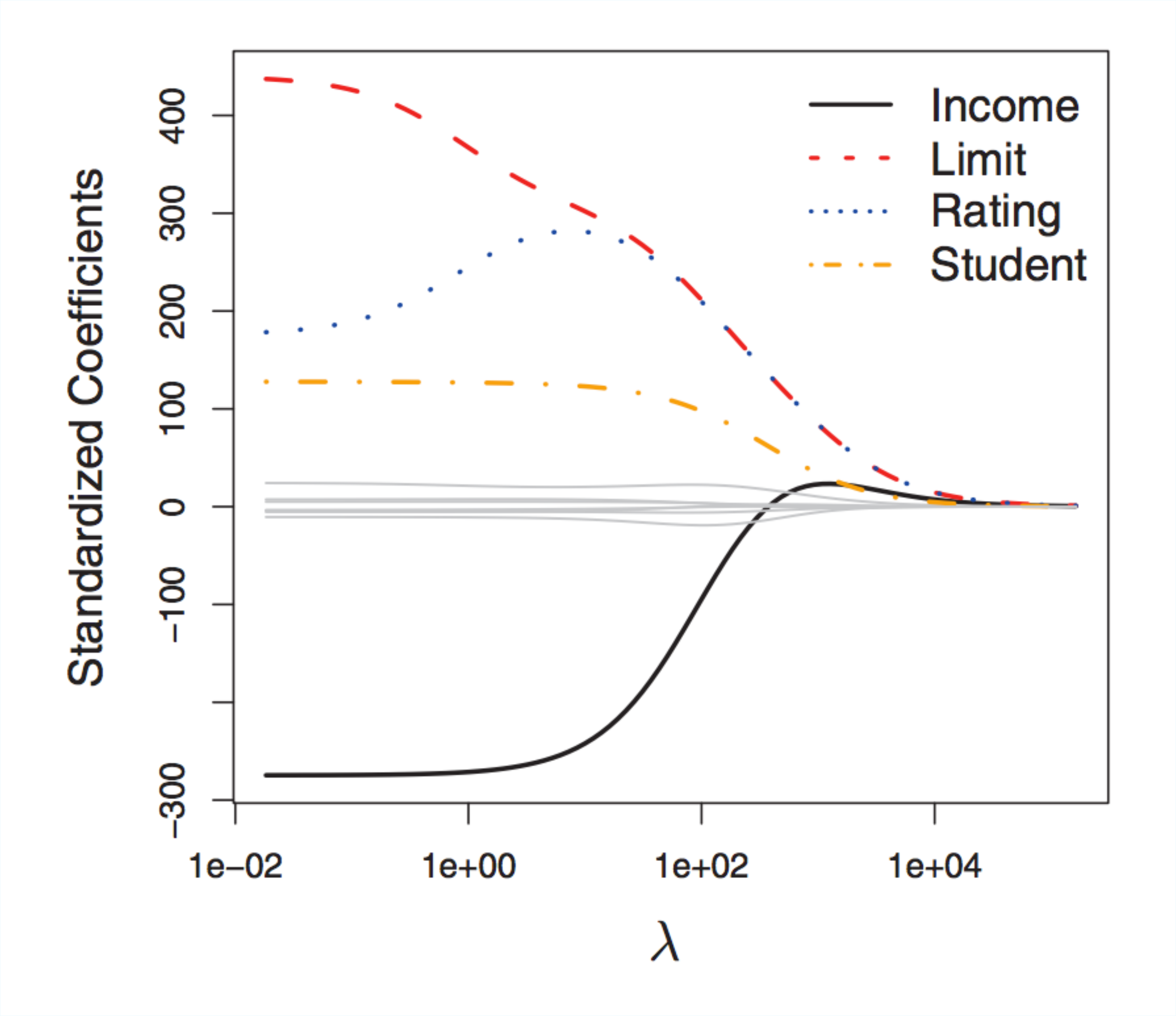
- Penalty term has impact of “shrinking” coefficients toward 0, increasing bias but reducing variance

