Linear Models for Regression

CSci 5525: Machine Learning

Instructor: Nicholas Johnson

September 15, 2020

Announcements

- HW0 due tonight at 11:59 pm central
  - Submit via Canvas
- Office hours posted
- Syllabus updated

Problem

Suppose you work at a restaurant and want to predict how much the customers tip. You are given the following data consisting of the total bill amount and the tip added.

Total Bill	Tip
16.99	1.01
10.34	1.66
21.01	3.50
23.68	3.31
24.59	3.61
25.29	4.71
8.77	2.00

One heuristic is to predict the average tip: \$2.83.

Can we do better?

Regression

- Dataset: $\mathcal{D} = \{(\text{total bill}_i, \text{tip}_i)\}_{i=1}^n = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$
- Features $\mathbf{x}_i$ and targets y_i
- Supervised learning problem
  - $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}$, $y \in \mathcal{Y} \subset \mathbb{R}$ (regression)
- Goal: find prediction function $f : \mathcal{X} \to \mathcal{Y}$

Two Approaches

Empirical Risk Minimization (ERM)

- Pick class of predictors $\mathcal{C}$ (linear in this lecture)
- Pick loss function $\ell(\cdot)$
- Minimize empirical risk over model/parameters

Maximum Likelihood Estimation (MLE)

- Choose statistical model (conditional distribution)
- Maximize likelihood function

Approach 1: ERM

- Loss function: least square loss for prediction $\hat{y} = \hat{f}(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$

$$\ell(y, \hat{y}) = (y - \hat{y})^2$$

- Goal: minimize least squares empirical risk:

$$\mathcal{R}(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^{n} (y_i - \mathbf{w}^\top \mathbf{x}_i)^2 = \frac{1}{n} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2$$

- If $\mathbf{X}$ is full-rank then we can invert so: $\mathbf{w}^* = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}$

Approach 2: MLE

- Statistical model:

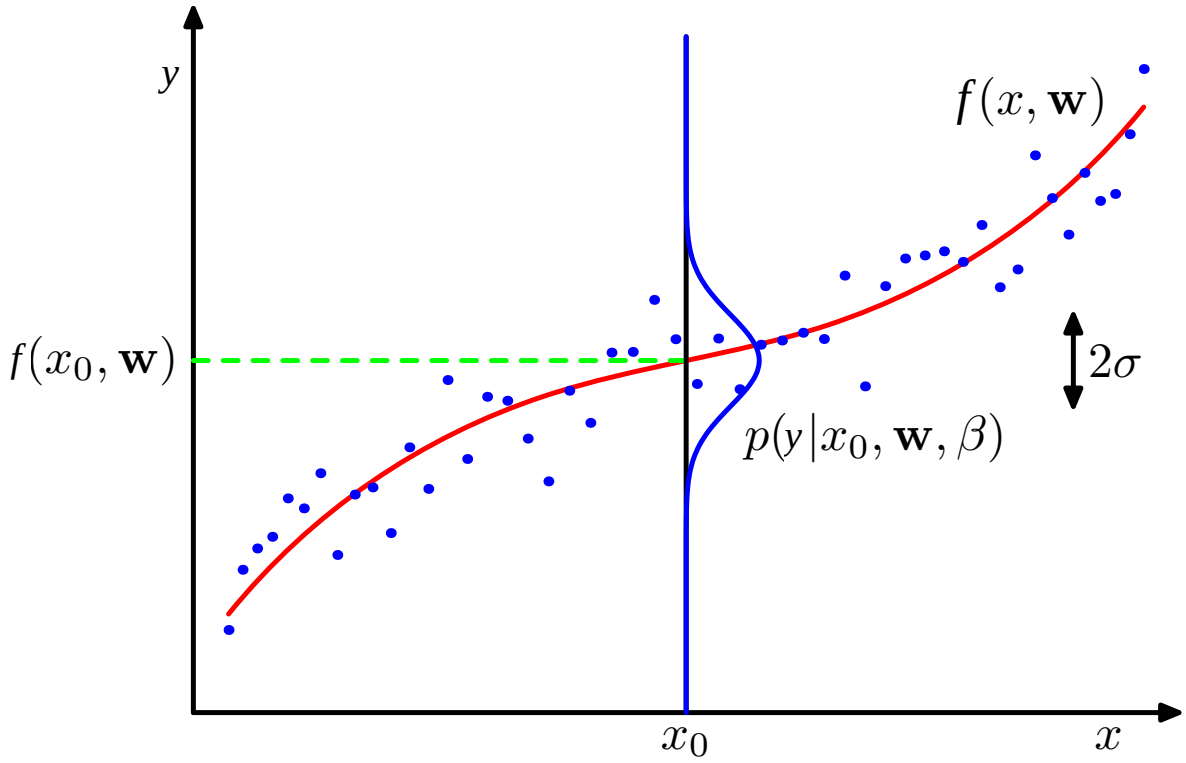
$$y_i = \mathbf{w}^\top \mathbf{x}_i + \epsilon_i \qquad \text{where } \epsilon \sim N(0, \sigma^2)$$

