

Trading off bias and variance, Cross-validation

Bias-variance tradeoff for least squares

If $Y_i = X_i^T w^* + \epsilon_i$ and $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$

$$\widehat{w}_{\text{MLE}} = w^* + (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \epsilon$$

$$\eta(x) = x^T w^*$$

$$\widehat{f}_{\mathcal{D}}(x) = x^T w^* + x^T (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \epsilon$$

- Variance: $\mathbb{E}_{\mathcal{D}} \left[\left(\widehat{f}_{\mathcal{D}}(x) - \mathbb{E}_{\mathcal{D}}[\widehat{f}_{\mathcal{D}}(x)] \right)^2 \right] = \mathbb{E}_{\mathcal{D}} [x^T (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \epsilon \epsilon^T \mathbf{X} (\mathbf{X}^T \mathbf{X})^{-1} x]$
 $= \sigma^2 \mathbb{E}_{\mathcal{D}} [x^T (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{X} (\mathbf{X}^T \mathbf{X})^{-1} x]$
 $= \sigma^2 x^T \mathbb{E}_{\mathcal{D}} [(\mathbf{X}^T \mathbf{X})^{-1}] x$
- To analyze this, let's assume that $X_i \sim \mathcal{N}(0, \mathbf{I})$ and number of samples, n , is large enough such that $\mathbf{X}^T \mathbf{X} = n\mathbf{I}$ with high probability and $\mathbb{E}[(\mathbf{X}^T \mathbf{X})^{-1}] \simeq \frac{1}{n} \mathbf{I}$, then
 - Variance is $\frac{\sigma^2 x^T x}{n}$, and decreases with increasing sample size n

Bias-Variance Properties of Ridge regression

- Recall: $\hat{\mathbf{w}}_{\text{ridge}} = (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^T \mathbf{y}$
- To analyze bias-variance tradeoff, we need to assume probabilistic generative model: $x_i \sim P_X$, $\mathbf{y} = \mathbf{X}\mathbf{w} + \epsilon$, $\epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$
- The true error at a sample with feature x is

$$\mathbb{E}_{y, \mathcal{D}_{\text{train}} | x} [(y - x^T \hat{\mathbf{w}}_{\text{ridge}})^2 | x]$$

Bias-Variance Properties of Ridge regression

- Recall: $\hat{\mathbf{w}}_{\text{ridge}} = (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^T \mathbf{y}$
- To analyze bias-variance tradeoff, we need to assume probabilistic generative model: $x_i \sim P_X$, $\mathbf{y} = \mathbf{X}\mathbf{w} + \epsilon$, $\epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$
- The true error at a sample with feature x is

$$\begin{aligned} & \mathbb{E}_{\mathbf{y}, \mathcal{D}_{\text{train}} | x} [(y - x^T \hat{\mathbf{w}}_{\text{ridge}})^2 | x] \\ &= \underbrace{\mathbb{E}_{y|x} [(y - \mathbb{E}[y|x])^2 | x]}_{\text{Irreducible Error}} + \underbrace{\mathbb{E}_{\mathcal{D}_{\text{train}}} [(\mathbb{E}[y|x] - x^T \hat{\mathbf{w}}_{\text{ridge}})^2 | x]}_{\text{Learning Error}} \end{aligned}$$

Bias-Variance Properties of Ridge regression

- Recall: $\hat{\mathbf{w}}_{\text{ridge}} = (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^T \mathbf{y}$
- To analyze bias-variance tradeoff, we need to assume probabilistic generative model: $x_i \sim P_X$, $\mathbf{y} = \mathbf{X}\mathbf{w} + \epsilon$, $\epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$
- The true error at a sample with feature x is

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

Bias-Variance Properties of Ridge regression

- Recall: $\hat{w}_{\text{ridge}} = (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^T \mathbf{y}$
- To analyze bias-variance tradeoff, we need to assume probabilistic generative model: $x_i \sim P_X$, $\mathbf{y} = \mathbf{X}w + \epsilon$, $\epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$
- The true error at a sample with feature x is

$$\mathbb{E}_{y, \mathcal{D}_{\text{train}} | x} [(y - x^T \hat{w}_{\text{ridge}})^2 | x]$$

$$= \mathbb{E}_{y|x} [(y - \mathbb{E}[y | x])^2 | x] + \mathbb{E}_{\mathcal{D}_{\text{train}}} [(\mathbb{E}[y | x] - x^T \hat{w}_{\text{ridge}})^2 | x]$$

$$= \mathbb{E}_{y|x} [(y - x^T w)^2 | x] + \mathbb{E}_{\mathcal{D}_{\text{train}}} [(x^T w - x^T \hat{w}_{\text{ridge}})^2 | x]$$

$$= \underbrace{\sigma^2}_{\text{Irreduc. Error}} + \underbrace{(x^T w - \mathbb{E}_{\mathcal{D}_{\text{train}}} [x^T \hat{w}_{\text{ridge}} | x])^2}_{\text{Bias-squared}} + \underbrace{\mathbb{E}_{\mathcal{D}_{\text{train}}} [(\mathbb{E}_{\tilde{\mathcal{D}}_{\text{train}}} [x^T \hat{w}_{\text{ridge}} | x] - x^T \hat{w}_{\text{ridge}})^2 | x]}_{\text{Variance}}$$

