

Bias-Variance Tradeoff

Machine Learning – CSE446
Kevin Jamieson
University of Washington

April 15, 2019

Optimal Prediction

Goal: Predict $Y \in \mathbb{R}^d$ given $X \in \mathbb{R}^d$ if $(X, Y) \sim P_{XY}$

Find function η that minimizes

n(X) takes in an X value and predicts a value of Y. Has no idea what Y is, only knows the value of X.

$$\mathbb{E}_{XY}[(Y - \eta(X))^2] = \mathbb{E}_X \left[\mathbb{E}_{Y|X}[(Y - \eta(x))^2 | X = x] \right]$$

(Hint: for any x , $\eta(x) = c_x$ where c_x minimizes $\mathbb{E}_{Y|X}[(Y - c_x)^2 | X = x]$)

$$0 = \frac{d}{dc_x} \mathbb{E}_{Y|X}[(Y - c_x)^2 | X = x]$$

NOTE: n(X) minimizes the error over all possible values of Y!
does NOT give the most likely choice of Y

$$= \mathbb{E}_{Y|X} \left[\frac{d}{dc_x} (Y - c_x)^2 | X = x \right]$$

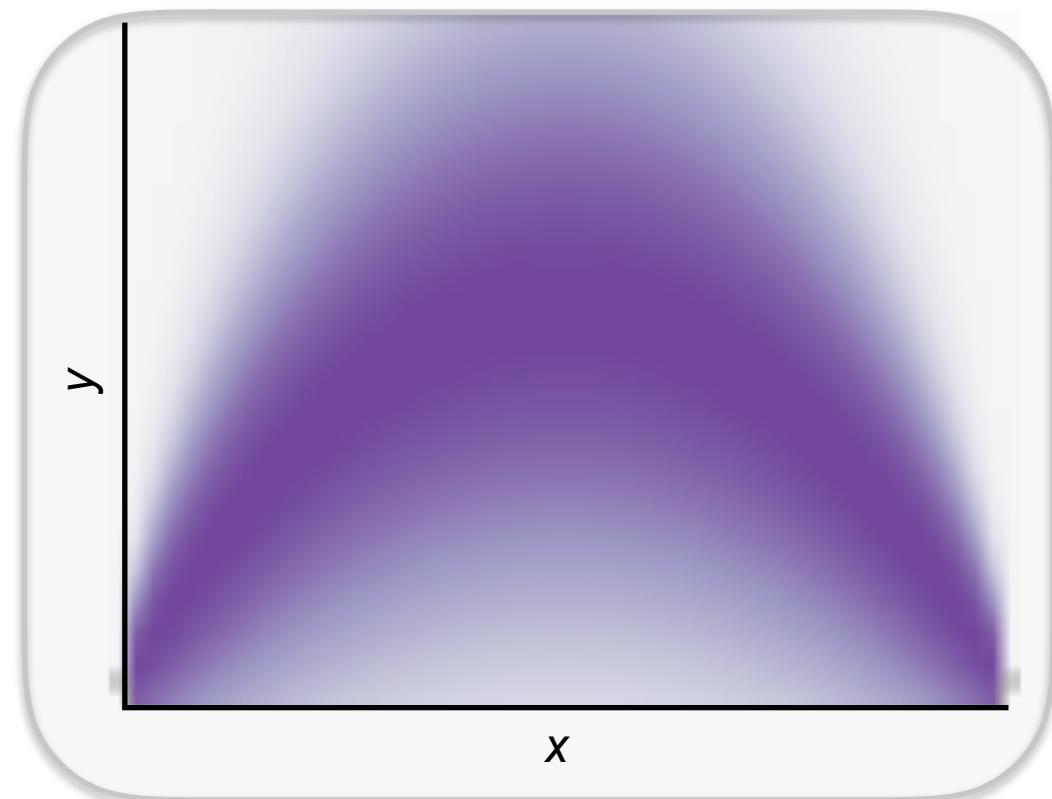
$$= \mathbb{E}_{Y|X}[-2(Y - c_x) | X = x] = -2\mathbb{E}_{Y|X}[Y | X = x] + 2c_x$$

Squared Error Optimal Predictor: $\eta(x) = \mathbb{E}_{Y|X}[Y | X = x]$

optimal estimation that will minimize the squared error.

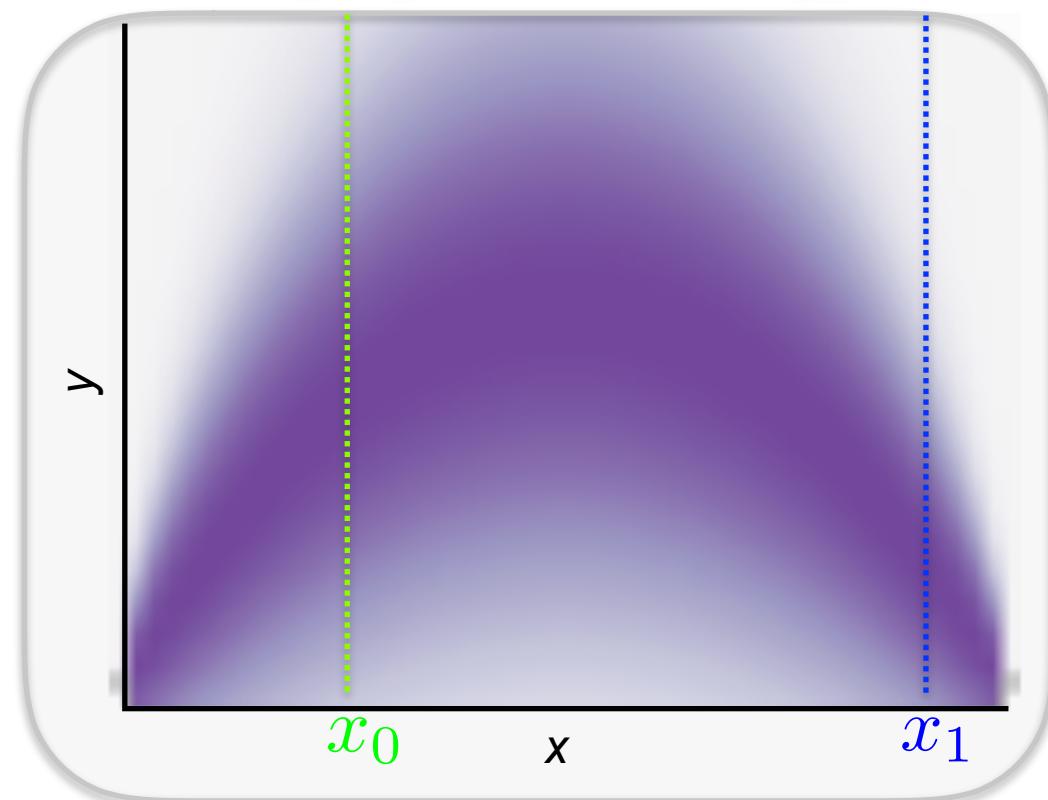
Statistical Machine Learning

$$P_{XY}(X = x, Y = y)$$



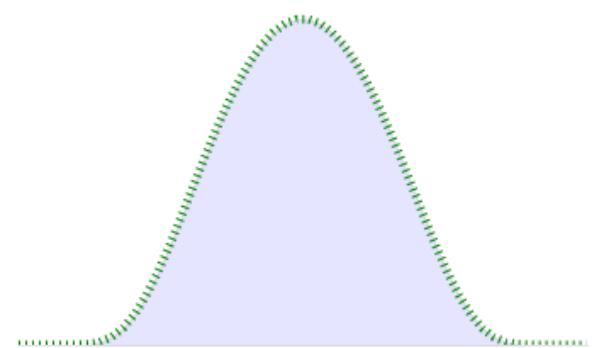
Statistical Machine Learning

$$P_{XY}(X = x, Y = y)$$

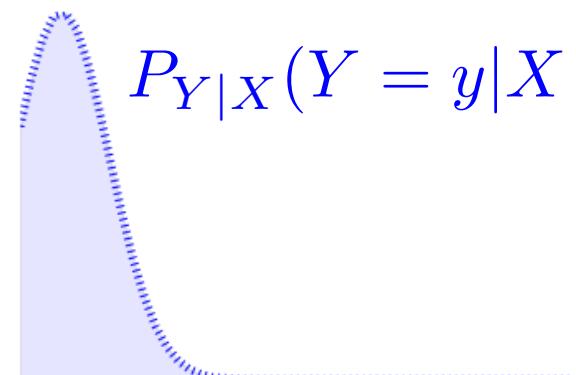


conditional distribution of Y given $X = x_0$

$$P_{Y|X}(Y = y|X = x_0)$$

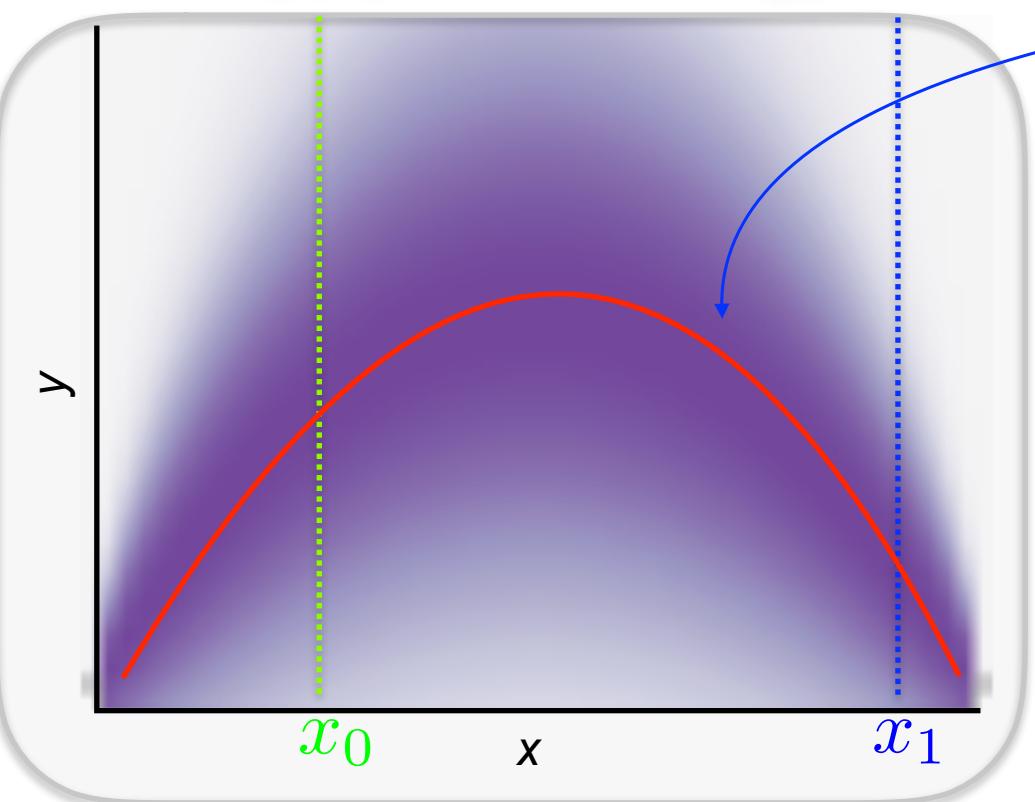


$$P_{Y|X}(Y = y|X = x_1)$$



Statistical Machine Learning

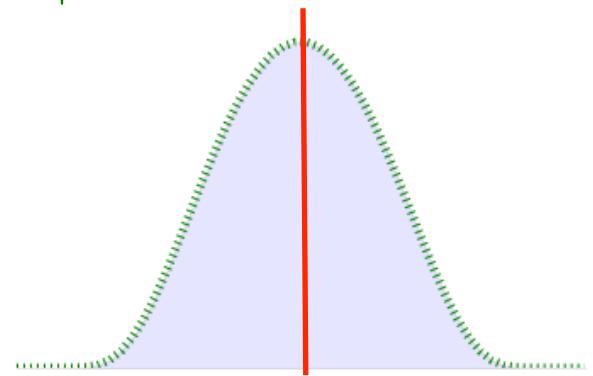
$$P_{XY}(X = x, Y = y)$$



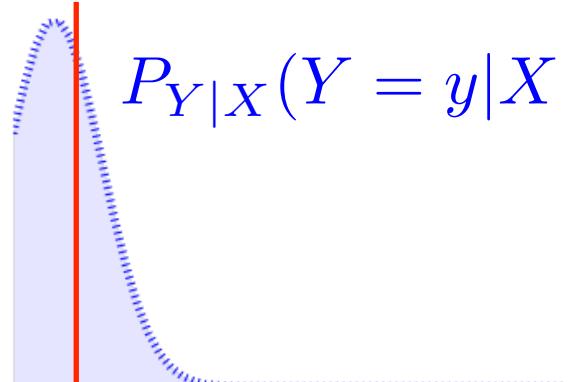
Ideally, we want to find:

