Linear Models for Supervised Learning

CS 540

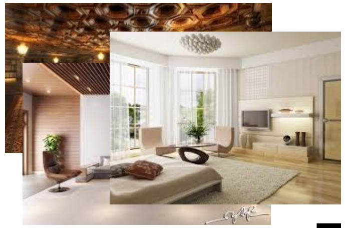
Yingyu Liang

Supervised Learning

Example: image classification

Raw training data

Label: indoor



Label: outdoor

Feature extraction

Training data

$$x_1 = [0.19, 0.23, \ldots], y_1 = +1$$

$$x_2 = [0.51, 0.33, \ldots], y_2 = -1$$

$$\vdots$$

Label: outdoor

Label: indoor

Raw test data

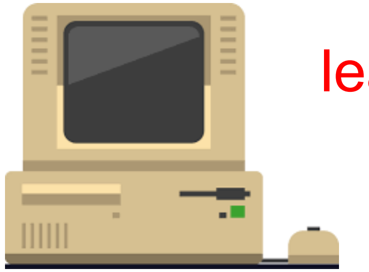
Feature extraction

Test data

$$x'_1 = [0.11, 0.13, \ldots], y'_1 = +1$$

$$x'_2 = [0.41, 0.35, \ldots], y'_2 = -1$$

$$\vdots$$



learning (i.e.,training)

testing

performance

Supervised Learning

- Input: training data set $\{(x_i, y_i): 1 \leq i \leq n\}$
- Output: model/function $y = f(x)$ learned on the training data
- Goal: learn f with good performance on future data

Supervised Learning

- Input: training data set $\{(x_i, y_i): 1 \leq i \leq n\}$
- Output: model/function $y = f(x)$ learned on the training data
- Goal: learn f with good performance on future data

Typically:

- Performance of f estimated on a test data set $\{(x'_i, y'_i): 1 \leq i \leq n'\}$.
- Assume both the training and test data are i.i.d. samples from an unknown data distribution D_{XY}
- Classification: discrete labels; Regression: continuous labels

Supervised Learning

- Input: training data set $\{(x_i, y_i): 1 \leq i \leq n\}$
- Output: model/function $y = f(x)$ learned on the training data
- Goal: learn f with good performance on future data

- Different kinds of functions correspond to different learning methods
- The focus of this lecture: linear models

Review: MLE and MAP

- Given $\{(x_i, y_i)\}$ from distribution D_θ with unknown parameter θ

- MLE: find θ that maximizes the likelihood

$$\max_\theta \; p(\{(x_i, y_i)\}|\theta) = \prod_i p(x_i, y_i|\theta)$$

- MAP: assume a prior $p(\theta)$ over θ, find θ that maximizes the posterior

$$\max_\theta \; p(\theta)p(\{(x_i, y_i)\}|\theta) = p(\theta)\prod_i p(x_i, y_i|\theta)$$

Review: Optimization

- For a differentiable function $L(\theta)$, the local maxima/minima satisfy

$$\nabla L(\theta) = 0$$

- For a convex function, all local minima are global minima. So setting the gradient to 0 gives the global minima.