Irreduc. Error

Bias-squared

Variance

Bias-Variance Properties of Ridge regression

- Recall: $\hat{w}_{\text{ridge}} = (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^T \mathbf{y}$
- To analyze bias-variance tradeoff, we need to assume probabilistic generative model: $x_i \sim P_X$, $\mathbf{y} = \mathbf{X}w + \epsilon$, $\epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$
- The true error at a sample with feature x is

$$\mathbb{E}_{y, \mathcal{D}_{\text{train}} | x} [(y - x^T \hat{w}_{\text{ridge}})^2 | x]$$

$$= \mathbb{E}_{y|x} [(y - \mathbb{E}[y | x])^2 | x] + \mathbb{E}_{\mathcal{D}_{\text{train}}} [(\mathbb{E}[y | x] - x^T \hat{w}_{\text{ridge}})^2 | x]$$

$$= \mathbb{E}_{y|x} [(y - x^T w)^2 | x] + \mathbb{E}_{\mathcal{D}_{\text{train}}} [(x^T w - x^T \hat{w}_{\text{ridge}})^2 | x]$$

$$= \underbrace{\sigma^2}_{\text{Irreduc. Error}} + \underbrace{(x^T w - \mathbb{E}_{\mathcal{D}_{\text{train}}} [x^T \hat{w}_{\text{ridge}} | x])^2}_{\text{Bias-squared}} + \underbrace{\mathbb{E}_{\mathcal{D}_{\text{train}}} [(\mathbb{E}_{\tilde{\mathcal{D}}_{\text{train}}} [x^T \hat{w}_{\text{ridge}} | x] - x^T \hat{w}_{\text{ridge}})^2 | x]}_{\text{Variance}}$$

Suppose $\mathbf{X}^T \mathbf{X} = n\mathbf{I}$, then $\hat{w}_{\text{ridge}} = (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^T (\mathbf{X}w + \epsilon)$

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

Bias-Variance Properties

Suppose $\mathbf{X}^T \mathbf{X} = n\mathbf{I}$, then

$$\hat{w}_{\text{ridge}} = \frac{n}{n + \lambda} w + \frac{1}{n + \lambda} \mathbf{X}^T \epsilon$$

- Recall: $\hat{w}_{\text{ridge}} = (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^T \mathbf{y}$
- To analyze bias-variance tradeoff, we need to assume probabilistic generative model: $x_i \sim P_X$, $\mathbf{y} = \mathbf{X}w + \epsilon$, $\epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$

- The true error at a sample with feature x is

$$\mathbb{E}_{y, \mathcal{D}_{\text{train}} | x} [(y - x^T \hat{w}_{\text{ridge}})^2 | x]$$

$$= \mathbb{E}_{y|x} [(y - \mathbb{E}[y | x])^2 | x] + \mathbb{E}_{\mathcal{D}_{\text{train}}} [(\mathbb{E}[y | x] - x^T \hat{w}_{\text{ridge}})^2 | x]$$

$$= \mathbb{E}_{y|x} [(y - x^T w)^2 | x] + \mathbb{E}_{\mathcal{D}_{\text{train}}} [(x^T w - x^T \hat{w}_{\text{ridge}})^2 | x]$$

$$= \sigma^2 + (x^T w - \mathbb{E}_{\mathcal{D}_{\text{train}}} [x^T \hat{w}_{\text{ridge}} | x])^2 + \mathbb{E}_{\mathcal{D}_{\text{train}}} [(\mathbb{E}_{\tilde{\mathcal{D}}_{\text{train}}} [x^T \hat{w}_{\text{ridge}} | x] - x^T \hat{w}_{\text{ridge}})^2 | x]$$

(verify at home)

$$= \sigma^2 + \frac{\lambda^2}{(n + \lambda)^2} (w^T x)^2 + \frac{\sigma^2 n}{(n + \lambda)^2} \|x\|_2^2$$