$$\eta(x) = \mathbb{E}_{Y|X}[Y|X = x]$$

$$P_{Y|X}(Y = y|X = x_0)$$

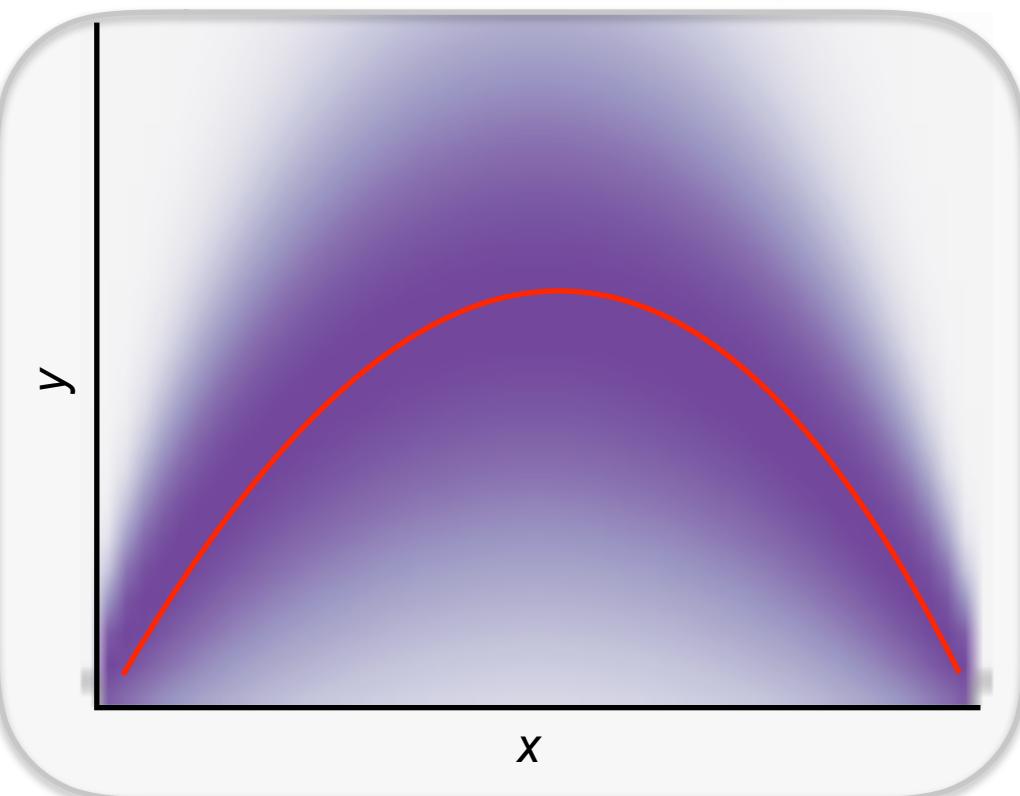


$$P_{Y|X}(Y = y|X = x_1)$$



Statistical Machine Learning

$$P_{XY}(X = x, Y = y)$$

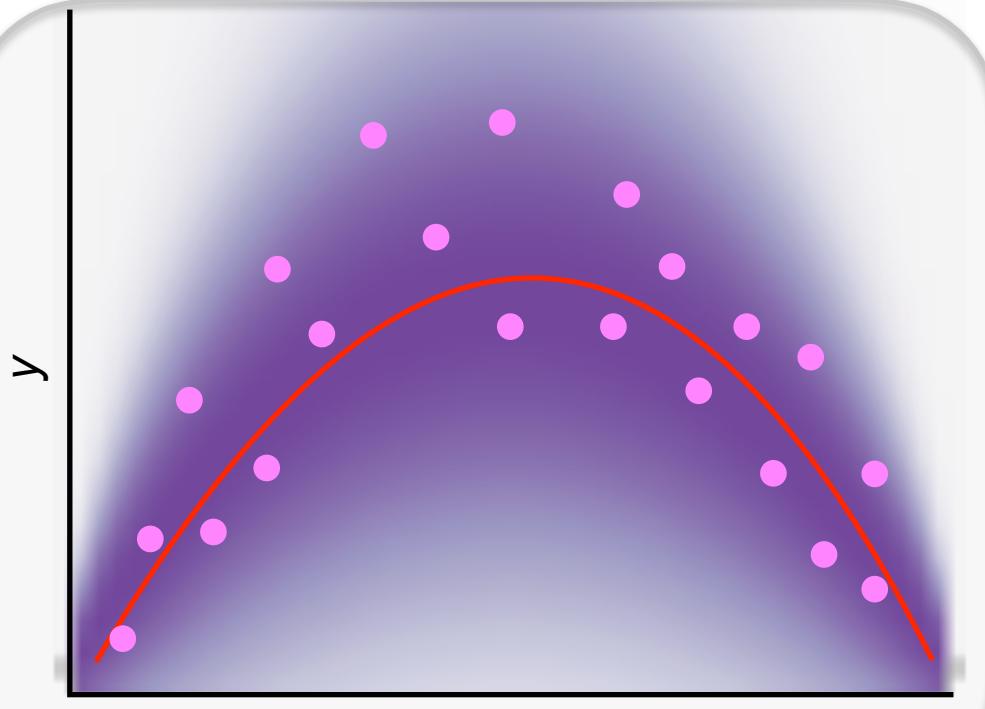


Ideally, we want to find:

$$\eta(x) = \mathbb{E}_{Y|X}[Y|X = x]$$

Statistical Machine Learning

$$P_{XY}(X = x, Y = y)$$



Ideally, we want to find:

$$\eta(x) = \mathbb{E}_{Y|X}[Y|X = x]$$

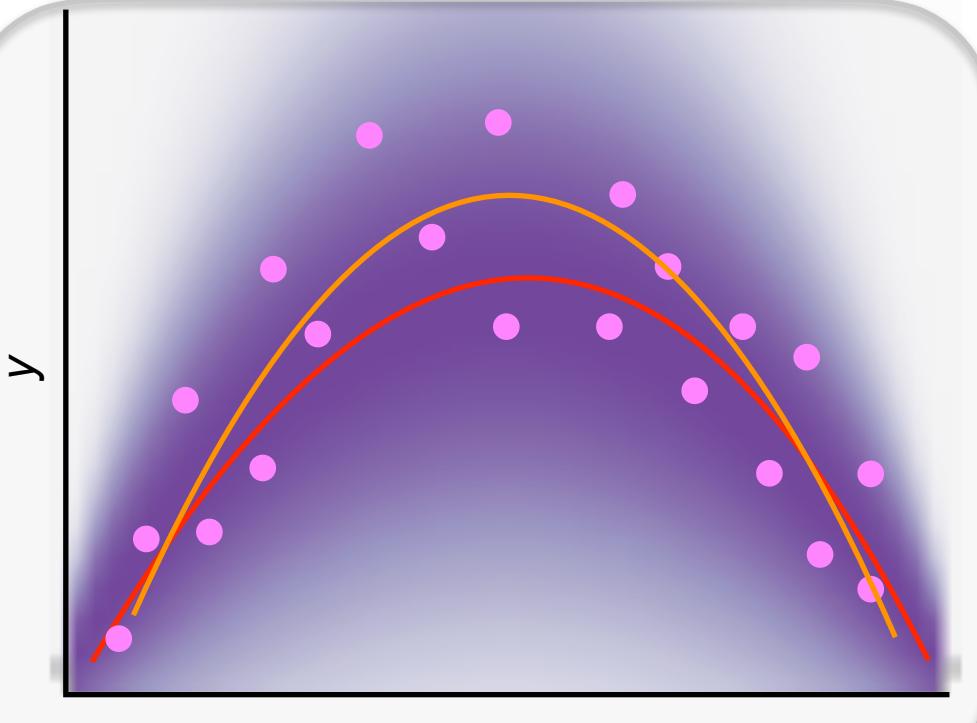
But we only have samples:

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n \stackrel{iid}{\sim} P_{XY}$$

do NOT know the true distributions. Only have a series of samples from P_{xy}

Statistical Machine Learning

$$P_{XY}(X = x, Y = y)$$



Ideally, we want to find:

$$\eta(x) = \mathbb{E}_{Y|X}[Y|X = x]$$

But we only have samples:

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n \stackrel{iid}{\sim} P_{XY}$$

and are restricted to a function class (e.g., linear)
so we compute:

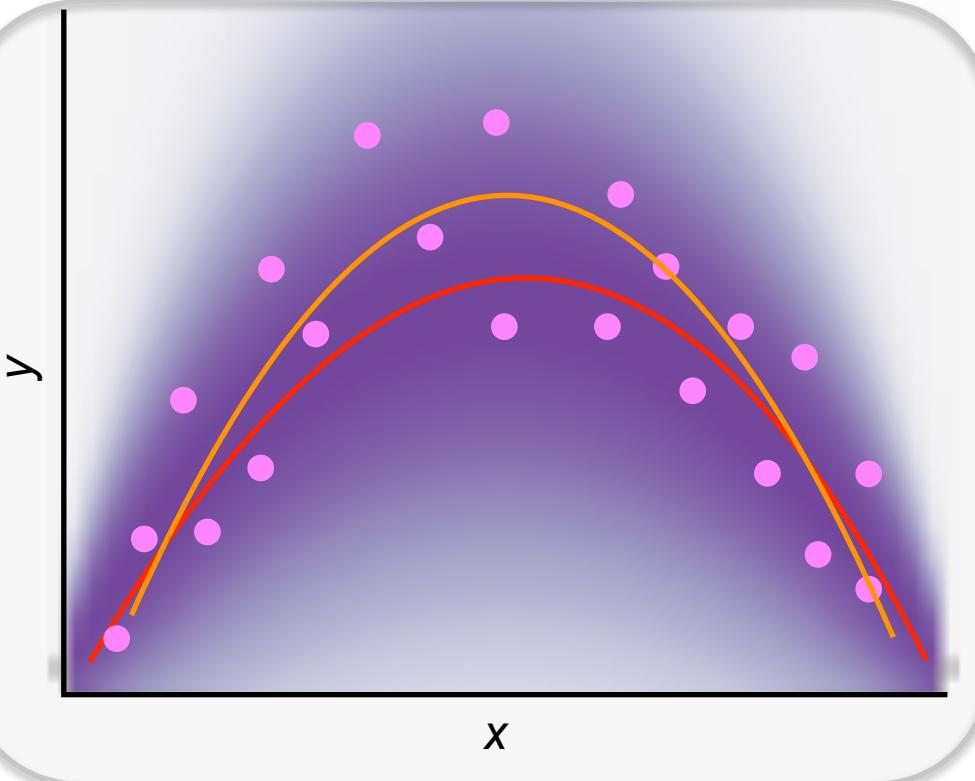
$$\hat{f}_{\mathcal{D}} = \arg \min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^n (y_i - f(x_i))^2$$

minimize the squared error
in the OBSERVED data

this is an approximation of
 $n(X)$.

Statistical Machine Learning

$$P_{XY}(X = x, Y = y)$$



Ideally, we want to find:

$$\eta(x) = \mathbb{E}_{Y|X}[Y|X = x]$$

But we only have samples:

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n \stackrel{iid}{\sim} P_{XY}$$

and are restricted to a function class (e.g., linear)
so we compute:

$$\hat{f}_{\mathcal{D}} = \arg \min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^n (y_i - f(x_i))^2$$

What about future predictions? $\mathbb{E}_{Y|X, \mathcal{D}}[(Y - \hat{f}_{\mathcal{D}}(x))^2 | X = x]$

how does this model that the data trained perform on future predictions for Y?

Bias-Variance Tradeoff

$$\eta(x) = \mathbb{E}_{Y|X}[Y|X = x]$$

$$\hat{f}_{\mathcal{D}} = \arg \min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^n (y_i - f(x_i))^2$$

$$\mathbb{E}_{Y|X}[\mathbb{E}_{\mathcal{D}}[(Y - \hat{f}_{\mathcal{D}}(x))^2] | X = x] = \mathbb{E}_{Y|X}[\mathbb{E}_{\mathcal{D}}[(Y - \eta(x) + \eta(x) - \hat{f}_{\mathcal{D}}(x))^2] | X = x]$$



data is a random variable

Bias-Variance Tradeoff

$$\eta(x) = \mathbb{E}_{Y|X}[Y|X = x]$$

$$\hat{f}_{\mathcal{D}} = \arg \min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^n (y_i - f(x_i))^2$$

$$\begin{aligned}\mathbb{E}_{Y|X}[\mathbb{E}_{\mathcal{D}}[(Y - \hat{f}_{\mathcal{D}}(x))^2] | X = x] &= \mathbb{E}_{Y|X}[\mathbb{E}_{\mathcal{D}}[(Y - \eta(x) + \eta(x) - \hat{f}_{\mathcal{D}}(x))^2] | X = x] \\ &= \mathbb{E}_{Y|X} \left[\mathbb{E}_{\mathcal{D}}[(Y - \eta(x))^2 + \underline{2(Y - \eta(x))(\eta(x) - \hat{f}_{\mathcal{D}}(x))} \right. \\ &\quad \left. + (\eta(x) - \hat{f}_{\mathcal{D}}(x))^2] | X = x \right] \quad \text{the expectation of this is zero when } X \text{ is given by the definition of } n(X) \\ &= \mathbb{E}_{Y|X}[(Y - \eta(x))^2 | X = x] + \mathbb{E}_{\mathcal{D}}[(\eta(x) - \hat{f}_{\mathcal{D}}(x))^2]\end{aligned}$$

irreducible error

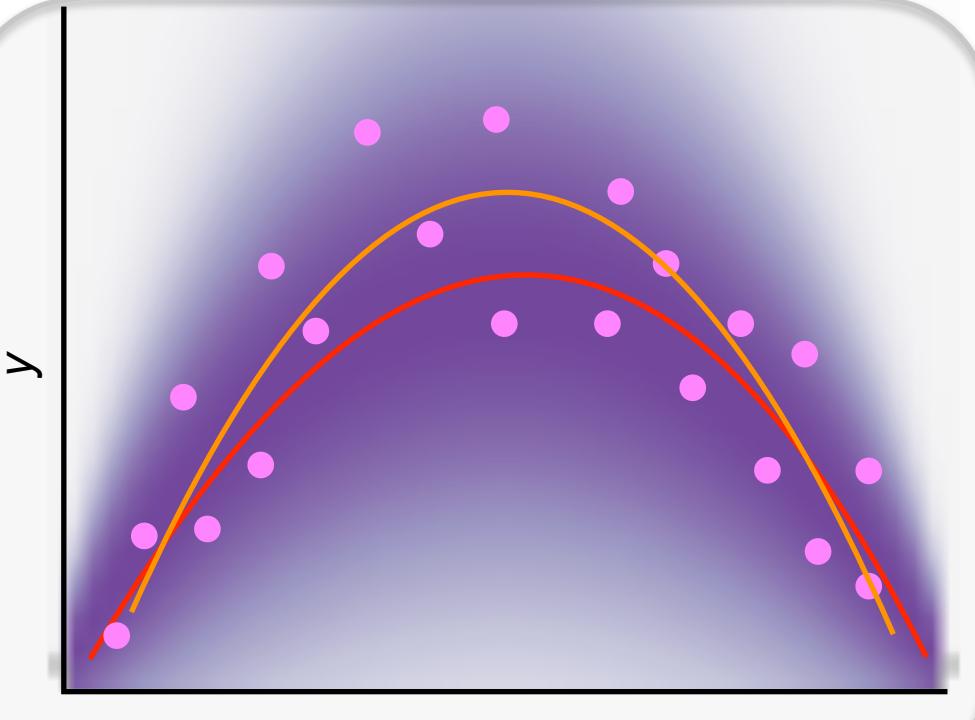
Caused by stochastic
label noise

learning error

Caused by either using too “simple”
of a model or not enough
data to learn the model accurately