Linear Regression

Linear regression in 1-dimension

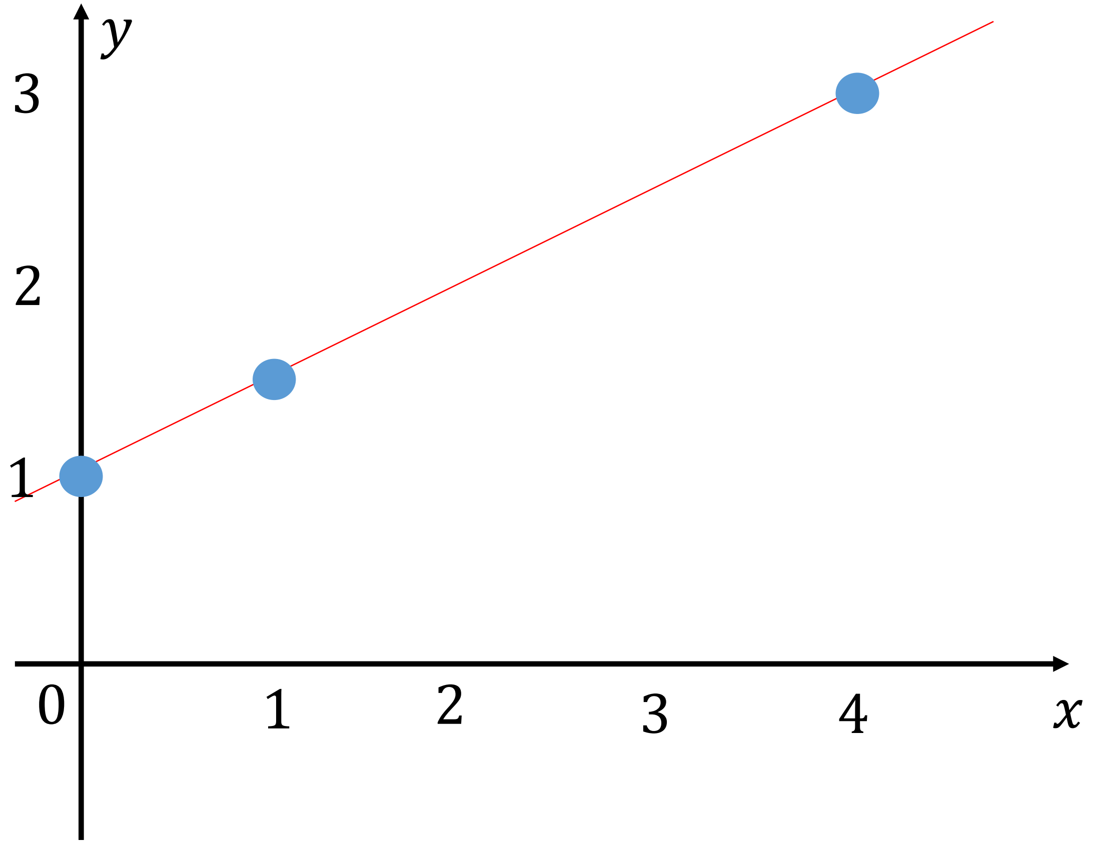
- $x \in R$: 1-dimension
- $y = f(x) = \beta_0 + \beta_1 x$: linear function in parameters β_0 , β_1

Terminologies:

- x : input variable (also called independent, predictor, explanatory variable)
- y : output variable (also called dependent, response variable)

