Least Squares

Pontus Giselsson

Outline

- **Supervised learning – Overview**
- Least squares – Basics
- Nonlinear features
- Generalization, overfitting, and regularization
- Cross validation
- Feature selection
- Training problem properties

Machine learning

- Machine learning can very roughly be divided into:
    - Supervised learning
    - Unsupervised learning
    - Semisupervised learning (between supervised and unsupervised)
    - Reinforcement learning
- We will focus on supervised learning

Supervised learning

- Let (x, y) represent object and label pairs
  - Object $x \in \mathcal{X} \subseteq \mathbb{R}^n$
  - Label $y \in \mathcal{Y} \subseteq \mathbb{R}^K$
- Available: Labeled training data (training set) $\{(x_i, y_i)\}_{i=1}^N$
  - Data $x_i \in \mathbb{R}^n$, or *examples* (often n large)
  - Labels $y_i \in \mathbb{R}^K$, or *response variables* (often $K = 1$)

Objective: Find a model (function) $m(x)$:

- that takes data (example, object) x as input
- and predicts corresponding label (response variable) y

How?:

- learn m from training data, but should *generalize* to all (x, y)

Relation to optimization

Training the “machine” m consists in solving optimization problem

Regression vs Classification

There are two main types of supervised learning tasks:

- Regression:
  - Predicts quantities
  - Real-valued labels $y \in \mathcal{Y} = \mathbb{R}^K$ (will mainly consider $K = 1$)
- Classification:
  - Predicts class belonging
  - Finite number of class labels, e.g., $y \in \mathcal{Y} = \{1, 2, \ldots, k\}$

Examples of data and label pairs

Data	Label	R/C
text in email	spam?	C
dna	blood cell concentration	R
dna	cancer?	C
image	cat or dog	C
advertisement display	click?	C
image of handwritten digit	digit	C
house address	selling cost	R
stock	price	R
sport analytics	winner	C
speech representation	spoken word	C

R/C is for regression or classification

In this course

Lectures will cover different supervised learning methods:

- Classical methods with convex training problems
  - Least squares (this lecture)
  - Logistic regression
  - Support vector machines
- Deep learning methods with nonconvex training problem

Highlight difference:

- Deep learning (specific) nonlinear model instead of linear

Notation

- (Primal) Optimization variable notation:
  - Optimization literature: x, y, z (as in first part of course)
  - Statistics literature: β
  - Machine learning literature: θ, w, b
- Reason: data, labels in statistics and machine learning are x, y
- Will use machine learning notation in these lectures
- We collect training data in matrices (one example per row)

$$X = \begin{bmatrix} x_1^T \\ \vdots \\ x_N^T \end{bmatrix} \qquad\qquad Y = \begin{bmatrix} y_1^T \\ \vdots \\ y_N^T \end{bmatrix}$$

- Columns X_j of data matrix $X = [X_1, \ldots, X_n]$ are called *features*

Outline

- Supervised learning – Overview
- **Least squares – Basics**
- Nonlinear features
- Generalization, overfitting, and regularization
- Cross validation
- Feature selection
- Training problem properties

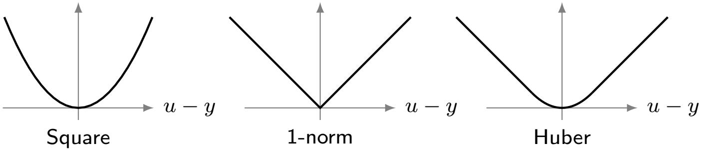
Regression training problem

- Objective: Find data model m such that for all (x, y):

$$m(x) - y \approx 0$$

- Let model output $u = m(x)$; Examples of data misfit losses

$$L(u, y) = \frac{1}{2}(u - y)^2$$
$$L(u, y) = |u - y|$$
$$L(u, y) = \begin{cases} \frac{1}{2}(u - y)^2 & \text{if } |u - v| \leq c \\ c(|u - y| - c/2) & \text{else} \end{cases}$$



- Training: find model m that minimizes sum of training set losses

$$\underset{m}{\text{minimize}} \sum_{i=1}^{N} L(m(x_i), y_i)$$

Supervised learning – Least squares

- Parameterize model m and set a linear (affine) structure

$$m(x;\theta) = w^T x + b$$

  where $\theta = (w, b)$ are *parameters* (also called *weights*)

- Training: find model parameters that minimize training cost

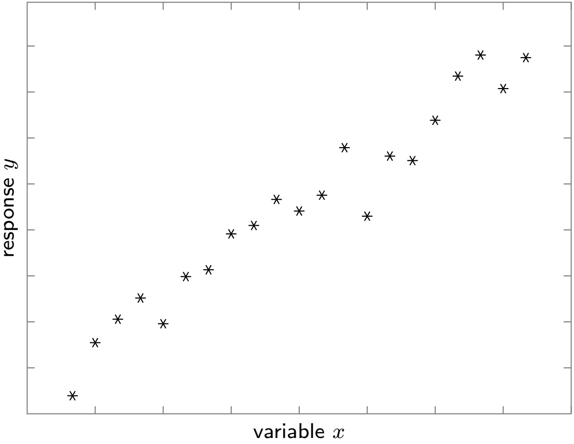
$$\underset{\theta}{\text{minimize}} \sum_{i=1}^{N} L(m(x_i;\theta), y_i) = \tfrac{1}{2}\sum_{i=1}^{N}(w^T x_i + b - y_i)^2$$

  (note: optimization over model *parameters* θ)

- Once trained, predict response of new input x as $\hat{y} = w^T x + b$

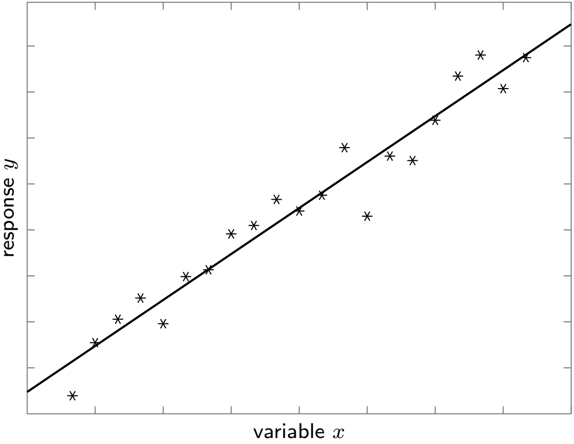
Example – Least squares

- Find affine function parameters that fit data:



Example – Least squares

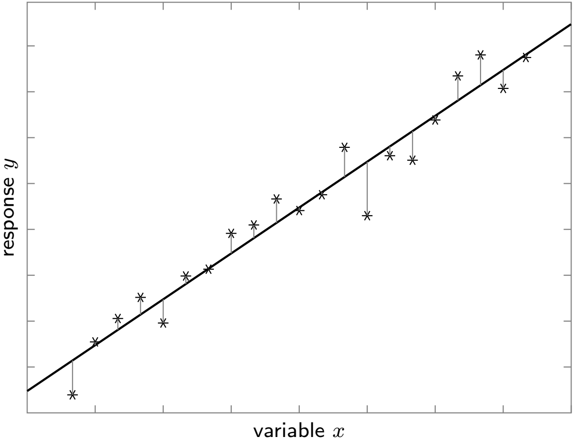
- Find affine function parameters that fit data:



- Data points (x, y) marked with ($*$), LS model $wx + b$ (——)

Example – Least squares

- Find affine function parameters that fit data:



- Data points (x, y) marked with ($*$), LS model $wx + b$ (——)
- Least squares finds affine function that minimizes squared distance

Solving for constant term

- Constant term b also called *bias term* or *intercept*
- What is optimal b?

$$\underset{w,b}{\text{minimize}}\ \tfrac{1}{2}\sum_{i=1}^{N}(w^T x_i + b - y_i)^2$$

- Optimality condition w.r.t. b (gradient w.r.t. b is 0):

$$0 = Nb + \sum_{i=1}^{N}(w^T x_i - y_i) \quad \Leftrightarrow \quad b = \bar{y} - w^T \bar{x}$$

where $\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$ and $\bar{y} = \frac{1}{N}\sum_{i=1}^{N} y_i$ are mean values

Equivalent problem

- Plugging in optimal $b = \bar{y} - w^T \bar{x}$ in least squares estimate gives

$$\underset{w,b}{\text{minimize}} \, \frac{1}{2} \sum_{i=1}^{N} (w^T x_i + b - y_i)^2 = \frac{1}{2} \sum_{i=1}^{N} (w^T (x_i - \bar{x}) - (y_i - \bar{y}))^2$$

- Let $\tilde{x}_i = x_i - \bar{x}$ and $\tilde{y}_i = y_i - \bar{y}$, then it is equivalent to solve

$$\underset{w}{\text{minimize}} \, \frac{1}{2} \sum_{i=1}^{N} (w^T \tilde{x}_i - \tilde{y}_i)^2 = \frac{1}{2} \| Xw - Y \|_2^2$$

  where X and Y now contain all $\tilde{x}_i$ and $\tilde{y}_i$ respectively
- Obviously $\tilde{x}_i$ and $\tilde{y}_i$ have zero averages (by construction)
- Will often assume averages subtracted from data and responses

Least squares – Solution

- Training problem

$$\underset{w}{\text{minimize}}\, \tfrac{1}{2}\|Xw - Y\|_2^2$$

- Strongly convex if X full column rank
  - Features linearly independent and more examples than features
  - Consequences: X^TX is invertible and solution exists and is unique
- Optimal w satisfies (set gradient to zero)

$$0 = X^TXw - X^TY$$

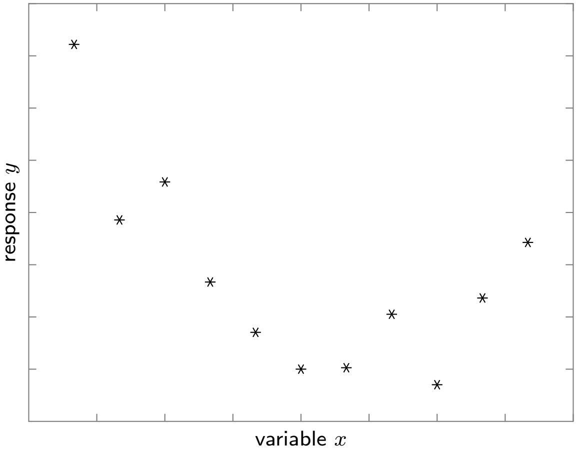
if X full column rank, then unique solution $w = (X^TX)^{-1}X^TY$

Outline

- Supervised learning – Overview
- Least squares – Basics
- **Nonlinear features**
- Generalization, overfitting, and regularization
- Cross validation
- Feature selection
- Training problem properties

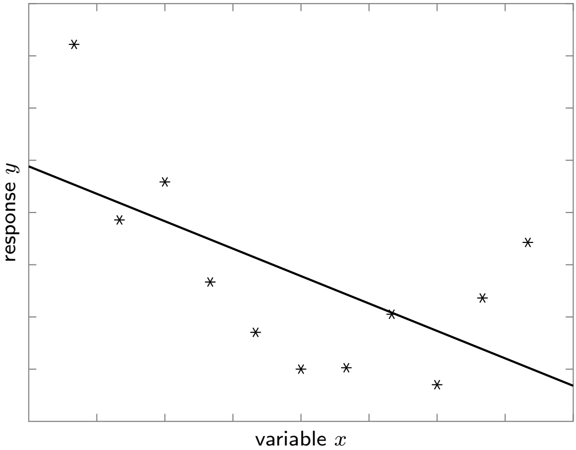
Nonaffine example

- What if data that cannot be well approximated by affine mapping?



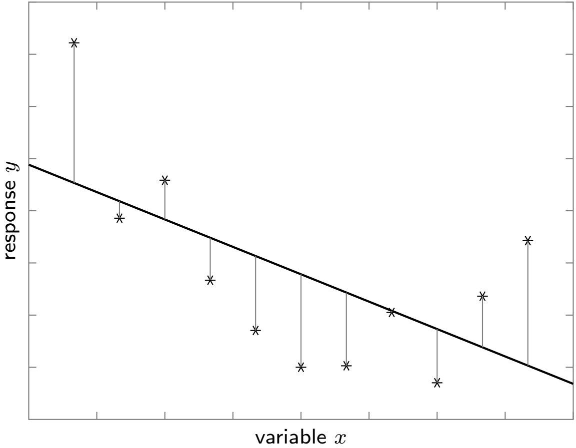
Nonaffine example

- What if data that cannot be well approximated by affine mapping?



Nonaffine example

- What if data that cannot be well approximated by affine mapping?



Adding nonlinear features

- A linear model is not rich enough to model relationship
- Try, e.g., a quadratic model

$$m(x;\theta) = b + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=1}^{i} q_{ij} x_i x_j$$

  where $x = (x_1, \ldots, x_n)$ and parameters $\theta = (b, w, q)$

- For $x \in \mathbb{R}^2$, the model is

$$m(x;\theta) = b + w_1 x_1 + w_2 x_2 + q_{11} x_1^2 + q_{12} x_1 x_2 + q_{22} x_2^2 = \theta^T \phi(x)$$

  where $x = (x_1, x_2)$ and

$$\begin{aligned} \theta &= (b, w_1, w_2, q_{11}, q_{12}, q_{22}) \\ \phi(x) &= (1, x_1, x_2, x_1^2, x_1 x_2, x_2^2) \end{aligned}$$

- Add *nonlinear features* $\phi(x)$, but model still *linear in parameter* θ

Least squares with nonlinear features

- Can, of course, use other nonlinear feature maps ϕ
- Gives models $m(x;\theta) = \theta^T\phi(x)$ with increased fitting capacity
- Use least squares estimate with new model

$$\underset{\theta}{\text{minimize}}\, \tfrac{1}{2}\sum_{i=1}^{N}(m(x_i;\theta) - y_i)^2 = \tfrac{1}{2}\sum_{i=1}^{N}(\theta^T\phi(x_i) - y_i)^2$$

  which is still convex since ϕ does not depend on θ!
- Build new data matrix (with one column per feature in ϕ)

$$X = \begin{bmatrix} \phi(x_1)^T \\ \vdots \\ \phi(x_N)^T \end{bmatrix}$$

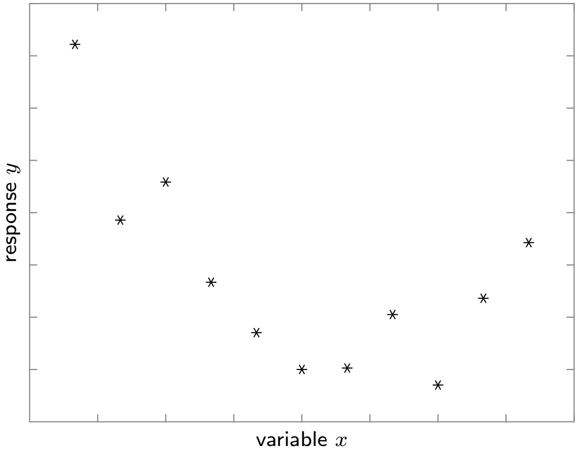
  to arrive at least squares formulation

$$\underset{\theta}{\text{minimize}}\, \tfrac{1}{2}\|X\theta - Y\|_2^2$$

- The more features, the more parameters θ to optimize (lifting)