Irreduc. Error

Bias-squared

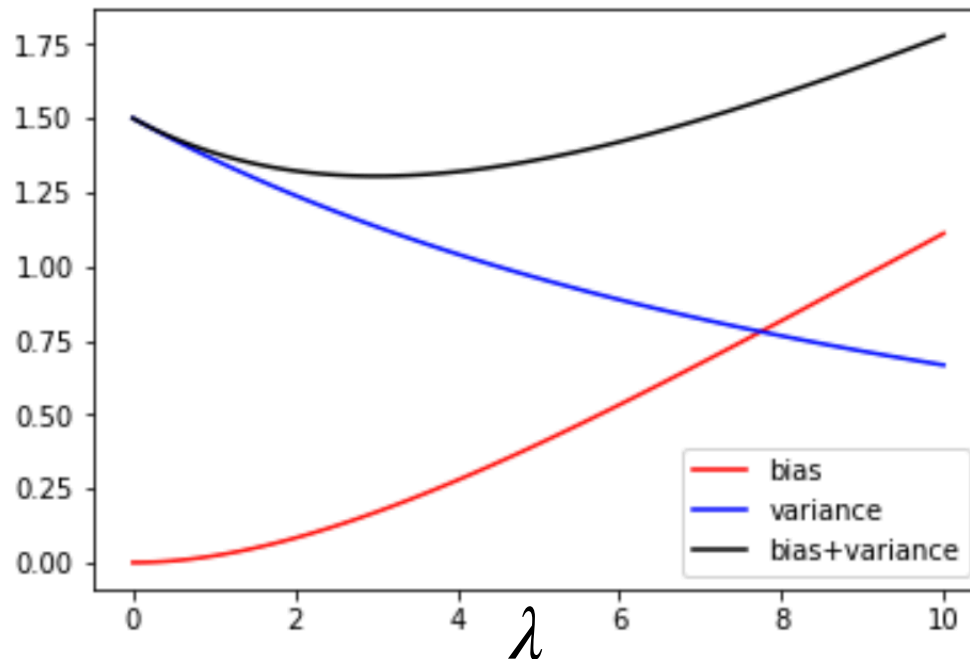
Variance

Bias-Variance Properties of Ridge regression

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

$$\mathbb{E}_{y, \mathcal{D}_{train}|x} [(y - x^T \hat{w}_{ridge})^2 | x] = \underbrace{\sigma^2 + \frac{\lambda^2}{(n + \lambda)^2} (w^T x)^2}_{\text{Bias-squared}} + \underbrace{\frac{\sigma^2 n}{(n + \lambda)^2} \|x\|_2^2}_{\text{Variance}}$$

$$d=10, n=20, \sigma^2 = 3.0, \|w\|_2^2 = 10$$



as $\lambda \rightarrow 0$,

$$\hat{w}_{ridge} \rightarrow \hat{w}_{LS}$$

as $\lambda \rightarrow \infty$

$$\hat{w}_{ridge} \rightarrow 0$$

What you need to know...

> Regularization

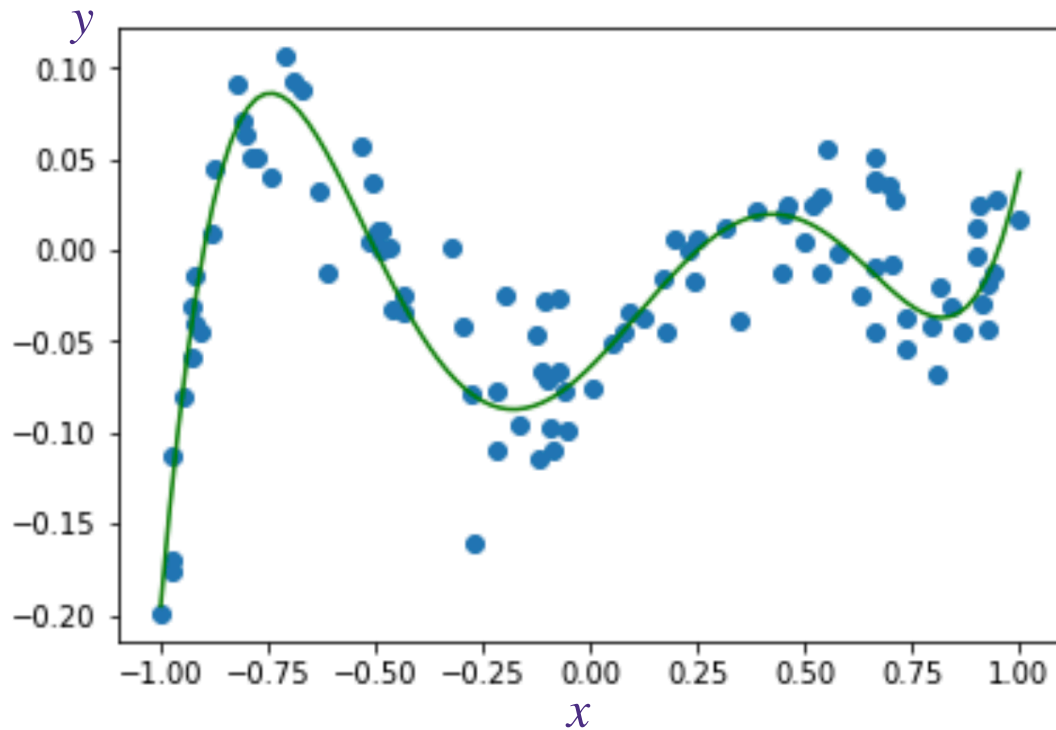
- Penalizes complex models towards preferred, simpler models

> Ridge regression

- L_2 penalized least-squares regression
- Regularization parameter trades off model complexity with training error
- Never regularize the offset!

Example: piecewise linear fit

- we fit a linear model:
$$f(x) = b + w_1h_1(x) + w_2h_2(x) + w_3h_3(x) + w_4h_4(x) + w_5h_5(x)$$
- with a specific choice of features using piecewise linear functions