Example 1

x	y
0	1
1	1.5
4	3

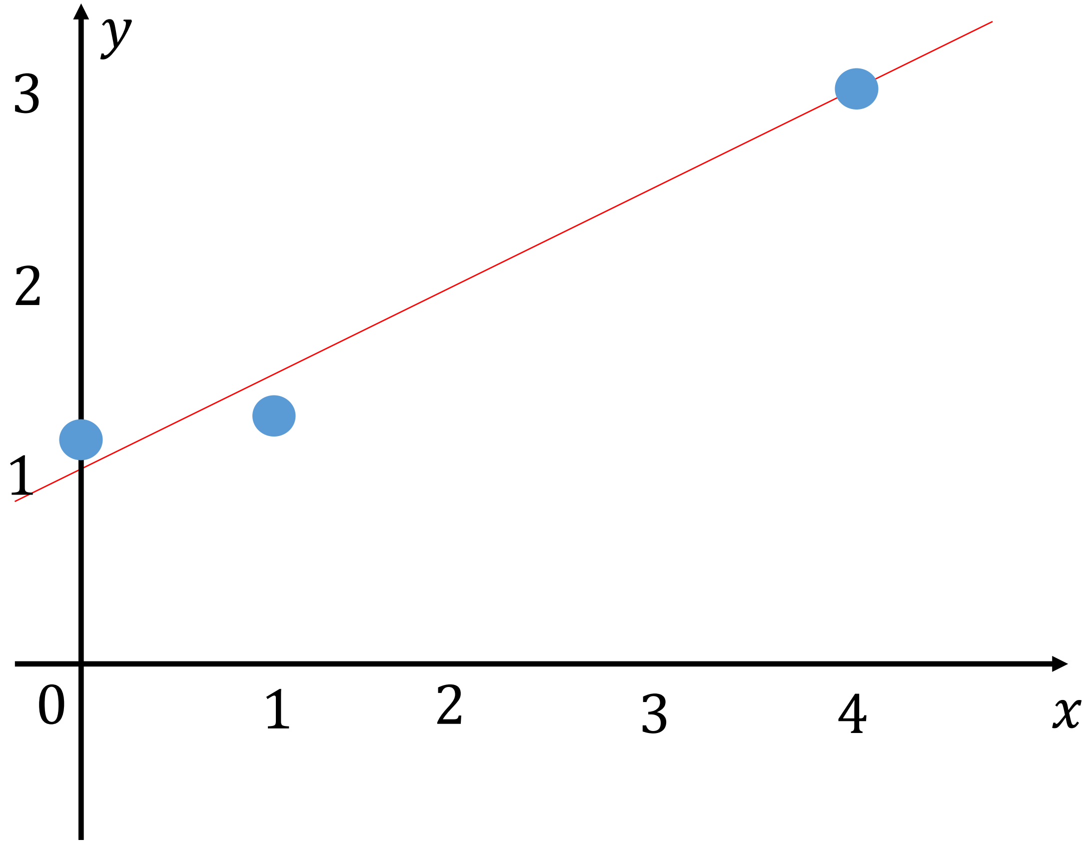


- Can just solve the equations:
- $\beta_0 + \beta_1 * 0 = 1$
- $\beta_0 + \beta_1 * 1 = 1.5$
- $\beta_0 + \beta_1 * 4 = 3$

$\Rightarrow$ $\beta_0 = 1,$ $\beta_1 = 0.5$

Example 2: with noise

x	y
0	1.1
1	1.4
4	3



- No solutions to the equations!
- $\beta_0 + \beta_1 * 0 = 1.1$
- $\beta_0 + \beta_1 * 1 = 1.4$
- $\beta_0 + \beta_1 * 4 = 3$

Linear regression in 1-dimension

- Why we use linear functions? How to handle noise? 
- Assumption on the data distribution: there are ground truth $\beta_0^*,\ \beta_1^*$ and

$$y = \beta_0^* + \beta_1^* x + \epsilon$$

where $\epsilon \sim N(0, \sigma^2)$.

Linear regression in 1-dimension

- Training: Maximum Likelihood Estimate (=least squares estimate)
- Since the noise ϵ is Gaussian:

$$likelihood(\beta_0, \beta_1 | \{x_i, y_i\}) = p(\{x_i, y_i\} | \beta_0, \beta_1) = \prod_i p(x_i) \frac{1}{\sqrt{2\pi}} e^{-\frac{(\beta_0 + \beta_1 x_i - y_i)^2}{2\sigma^2}}$$

$$loglikelihood(\beta_0, \beta_1 | \{x_i, y_i\}) = n \log \frac{1}{\sqrt{2\pi}} + \sum_i \log p(x_i) - \sum_i \frac{(\beta_0 + \beta_1 x_i - y_i)^2}{2\sigma^2}$$

So MLE leads to

$$\widehat{\beta_0}, \widehat{\beta_1} = \mathrm{argmin}_{\beta_0, \beta_1} \sum_i (\beta_0 + \beta_1 x_i - y_i)^2$$

Linear regression in 1-dimension

- Training: Maximum Likelihood Estimate (=least squares estimate)

$$\widehat{\beta_0}, \widehat{\beta_1} = \text{argmin}_{\beta_0,\beta_1} \sum_i (\beta_0 + \beta_1 x_i - y_i)^2$$

- Also called Ordinary Least Squares (OLS)
- Convex optimization
- Set gradient to 0 leads to closed-form solution

Gradient w.r.t. β_0 is $\sum_i 2(\beta_0 + \beta_1 x_i - y_i) = 0$

Gradient w.r.t. β_1 is $\sum_i 2(\beta_0 + \beta_1 x_i - y_i)\, x_i = 0$

Linear regression in 1-dimension

- Set gradient to 0 leads to closed-form solution

  Gradient w.r.t. β_0 is $\sum_i 2(\beta_0 + \beta_1 x_i - y_i) = 0$

  $n\beta_0 + \beta_1 \sum_i x_i - \sum_i y_i = 0$

  $\beta_0 = \frac{\sum_i y_i}{n} - \beta_1 \frac{\sum_i x_i}{n}$

  $\beta_0 = \bar{y} - \beta_1 \bar{x}$, where we define $\bar{x} = \frac{\sum_i x_i}{n}, \bar{y} = \frac{\sum_i y_i}{n}$

Linear regression in 1-dimension

- Set gradient to 0 leads to closed-form solution

$\beta_0 = \bar{y} - \beta_1 \bar{x}$, where we define $\bar{x} = \frac{\sum_i x_i}{n}, \bar{y} = \frac{\sum_i y_i}{n}$