- Choose lambda to minimize validation error

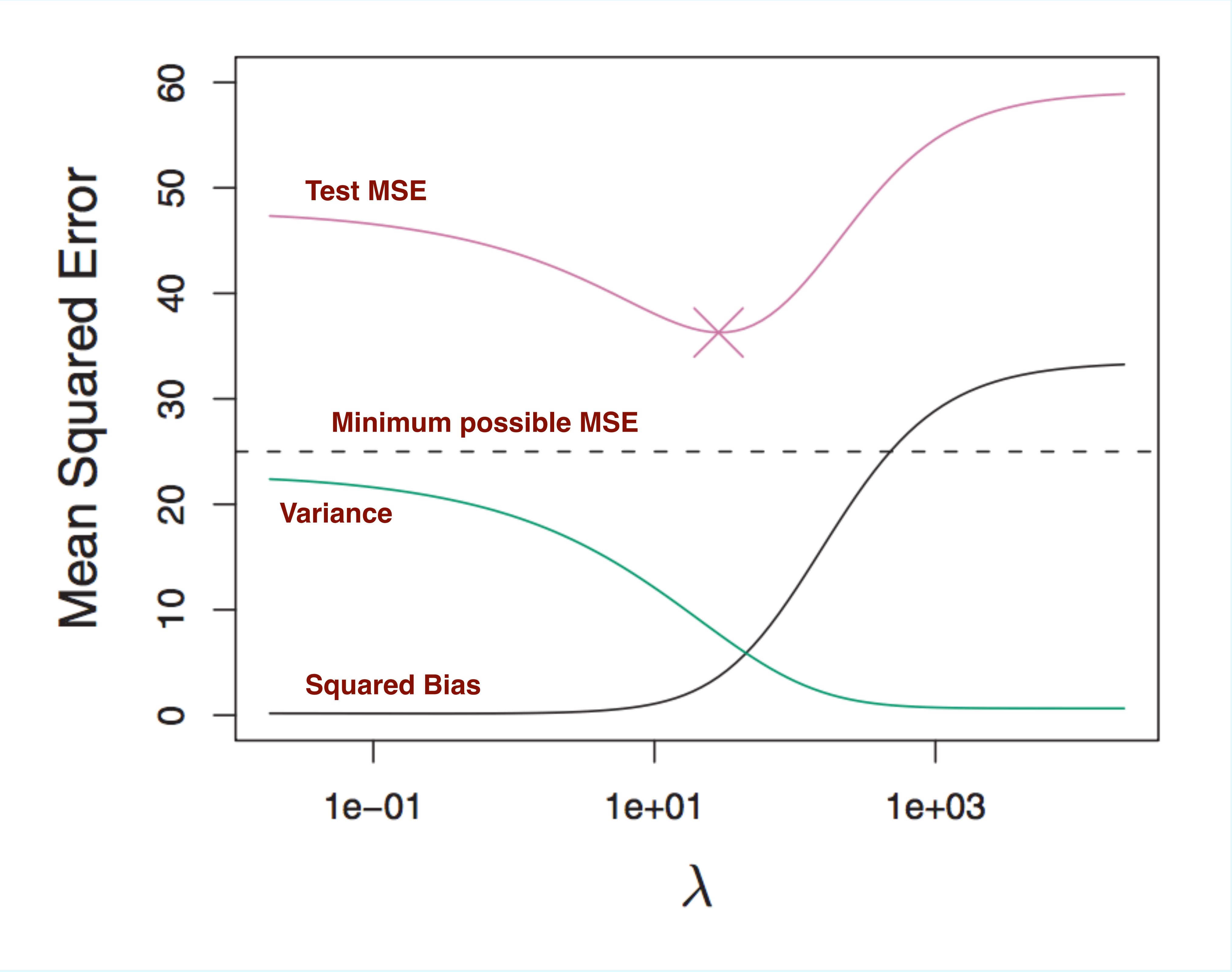
- <u>Warning: scale matters!</u>: standardize your features

$$x' = \frac{\overset{\text{Original}}{x} - \overset{\text{Mean}}{\bar{x}}}{\underset{\text{Standard deviation}}{\sigma}}$$

# Ridge regression coefficient shrinking as lambda increases



# Ridge regression bias-variance tradeoff: increasing lambda reduces generalization error, up to a point!



# An Alternative: Lasso Regularization

- Lasso regression: penalty term = sum of absolute value of coefficients

**Fit model by minimizing:**

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j| = \text{RSS} + \lambda\sum_{j=1}^{p}|\beta_j|.$$

- Like ridge regression, increasing lambda raises bias but lowers variance.