- The distribution of y_i given $\mathbf{x}_i$ is:

$$y_i | \mathbf{x}_i \sim N(\mathbf{w}^\top \mathbf{x}_i, \sigma^2)$$

$$\Rightarrow P(y_i | \mathbf{x}_i, \mathbf{w}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{ -\frac{(\mathbf{w}^\top \mathbf{x}_i - y_i)^2}{2\sigma^2} \right\}$$

Conditional Distribution



A Statistical View

- MLE aims to maximize $P(\text{observed data}|\text{model parameters})$

$$
\begin{aligned}
\mathbf{w} &= \text{argmax}_{\mathbf{w}} \, P(y_1, \mathbf{x}_1, \ldots, y_n, \mathbf{x}_n | \mathbf{w}) \\
&= \text{argmax}_{\mathbf{w}} \sum_{i=1}^{n} \log P(y_i | \mathbf{x}_i, \mathbf{w}) \\
&= \text{argmax}_{\mathbf{w}} \sum_{i=1}^{n} \log \frac{1}{\sqrt{2\pi\sigma^2}} + \log \exp \left\{ -\frac{(\mathbf{w}^\top \mathbf{x}_i - y_i)^2}{2\sigma^2} \right\} \\
&= \text{argmin}_{\mathbf{w}} \frac{1}{n} \sum_{i=1}^{n} (\mathbf{w}^\top \mathbf{x}_i - y_i)^2
\end{aligned}
$$

(Equivalent to minimizing least squares risk)

Structural Risk Minimization (SRM)

- SRM is similar to ERM but takes into account model complexity
- SRM balances fitting to training data and model complexity
- Typical problem:

$$\text{argmin}_\theta \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, \hat{f}(\mathbf{x}_i)) + \lambda R(\theta)$$

- First term is loss on training data
- Second term measures model complexity (i.e., regularizer)

SRM

- SRM recipe:
  - Pick class of predictors $\mathcal{C}$ (linear in this lecture)
  - Pick loss function $\ell(\cdot)$
  - Pick regularizer $R(\cdot)$
  - Minimize structural risk over model/parameters

Ridge Regression

- Predictor class $\mathcal{C}$ = linear functions
- Loss: squared error $\ell(y, \mathbf{w}^\top \mathbf{x}) = (y - \mathbf{w}^\top \mathbf{x})^2$
- Regularizer: $R(\mathbf{w}) = \|\mathbf{w}\|^2 = \mathbf{w}^\top \mathbf{w} = \sum_i (w(i))^2$
- Structural risk minimization:

$$\begin{aligned} \mathbf{w}^* &= \text{argmin}_{\mathbf{w}} \frac{1}{n} \sum_{i=1}^{n} (y_i - \mathbf{w}^\top \mathbf{x}_i)^2 + \lambda \|\mathbf{w}\|^2 \\ &= \text{argmin}_{\mathbf{w}} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 + \lambda \|\mathbf{w}\|^2 \end{aligned}$$

Ridge Regression

- Solve similarly as least squares (minimizer is stationary point)

$$\mathbf{w}^* = (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^\top \mathbf{y}$$

  where $\mathbf{I}$ is $p \times p$ identify matrix
- Solution always unique even if $\mathbf{X}$ is not full rank (e.g., $n < p$)
- Regularizer $\lambda \|\mathbf{w}\|^2$ encourages "shorter" solutions
- λ manages tradeoff between fitting to data and magnitude of $\mathbf{w}$ (model complexity)

SRM

- Why do we care about model complexity (or making $\mathbf{w}$ small)?
- Bounding model complexity can prevent overfitting - model has small training error but large test error
- Setting λ can be tricky
  - Too small and we can overfit
  - Too large and we can underfit
  - Typically λ chosen via cross validation