Statistical Machine Learning

$$P_{XY}(X = x, Y = y)$$



Ideally, we want to find:

$$\eta(x) = \mathbb{E}_{Y|X}[Y|X = x]$$

But we only have samples:

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n \stackrel{iid}{\sim} P_{XY}$$

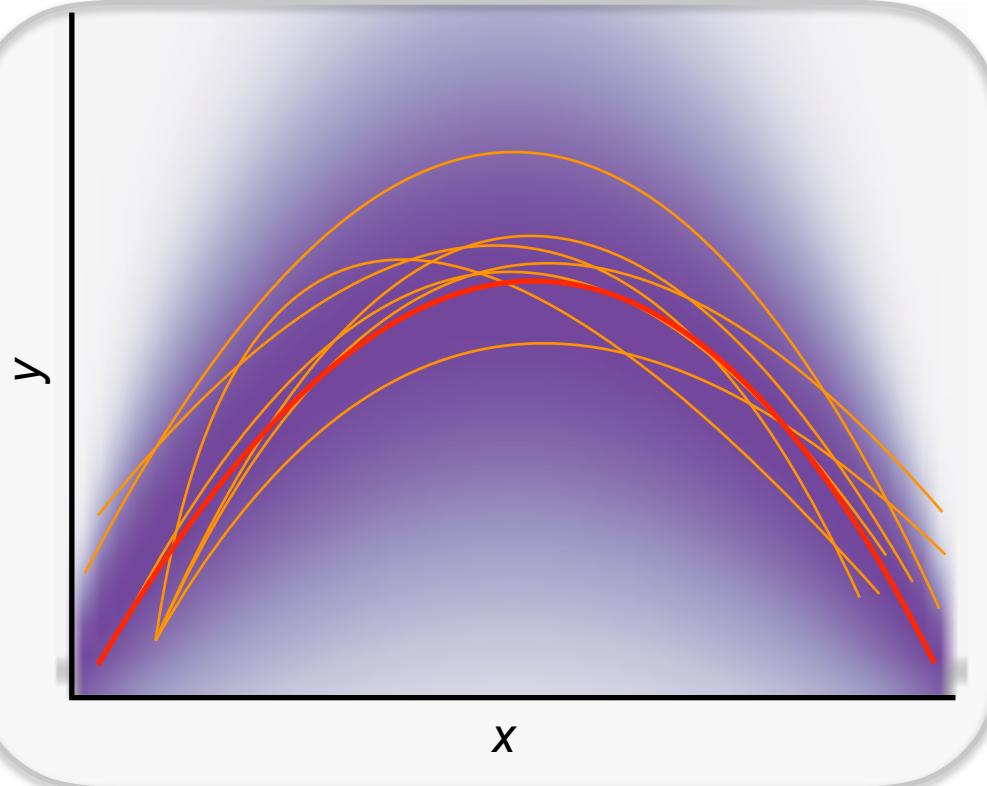
and are restricted to a function class (e.g., linear)
so we compute:

$$\hat{f}_{\mathcal{D}} = \arg \min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^n (y_i - f(x_i))^2$$

$$\mathbb{E}_{Y|X,\mathcal{D}}[(Y - \hat{f}_{\mathcal{D}}(x))^2 | X = x] = \mathbb{E}_{Y|X}[(Y - \eta(x))^2 | X = x] + \mathbb{E}_{\mathcal{D}}[(\eta(x) - \hat{f}_{\mathcal{D}}(x))^2]$$

Statistical Machine Learning

$$P_{XY}(X = x, Y = y)$$



Ideally, we want to find:

$$\eta(x) = \mathbb{E}_{Y|X}[Y|X = x]$$

But we only have samples:

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n \stackrel{iid}{\sim} P_{XY}$$

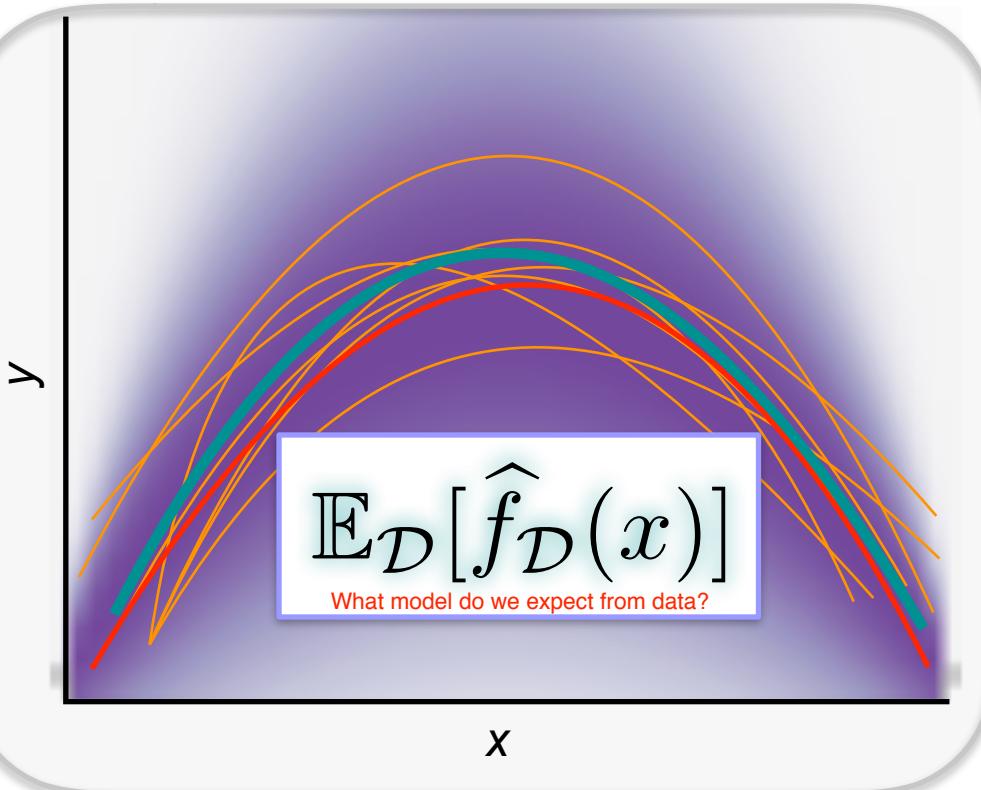
and are restricted to a function class (e.g., linear)
so we compute:

$$\hat{f}_{\mathcal{D}} = \arg \min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^n (y_i - f(x_i))^2$$

Each draw $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ results in different \hat{f}

Statistical Machine Learning

$$P_{XY}(X = x, Y = y)$$



Ideally, we want to find:

$$\eta(x) = \mathbb{E}_{Y|X}[Y|X = x]$$

But we only have samples:

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n \stackrel{iid}{\sim} P_{XY}$$

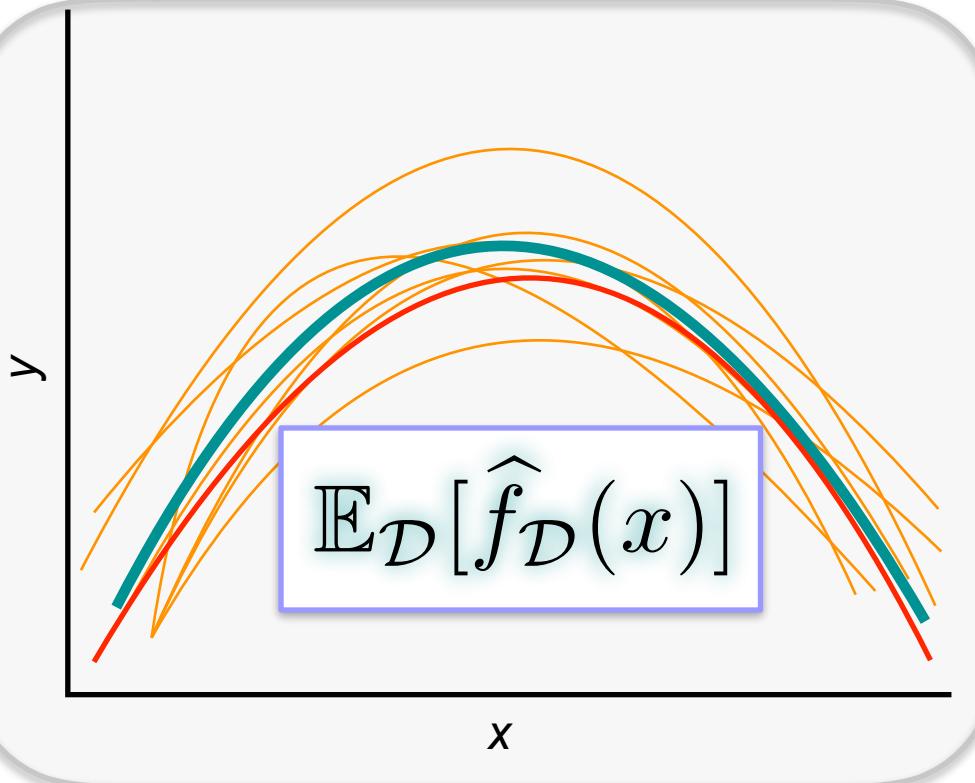
and are restricted to a function class (e.g., linear) so we compute:

$$\hat{f}_{\mathcal{D}} = \arg \min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^n (y_i - f(x_i))^2$$

Each draw $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ results in different \hat{f}

Statistical Machine Learning

$$P_{XY}(X = x, Y = y)$$



Ideally, we want to find:

$$\eta(x) = \mathbb{E}_{Y|X}[Y|X = x]$$

But we only have samples:

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n \stackrel{iid}{\sim} P_{XY}$$

and are restricted to a function class (e.g., linear)
so we compute:

$$\hat{f}_{\mathcal{D}} = \arg \min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^n (y_i - f(x_i))^2$$

Each draw $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ results in different \hat{f}

Bias-Variance Tradeoff

$$\eta(x) = \mathbb{E}_{Y|X}[Y|X = x] \quad \mathbb{E}_{\mathcal{D}}[\hat{f}_{\mathcal{D}}(x)] \quad \hat{f}_{\mathcal{D}} = \arg \min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^n (y_i - f(x_i))^2$$

$$\mathbb{E}_{\mathcal{D}}[(\eta(x) - \hat{f}_{\mathcal{D}}(x))^2] = \mathbb{E}_{\mathcal{D}}[(\eta(x) - \mathbb{E}_{\mathcal{D}}[\hat{f}_{\mathcal{D}}(x)]) + \mathbb{E}_{\mathcal{D}}[\hat{f}_{\mathcal{D}}(x)] - \hat{f}_{\mathcal{D}}(x))^2]$$

Bias-Variance Tradeoff

$$\eta(x) = \mathbb{E}_{Y|X}[Y|X = x] \quad \mathbb{E}_{\mathcal{D}}[\hat{f}_{\mathcal{D}}(x)] \quad \hat{f}_{\mathcal{D}} = \arg \min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^n (y_i - f(x_i))^2$$

$$\begin{aligned} \mathbb{E}_{\mathcal{D}}[(\eta(x) - \hat{f}_{\mathcal{D}}(x))^2] &= \mathbb{E}_{\mathcal{D}}[(\eta(x) - \mathbb{E}_{\mathcal{D}}[\hat{f}_{\mathcal{D}}(x)]) + \mathbb{E}_{\mathcal{D}}[\hat{f}_{\mathcal{D}}(x)] - \hat{f}_{\mathcal{D}}(x))^2] \\ &= \mathbb{E}_{\mathcal{D}}[(\eta(x) - \mathbb{E}_{\mathcal{D}}[\hat{f}_{\mathcal{D}}(x)])^2 + 2(\eta(x) - \mathbb{E}_{\mathcal{D}}[\hat{f}_{\mathcal{D}}(x)])(\mathbb{E}_{\mathcal{D}}[\hat{f}_{\mathcal{D}}(x)] - \hat{f}_{\mathcal{D}}(x)) \\ &\quad + (\mathbb{E}_{\mathcal{D}}[\hat{f}_{\mathcal{D}}(x)] - \hat{f}_{\mathcal{D}}(x))^2] \\ &= \underbrace{(\eta(x) - \mathbb{E}_{\mathcal{D}}[\hat{f}_{\mathcal{D}}(x)])^2}_{\text{biased squared}} + \underbrace{\mathbb{E}_{\mathcal{D}}[(\mathbb{E}_{\mathcal{D}}[\hat{f}_{\mathcal{D}}(x)] - \hat{f}_{\mathcal{D}}(x))^2]}_{\text{variance}} \end{aligned}$$

goes to zero

Bias-Variance Tradeoff

Our measurements Y come from a RANDOM distribution.
We can predict the overall trends in this distribution, but
never fully capture this randomness.

$$\mathbb{E}_{Y|X}[\mathbb{E}_{\mathcal{D}}[(Y - \hat{f}_{\mathcal{D}}(x))^2] | X = x] = \underline{\mathbb{E}_{Y|X}[(Y - \eta(x))^2 | X = x]}$$

irreducible error

$$+ \underline{(\eta(x) - \mathbb{E}_{\mathcal{D}}[\hat{f}_{\mathcal{D}}(x)])^2} + \underline{\mathbb{E}_{\mathcal{D}}[(\mathbb{E}_{\mathcal{D}}[\hat{f}_{\mathcal{D}}(x)] - \hat{f}_{\mathcal{D}}(x))^2]}$$

biased squared **variance**

At the given x value, how well is our model expected to capture the true distribution (i.e. $E[Y | X = x]$)

How much will our model changed based on the data it sees. How variable is it with respect to the training dataset?