Gradient w.r.t. β_1 is $\sum_i 2(\beta_0 + \beta_1 x_i - y_i)\, x_i = 0$

$\sum_i (\bar{y} - \beta_1 \bar{x} + \beta_1 x_i - y_i)\, x_i = 0$

$\sum_i \bar{y} x_i - \sum_i \beta_1 \bar{x} x_i + \sum_i \beta_1 x_i^2 - \sum_i y_i x_i = 0$

$\beta_1 \left(\sum_i x_i^2 - \sum_i \bar{x} x_i\right) + \sum_i \bar{y} x_i - \sum_i y_i x_i = 0$

$\beta_1 = (\sum_i y_i x_i - \sum_i \bar{y} x_i)/\left(\sum_i x_i^2 - \sum_i \bar{x} x_i\right)$

Linear regression in 1-dimension

- Set gradient to 0 leads to closed-form solution

$\beta_0 = \bar{y} - \beta_1 \bar{x}$, where we define $\bar{x} = \frac{\sum_i x_i}{n}, \bar{y} = \frac{\sum_i y_i}{n}$

$\beta_1 = (\sum_i y_i x_i - \sum_i \bar{y} x_i) / \left(\sum_i x_i^2 - \sum_i \bar{x} x_i\right)$

We have:

$$\begin{aligned}
\sum_i (x_i - \bar{x})^2 &= \sum_i (x_i^2 - 2x_i\bar{x} + \bar{x}^2) \\
&= \sum_i x_i^2 - \sum_i 2x_i\bar{x} \ + n\bar{x}^2 \\
&= \sum_i x_i^2 - \sum_i 2x_i\bar{x} \ + \sum_i x_i\bar{x} \\
&= \sum_i x_i^2 - \sum_i x_i\bar{x}
\end{aligned}$$

Linear regression in 1-dimension

- Set gradient to 0 leads to closed-form solution

$\beta_0 = \bar{y} - \beta_1 \bar{x}$, where we define $\bar{x} = \frac{\sum_i x_i}{n}, \bar{y} = \frac{\sum_i y_i}{n}$

$\beta_1 = (\sum_i y_i x_i - \sum_i \bar{y} x_i)/(\sum_i x_i^2 - \sum_i \bar{x} x_i)$

We have:

$\sum_i (x_i - \bar{x})^2 = \sum_i x_i^2 - \sum_i x_i \bar{x}$

Similarly (please check it offline!):

$\sum_i (x_i - \bar{x})(y_i - \bar{y}) = \sum_i y_i x_i - \sum_i \bar{y} x_i$

Linear regression in 1-dimension

- Set gradient to 0 leads to closed-form solution

$\widehat{\beta_0} = \bar{y} - \widehat{\beta_1}\bar{x}$, where we define $\bar{x} = \frac{\sum_i x_i}{n}, \bar{y} = \frac{\sum_i y_i}{n}$

$\widehat{\beta_1} = \sum_i (x_i - \bar{x})(y_i - \bar{y}) \,/\, \sum_i (x_i - \bar{x})^2$

Linear regression in 1-dimension

- Given the closed-form solution $\widehat{\beta_0}, \widehat{\beta_1}$, denote the prediction