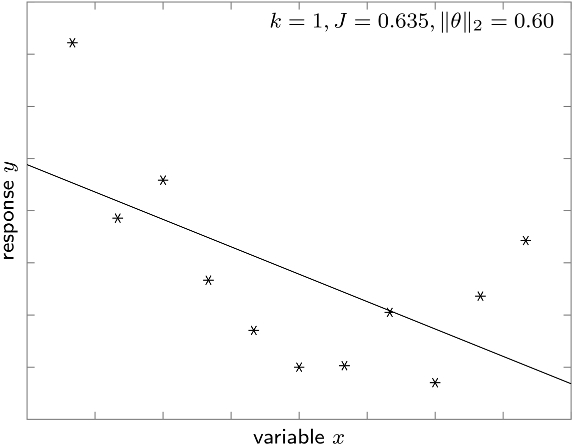
Nonaffine example

- Fit polynomial of degree k to data using LS (J is cost):



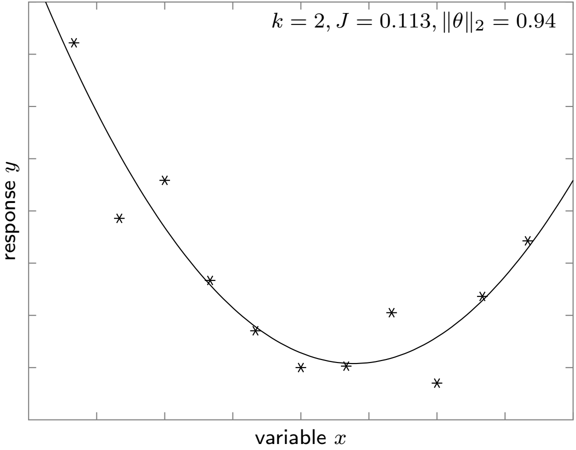
Nonaffine example

- Fit polynomial of degree k to data using LS (J is cost):



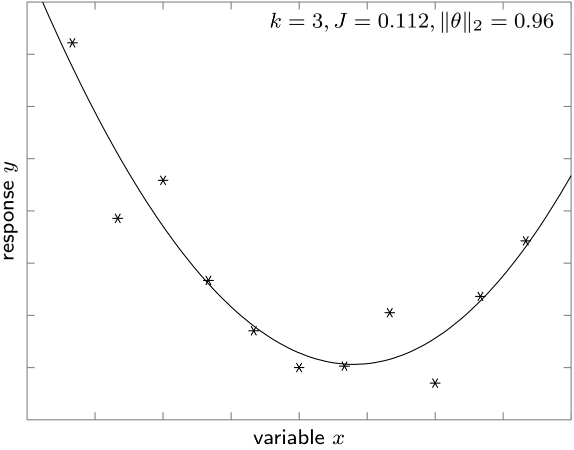
Nonaffine example

- Fit polynomial of degree k to data using LS (J is cost):



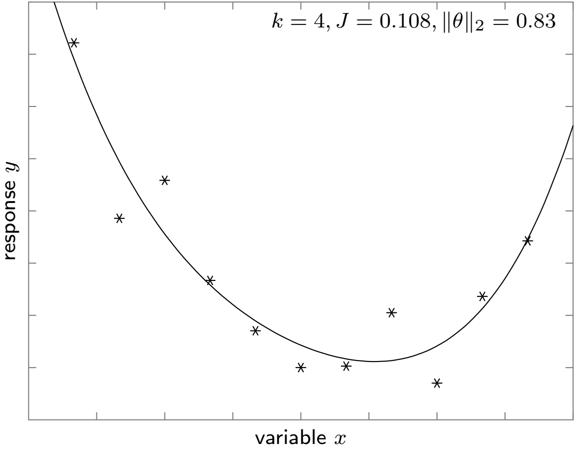
Nonaffine example

- Fit polynomial of degree k to data using LS (J is cost):



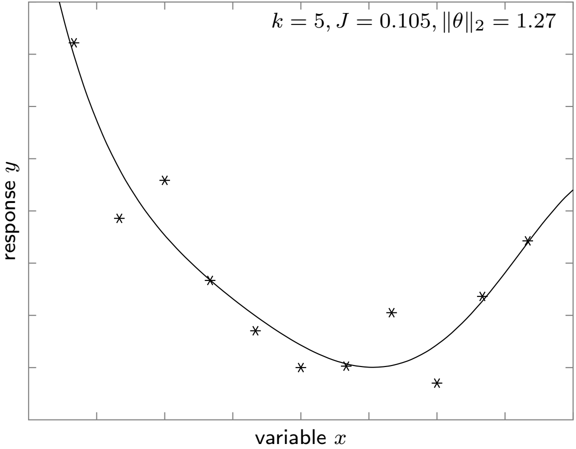
Nonaffine example

- Fit polynomial of degree k to data using LS (J is cost):



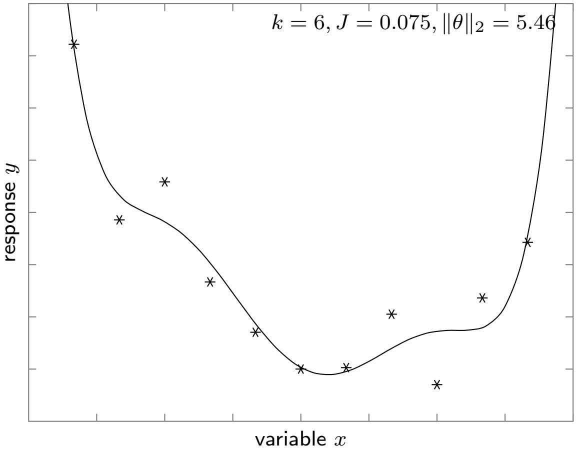
Nonaffine example

- Fit polynomial of degree k to data using LS (J is cost):



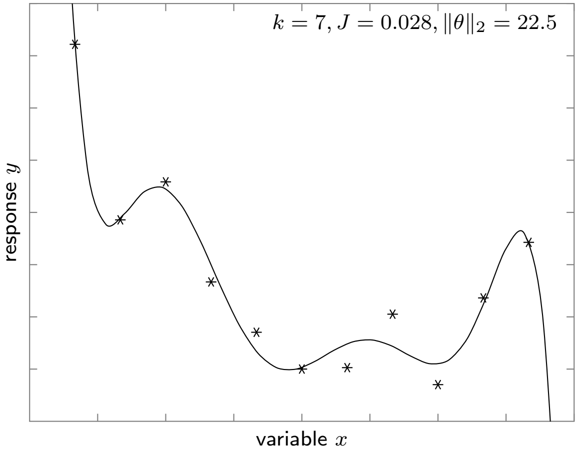
Nonaffine example

- Fit polynomial of degree k to data using LS (J is cost):