- Unlike ridge regression, lasso performs variable selection: coefficients are forced to 0 as lambda increases

- Math aside: penalties are L1 and L2 norms

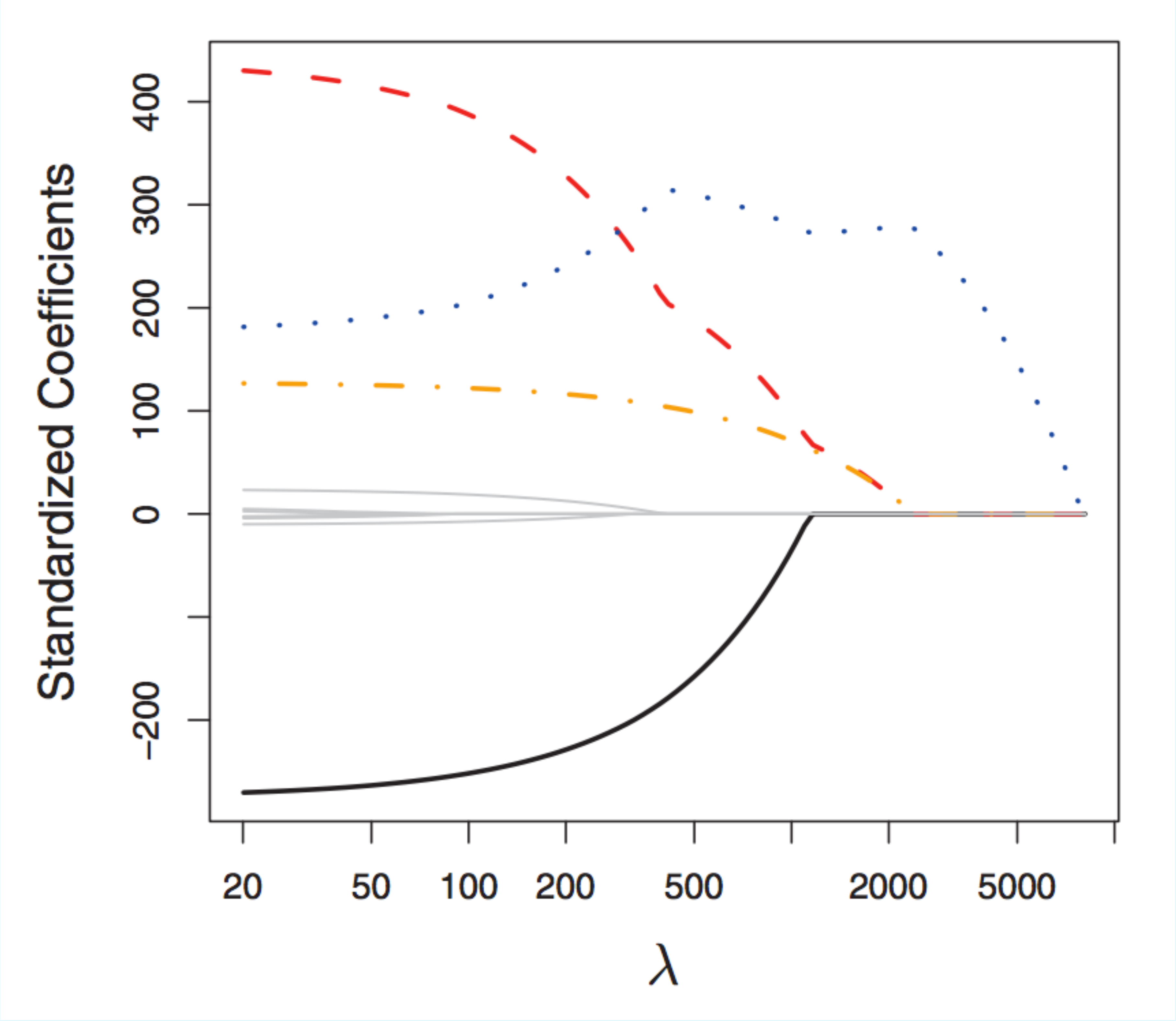
**Lasso - L1**

$$\|\beta\|_1 = \sum|\beta_j|$$

**Ridge - L2**

$$\|\beta\|_2 = \sqrt{\sum_{j=1}^{p}\beta_j{}^2}$$

# Lasso regression: feature selection as lambda increases



# Lasso vs. Ridge

- Everything is data dependent: always validate
- Lasso performs feature selection (interpretability bonus), but may underperform if the target is truly dependent on many features
- Also a hybrid model: elastic net

$$\lambda \sum_{j=1}^{p} \left( \alpha \beta_j^2 + (1 - \alpha) |\beta_j| \right)$$