Bias-Variance Tradeoff

$$\mathbb{E}_{Y|X}[\mathbb{E}_{\mathcal{D}}[(Y - \hat{f}_{\mathcal{D}}(x))^2] | X = x] = \underline{\mathbb{E}_{Y|X}[(Y - \eta(x))^2 | X = x]}$$

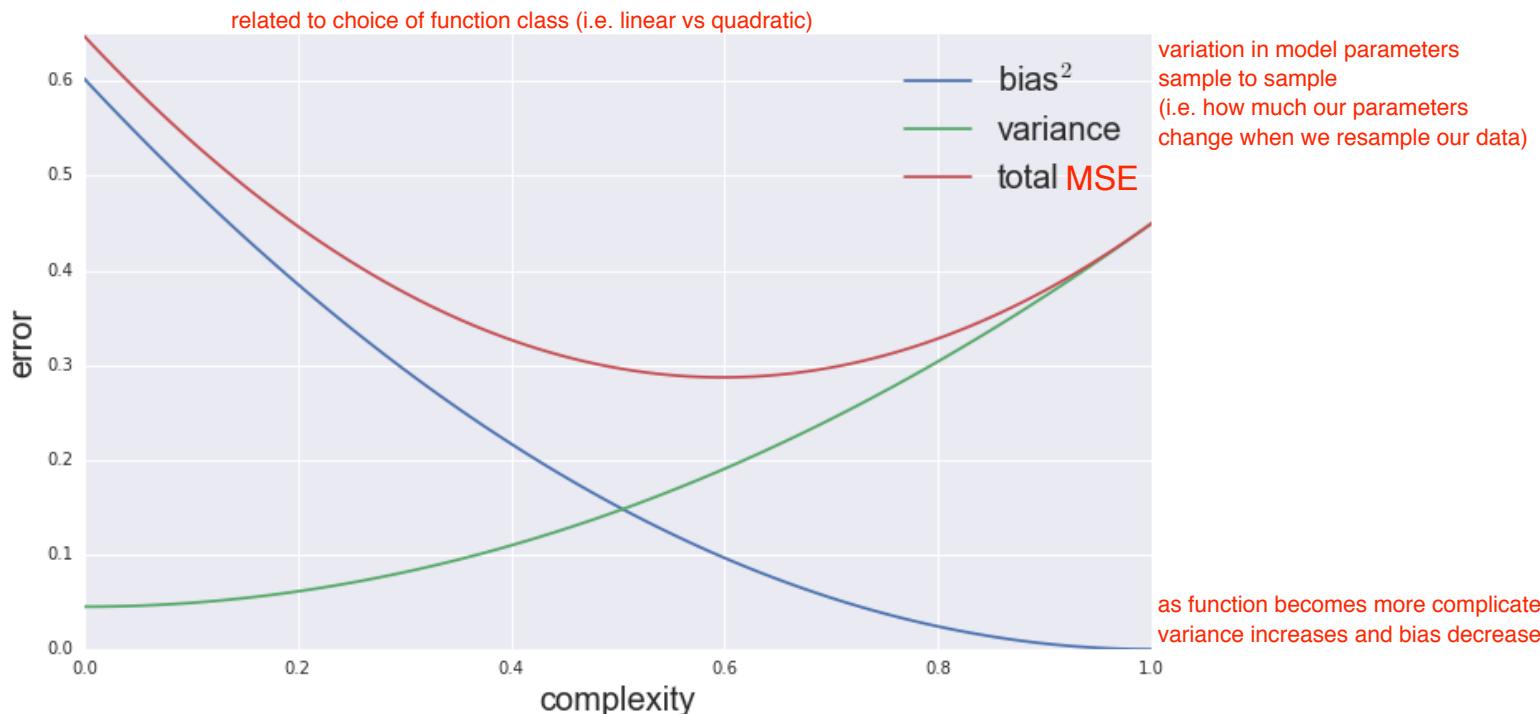
irreducible error

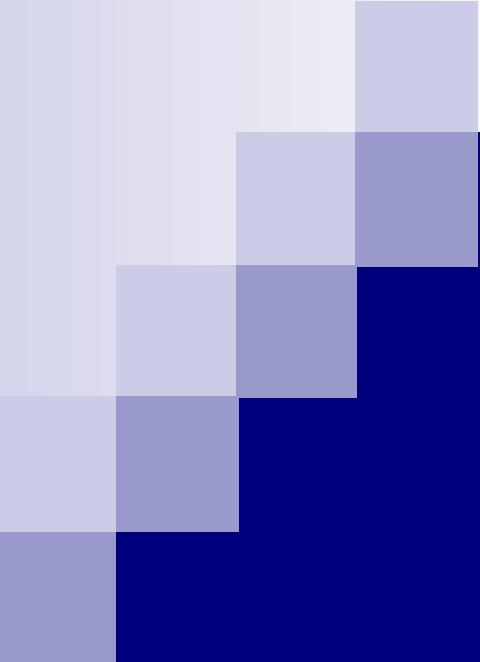
$$+ \underline{(\eta(x) - \mathbb{E}_{\mathcal{D}}[\hat{f}_{\mathcal{D}}(x)])^2} + \underline{\mathbb{E}_{\mathcal{D}}[(\mathbb{E}_{\mathcal{D}}[\hat{f}_{\mathcal{D}}(x)] - \hat{f}_{\mathcal{D}}(x))^2]}$$

biased squared

variance

How well our choice of function class can capture the true function $n(X)$. i.e. if $n(X)$ is quadratic than no way a linear function could capture all its dynamics so I will have a nonzero bias. But if I choose a quadratic function class then I could possibly get the true function $n(X)$





Regularization

Machine Learning – CSE546
Kevin Jamieson
University of Washington

April 15, 2019

Regularization in Linear Regression

find w that minimizes the error $\|Xw - y\|^2$

$$\begin{aligned} \text{Recall Least Squares: } \hat{w}_{LS} &= \arg \min_w \sum_{i=1}^n (y_i - x_i^T w)^2 \\ &= \arg \min_w (\mathbf{y} - \mathbf{X}w)^T (\mathbf{y} - \mathbf{X}w) \\ \text{when } (\mathbf{X}^T \mathbf{X})^{-1} \text{ exists....} &= (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y} \end{aligned}$$

this inverse does not
always exist.

Regularization in Linear Regression

Recall Least Squares: $\hat{w}_{LS} = \arg \min_w \sum_{i=1}^n (y_i - x_i^T w)^2$

In general: $= \arg \min_w (\mathbf{y} - \mathbf{X}w)^T (\mathbf{y} - \mathbf{X}w)$

$$= \arg \min_w w^T (\mathbf{X}^T \mathbf{X}) w - 2y^T \mathbf{X} w$$

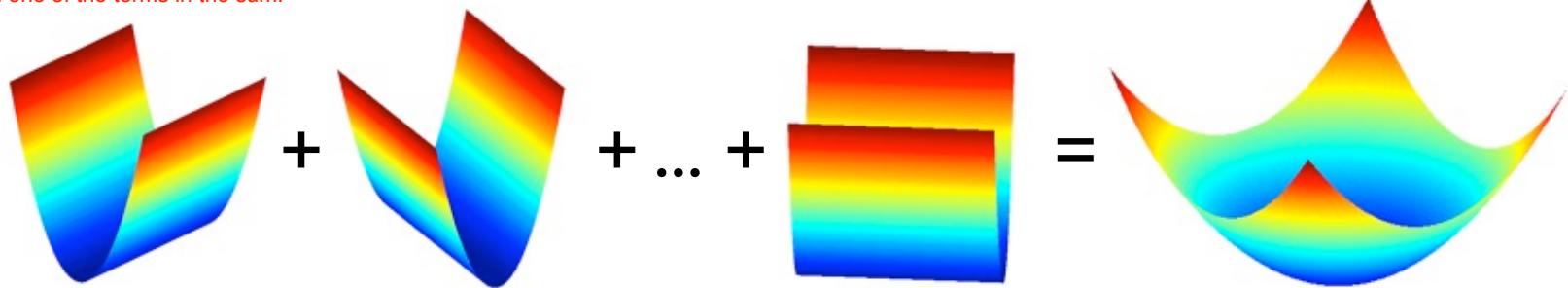
Regularization in Linear Regression

Recall Least Squares: $\hat{w}_{LS} = \arg \min_w \sum_{i=1}^n (y_i - x_i^T w)^2$

In general: $= \arg \min_w (\mathbf{y} - \mathbf{X}w)^T (\mathbf{y} - \mathbf{X}w)$

$= \arg \min_w w^T (\mathbf{X}^T \mathbf{X})w - 2y^T \mathbf{X}w$

Each one of the terms in the sum.

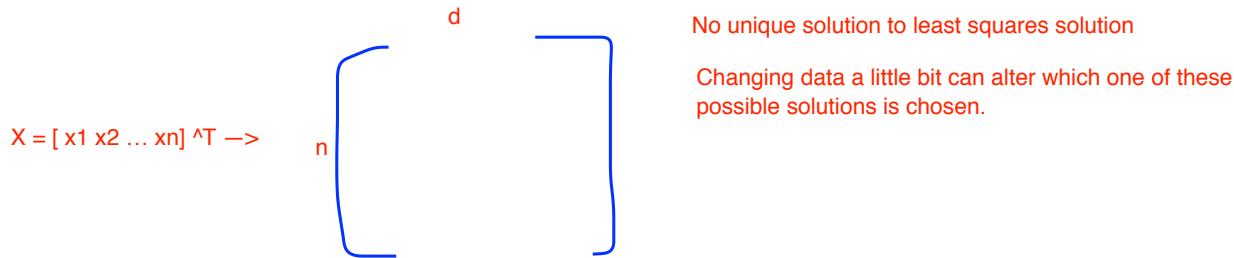


$$(y_1 - x_1^T w)^2 + (y_2 - x_2^T w)^2 + \cdots + (y_n - x_n^T w)^2 = \sum_{i=1}^n (y_i - x_i^T w)^2$$

Regularization in Linear Regression

Recall Least Squares: $\hat{w}_{LS} = \arg \min_w \sum_{i=1}^n (y_i - x_i^T w)^2$

When $x_i \in \mathbb{R}^d$ and $d > n$ the objective function is flat in some directions:



Regularization in Linear Regression

Recall Least Squares: $\hat{w}_{LS} = \arg \min_w \sum_{i=1}^n (y_i - x_i^T w)^2$

When $x_i \in \mathbb{R}^d$ and $d > n$ the objective function is flat in some directions:

Implies optimal solution is *underconstrained* and unstable due to lack of curvature:

- small changes in training data result in large changes in solution Suggests w does not generalize well
- often the *magnitudes* of w are “very large”

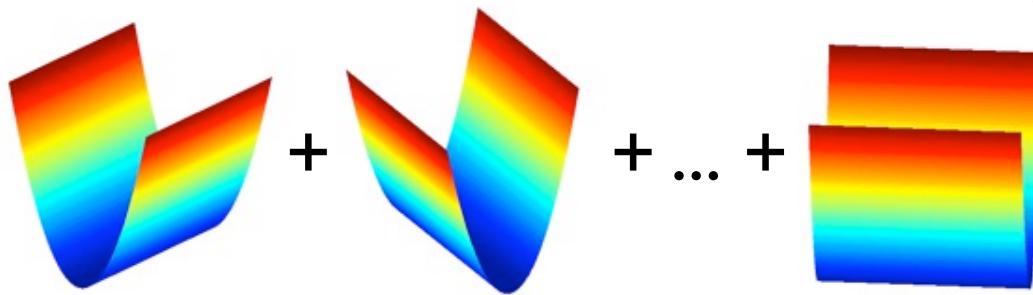
If the entries in w have large magnitude, a small change in one of the inputs can have a drastic change in the prediction made by the model. Not ideal



Regularization imposes “simpler” solutions by a “complexity” penalty

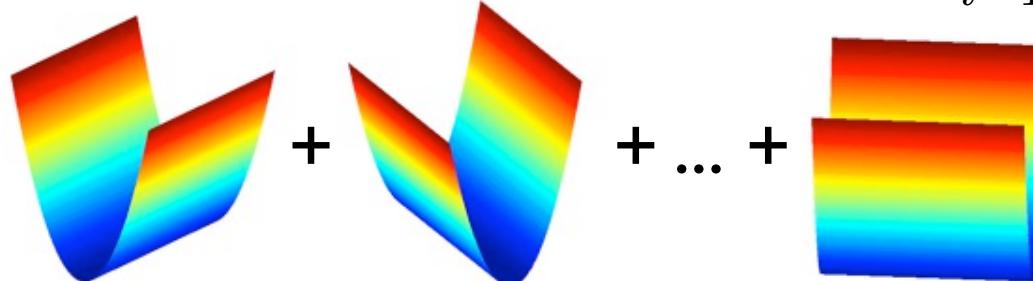
Ridge Regression

- Old Least squares objective:



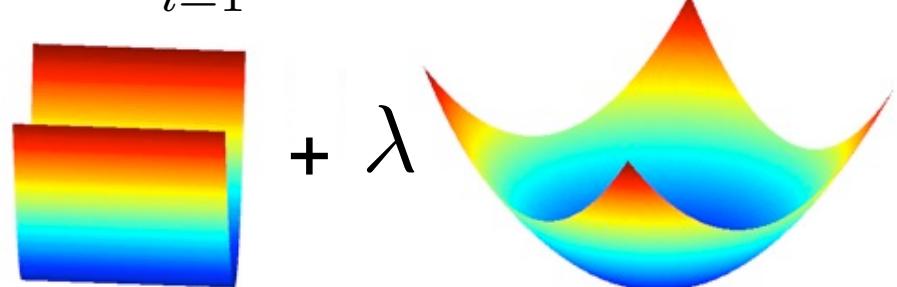
$$\hat{w}_{LS} = \arg \min_w \sum_{i=1}^n (y_i - x_i^T w)^2$$

- Ridge Regression objective:



$$\hat{w}_{ridge} = \arg \min_w \sum_{i=1}^n (y_i - x_i^T w)^2 + \lambda ||w||_2^2$$

$$+ \lambda$$



Minimizing the Ridge Regression Objective

$$\hat{w}_{ridge} = \arg \min_w \sum_{i=1}^n (y_i - x_i^T w)^2 + \lambda ||w||_2^2$$

$$0 = \nabla_w \left(\sum_{i=1}^n (y_i - x_i^T w)^2 + \lambda ||w||_2^2 \right)$$

$$= - \sum_{i=1}^n 2x_i (y_i - x_i^T w) + 2\lambda w$$

$$= -2 \left(\sum_{i=1}^n x_i y_i \right) + 2 \left(\sum_{i=1}^n x_i x_i^T + \lambda I \right) w$$

$$\hat{w}_{ridge} = \left(\sum_{i=1}^n x_i x_i^T + \lambda I \right)^{-1} \left(\sum_{i=1}^n x_i y_i \right)$$

Matrix is always
invertible for lambda > 0
 XX^T is positive semi-definite

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

Shrinkage Properties

$$\hat{w}_{ridge} = (\mathbf{X}^T \mathbf{X} + \lambda I)^{-1} \mathbf{X}^T \mathbf{y}$$

- **Assume:** $\mathbf{X}^T \mathbf{X} = nI$ NOT valid in all cases...

$$\begin{aligned}\hat{w}_{ridge} &= (\mathbf{X}^T \mathbf{X} + \lambda I)^{-1} \mathbf{X}^T \mathbf{y} \\ &= (\mathbf{X}^T \mathbf{X} + \lambda \frac{1}{n} \mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y} \\ &= (1 + \lambda \frac{1}{n})^{-1} (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y} \\ &= \frac{n}{n + \lambda} \hat{w}_{LS}\end{aligned}$$

Shrinking \hat{w} towards the origin

In this special case, Ridge is a “shrunk” Least squares!