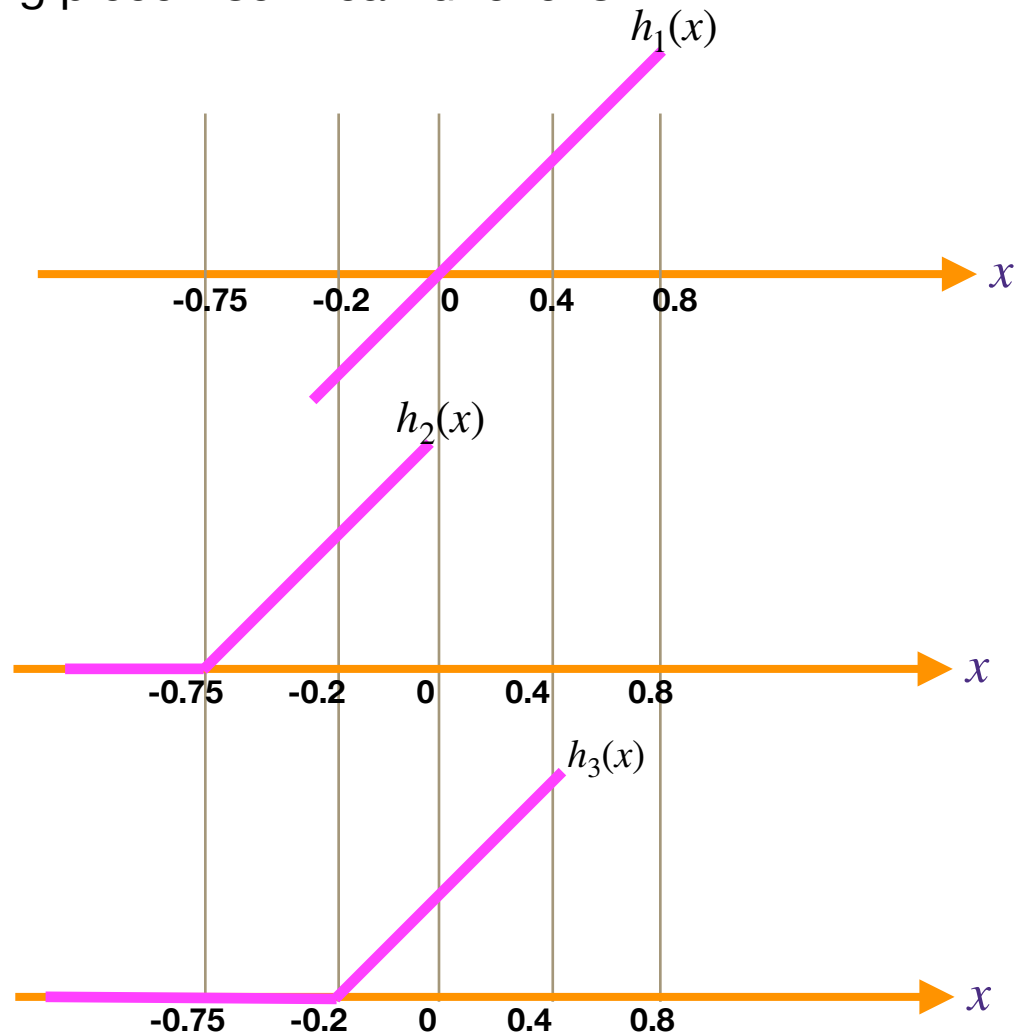


Example: piecewise linear fit

- we fit a linear model:
$$f(x) = b + w_1 h_1(x) + w_2 h_2(x) + w_3 h_3(x) + w_4 h_4(x) + w_5 h_5(x)$$
- with a specific choice of features using piecewise linear functions

$$h(x) = \begin{bmatrix} h_1(x) \\ h_2(x) \\ h_3(x) \\ h_4(x) \\ h_5(x) \end{bmatrix} = \begin{bmatrix} x \\ [x + 0.75]^+ \\ [x + 0.2]^+ \\ [x - 0.4]^+ \\ [x - 0.8]^+ \end{bmatrix}$$