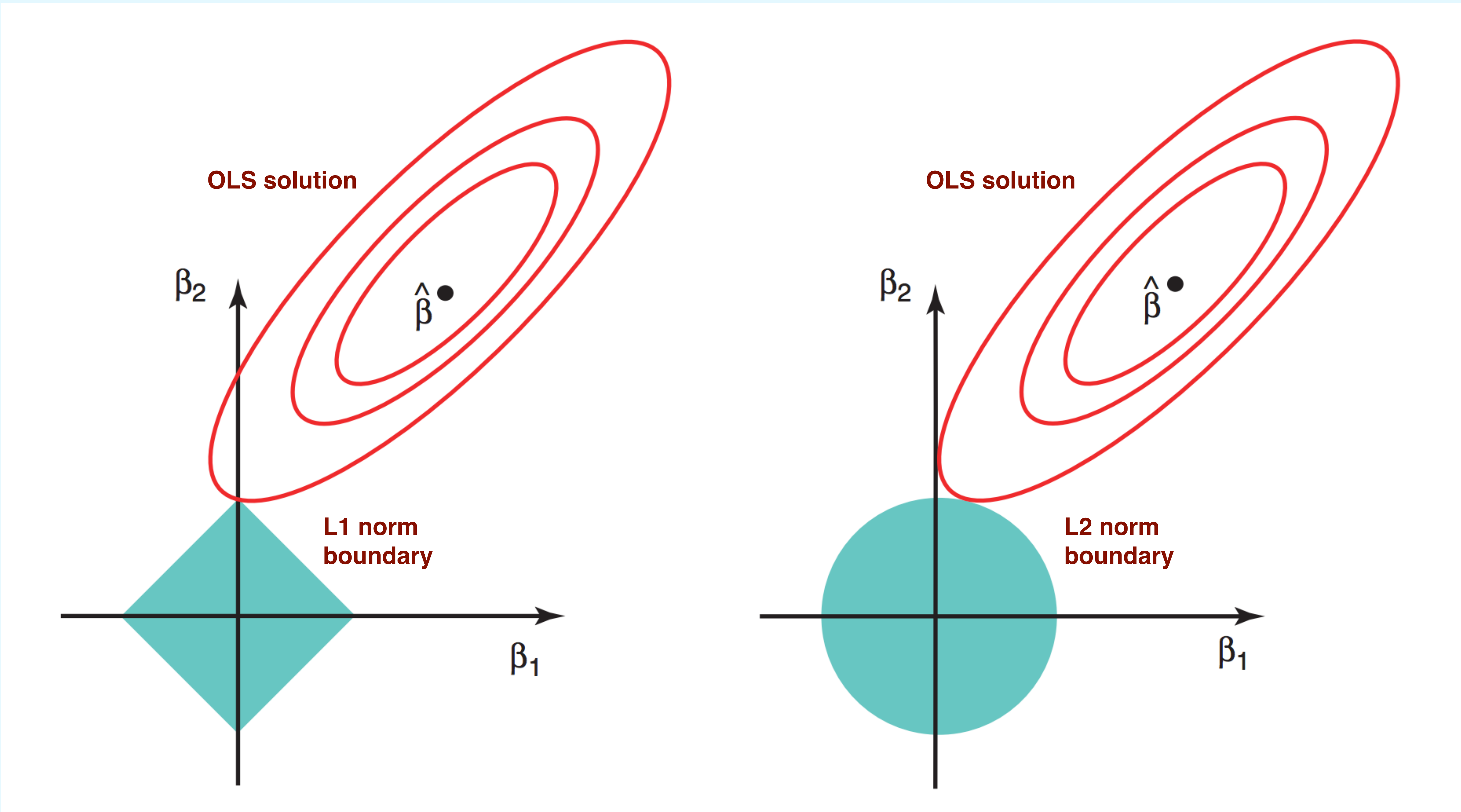
# So why does it work?