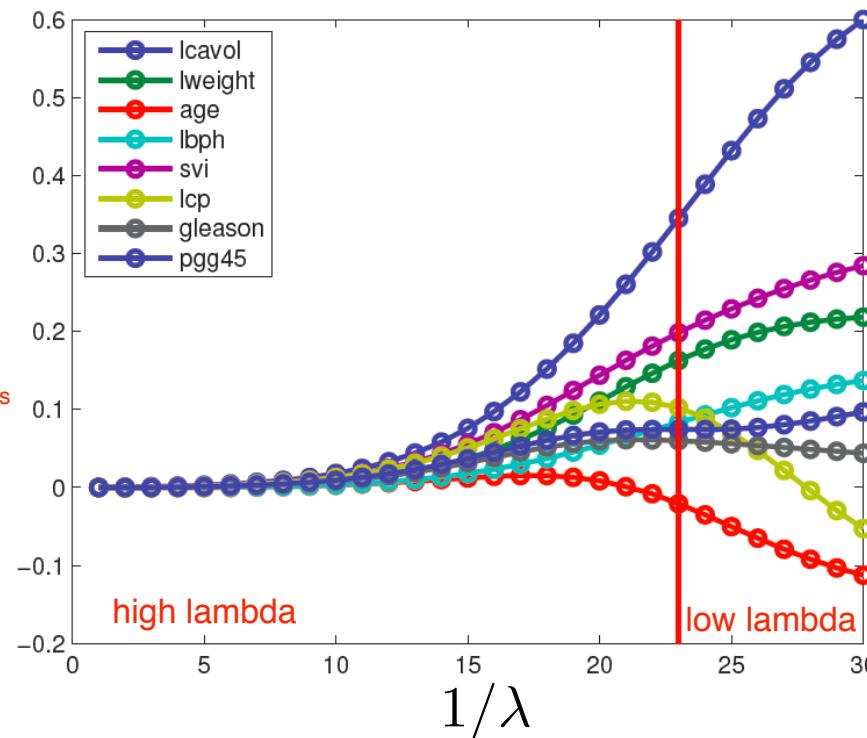
Ridge Coefficient Path

$\mathbf{X}^T \mathbf{X}$ in general

plotted individual components in
the solution vector w

As $\lambda \rightarrow 0$ we get the
least squares solution.

Note: NOT monotonic like in
the case we studied above using
our assumption. Changes solutions
in a nontrivial way.



From
Kevin Murphy
textbook

- Typical approach: select λ using cross validation, up next

Bias-Variance Properties

$$\hat{w}_{ridge} = (\mathbf{X}^T \mathbf{X} + \lambda I)^{-1} \mathbf{X}^T \mathbf{y}$$

our data points have zero mean gaussian error.
Our model given by parameters in \mathbf{X} would be enough
to predict \mathbf{y} , besides some noise.

- **Assume:** $\mathbf{X}^T \mathbf{X} = nI$ **and** $\mathbf{y} = \mathbf{X}w + \boldsymbol{\epsilon}$ $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma^2 I)$

If $x \in \mathbb{R}^d$ and $Y \sim \mathcal{N}(x^T w, \sigma^2)$, what is $\mathbb{E}_{Y|x, \text{train}}[(Y - x^T \hat{w}_{ridge})^2 | X = x]$?

how well does the ridge predictor estimate the true
value that Y takes on?

Bias-Variance Properties

$$\hat{w}_{ridge} = (\mathbf{X}^T \mathbf{X} + \lambda I)^{-1} \mathbf{X}^T \mathbf{y}$$

- **Assume:** $\mathbf{X}^T \mathbf{X} = nI$ **and** $\mathbf{y} = \mathbf{X}w + \boldsymbol{\epsilon}$ $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma^2 I)$

If $x \in \mathbb{R}^d$ and $Y \sim \mathcal{N}(x^T w, \sigma^2)$, what is $\mathbb{E}_{Y|x, \text{train}}[(Y - x^T \hat{w}_{ridge})^2 | X = x]$?

$$\mathbb{E}_{Y|X, \mathcal{D}}[(Y - x^T \hat{w}_{ridge})^2 | X = x] \quad \text{Same math as before.}$$

$$= \underbrace{\mathbb{E}_{Y|X}[(Y - \mathbb{E}_{Y|X}[Y|X = x])^2 | X = x]}_{\text{Irreducible Error}} + \underbrace{\mathbb{E}_{\mathcal{D}}[(\mathbb{E}_{Y|X}[Y|X = x] - x^T \hat{w}_{ridge})^2]}_{\text{Learning Error}}$$

$$= \sigma^2$$

$Y \sim N(x^T w, \sigma^2)$
so this is $\mu = x^T w$

Bias-Variance Properties

$$\hat{w}_{ridge} = (\mathbf{X}^T \mathbf{X} + \lambda I)^{-1} \mathbf{X}^T \mathbf{y}$$

- **Assume:** $\mathbf{X}^T \mathbf{X} = nI$ **and** $\mathbf{y} = \mathbf{X}w + \epsilon^{\text{Text}}$ $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$

If $x \in \mathbb{R}^d$ and $Y \sim \mathcal{N}(x^T w, \sigma^2)$, what is $\mathbb{E}_{Y|x, \text{train}}[(Y - x^T \hat{w}_{ridge})^2 | X = x]$?

$$\begin{aligned}\mathbb{E}_{Y|X, \mathcal{D}}[(Y - x^T \hat{w}_{ridge})^2 | X = x] &\leftarrow \text{expected prediction error at } \mathbf{X} = \mathbf{x} \\&= \mathbb{E}_{Y|X}[(Y - \mathbb{E}_{Y|X}[Y|X = x])^2 | X = x] + \mathbb{E}_{\mathcal{D}}[(\mathbb{E}_{Y|X}[Y|X = x] - x^T \hat{w}_{ridge})^2] \\&= \mathbb{E}_{Y|X}[(Y - x^T w)^2 | X = x] + \mathbb{E}_{\mathcal{D}}[(x^T w - x^T \hat{w}_{ridge})^2] \\&= \underline{\sigma^2} + \underline{(x^T w - \mathbb{E}_{\mathcal{D}}[x^T \hat{w}_{ridge}])^2} + \underline{\mathbb{E}_{\mathcal{D}}[(\mathbb{E}_{\mathcal{D}}[x^T \hat{w}_{ridge}] - x^T \hat{w}_{ridge})^2]}\end{aligned}$$

Irreduc. Error Bias-squared Variance

Bias-Variance Properties

$$\hat{w}_{ridge} = (\mathbf{X}^T \mathbf{X} + \lambda I)^{-1} \mathbf{X}^T \mathbf{y}$$

- **Assume:** $\mathbf{X}^T \mathbf{X} = nI$ **and** $\mathbf{y} = \mathbf{X}w + \boldsymbol{\epsilon}$ $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma^2 I)$

If $x \in \mathbb{R}^d$ and $Y \sim \mathcal{N}(x^T w, \sigma^2)$, what is $\mathbb{E}_{Y|x, \text{train}}[(Y - x^T \hat{w}_{ridge})^2 | X = x]$?

$$\begin{aligned}\mathbb{E}_{Y|X, \mathcal{D}}[(Y - x^T \hat{w}_{ridge})^2 | X = x] &= \mathbb{E}_{Y|X}[(Y - \mathbb{E}_{Y|X}[Y|X = x])^2 | X = x] + \mathbb{E}_{\mathcal{D}}[(\mathbb{E}_{Y|X}[Y|X = x] - x^T \hat{w}_{ridge})^2] \\ &= \mathbb{E}_{Y|X}[(Y - x^T w)^2 | X = x] + \mathbb{E}_{\mathcal{D}}[(x^T w - x^T \hat{w}_{ridge})^2] \text{ do the trick where you add and subtract } \mathbb{E}_{\mathcal{D}}[x^T w_{\text{ridge}}] \text{ from this} \\ &= \underline{\sigma^2} + \underline{(x^T w - \mathbb{E}_{\mathcal{D}}[x^T \hat{w}_{ridge}])^2} + \underline{\mathbb{E}_{\mathcal{D}}[(\mathbb{E}_{\mathcal{D}}[x^T \hat{w}_{ridge}])^2]} \text{ This is the MSE}\end{aligned}$$

Irreduc. Error

Bias squared

Variance

plugging this in
we can calc
 $\mathbb{E}_{\mathcal{D}}[x^T w_{\text{ridge}}]$

$\mathbb{E}[\boldsymbol{\epsilon}] = 0$ so
the right term
disappears

Total result on next slide

$$\hat{w}_{ridge} = (\mathbf{X}^T \mathbf{X} + \lambda I)^{-1} \mathbf{X}^T (\mathbf{X}w + \boldsymbol{\epsilon})$$

$$= \frac{n}{n + \lambda} w + \frac{1}{n + \lambda} \mathbf{X}^T \boldsymbol{\epsilon}$$

ONLY TRUE
UNDER THE
ABOVE ASSUMPTIONS

Bias-Variance Properties

$$\hat{w}_{ridge} = (\mathbf{X}^T \mathbf{X} + \lambda I)^{-1} \mathbf{X}^T \mathbf{y}$$

- **Assume:** $\mathbf{X}^T \mathbf{X} = nI$ **and** $\mathbf{y} = \mathbf{X}w + \boldsymbol{\epsilon}$ $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma^2 I)$

If $x \in \mathbb{R}^d$ and $Y \sim \mathcal{N}(x^T w, \sigma^2)$, what is $\mathbb{E}_{Y|x, \text{train}}[(Y - x^T \hat{w}_{ridge})^2 | X = x]$?

$$\begin{aligned}\mathbb{E}_{Y|X, \mathcal{D}}[(Y - x^T \hat{w}_{ridge})^2 | X = x] &= \mathbb{E}_{Y|X}[(Y - \mathbb{E}_{Y|X}[Y|X = x])^2 | X = x] + \mathbb{E}_{\mathcal{D}}[(\mathbb{E}_{Y|X}[Y|X = x] - x^T \hat{w}_{ridge})^2] \\ &= \mathbb{E}_{Y|X}[(Y - x^T w)^2 | X = x] + \mathbb{E}_{\mathcal{D}}[(x^T w - x^T \hat{w}_{ridge})^2] \\ &= \sigma^2 + (x^T w - \mathbb{E}_{\mathcal{D}}[x^T \hat{w}_{ridge}])^2 + \mathbb{E}_{\mathcal{D}}[(\mathbb{E}_{\mathcal{D}}[x^T \hat{w}_{ridge}] - x^T \hat{w}_{ridge})^2]\end{aligned}$$

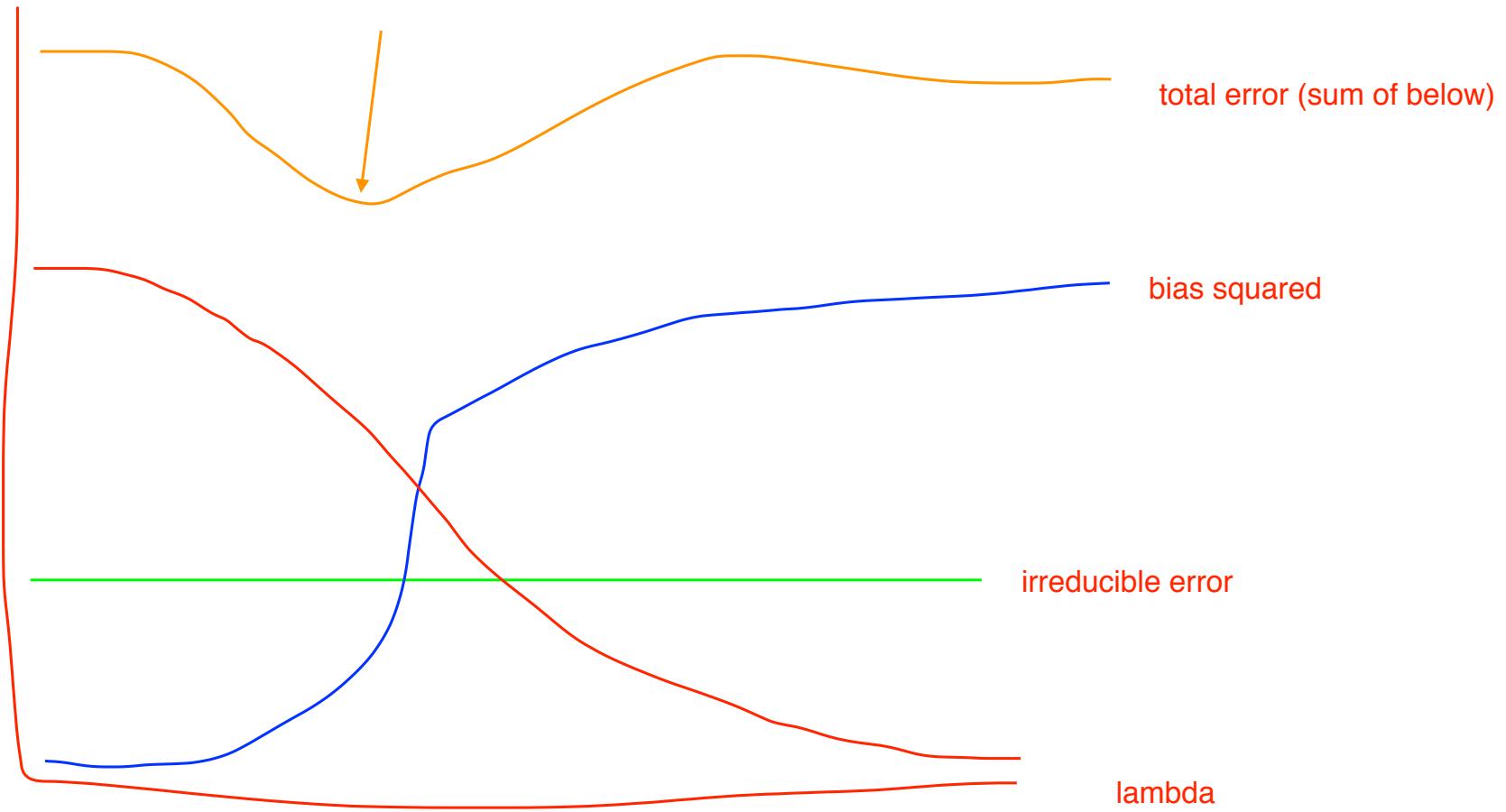
expected (total) error
in predicting Y given $X = x$ = $\frac{\text{Irreduc. Error}}{\text{Bias-squared}} + \frac{\text{Variance}}{d}$

d is the number of features in x. i.e. $x \in \mathbb{R}^d$
(verify at home)