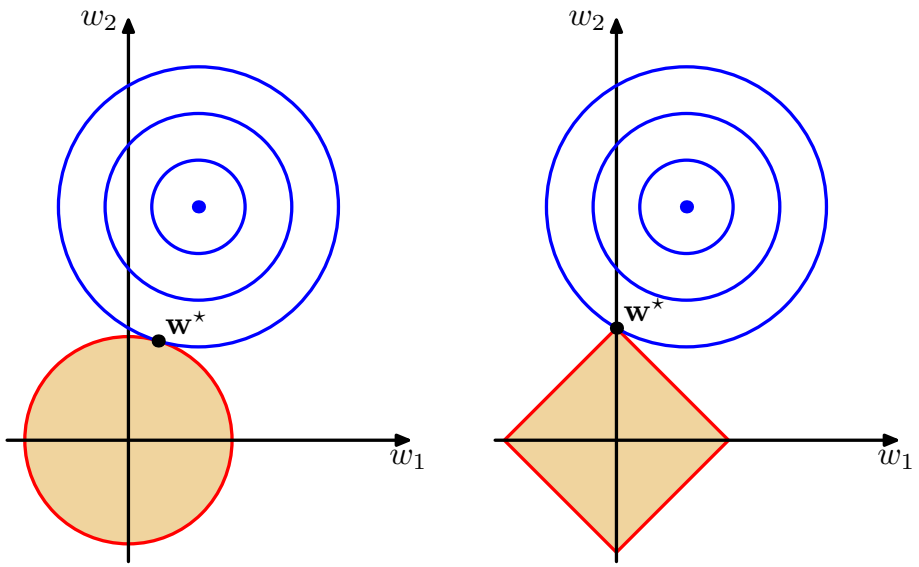
Lasso

- Predictor class $\mathcal{C}$ = linear functions
- Loss: squared error $\ell(y, \mathbf{w}^\top \mathbf{x}) = (y - \mathbf{w}^\top \mathbf{x})^2$
- Regularizer: $R(\mathbf{w}) = \|\mathbf{w}\|_1 = \sum_i |w(i)|$
- Structural risk minimization:

$$\mathbf{w}^* = \text{argmin}_{\mathbf{w}} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 + \lambda \|\mathbf{w}\|_1$$

- Lasso encourages sparse solutions, often used when $n \ll p$
- No closed-form solution

Regularized Least Squares



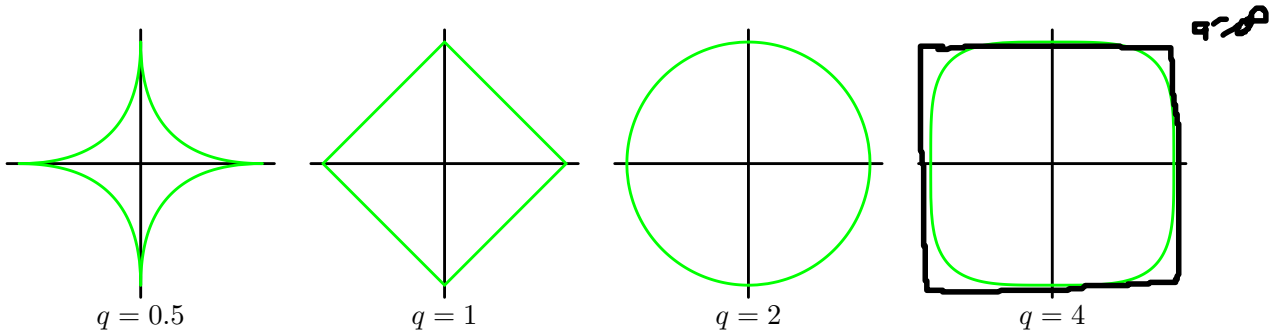
- Regression with L_2 regularization: Ridge Regression

$$\text{argmin}_{\mathbf{w}} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 + \lambda\|\mathbf{w}\|^2$$

- Regression with L_1 regularization: Lasso

$$\text{argmin}_{\mathbf{w}} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 + \lambda\|\mathbf{w}\|_1$$

Regularized Least Squares



General classes of regularizers:

$$\mathbf{w}^* = \text{argmin}_{\mathbf{w}} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 + \lambda \sum_i |w(i)|^q$$