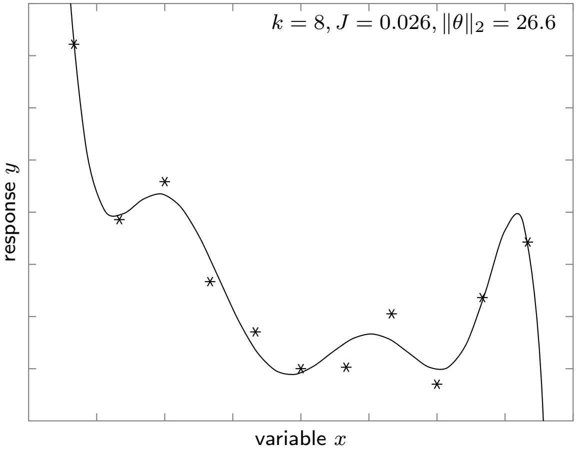
Nonaffine example

- Fit polynomial of degree k to data using LS (J is cost):



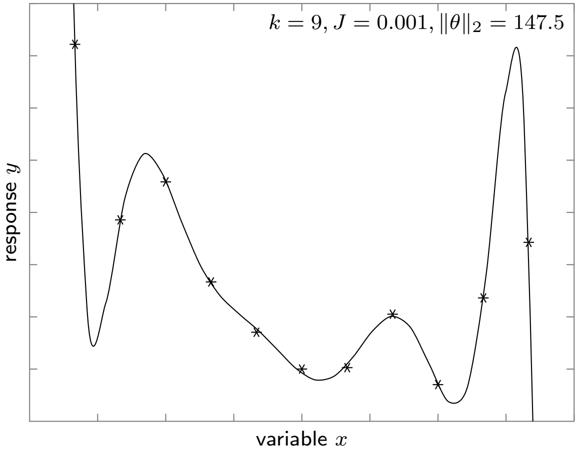
Nonaffine example

- Fit polynomial of degree k to data using LS (J is cost):



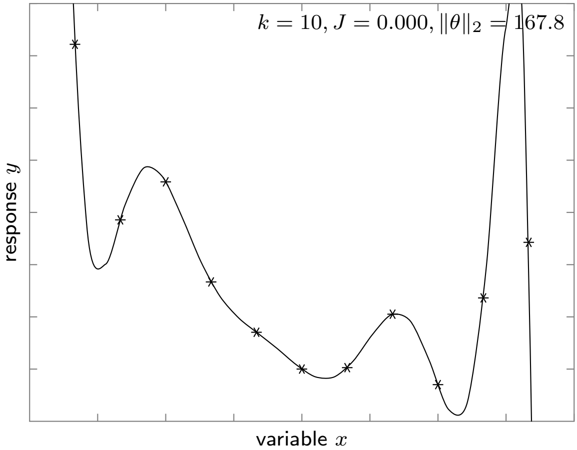
Nonaffine example

- Fit polynomial of degree k to data using LS (J is cost):



Nonaffine example

- Fit polynomial of degree k to data using LS (J is cost):



Outline

- Supervised learning – Overview
- Least squares – Basics
- Nonlinear features
- **Generalization, overfitting, and regularization**
- Cross validation
- Feature selection
- Training problem properties

Generalization and overfitting

- *Generalization*: How well does model perform on unseen data
- *Overfitting*: Model explains training data, but not unseen data
- How to reduce overfitting/improve generalization?

Tikhonov Regularization

- Example indicates: Reducing $\|\theta\|_2$ seems to reduce overfitting
- Least squares with *Tikhonov regularization*:

$$\underset{\theta}{\text{minimize}} \, \frac{1}{2}\|X\theta - Y\|_2^2 + \frac{\lambda}{2}\|\theta\|_2^2$$

- Regularization parameter $\lambda \geq 0$ controls fit vs model expressivity
- Optimization problem called ridge regression in statistics
- (Could regularize with $\|\theta\|_2$, but square easier to solve)
- (Don't regularize b – constant data offset gives different solution)

Ridge Regression – Solution

- Recall ridge regression problem for given λ:

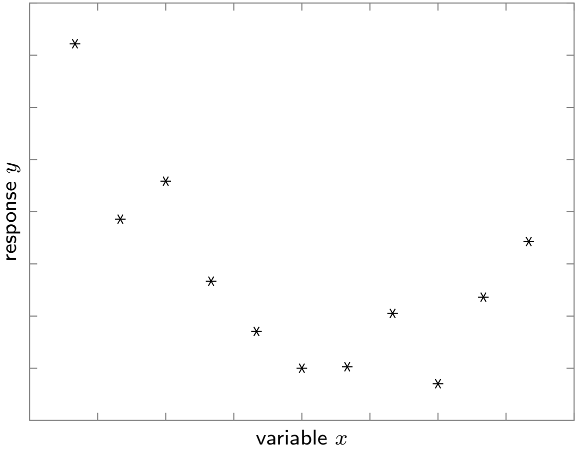
$$\underset{\theta}{\text{minimize}}\, \tfrac{1}{2}\|X\theta - Y\|_2^2 + \tfrac{\lambda}{2}\|\theta\|_2^2$$

- Objective λ-strongly convex for all $\lambda > 0$, hence unique solution
- Objective is differentiable, Fermat's rule:

$$\begin{aligned} 0 = X^T(X\theta - Y) + \lambda\theta \qquad &\iff \qquad (X^TX + \lambda I)\theta = X^TY \\ &\iff \qquad \theta = (X^TX + \lambda I)^{-1}X^TY \end{aligned}$$

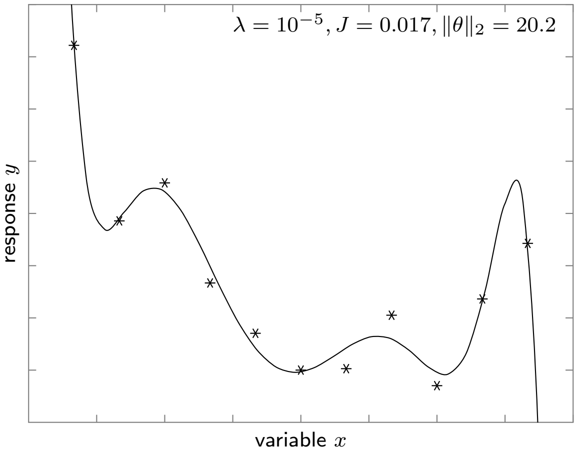
Ridge Regression – Example

- Same problem data as before
- Fit 10-degree polynomial with Tikhonov regularization
- λ: regularization parameter, J LS cost, $\|\theta\|_2$ norm of weights



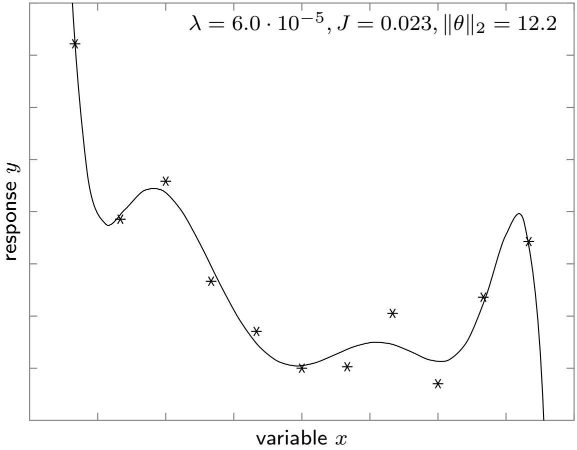
Ridge Regression – Example

- Same problem data as before
- Fit 10-degree polynomial with Tikhonov regularization
- λ: regularization parameter, J LS cost, $\|\theta\|_2$ norm of weights