$\text{Irreduc. Error} = \sigma^2 + \frac{\lambda^2}{(n + \lambda)^2} (w^T x)^2 + \frac{d\sigma^2 n}{(n + \lambda)^2} \|x\|_2^2$

can never predict Y better than the irreducible error \rightarrow error will never go to zero unless Irreducible error = 0

choose lambda that minimizes total error



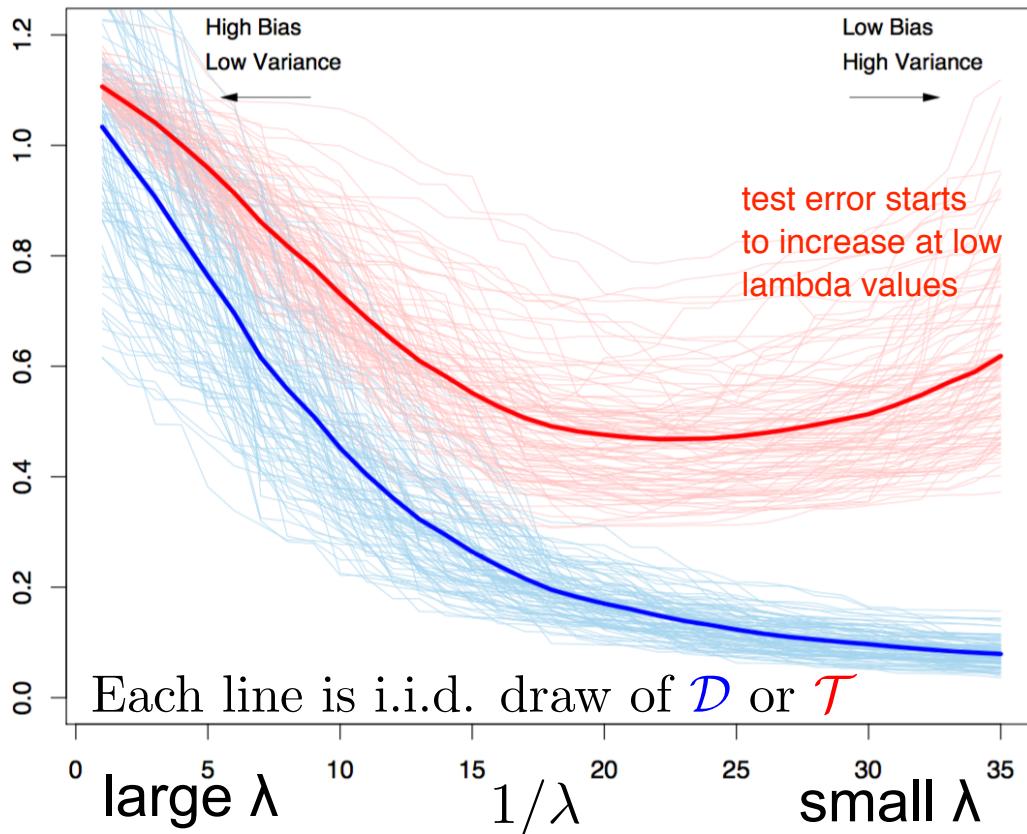
Ridge Regression: Effect of Regularization

$$\hat{w}_{ridge} = \arg \min_w \sum_{i=1}^n (y_i - x_i^T w)^2 + \lambda ||w||_2^2$$

- Solution is indexed by the regularization parameter λ
- Larger λ Bias increases; Variance decreases
- Smaller λ
- As $\lambda \rightarrow 0$, $\hat{w}_{ridge} \rightarrow$ w_LS = least squares solution
- As $\lambda \rightarrow \infty$, $\hat{w}_{ridge} \rightarrow$ zero vector

Ridge Regression: Effect of Regularization

$$\mathcal{D} \stackrel{i.i.d.}{\sim} P_{XY} \quad \widehat{w}_{\mathcal{D}, \text{ridge}}^{(\lambda)} = \arg \min_w \frac{1}{|\mathcal{D}|} \sum_{(x_i, y_i) \in \mathcal{D}} (y_i - x_i^T w)^2 + \lambda \|w\|_2^2$$



TRAIN error:

$$\frac{1}{|\mathcal{D}|} \sum_{(x_i, y_i) \in \mathcal{D}} (y_i - x_i^T \widehat{w}_{\mathcal{D}, \text{ridge}}^{(\lambda)})^2$$

TEST error:

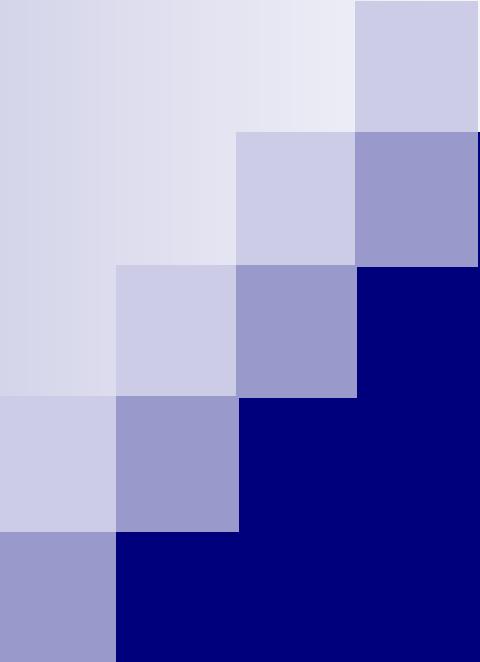
$$\mathcal{T} \stackrel{i.i.d.}{\sim} P_{XY}$$

$$\frac{1}{|\mathcal{T}|} \sum_{(x_i, y_i) \in \mathcal{T}} (y_i - x_i^T \widehat{w}_{\mathcal{D}, \text{ridge}}^{(\lambda)})^2$$

Important: $\mathcal{D} \cap \mathcal{T} = \emptyset$

What you need to know...

- Regularization
 - Penalizes for complex models
- Ridge regression
 - L_2 penalized least-squares regression
 - Regularization parameter trades off model complexity with training error



Cross-Validation

Machine Learning – CSE546
Kevin Jamieson
University of Washington

April 15, 2016

How... How... How???????

- *How do we pick the regularization constant λ ...*
- *How do we pick the number of basis functions...*

- We could use the test data, but...

How... How... How???????

- *How do we pick the regularization constant λ ...*
- *How do we pick the number of basis functions...*
- We could use the test data, but...
- Never ever ever ever ever ever ever
ever ever ever ever ever ever ever ever
ever ever ever ever ever ever ever ever
train on the test data

(LOO) Leave-one-out cross validation

- Consider a **validation set with 1 example**:
 - D – training data
 - $D \setminus j$ – training data with j th data point $(\mathbf{x}_j, \mathbf{y}_j)$ moved to validation set
- **Learn classifier $f_{D \setminus j}$ with $D \setminus j$ dataset**
- **Estimate true error** as squared error on predicting \mathbf{y}_j :
 - Unbiased estimate of error_{true}($f_{D \setminus j}$)!

□

(LOO) Leave-one-out cross validation

- Consider a **validation set with 1 example**:
 - D – training data
 - $D \setminus j$ – training data with j th data point (x_j, y_j) moved to validation set
- **Learn classifier $f_{D \setminus j}$ with $D \setminus j$ dataset**
- **Estimate true error** as squared error on predicting y_j :
 - Unbiased estimate of error_{true}($f_{D \setminus j}$)!
- **LOO cross validation**: Average over all data points j :
 - For each data point you leave out, learn a new classifier $f_{D \setminus j}$
 - Estimate error as:

$$\text{error}_{LOO} = \frac{1}{n} \sum_{j=1}^n (y_j - f_{D \setminus j}(x_j))^2$$

LOO cross validation is (almost) unbiased estimate of true error of h_D !

- When computing **LOOCV error**, we only use $N-1$ data points
 - So it's not estimate of true error of learning with N data points
 - Usually pessimistic, though – learning with less data typically gives worse answer
- **LOO is almost unbiased! Use LOO error for model selection!!!**
 - E.g., picking λ

Computational cost of LOO

- Suppose you have 100,000 data points
- You implemented a great version of your learning algorithm
 - Learns in only 1 second
- Computing LOO will take about 1 day!!!
 -

Use k -fold cross validation

- Randomly divide training data into k equal parts
 - D_1, \dots, D_k
- For each i
 - Learn classifier $f_{D \setminus D_i}$ using data point not in D_i
 - Estimate error of $f_{D \setminus D_i}$ on validation set D_i :
- $$\text{error}_{D_i} = \frac{1}{|D_i|} \sum_{(x_j, y_j) \in D_i} (y_j - f_{D \setminus D_i}(x_j))^2$$



Use k -fold cross validation

- Randomly divide training data into k equal parts
 - D_1, \dots, D_k

- For each i

- Learn classifier $f_{D \setminus D_i}$ using data point not in D_i
 - Estimate error of $f_{D \setminus D_i}$ on validation set D_i :

$$\text{error}_{\mathcal{D}_i} = \frac{1}{|\mathcal{D}_i|} \sum_{(x_j, y_j) \in \mathcal{D}_i} (y_j - f_{\mathcal{D} \setminus \mathcal{D}_i}(x_j))^2$$

- **k -fold cross validation error is average** over data splits:

$$\text{error}_{k\text{-}fold} = \frac{1}{k} \sum_{i=1}^k \text{error}_{\mathcal{D}_i}$$

- k -fold cross validation properties:

- **Much faster to compute** than LOO
 - **More (pessimistically) biased** – using much less data, only $n(k-1)/k$
 - **Usually, $k = 10$**

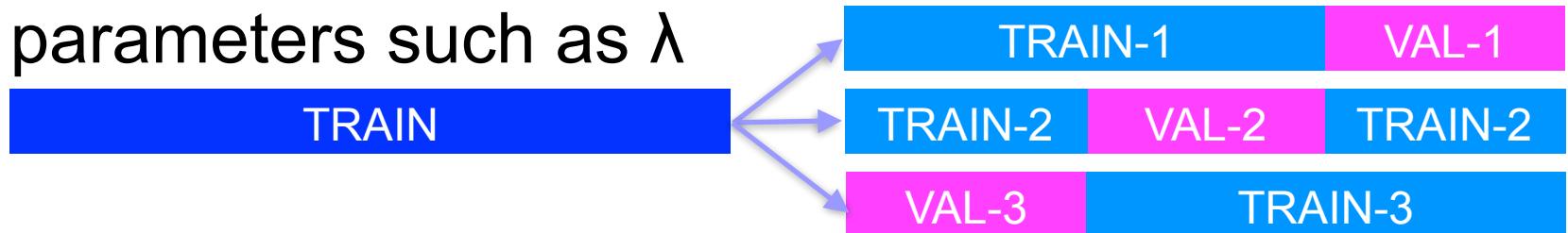


Recap

- Given a dataset, begin by splitting into



- Model selection:** Use k-fold cross-validation on **TRAIN** to train predictor and choose magic parameters such as λ



- Model assessment:** Use **TEST** to assess the accuracy of the model you output
 - Never ever ever ever train or choose parameters based on the test data

Example

- Given 10,000-dimensional data and n examples, we pick a subset of 50 dimensions that have the highest correlation with labels in the training set:

50 indices j that have largest

$$\frac{|\sum_{i=1}^n x_{i,j} y_i|}{\sqrt{\sum_{i=1}^n x_{i,j}^2}}$$

- After picking our 50 features, we then use CV to train ridge regression with regularization λ
- What's wrong with this procedure?

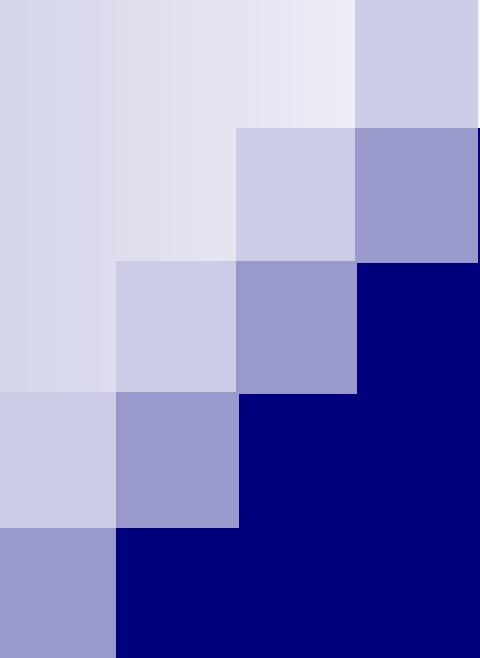
Recap

- Learning is...
 - Collect some data
 - E.g., housing info and sale price
 - Randomly split dataset into TRAIN, VAL, and TEST
 - E.g., 80%, 10%, and 10%, respectively
 - Choose a hypothesis class or model
 - E.g., linear with non-linear transformations
 - Choose a loss function
 - E.g., least squares with ridge regression penalty on TRAIN
 - Choose an optimization procedure
 - E.g., set derivative to zero to obtain estimator, cross-validation on VAL to pick num. features and amount of regularization
 - Justifying the accuracy of the estimate
 - E.g., report TEST error

Warm up

Fix any $a, b, c > 0$.