$$[a]^+ \triangleq \max\{a, 0\}$$

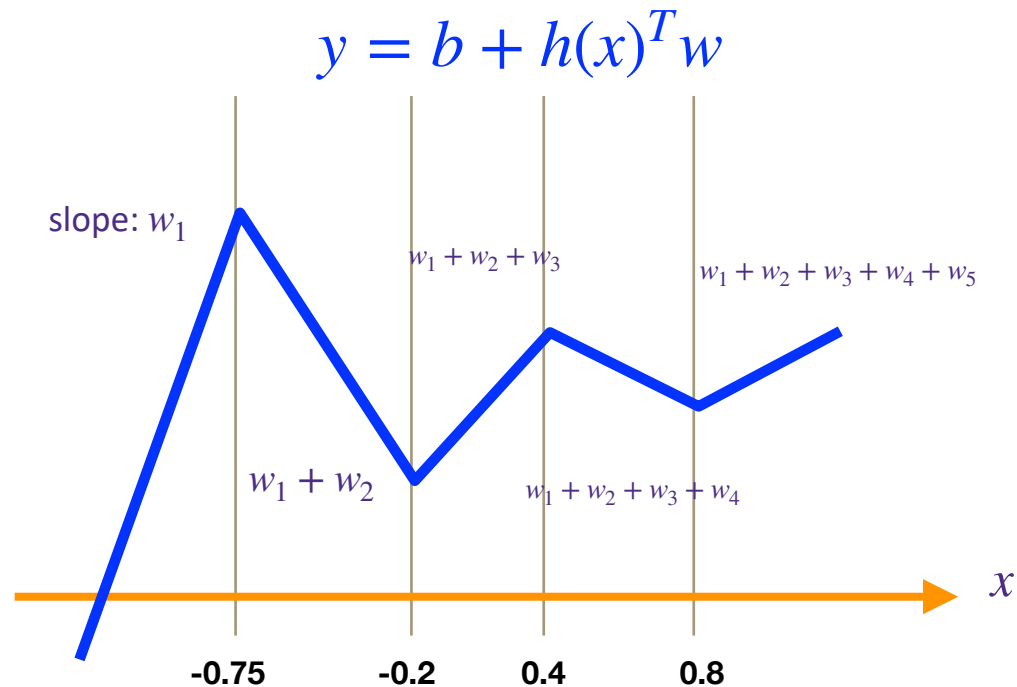


Example: piecewise linear fit

- we fit a linear model:
 $f(x) = b + w_1 h_1(x) + w_2 h_2(x) + w_3 h_3(x) + w_4 h_4(x) + w_5 h_5(x)$
- with a specific choice of features using piecewise linear functions

$$h(x) = \begin{bmatrix} h_1(x) \\ h_2(x) \\ h_3(x) \\ h_4(x) \\ h_5(x) \end{bmatrix} = \begin{bmatrix} x \\ [x + 0.75]^+ \\ [x + 0.2]^+ \\ [x - 0.4]^+ \\ [x - 0.8]^+ \end{bmatrix}$$