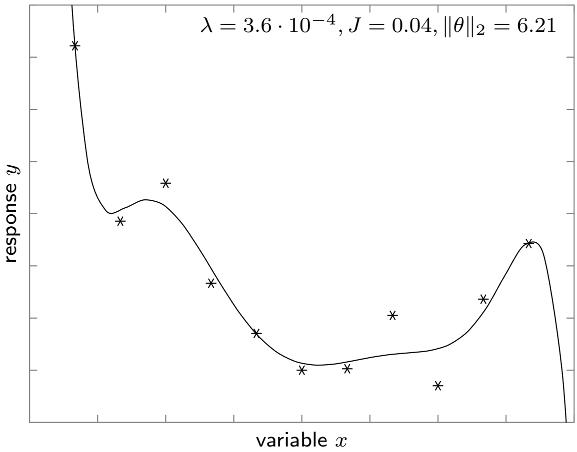
Ridge Regression – Example

- Same problem data as before
- Fit 10-degree polynomial with Tikhonov regularization
- λ: regularization parameter, J LS cost, $\|\theta\|_2$ norm of weights



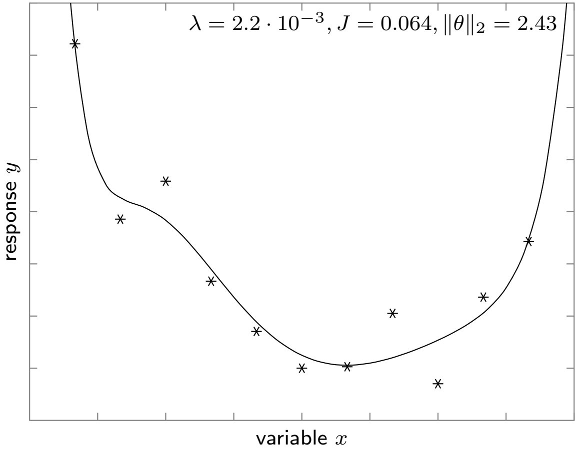
Ridge Regression – Example

- Same problem data as before
- Fit 10-degree polynomial with Tikhonov regularization
- λ: regularization parameter, J LS cost, $\|\theta\|_2$ norm of weights



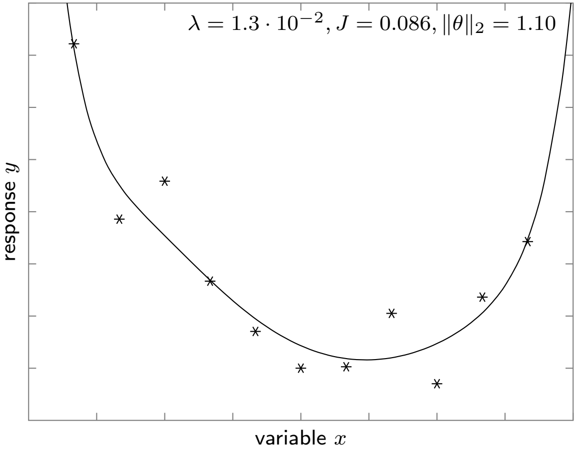
Ridge Regression – Example

- Same problem data as before
- Fit 10-degree polynomial with Tikhonov regularization
- λ: regularization parameter, J LS cost, $\|\theta\|_2$ norm of weights



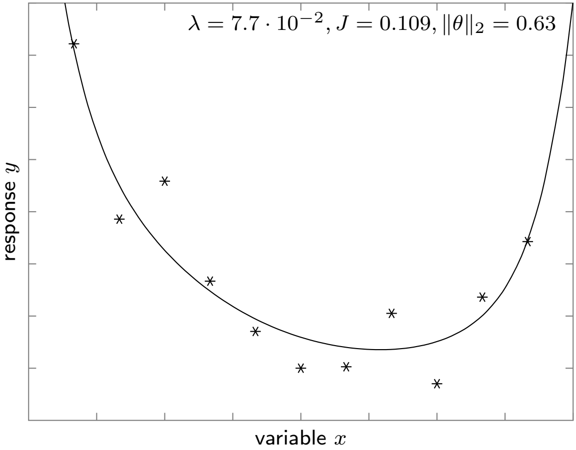
Ridge Regression – Example

- Same problem data as before
- Fit 10-degree polynomial with Tikhonov regularization
- λ: regularization parameter, J LS cost, $\|\theta\|_2$ norm of weights



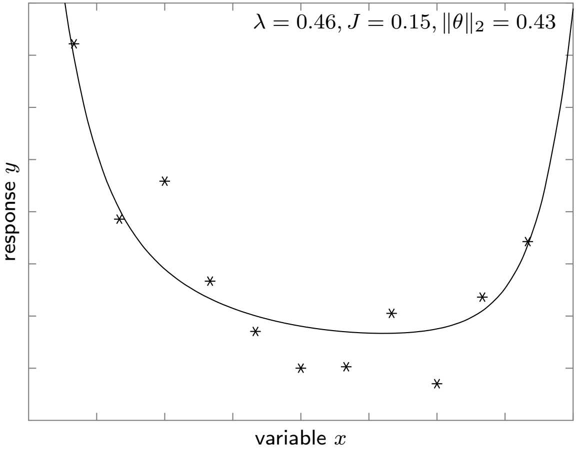
Ridge Regression – Example

- Same problem data as before
- Fit 10-degree polynomial with Tikhonov regularization
- λ: regularization parameter, J LS cost, $\|\theta\|_2$ norm of weights



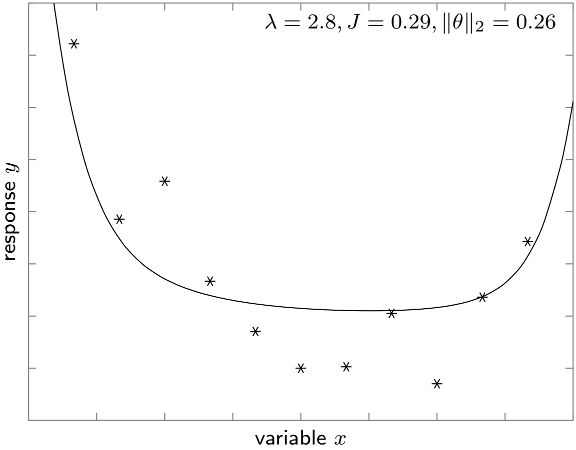
Ridge Regression – Example

- Same problem data as before
- Fit 10-degree polynomial with Tikhonov regularization
- λ: regularization parameter, J LS cost, $\|\theta\|_2$ norm of weights



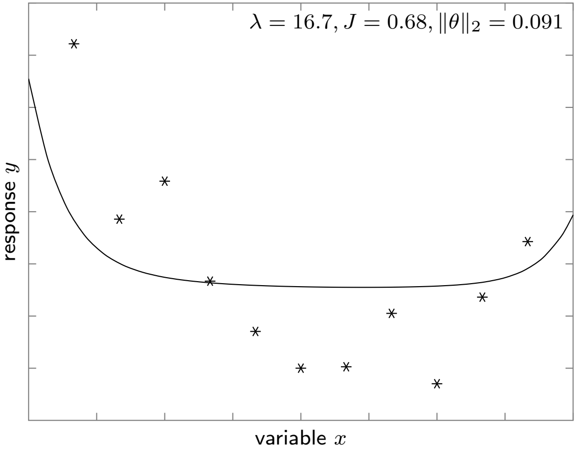
Ridge Regression – Example

- Same problem data as before
- Fit 10-degree polynomial with Tikhonov regularization
- λ: regularization parameter, J LS cost, $\|\theta\|_2$ norm of weights