1. What is the $x \in \mathbb{R}$ that minimizes $ax^2 + bx + c$
2. What is the $x \in \mathbb{R}$ that minimizes $\max\{-ax + b, cx\}$



Simple Variable Selection LASSO: Sparse Regression

Machine Learning – CSE546
Kevin Jamieson
University of Washington

April 15, 2019

Sparsity

$$\hat{w}_{LS} = \arg \min_w \sum_{i=1}^n (y_i - x_i^T w)^2$$

- Vector w is sparse, if many entries are zero

Sparsity

$$\hat{w}_{LS} = \arg \min_w \sum_{i=1}^n (y_i - x_i^T w)^2$$

- Vector w is sparse, if many entries are zero
- Very useful for many tasks, e.g.,
 - **Efficiency:** If $\text{size}(w) = 100$ Billion, each prediction is expensive:
 - If w is sparse, prediction computation only depends on number of non-zeros
 - **Interpretability:** What are the relevant dimensions to make a prediction?
 - E.g., what are the parts of the brain associated with particular words?

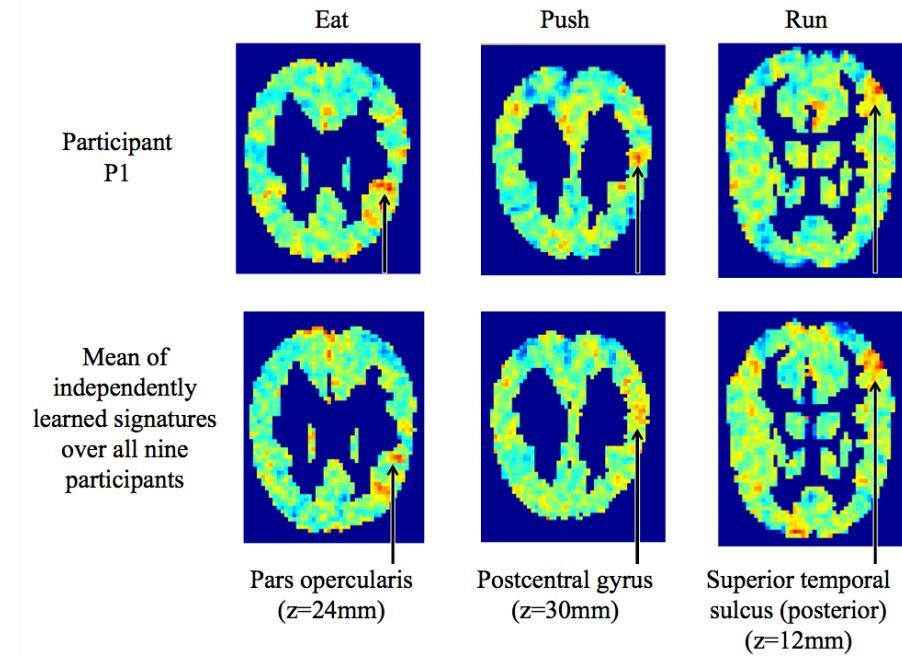


Figure from Tom Mitchell

Sparsity

$$\hat{w}_{LS} = \arg \min_w \sum_{i=1}^n (y_i - x_i^T w)^2$$

- Vector w is sparse, if many entries are zero
- Very useful for many tasks, e.g.,
 - **Efficiency:** If $\text{size}(w) = 100$ Billion, each prediction is expensive:
 - If w is sparse, prediction computation only depends on number of non-zeros
 - **Interpretability:** What are the relevant dimensions to make a prediction?
 - E.g., what are the parts of the brain associated with particular words?
- How do we find “best” subset among all possible?

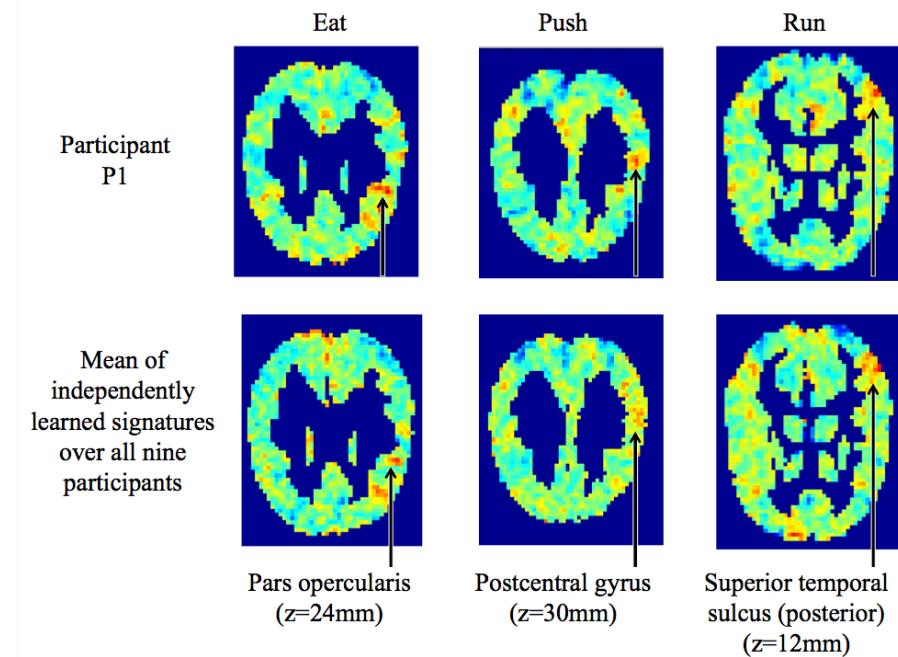


Figure from Tom Mitchell

Greedy model selection algorithm



- Pick a dictionary of features
 - e.g., cosines of random inner products
- Greedy heuristic:
 - Start from empty (or simple) set of features $F_0 = \emptyset$
 - Run learning algorithm for current set of features F_t
 - Obtain weights for these features
 - Select **next best feature $h_i(x)^*$**
 - e.g., $h_j(x)$ that results in lowest training error learner when using $F_t + \{h_j(x)^*\}$
 - $F_{t+1} \leftarrow F_t + \{h_i(x)^*\}$
 - Recurse

Greedy model selection

- Applicable in many other settings:
 - Considered later in the course:
 - Logistic regression: Selecting features (basis functions)
 - Naïve Bayes: Selecting (independent) features $P(X_i|Y)$
 - Decision trees: Selecting leaves to expand
- Only a heuristic!
 - **Finding the best set of k features is computationally intractable!**
 - Sometimes you can prove something strong about it...
-

When do we stop???

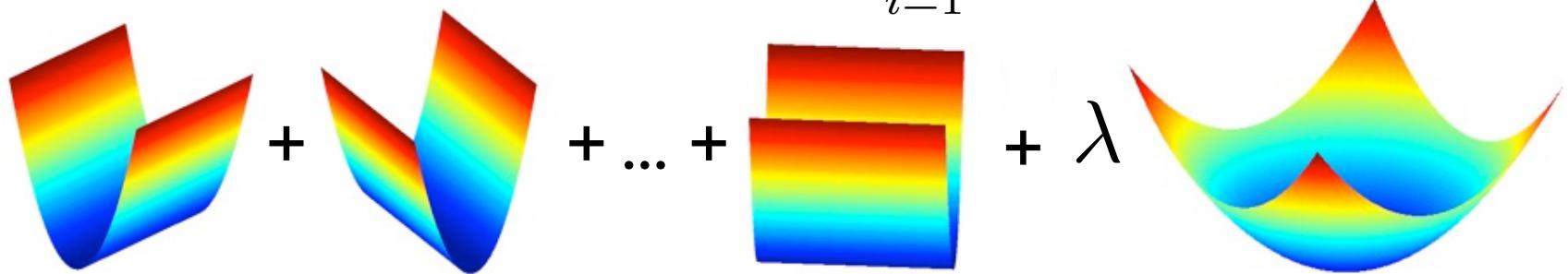
- Greedy heuristic:
 - ...
 - Select **next best feature X_i^***
 - E.g. $h_j(x)$ that results in lowest training error learner when using $F_t + \{h_j(x)^*\}$
 - Recurse **When do you stop???**
 - When training error is low enough?
 - When test set error is low enough?
 - Using cross validation?

Is there a more principled approach?

Recall Ridge Regression

- Ridge Regression objective:

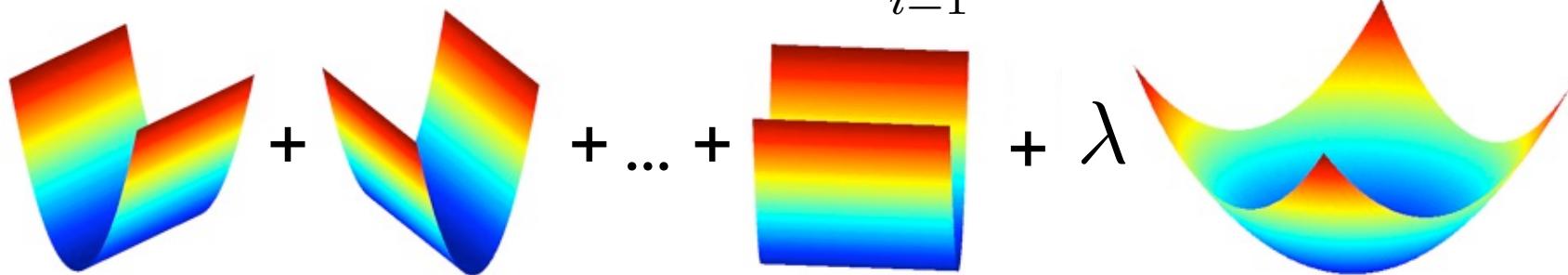
$$\hat{w}_{ridge} = \arg \min_w \sum_{i=1}^n (y_i - x_i^T w)^2 + \lambda ||w||_2^2$$



Ridge vs. Lasso Regression

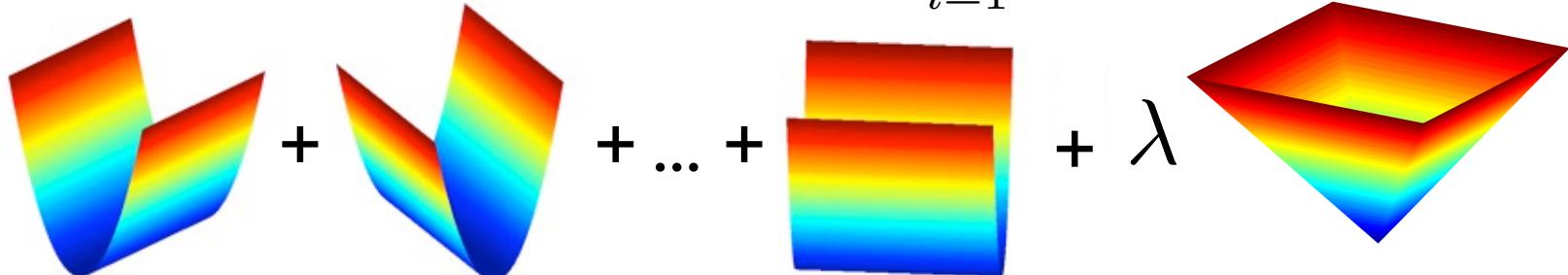
- Ridge Regression objective:

$$\hat{w}_{ridge} = \arg \min_w \sum_{i=1}^n (y_i - x_i^T w)^2 + \lambda ||w||_2^2$$



- Lasso objective:

$$\hat{w}_{lasso} = \arg \min_w \sum_{i=1}^n (y_i - x_i^T w)^2 + \lambda ||w||_1$$



Penalized Least Squares

$$\text{Ridge : } r(w) = \|w\|_2^2 \quad \text{Lasso : } r(w) = \|w\|_1$$

$$\widehat{w}_r = \arg \min_w \sum_{i=1}^n (y_i - x_i^T w)^2 + \lambda r(w)$$

Penalized Least Squares

$$\text{Ridge : } r(w) = \|w\|_2^2 \quad \text{Lasso : } r(w) = \|w\|_1$$

$$\widehat{w}_r = \arg \min_w \sum_{i=1}^n (y_i - x_i^T w)^2 + \lambda r(w)$$

For any $\lambda \geq 0$ for which \widehat{w}_r achieves the minimum, there exists a $\nu \geq 0$ such that

$$\widehat{w}_r = \arg \min_w \sum_{i=1}^n (y_i - x_i^T w)^2 \quad \text{subject to } r(w) \leq \nu$$

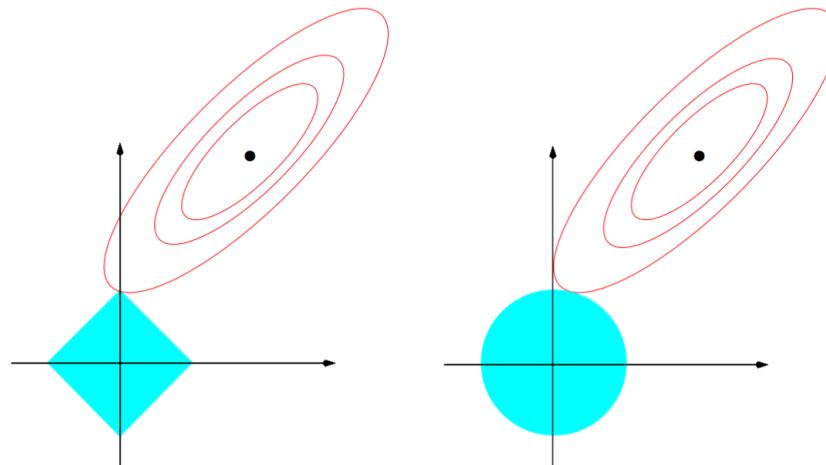
Penalized Least Squares

$$\text{Ridge : } r(w) = \|w\|_2^2 \quad \text{Lasso : } r(w) = \|w\|_1$$

$$\hat{w}_r = \arg \min_w \sum_{i=1}^n (y_i - x_i^T w)^2 + \lambda r(w)$$

For any $\lambda \geq 0$ for which \hat{w}_r achieves the minimum, there exists a $\nu \geq 0$ such that

$$\hat{w}_r = \arg \min_w \sum_{i=1}^n (y_i - x_i^T w)^2 \quad \text{subject to } r(w) \leq \nu$$



Optimizing the LASSO Objective

- LASSO solution:

$$\hat{w}_{lasso}, \hat{b}_{lasso} = \arg \min_{w,b} \sum_{i=1}^n (y_i - (x_i^T w + b))^2 + \lambda ||w||_1$$

$$\hat{b}_{lasso} = \arg \min_{w,b} \frac{1}{n} \sum_{i=1}^n (y_i - x_i^T \hat{w}_{lasso})$$

Optimizing the LASSO Objective

- LASSO solution:

$$\hat{w}_{lasso}, \hat{b}_{lasso} = \arg \min_{w,b} \sum_{i=1}^n (y_i - (x_i^T w + b))^2 + \lambda ||w||_1$$

$$\hat{b}_{lasso} = \arg \min_{w,b} \frac{1}{n} \sum_{i=1}^n (y_i - x_i^T \hat{w}_{lasso})$$

So as usual, preprocess to make sure that $\frac{1}{n} \sum_{i=1}^n y_i = 0, \frac{1}{n} \sum_{i=1}^n x_i = \mathbf{0}$
so we don't have to worry about an offset.