Maximum A Posteriori (MAP) Estimation

- MAP estimation solves $P(\text{model parameter}|\text{observed data})$
- Statistical model:

$$y_i = \mathbf{w}^\top \mathbf{x}_i + \epsilon_i \qquad \text{where } \epsilon \sim N(0, \sigma^2)$$

- The distribution of y_i given $\mathbf{x}_i$ is:

$$y_i|\mathbf{x}_i \sim N(\mathbf{w}^\top \mathbf{x}_i, \sigma^2)$$

$$\Rightarrow P(y_i|\mathbf{x}_i, \mathbf{w}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(\mathbf{w}^\top \mathbf{x}_i - y_i)^2}{2\sigma^2}\right)$$

- A priori distribution of $\mathbf{w}$ is $P(\mathbf{w}) = \frac{1}{\sqrt{2\pi\tau^2}} \exp\left(\frac{-\mathbf{w}^\top \mathbf{w}}{2\tau^2}\right)$

Maximum A Posteriori (MAP) Estimation

$$
\begin{aligned}
\mathbf{w}^* &= \text{argmax}_{\mathbf{w}} \, P(\mathbf{w} | y_1, \mathbf{x}_1, \ldots, y_n, \mathbf{x}_n) \\
&= \text{argmax}_{\mathbf{w}} \, \frac{P(y_1, \mathbf{x}_1, \ldots, y_n, \mathbf{x}_n | \mathbf{w}) P(\mathbf{w})}{P(y_1, \mathbf{x}_1, \ldots, y_n, \mathbf{x}_n)} \\
&= \text{argmax}_{\mathbf{w}} \, P(y_1, \mathbf{x}_1, \ldots, y_n, \mathbf{x}_n | \mathbf{w}) P(\mathbf{w}) \\
&= \text{argmax}_{\mathbf{w}} \left[\prod_{i=1}^{n} P(y_i, \mathbf{x}_i | \mathbf{w}) \right] P(\mathbf{w}) \\
&= \text{argmax}_{\mathbf{w}} \left[\prod_{i=1}^{n} P(y_i | \mathbf{x}_i, \mathbf{w}) P(\mathbf{x}_i | \mathbf{w}) \right] P(\mathbf{w}) \\
&= \text{argmax}_{\mathbf{w}} \left[\prod_{i=1}^{n} P(y_i | \mathbf{x}_i, \mathbf{w}) P(\mathbf{x}_i) \right] P(\mathbf{w})
\end{aligned}
$$

Maximum A Posteriori (MAP) Estimation

$$= \text{argmax}_{\mathbf{w}} \left[\prod_{i=1}^{n} P(y_i|\mathbf{x}_i, \mathbf{w}) P(\mathbf{x}_i) \right] P(\mathbf{w})$$

$$= \text{argmax}_{\mathbf{w}} \left[\prod_{i=1}^{n} P(y_i|\mathbf{x}_i, \mathbf{w}) \right] P(\mathbf{w})$$

$$= \text{argmax}_{\mathbf{w}} \sum_{i=1}^{n} \log P(y_i|\mathbf{x}_i, \mathbf{w}) + \log P(\mathbf{w})$$

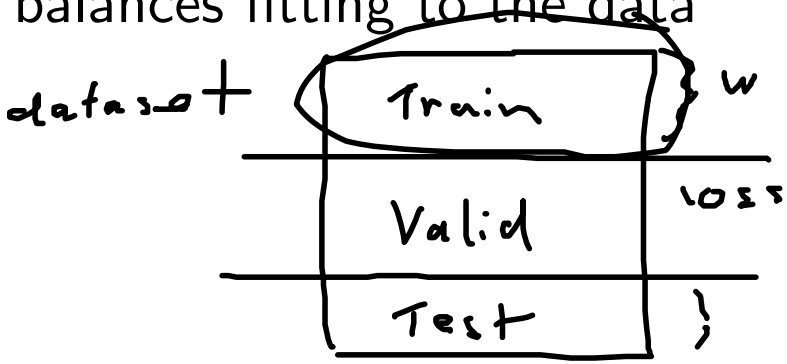
$$= \text{argmin}_{\mathbf{w}} \frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - \mathbf{w}^\top \mathbf{x}_i)^2 + \frac{1}{2\tau^2} \mathbf{w}^\top \mathbf{w}$$

$$= \text{argmin}_{\mathbf{w}} \frac{1}{n} \sum_{i=1}^{n} (y_i - \mathbf{w}^\top \mathbf{x}_i)^2 + \lambda \|\mathbf{w}\|^2$$