- First some geometry — equivalent formulations of minimizing lasso and ridge cost functions:

$$\underset{\beta}{\text{minimize}} \left\{ \sum_{i=1}^{n} \left( y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \right\} \quad \text{subject to} \quad \sum_{j=1}^{p} |\beta_j| \leq s$$

$$\underset{\beta}{\text{minimize}} \left\{ \sum_{i=1}^{n} \left( y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \right\} \quad \text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 \leq s$$

# Cost function minimum: intersection of penalty boundary and best ordinary least squares contour




**Lasso - L1** **Ridge - L2**

# And now some Bayes: regularization is just imposing a certain prior on coefficients

- Letting f be the likelihood (probability of target given parameter vector beta) and p(beta) be the prior distribution of beta, we get the posterior of beta

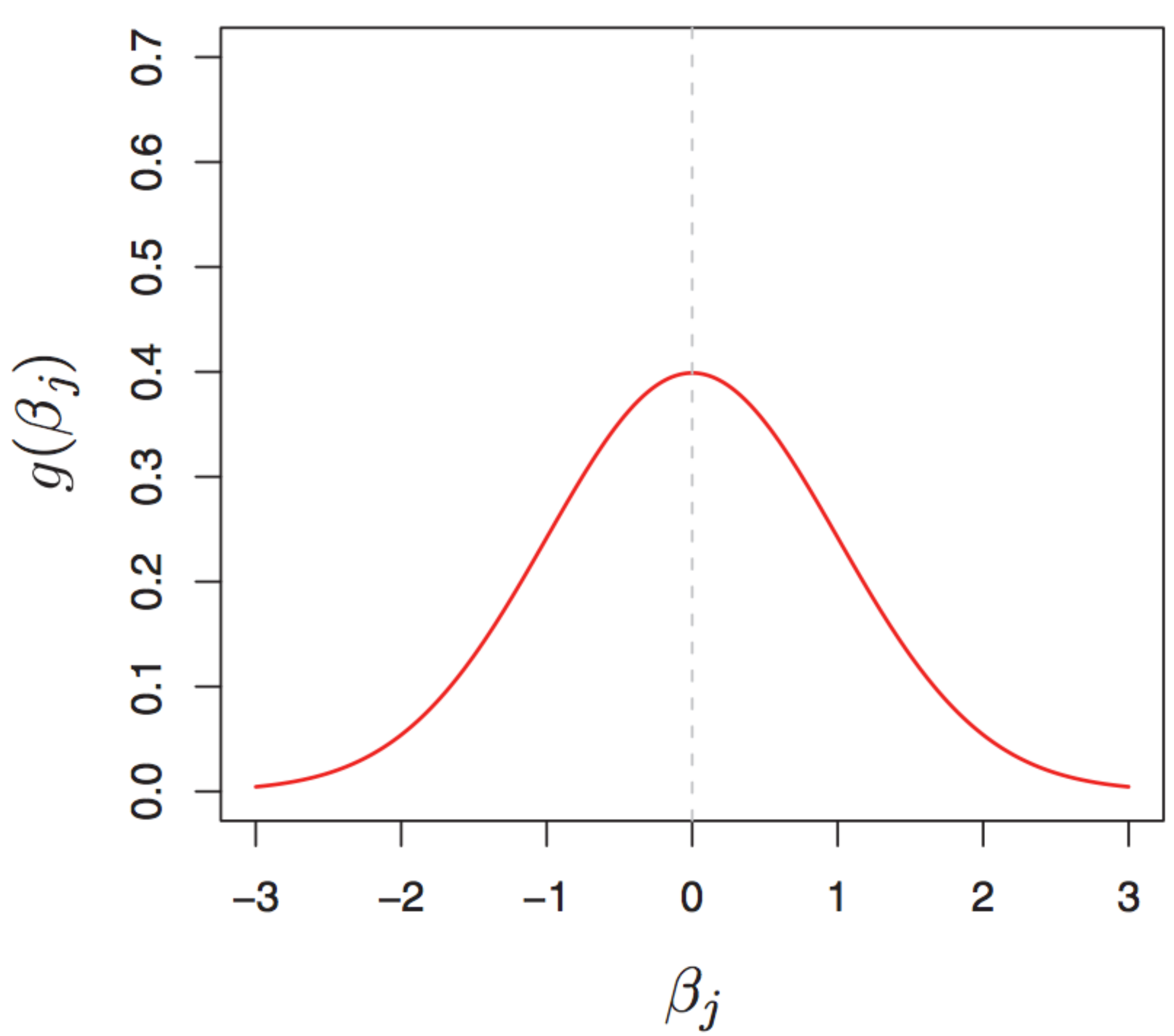
- p(beta) is derived from independent draws of a <u>prior coefficient density function g</u> that we choose when regularizing