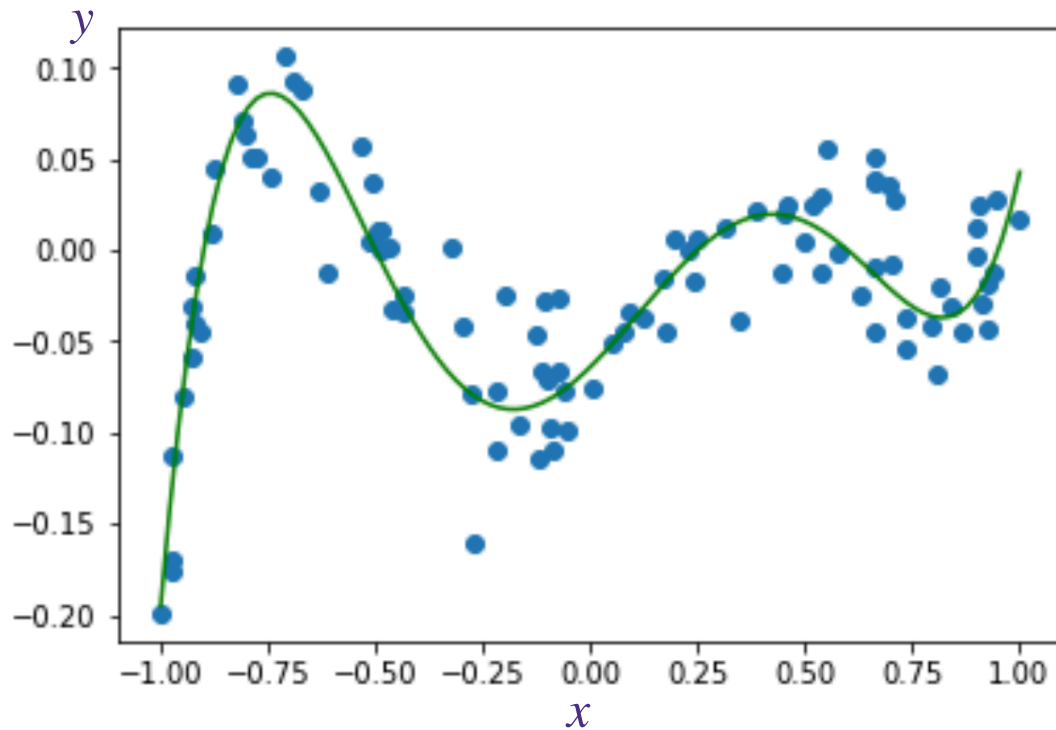
$$[a]^+ \triangleq \max\{a, 0\}$$



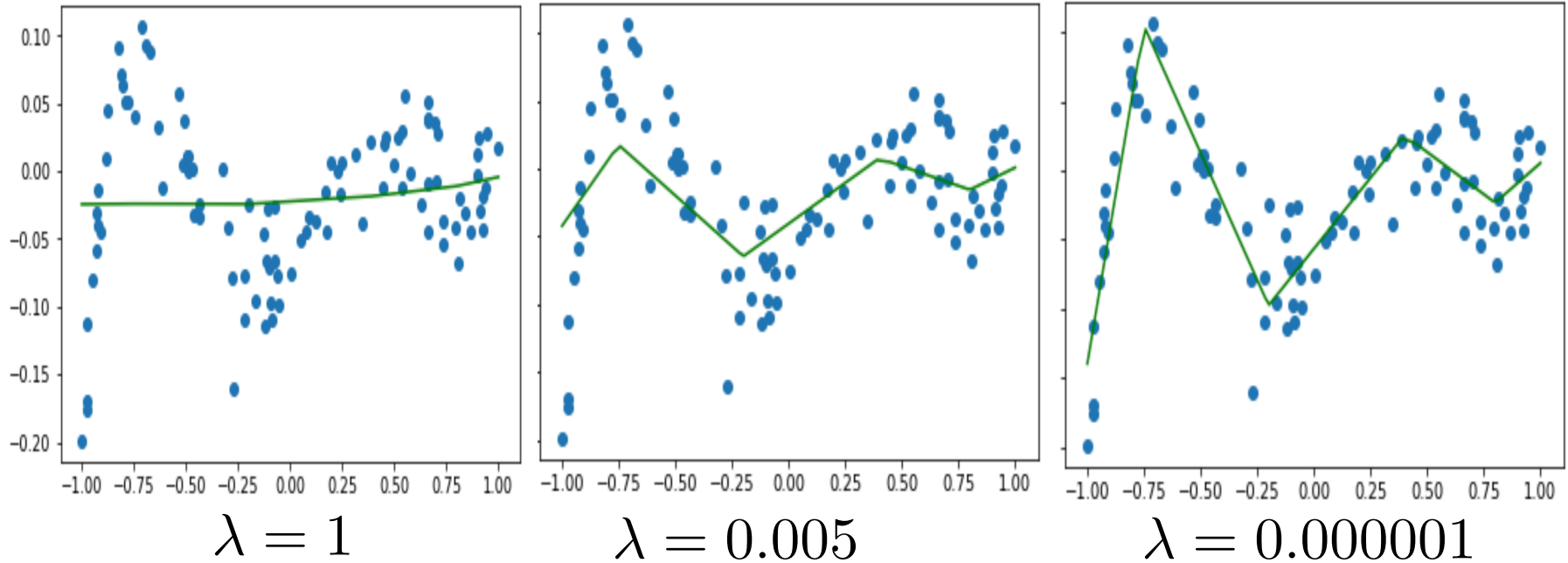
the weights capture the change in the slopes

Example: piecewise linear fit

- we fit a linear model:
$$f(x) = b + w_1h_1(x) + w_2h_2(x) + w_3h_3(x) + w_4h_4(x) + w_5h_5(x)$$
- with a specific choice of features using piecewise linear functions

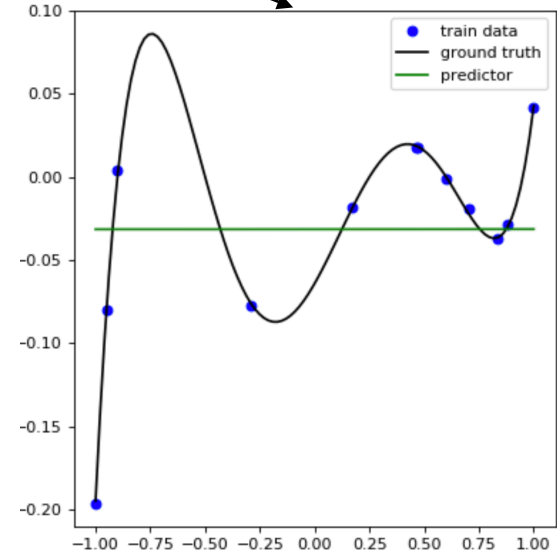
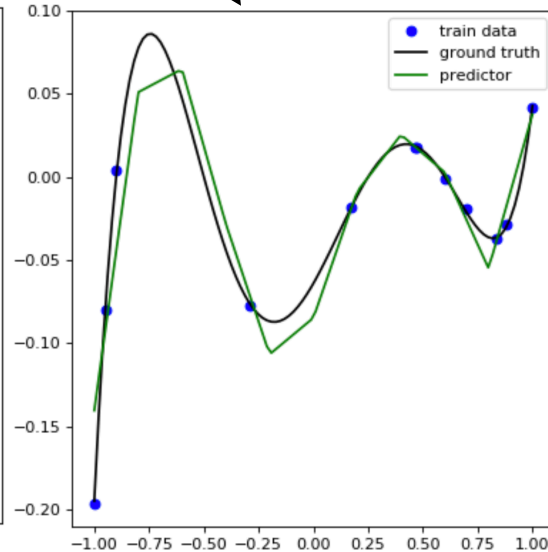
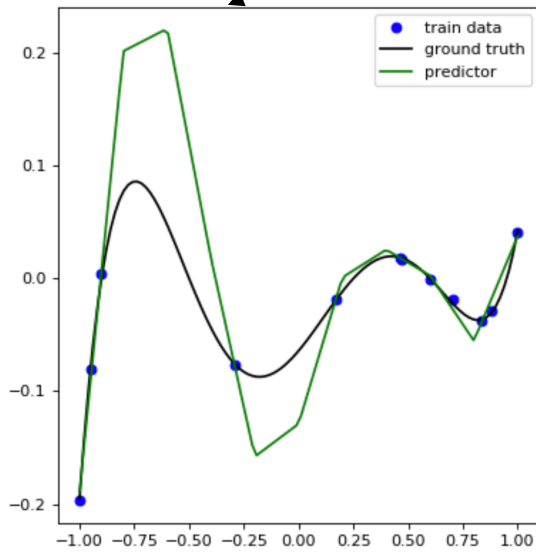
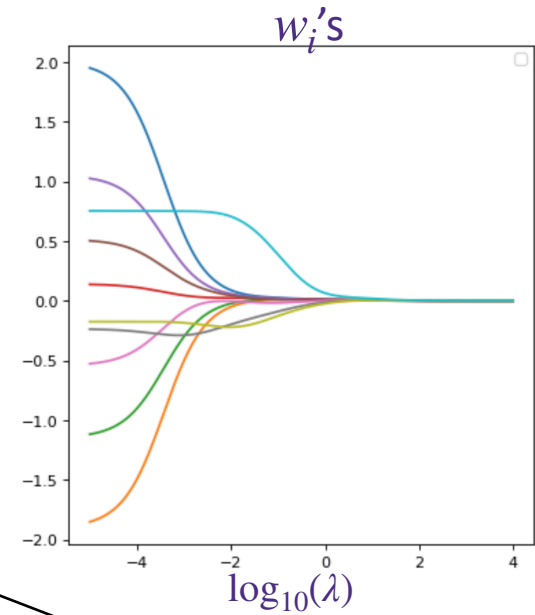
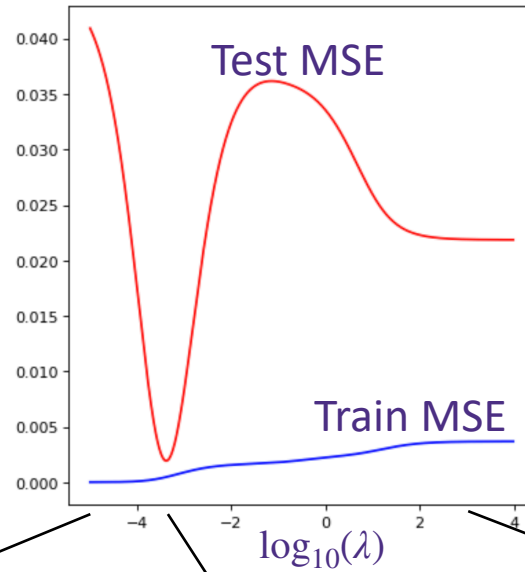