Optimizing the LASSO Objective

- LASSO solution:

$$\hat{w}_{lasso}, \hat{b}_{lasso} = \arg \min_{w,b} \sum_{i=1}^n (y_i - (x_i^T w + b))^2 + \lambda ||w||_1$$

$$\hat{b}_{lasso} = \arg \min_{w,b} \frac{1}{n} \sum_{i=1}^n (y_i - x_i^T \hat{w}_{lasso})$$

So as usual, preprocess to make sure that $\frac{1}{n} \sum_{i=1}^n y_i = 0, \frac{1}{n} \sum_{i=1}^n x_i = \mathbf{0}$
so we don't have to worry about an offset.

$$\hat{w}_{lasso} = \arg \min_w \sum_{i=1}^n (y_i - x_i^T w)^2 + \lambda ||w||_1$$

How do we solve this?

Coordinate Descent

- Given a function, we want to find minimum
- Often, it is easy to find minimum along a single coordinate:
- How do we pick next coordinate?
- Super useful approach for ***many*** problems
 - Converges to optimum in some cases, such as LASSO

Optimizing LASSO Objective One Coordinate at a Time

Fix any $j \in \{1, \dots, d\}$

$$\begin{aligned} \sum_{i=1}^n (y_i - x_i^T w)^2 + \lambda ||w||_1 &= \sum_{i=1}^n \left(y_i - \sum_{k=1}^d x_{i,k} w_k \right)^2 + \lambda \sum_{k=1}^d |w_k| \\ &= \sum_{i=1}^n \left(\left(y_i - \sum_{k \neq j} x_{i,k} w_k \right) - x_{i,j} w_j \right)^2 + \lambda \sum_{k \neq j} |w_k| + \lambda |w_j| \end{aligned}$$

Optimizing LASSO Objective One Coordinate at a Time

Fix any $j \in \{1, \dots, d\}$

$$\begin{aligned} \sum_{i=1}^n (y_i - x_i^T w)^2 + \lambda ||w||_1 &= \sum_{i=1}^n \left(y_i - \sum_{k=1}^d x_{i,k} w_k \right)^2 + \lambda \sum_{k=1}^d |w_k| \\ &= \sum_{i=1}^n \left(\left(y_i - \sum_{k \neq j} x_{i,k} w_k \right) - x_{i,j} w_j \right)^2 + \lambda \sum_{k \neq j} |w_k| + \lambda |w_j| \end{aligned}$$

Initialize $\hat{w}_k = 0$ for all $k \in \{1, \dots, d\}$

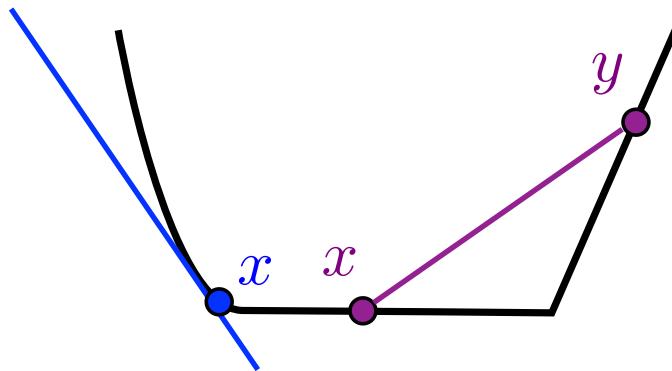
Loop over $j \in \{1, \dots, n\}$:

$$r_i^{(j)} = \sum_{k \neq j} x_{i,j} \hat{w}_k$$

$$\hat{w}_j = \arg \min_{w_j} \sum_{i=1}^n \left(r_i^{(j)} - x_{i,j} w_j \right)^2 + \lambda |w_j|$$

Convex Functions

- Equivalent definitions of convexity:



f convex:

$$f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y) \quad \forall x, y, \lambda \in [0, 1]$$

$$f(y) \geq f(x) + \nabla f(x)^T(y - x) \quad \forall x, y$$

- **Gradients** lower bound convex functions and are unique at \mathbf{x} iff function differentiable at \mathbf{x}
- **Subgradients** generalize gradients to non-differentiable points:
 - Any supporting hyperplane at \mathbf{x} that lower bounds entire function

g is a subgradient at x if $f(y) \geq f(x) + g^T(y - x)$

Taking the Subgradient

$$\widehat{w}_j = \arg \min_{w_j} \sum_{i=1}^n \left(r_i^{(j)} - x_{i,j} w_j \right)^2 + \lambda |w_j|$$

g is a subgradient at x if $f(y) \geq f(x) + g^T(y - x)$

- Convex function is minimized at w if 0 is a sub-gradient at w .

$$\partial_{w_j} |w_j| =$$

$$\partial_{w_j} \sum_{i=1}^n \left(r_i^{(j)} - x_{i,j} w_j \right)^2 =$$

Setting Subgradient to 0

$$\partial_{w_j} \left(\sum_{i=1}^n \left(r_i^{(j)} - x_{i,j} w_j \right)^2 + \lambda |w_j| \right) = \begin{cases} a_j w_j - c_j - \lambda & \text{if } w_j < 0 \\ [-c_j - \lambda, -c_j + \lambda] & \text{if } w_j = 0 \\ a_j w_j - c_j + \lambda & \text{if } w_j > 0 \end{cases}$$

$$a_j = \left(\sum_{i=1}^n x_{i,j}^2 \right) \quad c_j = 2 \left(\sum_{i=1}^n r_i^{(j)} x_{i,j} \right)$$

Setting Subgradient to 0

$$\partial_{w_j} \left(\sum_{i=1}^n \left(r_i^{(j)} - x_{i,j} w_j \right)^2 + \lambda |w_j| \right) = \begin{cases} a_j w_j - c_j - \lambda & \text{if } w_j < 0 \\ [-c_j - \lambda, -c_j + \lambda] & \text{if } w_j = 0 \\ a_j w_j - c_j + \lambda & \text{if } w_j > 0 \end{cases}$$

$$a_j = \left(\sum_{i=1}^n x_{i,j}^2 \right) \quad c_j = 2 \left(\sum_{i=1}^n r_i^{(j)} x_{i,j} \right)$$

$$\widehat{w}_j = \arg \min_{w_j} \sum_{i=1}^n \left(r_i^{(j)} - x_{i,j} w_j \right)^2 + \lambda |w_j|$$

w is a minimum if
0 is a sub-gradient at w

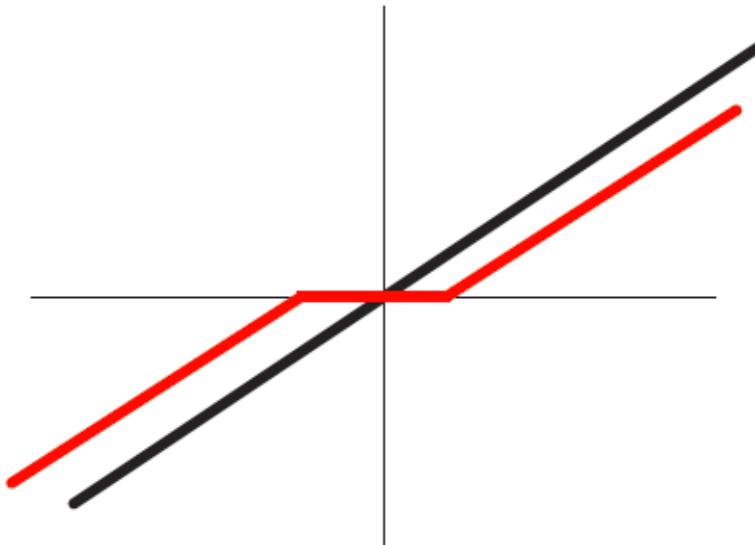
$$\widehat{w}_j = \begin{cases} (c_j + \lambda)/a_j & \text{if } c_j < -\lambda \\ 0 & \text{if } |c_j| \leq \lambda \\ (c_j - \lambda)/a_j & \text{if } c_j > \lambda \end{cases}$$

Soft Thresholding

$$\hat{w}_j = \begin{cases} (c_j + \lambda)/a_j & \text{if } c_j < -\lambda \\ 0 & \text{if } |c_j| \leq \lambda \\ (c_j - \lambda)/a_j & \text{if } c_j > \lambda \end{cases}$$

$$a_j = \sum_{i=1}^n x_{i,j}^2$$

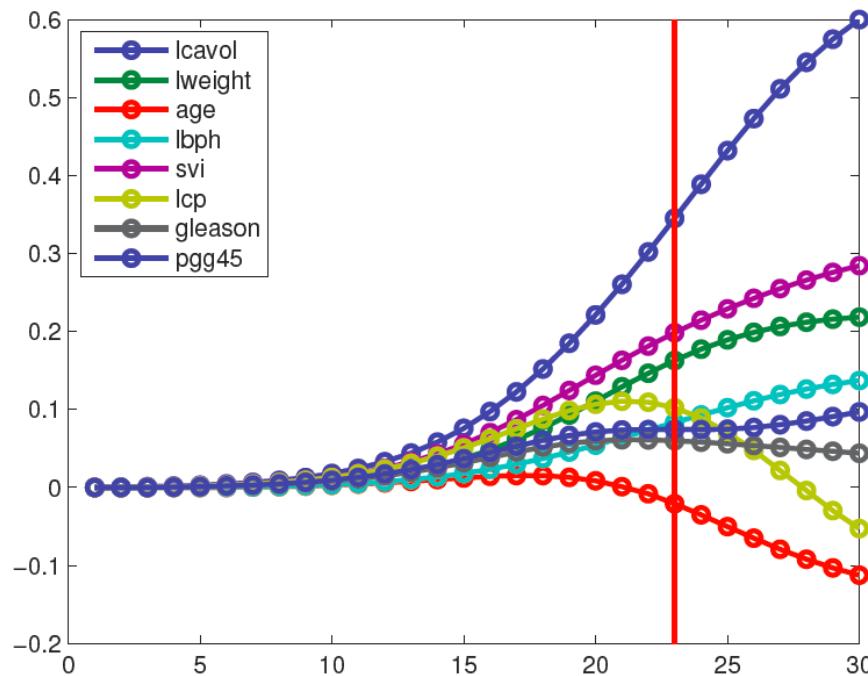
$$c_j = 2 \sum_{i=1}^n \left(y_i - \sum_{k \neq j} x_{i,k} w_k \right) x_{i,j}$$



Coordinate Descent for LASSO (aka Shooting Algorithm)

- Repeat until convergence
 - Pick a coordinate j at (random or sequentially)
 - Set:
$$\hat{w}_j = \begin{cases} (c_j + \lambda)/a_j & \text{if } c_j < -\lambda \\ 0 & \text{if } |c_j| \leq \lambda \\ (c_j - \lambda)/a_j & \text{if } c_j > \lambda \end{cases}$$
 - Where:
$$a_j = \sum_{i=1}^n x_{i,j}^2 \quad c_j = 2 \sum_{i=1}^n \left(y_i - \sum_{k \neq j} x_{i,k} w_k \right) x_{i,j}$$
 - For convergence rates, see Shalev-Shwartz and Tewari 2009
- Other common technique = LARS
 - Least angle regression and shrinkage, Efron et al. 2004

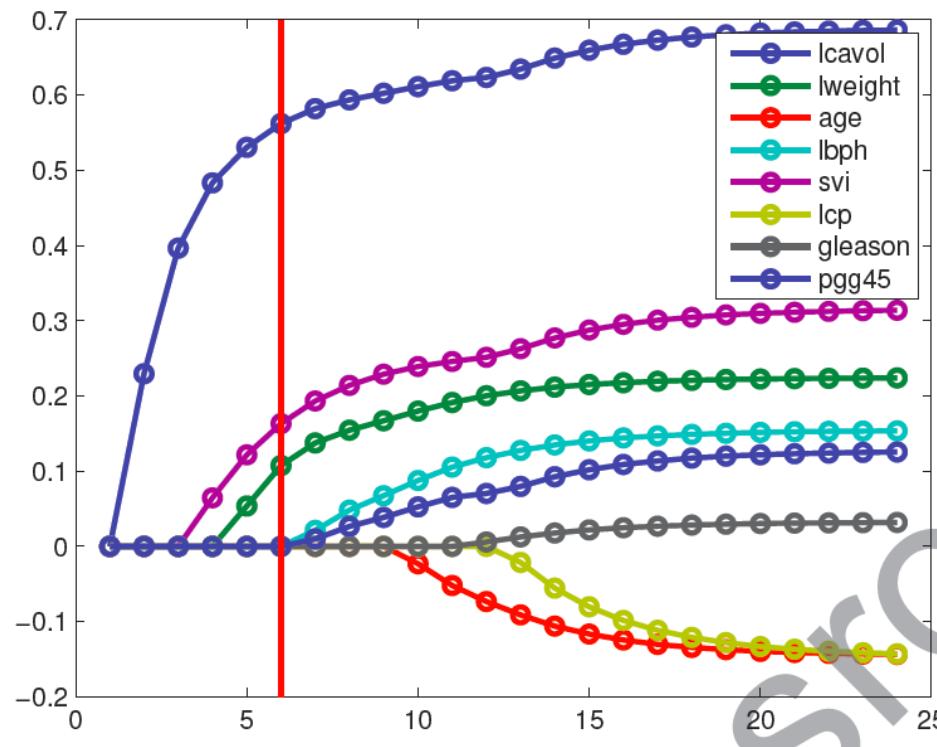
Recall: *Ridge Coefficient Path*



From
Kevin Murphy
textbook

- Typical approach: select λ using cross validation

Now: LASSO Coefficient Path



From
Kevin Murphy
textbook

What you need to know

- Variable Selection: find a sparse solution to learning problem
- L_1 regularization is one way to do variable selection
 - Applies beyond regression
 - Hundreds of other approaches out there
- LASSO objective non-differentiable, **but convex** → Use subgradient
- No closed-form solution for minimization → Use coordinate descent
- Shooting algorithm is simple approach for solving LASSO