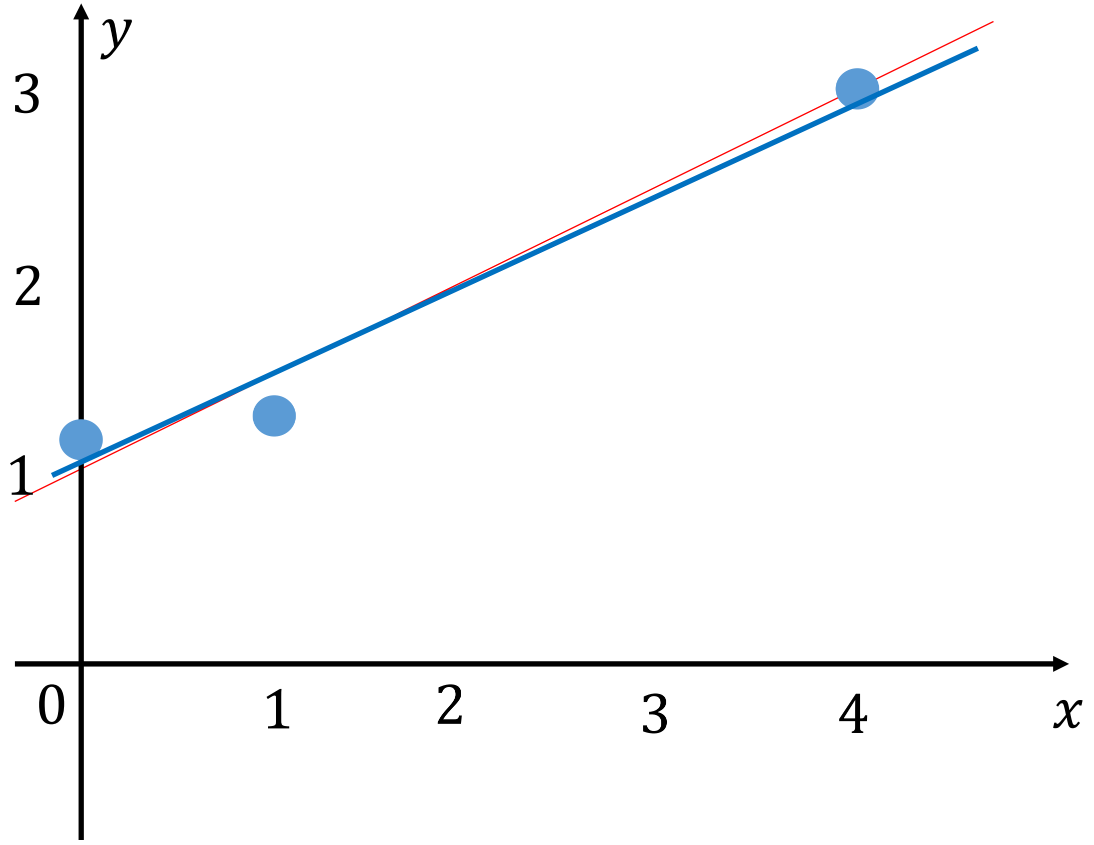
$$\hat{y}_i = \widehat{\beta_0} + \widehat{\beta_1} x_i$$

- The residue on x_i is $y_i - \hat{y}_i$
- The residue sum of squares is $\sum_i (y_i - \hat{y}_i)^2$

- Another way to evaluate the fit: compared to fit using a constant
- Best constant fit is $\bar{y}$
- The coefficient of determination is $r^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}$

Example 2: revisit

x	y
0	1.1
1	1.4
4	3



- Plug in the closed-form solution:
- $\bar{x} = \frac{5}{3}, \bar{y} = \frac{5.5}{3}$
- $\widehat{\beta_1} = \frac{\left(0-\frac{5}{3}\right)\left(1.1-\frac{5.5}{3}\right)+\left(1-\frac{5}{3}\right)\left(1.4-\frac{5.5}{3}\right)+\left(4-\frac{5}{3}\right)\left(3-\frac{5.5}{3}\right)}{\left(0-\frac{5}{3}\right)^2+\left(1-\frac{5}{3}\right)^2+\left(4-\frac{5}{3}\right)^2} \approx 0.4885,\ \ \widehat{\beta_0} = \frac{5.5}{3} - \frac{5}{3}\widehat{\beta_1} \approx 1.019$
- Can also compute residue ≈ 0.0188, and $r^2 \approx 0.991$

Linear regression in multi-dimension

- $x = \left(x_0 = 1, x_1, \ldots, x_p\right) \in R^{p+1}$
- $y = f(x) = \beta^T x, \beta = \left(\beta_0, \beta_1, \ldots, \beta_p\right) \in R^{p+1}$: linear function in β

- Assumption on the data distribution: there is ground truth β^* and

$$y = (\beta^*)^T x + \epsilon$$

where $\epsilon \sim N(0, \sigma^2)$.