Example: piecewise linear fit (ridge regression)



We do not observe overfitting, as $d=5$ and $n=100$

Piecewise linear with $w \in \mathbb{R}^{10}$ and $n=11$ samples

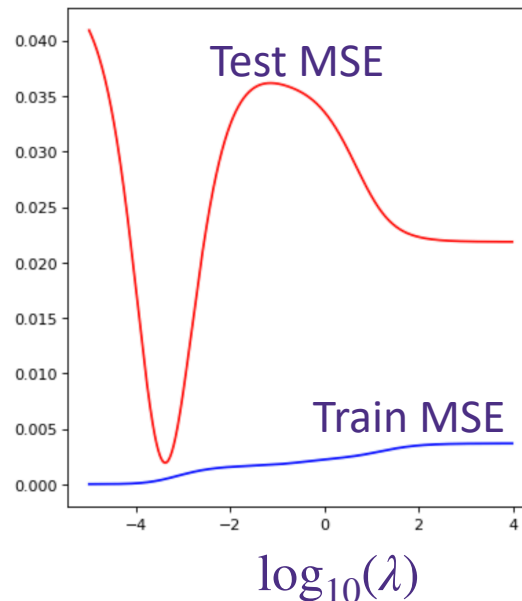


Model selection using Cross-validation



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

- > Ridge regression:
How do we pick the regularization constant λ ...
- > Polynomial features:
How do we pick the number of basis functions...
- > We could use the test data, but...



(LOO) Leave-one-out cross validation

- > Consider a validation set with 1 example:
 - \mathcal{D} : training data
 - $\mathcal{D} \setminus j$: training data with j -th data point (x_j, y_j) moved to validation set
- > Learn model $f_{\mathcal{D} \setminus j}$ with $\mathcal{D} \setminus j$ dataset
- > The squared error on predicting y_j : $(y_j - f_{\mathcal{D} \setminus j}(x_j))^2$

is an unbiased estimate of the **true error**

$$\text{error}_{\text{true}}(f_{\mathcal{D} \setminus j}) = \mathbb{E}_{(x,y) \sim P_{x,y}} [(y - f_{\mathcal{D} \setminus j}(x))^2]$$

but, variance of $(y_j - f_{\mathcal{D} \setminus j}(x_j))^2$ is too large

(LOO) Leave-one-out cross validation

- > Consider a validation set with 1 example:
 - \mathcal{D} : training data
 - $\mathcal{D} \setminus j$: training data with j -th data point (x_j, y_j) moved to validation set
- > Learn model $f_{\mathcal{D} \setminus j}$ with $\mathcal{D} \setminus j$ dataset

> The squared error on predicting y_j : $(y_j - f_{\mathcal{D} \setminus j}(x_j))^2$

is an unbiased estimate of the **true error**

$$\text{error}_{\text{true}}(f_{\mathcal{D} \setminus j}) = \mathbb{E}_{(x,y) \sim P_{x,y}} [(y - f_{\mathcal{D} \setminus j}(x))^2]$$

but variance of $(y_j - f_{\mathcal{D} \setminus j}(x_j))^2$ is too large, so instead