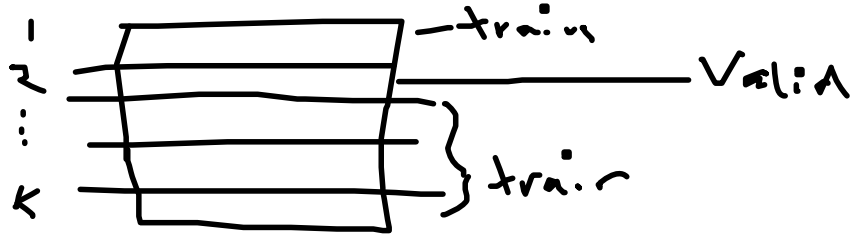
(Equivalent to ridge regression)

Tuning Hyperparameters

- The regularization parameter λ balances fitting to the data and model complexity
- λ is a hyperparameter
- How do we choose the best λ?
- Split data into 3 sets:
  - Training set: learn predictor $\hat{f}$ by fitting this dataset
  - Validation set: tune hyperparameters; use loss on this set to find the best hyperparameter
  - Test set: assess **risk** of model: $\mathcal{R}(f) = E_{(X,Y)\sim P}[\ell(Y, f(X))]$
- We want to predict well on future data; goal is to find $\hat{f}$ that minimizes risk (instead of empirical risk on training set)



Cross Validation



- What if we do not have a validation set? Create one!
- k-fold cross validation: split training set into k folds of (roughly) equal size $F_1, \ldots, F_k$
  - For $j = 1, \ldots, k$
    - Train on union of folds $F_{-j} = \cup_{j' \neq j} F_{j'}$ and validate on fold F_j
    - For each tuning parameter $\theta \in \{\theta_1, \ldots, \theta_m\}$ train on F_{-j} to obtain predictor $\hat{f}_\theta^{-j}$ and record loss on validation set $\hat{\mathcal{R}}(\hat{f}_\theta^{-j})$
  - For each θ, compute average loss over all folds
  $\hat{\mathcal{R}}_{CV}(\theta) = \frac{1}{k} \sum_{j=1}^{k} \hat{\mathcal{R}}(\hat{f}_\theta^{-j})$
  - Choose parameter θ that minimizes $\hat{\mathcal{R}}_{CV}(\theta)$

Feature Transformation/Representation

- Enrich linear regression by transforming features $\mathbf{x}$ into $\phi(\mathbf{x})$
- Predict with transformed features: $\hat{f}(\mathbf{x}) = \mathbf{w}^\top \phi(\mathbf{x})$
- Examples:

  $x \in \mathbb{R}, \phi(x) = \ln(1 + x)$

  $\mathbf{x} \in \mathbb{R}^p, \phi(\mathbf{x}) = (1, x(1), \ldots, x(p), x(1)^2, \ldots, x(p)^2, x(1)x(2), \ldots, x(p-1)x(p))$

  $x \in \mathbb{R}, \phi(x) = (1, \sin(x), \cos(x), \sin(2x), \cos(2x), \ldots)$

- Choice of representations: fixed, implicit, learned
- We will often use $\mathbf{x}$ (instead of $\phi(\mathbf{x})$) to denote the feature representation

Loss Decomposition

- Total expected loss

$$E_{(\mathbf{x},y)}[\ell(f(\mathbf{x}),y)] = \int\int \ell(f(\mathbf{x}),y)p(\mathbf{x},y)d\mathbf{x}dy$$
$$= \int\int (y - f(\mathbf{x}))^2 p(\mathbf{x},y)d\mathbf{x}dy$$

- Solution $f(\mathbf{x}) = \int yp(y|\mathbf{x}) = E[y|\mathbf{x}]$
- Loss decomposition

$$E_{(\mathbf{x},y)}[\ell(f(\mathbf{x}),y)] = \int (f(\mathbf{x}) - E[y|\mathbf{x}])^2 p(\mathbf{x})d\mathbf{x} + \underbrace{\int\int (E[y|\mathbf{x}] - y)^2 p(\mathbf{x},y)d\mathbf{x}dy}_{\text{distribution variance}}$$