$$p(\beta|X,Y) \propto f(Y|X,\beta)p(\beta|X) = f(Y|X,\beta)p(\beta)$$

$$p(\beta) = \prod_{j=1}^{p} g(\beta_j)$$

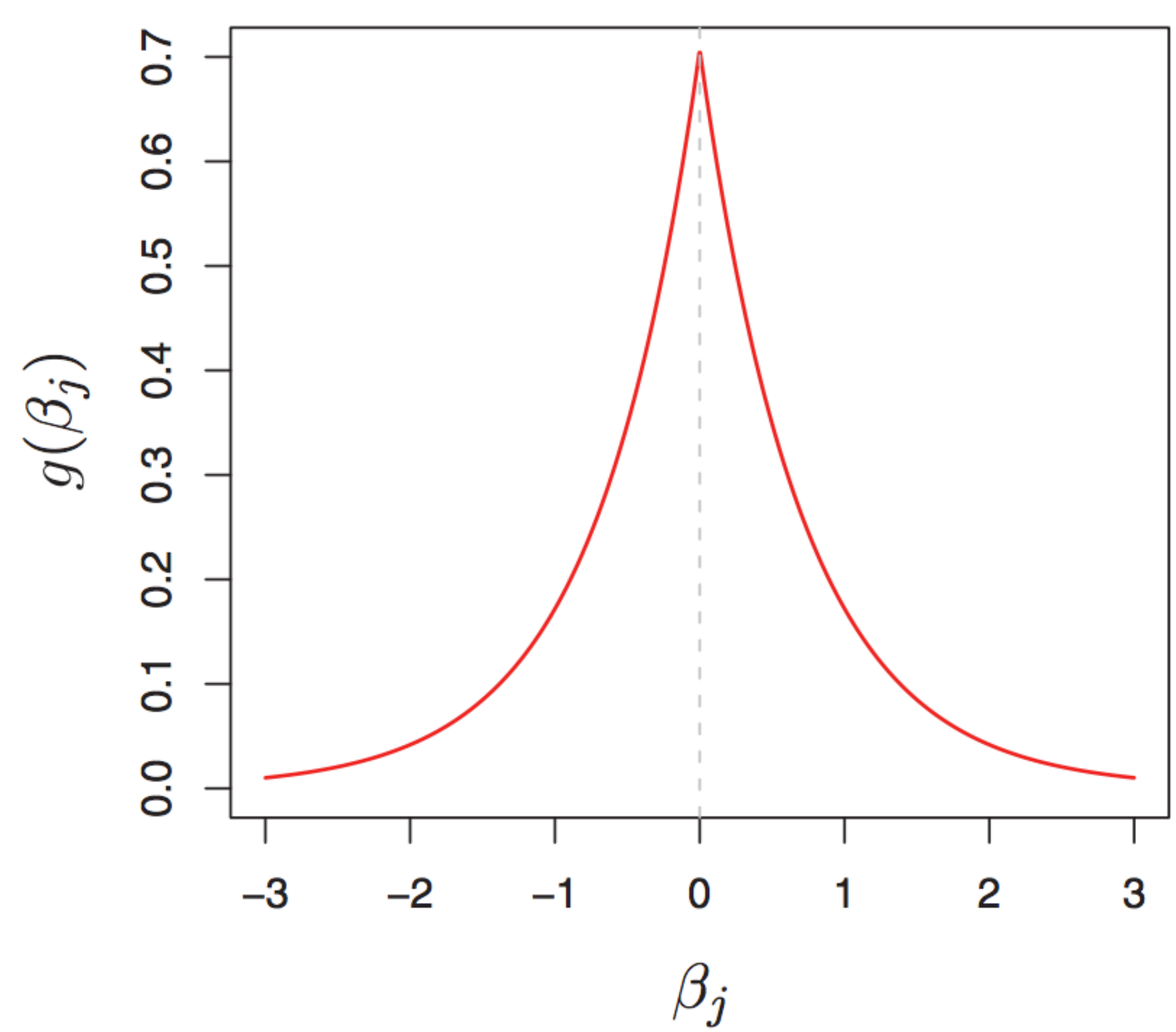
# Assumed prior distributions of coefficients

**Ridge - L2**




**Gaussian**

**Lasso - L1**




**Laplace**

# Sources

- Page 1: Analytics Vidhya
- Page 3: Introduction to Statistical Learning with Applications in R; Stack Overflow; Wikipedia
- Page 4: Deniz Yuret
- Page 5: Justin Domke
- Page 6: ISLR, wikipedia
- Pages 7-10: ISLR
- Page 11: The Elements of Statistical Learning
- Pages 12-15: ISLR