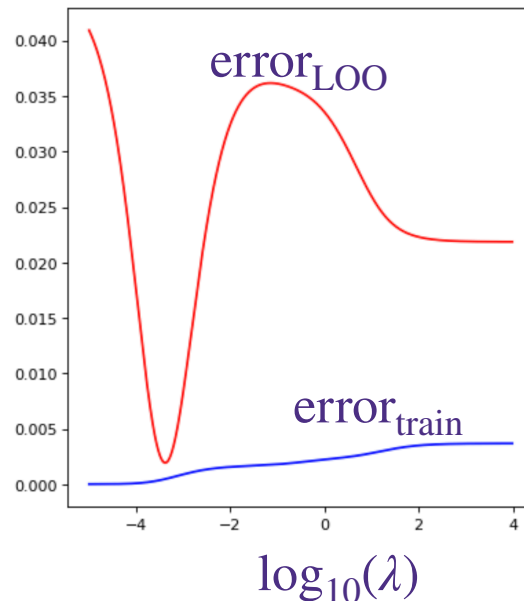
- > **LOO cross validation:** Average over all data points j :
 - Train n times:
for each data point you leave out, learn a new classifier $f_{\mathcal{D} \setminus j}$

- **Estimate the true error as:**

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

LOO cross validation is (almost) unbiased estimate!

- > When computing LOOCV error, we only use $n - 1$ data points to train
 - So it's not estimate of true error of learning with n data points
 - Usually pessimistic – learning with less data typically gives worse answer. (Leads to an over estimation of the error)
- > 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
- > say, you implemented a fast 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

$$\mathcal{D} = \mathcal{D}_1 \mathcal{D}_2 \mathcal{D}_3 \mathcal{D}_4 \mathcal{D}_5$$

> For each i

- Learn model $f_{\mathcal{D} \setminus \mathcal{D}_i}$ using data point not in \mathcal{D}_i

- Estimate error of $f_{\mathcal{D} \setminus \mathcal{D}_i}$ on validation set \mathcal{D}_i :

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

$$f_{\mathcal{D} \setminus \mathcal{D}_3}$$

Train	Train	Validation	Train	Train
-------	-------	------------	-------	-------

>

>

Use k -fold cross validation

> Randomly divide training data into k equal parts

- $\mathcal{D}_1, \dots, \mathcal{D}_k$

> For each i

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

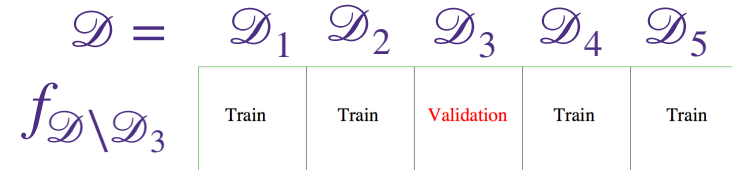
$$\text{error}_{\mathcal{D}_i} = \frac{1}{|\mathcal{D}_i|} \sum_{(x_j, y_j) \in \mathcal{D}_i} (y_j - f_{\mathcal{D} \setminus \mathcal{D}_i}(x_j))^2$$

> 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 as $k \ll n$
- More (pessimistically) biased – using much less data, only $n - \frac{n}{k}$
- Usually, $k = 10$

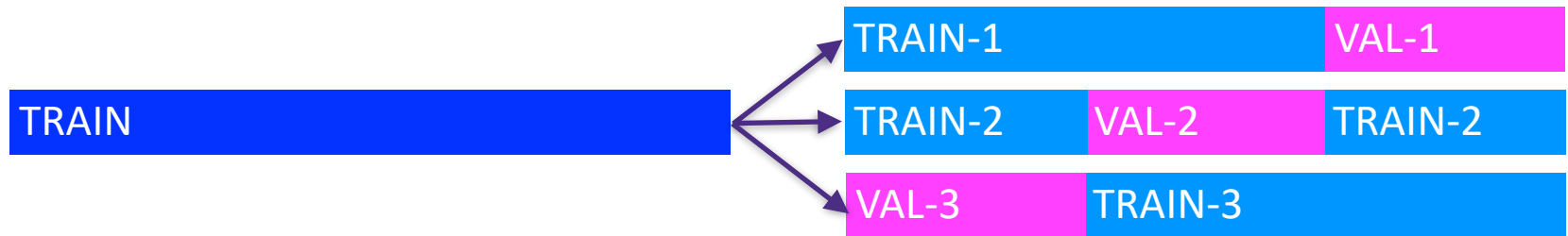


Recap

- > Given a dataset, begin by splitting into



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



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

Model selection using cross validation

> **For** $\lambda \in \{0.001, 0.01, 0.1, 1, 10\}$

> **For** $j \in \{1, \dots, k\}$

>

$$\hat{w}_{\lambda, \text{Train}-j} \leftarrow \arg \min_w \sum_{i \in \text{Train}-j} (y_i - w^T x_i)^2 + \lambda \|w\|_2^2$$

>

$$\hat{\lambda} \leftarrow \arg \min_{\lambda} \frac{1}{k} \sum_{j=1}^k \sum_{i \in \text{Val}-j} (y_i - \hat{w}_{\lambda, \text{Train}-j}^T x_i)^2$$

Example 1

- > You wish to predict the stock price of zoom.us given historical stock price data y_i 's (for each i -th day) and the historical news articles x_i 's
- > You use all daily stock price up to Jan 1, 2020 as **TRAIN** and Jan 2, 2020 - April 13, 2020 as **TEST**
- > What's wrong with this procedure?

Example 2

- > 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 with the training set 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