Bias Variance Decomposition

- Let $h(\mathbf{x}) = E[y|\mathbf{x}]$, the best we can do
- For a particular dataset D, we learn $f(\mathbf{x}) = f(\mathbf{x}; D)$
- Taking expectation over all such datasets

$$E_D[(f(\mathbf{x}, D) - h(\mathbf{x}))^2] = \underbrace{(E_D[f(\mathbf{x}; D)] - h(x))^2}_{(\text{bias})^2} + \underbrace{E_D[(f(x; D) - E_D[f(x; D)])^2]}_{\text{variance}}$$

- The overall loss decomposition

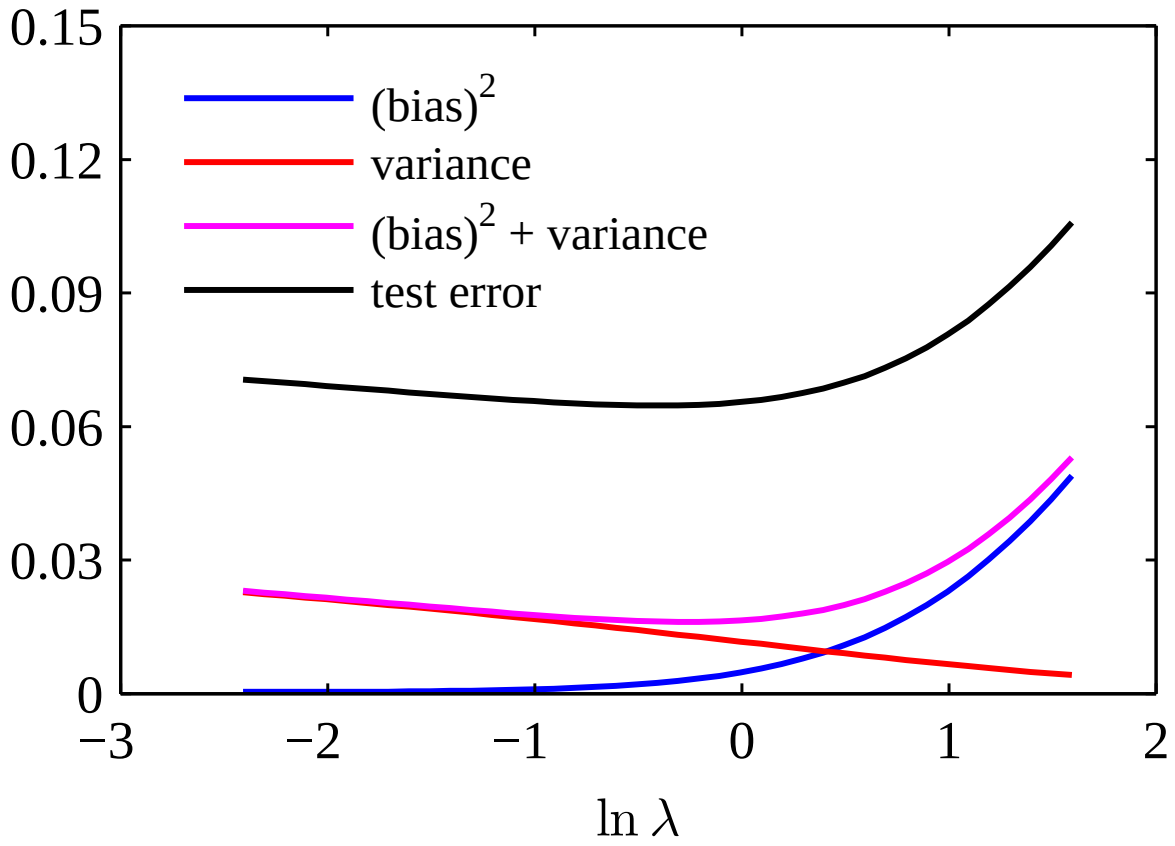
$$E_(\mathbf{x}, y)[\ell(f(\mathbf{x}), y)] = (\text{bias})^2 + \text{variance} + \text{distribution variance}$$

$$(\text{bias})^2 = \int (E_D[f(\mathbf{x}; D)] - h(x))^2 p(\mathbf{x}) d\mathbf{x}$$

$$\text{variance} = \int E_D[(f(\mathbf{x}; D) - E_D[f(\mathbf{x}; D)])^2] p(\mathbf{x}) d\mathbf{x}$$

$$\text{distribution variance} = \int \int (h(\mathbf{x}) - y)^2 p(\mathbf{x}, y) d\mathbf{x} dy$$

Bias Variance Tradeoff



General classes of regularizers:

$$\|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 + \lambda R(\mathbf{w})$$