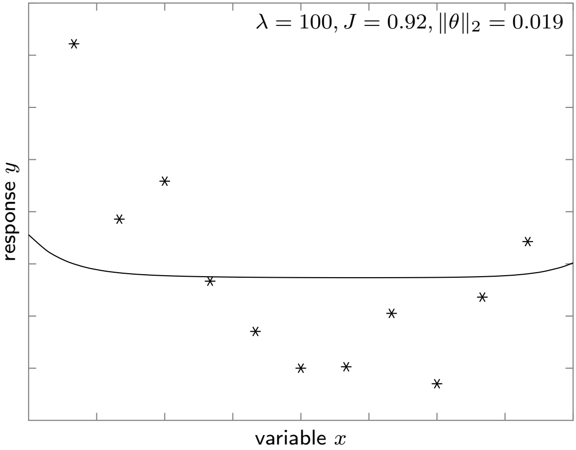
Ridge Regression – Example

- Same problem data as before
- Fit 10-degree polynomial with Tikhonov regularization
- λ: regularization parameter, J LS cost, $\|\theta\|_2$ norm of weights



Ridge Regression – Example

- Same problem data as before
- Fit 10-degree polynomial with Tikhonov regularization
- λ: regularization parameter, J LS cost, $\|\theta\|_2$ norm of weights



Outline

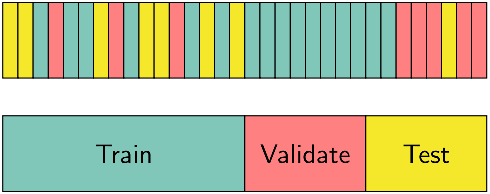
- Supervised learning – Overview
- Least squares – Basics
- Nonlinear features
- Generalization, overfitting, and regularization
- **Cross validation**
- Feature selection
- Training problem properties

Selecting model hyperparameters

- Parameters in machine learning models are called *hyperparameters*
- Ridge model has polynomial order and λ as hyperparameters
- How to select hyperparameters?

Holdout

- Randomize data and assign to train, validate, or test set



Training set:

- Solve training problems with different hyperparameters

Validation set:

- Estimate generalization performance of all trained models
- Use this to select model that seems to generalize best

Test set:

- Final assessment on how chosen model generalizes to unseen data
- *Not* for model selection, then final assessment too optimistic

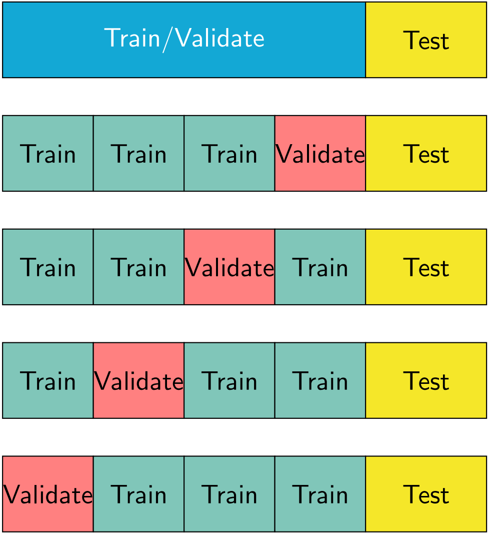
Holdout – Comments

- Typical division between sets 50/25/25 (or 70/20/10)
- Sometimes no test set (then no assessment of final model)
- If no test set, then validation set often called test set
- Can work well if lots of data, if less, use *(k-fold) cross validation*

k-fold cross validation

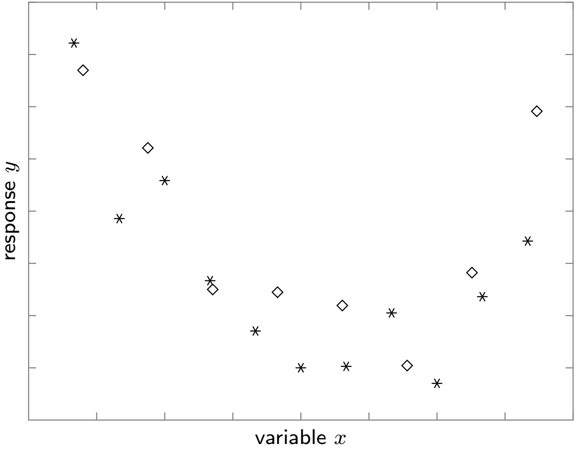
- Similar to hold out – divide first into training/validate and test set
- Divide training/validate set into k data chunks
- Train k models with $k-1$ chunks, use k:th chunk for validation
- Loop
  1. Set hyperparameters and train all k models
  2. Evaluate generalization score on its validation data
  3. Sum scores to get model performance
- Select final model hyperparameters based on best score
- Simpler model with slightly worse score may generalize better
- Estimate generalization performance via test set

4-fold cross validation – Graphics



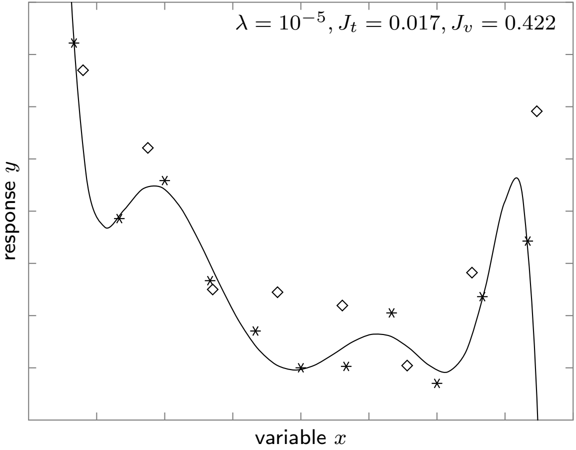
Evaluate generalization score/performance

- Ridge regression example generalization, validation data ($\diamond$)
- λ: regularization parameter, J_t train cost, J_v validation cost



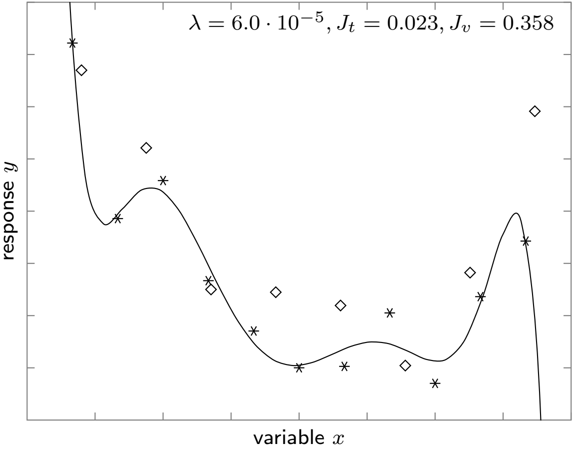
Evaluate generalization score/performance

- Ridge regression example generalization, validation data ($\diamond$)
- λ: regularization parameter, J_t train cost, J_v validation cost



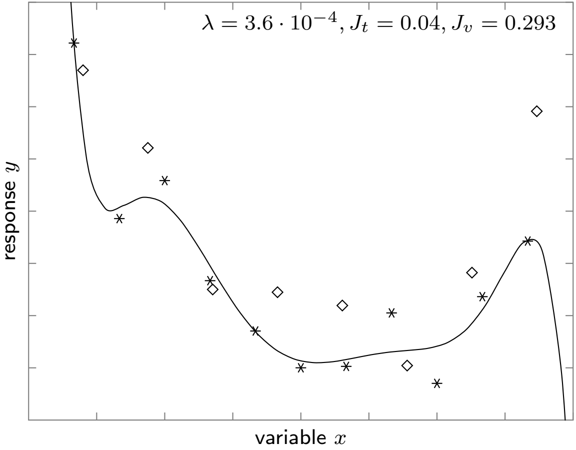
Evaluate generalization score/performance

- Ridge regression example generalization, validation data ($\diamond$)
- λ: regularization parameter, J_t train cost, J_v validation cost



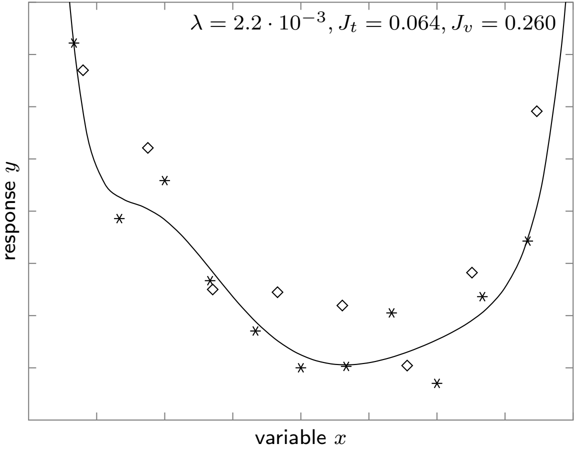
Evaluate generalization score/performance

- Ridge regression example generalization, validation data ($\diamond$)
- λ: regularization parameter, J_t train cost, J_v validation cost



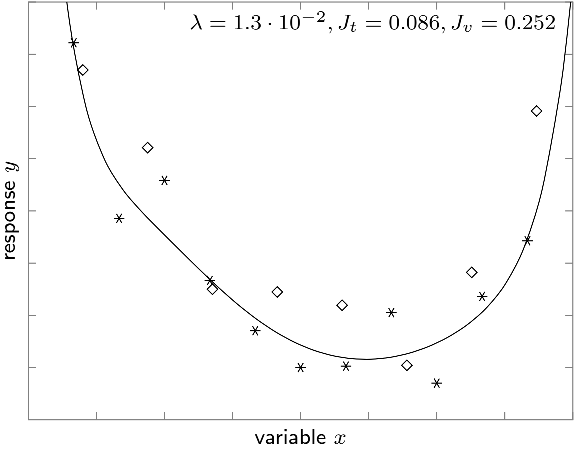
Evaluate generalization score/performance

- Ridge regression example generalization, validation data ($\diamond$)
- λ: regularization parameter, J_t train cost, J_v validation cost



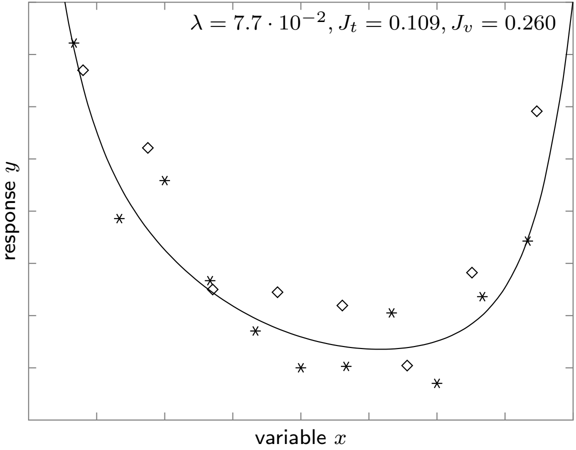
Evaluate generalization score/performance

- Ridge regression example generalization, validation data ($\diamond$)
- λ: regularization parameter, J_t train cost, J_v validation cost



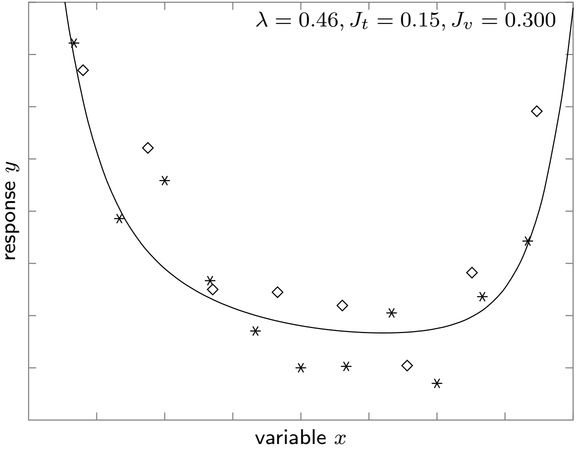
Evaluate generalization score/performance

- Ridge regression example generalization, validation data ($\diamond$)
- λ: regularization parameter, J_t train cost, J_v validation cost



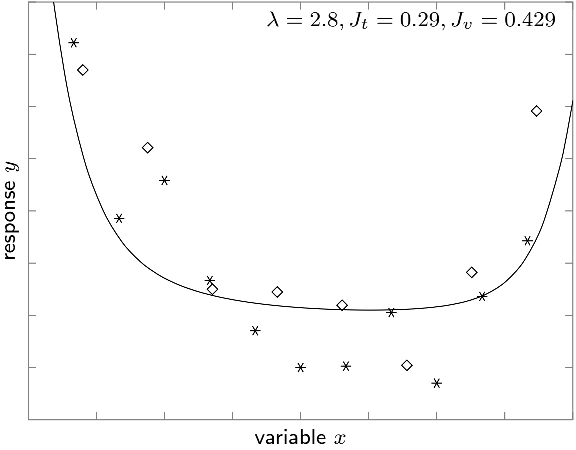
Evaluate generalization score/performance

- Ridge regression example generalization, validation data ($\diamond$)
- λ: regularization parameter, J_t train cost, J_v validation cost



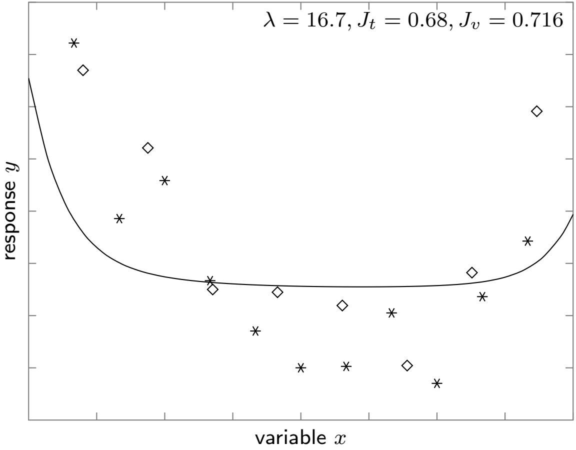
Evaluate generalization score/performance

- Ridge regression example generalization, validation data ($\diamond$)
- λ: regularization parameter, J_t train cost, J_v validation cost



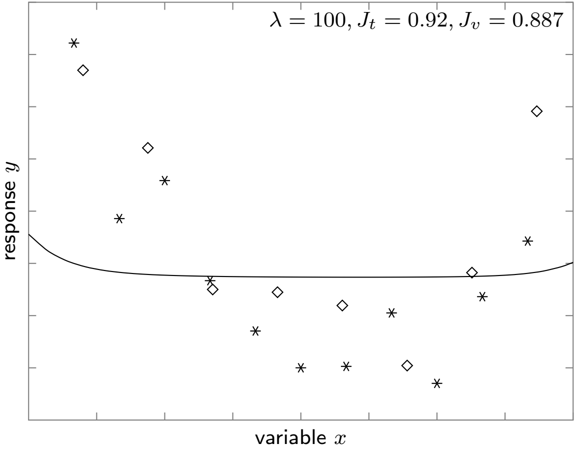
Evaluate generalization score/performance

- Ridge regression example generalization, validation data ($\diamond$)
- λ: regularization parameter, J_t train cost, J_v validation cost



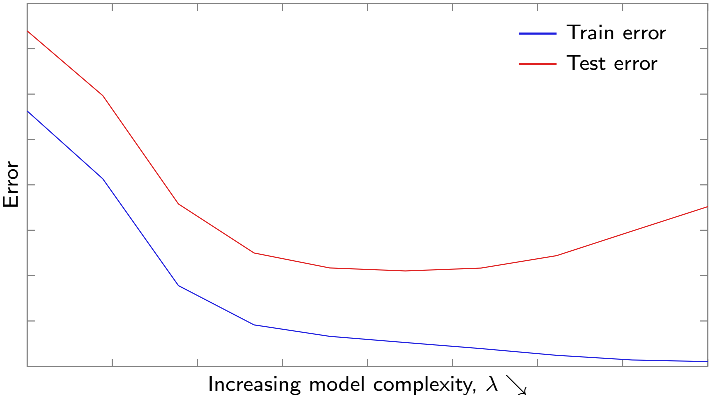
Evaluate generalization score/performance

- Ridge regression example generalization, validation data ($\diamond$)
- λ: regularization parameter, J_t train cost, J_v validation cost



Selecting model

- Average training and test error vs model complexity
- Average training error smaller than average test error
- Large λ (left) model not rich enough
- Small λ (right) model too rich (overfitting)



Outline

- Supervised learning – Overview
- Least squares – Basics
- Nonlinear features
- Generalization, overfitting, and regularization
- Cross validation
- **Feature selection**
- Training problem properties

Feature selection

- Assume $X \in \mathbb{R}^{m \times n}$ with $m < n$ (fewer examples than features)
- Want to find a subset of features that explains data well
- Example: Which genes in genome control eyecolor

Lasso

- Feature selection by regularizing least squares with 1-norm:

$$\underset{w}{\text{minimize}}\ \tfrac{1}{2}\|Xw - Y\|_2^2 + \lambda\|w\|_1$$

- Problem can be written as

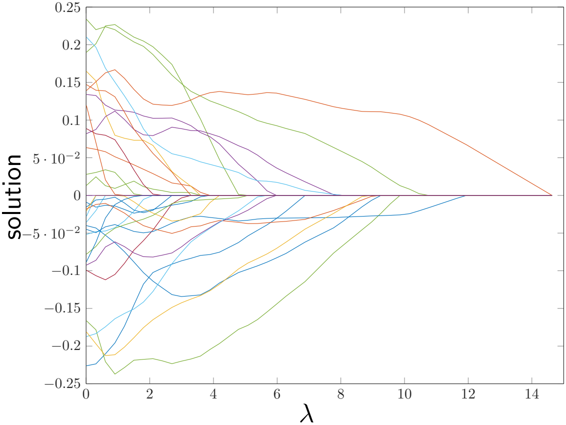
$$\underset{w}{\text{minimize}}\ \tfrac{1}{2}\left\|\sum_{i=1}^{n} w_i X_i - Y\right\|_2^2 + \lambda\|w\|_1$$

  if $w_i = 0$, then feature X_i not important

- The 1-norm promotes sparsity (many 0 variables) in solution
- It also reduces size (shrinks) w (like $\|\cdot\|_2^2$ regularization)
- Problem is called the *Lasso* problem

Example – Lasso

- Data $X \in \mathbb{R}^{30\times 200}$, Lasso solution for different λ



- For large enough λ solution $w = 0$
- More nonzero elements in solution as λ decreases
- For small λ, 30 (nbr examples) nonzero w_i (i.e., 170 $w_i = 0$)

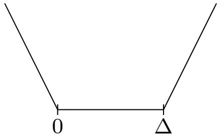
Lasso and correlated features

- Assume two equal features exist, e.g., $X_1 = X_2$, lasso problem is

$$\text{minimize}\ \tfrac{1}{2}\left\|(w_1 + w_2)X_1 + \sum_{i=3}^{n} w_i X_i - Y\right\|_2^2 + \lambda(|w_1| + |w_2| + \|w_{3:n}\|_1)$$

- Assume w^* solves the problem and let $\Delta := w_1^* + w_2^* > 0$ (wlog)
- Then all $w_1 \in [0, \Delta]$ with $w_2 = \Delta - w_1$ solves problem:
  - quadratic cost unchanged since sum $w_1 + w_2$ still Δ
  - the remainder of the regularization part reduces to

$$\min_{w_1} \lambda(|w_1| + |\Delta - w_1|)$$



- For almost correlated features:
  - often only w_1 or w_2 nonzero (the one with slightly better fit)
  - however, features highly correlated, if X_1 explains data so does X_2

Elastic net

- Add Tikhonov regularization to the Lasso

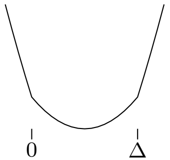
$$\text{minimize}\,\tfrac{1}{2}\|Xw - Y\|^2 + \lambda_1\|w\|_1 + \tfrac{\lambda_2}{2}\|w\|_2^2$$

- This problem is called *elastic net* in statistics
- Can perform better with correlated features

Elastic net and correlated features

- Assume equal features $X_1 = X_2$ and that w^* solves the elastic net
- Let $\Delta := w_1^* + w_2^* > 0$ (wlog), then $w_1^* = w_2^* = \frac{\Delta}{2}$
  - Data fit cost still unchanged for $w_2 = \Delta - w_1$ with $w_1 \in [0, \Delta]$
  - Remaining (regularization) part is

$$\min_{w_1} \lambda_1(|w_1| + |\Delta - w_1|) + \lambda_2(w_1^2 + (\Delta - w_1)^2)$$



which is minimized in the middle at $w_1 = w_2 = \frac{\Delta}{2}$

- For highly correlated features, both (or none) probably selected

Group lasso

- Sometimes want groups of variables to be 0 or nonzero
- Introduce blocks $w = (w_1, \ldots, w_p)$ where $w_i \in \mathbb{R}^{n_i}$
- The group Lasso problem is

$$\text{minimize } \tfrac{1}{2}\|Xw - Y\|_2^2 + \lambda \sum_{i=1}^{p} \|w_i\|_2$$

  (note $\|\cdot\|_2$-norm without square)
- With all $n_i = 1$, it reduces to the Lasso
- Promotes block sparsity, meaning full block $w_i \in \mathbb{R}^{n_i}$ would be 0

Outline

- Supervised learning – Overview
- Least squares – Basics
- Nonlinear features
- Generalization, overfitting, and regularization
- Cross validation
- Feature selection
- **Training problem properties**

Composite optimization

- Least squares problems are convex problems of the form

$$\underset{\theta}{\text{minimize}}\, f(X\theta) + g(\theta),$$

  where
  - $f = \frac{1}{2}\| \cdot - Y\|_2^2$ is data misfit term
  - X is training data matrix (potentially extended with features)
  - g is regularization term (1-norm, squared 2-norm, group lasso)
- Function properties
  - f is 1-strongly convex and 1-smooth and $f \circ X$ is $\|X\|_2^2$-smooth
  - g is convex and possibly nondifferentiable
- Gradient $\nabla(f \circ X)(\theta) = X^T(X\theta - Y)$