Linear regression in multi-dimension

- Training: Maximum Likelihood Estimate (=Ordinary Least Squares)

$$\hat{\beta} = \text{argmin}_\beta \sum_i (\beta^T x_i - y_i)^2$$

- Let $X \in R^{n \times (p+1)}$ be a matrix where the i-th row is x_i
- Let $\boldsymbol{y} \in R^n$ be a vector where the i-th entry is y_i
- Then OLS is

$$\hat{\beta} = \text{argmin}_\beta \|\boldsymbol{y} - X\beta\|_2^2$$

where $\|v\|_2^2 = v^T v = \sum_i v_i^2$

Linear regression in multi-dimension

- Training: Maximum Likelihood Estimate (=Ordinary Least Squares)

$$\min_\beta \|\boldsymbol{y} - X\beta\|_2^2$$

$$\min_\beta (\boldsymbol{y} - X\beta)^T(\boldsymbol{y} - X\beta)$$

$$\min_\beta \boldsymbol{y}^T\boldsymbol{y} - 2\beta^T X^T \boldsymbol{y} + \beta^T X^T X\beta$$

- Convex optimization. Set gradient to 0 leads to closed-form solution

  Gradient w.r.t. β is $-2X^T\boldsymbol{y} + 2X^T X\beta = 0$
- Suppose $X^T X$ invertible, then $\hat{\beta} = (X^T X)^{-1} X^T \boldsymbol{y}$

What if $X^T X$ not invertible?

Linear regression in multi-dimension

- Training: Maximum A Posteriori (leads to Ridge Regression)
- Assume prior $\beta \sim N(0, \frac{\sigma^2}{\lambda} I)$ for some $\lambda > 0$
- MAP leads to

$$\min_\beta ||\boldsymbol{y} - X\beta||_2^2 + \lambda||\beta||_2^2$$

- Ridge Regression = OLS + ℓ_2 regularization
- Convex optimization. Set gradient to 0 leads to closed-form solution

  Gradient w.r.t. β is $-2X^T\boldsymbol{y} + 2X^TX\beta + 2\lambda\beta = 0$
- $X^TX + \lambda I$ is always invertible, then $\hat{\beta} = (X^TX + \lambda I)^{-1}X^T\boldsymbol{y}$

Example 3

- $\beta^* = [1, \frac{1}{2}, \frac{1}{3}]$

$\boldsymbol{x}$	$(\boldsymbol{\beta}^*)^T\boldsymbol{x}$	$\boldsymbol{y}$
(1,0,0)	1	1
(1,1,1)	11/6	2
(1,1,2)	13/6	7/3

- $X = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 2 \end{bmatrix}, \boldsymbol{y} = \begin{bmatrix} 1 \\ 2 \\ 7/3 \end{bmatrix}$

Example 3

- $\beta^* = [1, \frac{1}{2}, \frac{1}{3}]$

$\boldsymbol{x}$	$(\boldsymbol{\beta}^*)^T \boldsymbol{x}$	$\boldsymbol{y}$	$\widehat{\boldsymbol{y}}$
(1,0,0)	1	1	0.997
(1,1,1)	11/6	2	1.99
(1,1,2)	13/6	7/3	2.34

- $X = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 2 \end{bmatrix}, \boldsymbol{y} = \begin{bmatrix} 1 \\ 2 \\ 7/3 \end{bmatrix}$
- $\lambda = 0.01, \hat{\beta} \approx [0.997, 0.648, 0.346]$

Example 3

- $\beta^* = [1, \frac{1}{2}, \frac{1}{3}]$

$\boldsymbol{x}$	$(\boldsymbol{\beta}^*)^T \boldsymbol{x}$	$\boldsymbol{y}$	$\widehat{\boldsymbol{y}}$
(1,0,0)	1	1	1
(1,1,1)	11/6	2	2
(1,1,2)	13/6	7/3	2.33

- $X = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 2 \end{bmatrix}, \boldsymbol{y} = \begin{bmatrix} 1 \\ 2 \\ 7/3 \end{bmatrix}$
- $\lambda = 0, \hat{\beta} \approx [1, 0.667, 0.333]$

Example 3

- $\beta^* = [1, \frac{1}{2}, \frac{1}{3}]$

$\boldsymbol{x}$	$(\boldsymbol{\beta}^*)^T \boldsymbol{x}$	$\boldsymbol{y}$	$\widehat{\boldsymbol{y}}$
(1,0,0)	1	1	0.049
(1,1,1)	11/6	2	0.150
(1,1,2)	13/6	7/3	0.211

- $X = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 2 \end{bmatrix}, \boldsymbol{y} = \begin{bmatrix} 1 \\ 2 \\ 7/3 \end{bmatrix}$
- $\lambda = 100, \hat{\beta} \approx [0.049, 0.040, 0.061]$

Linear regression using nonlinear features

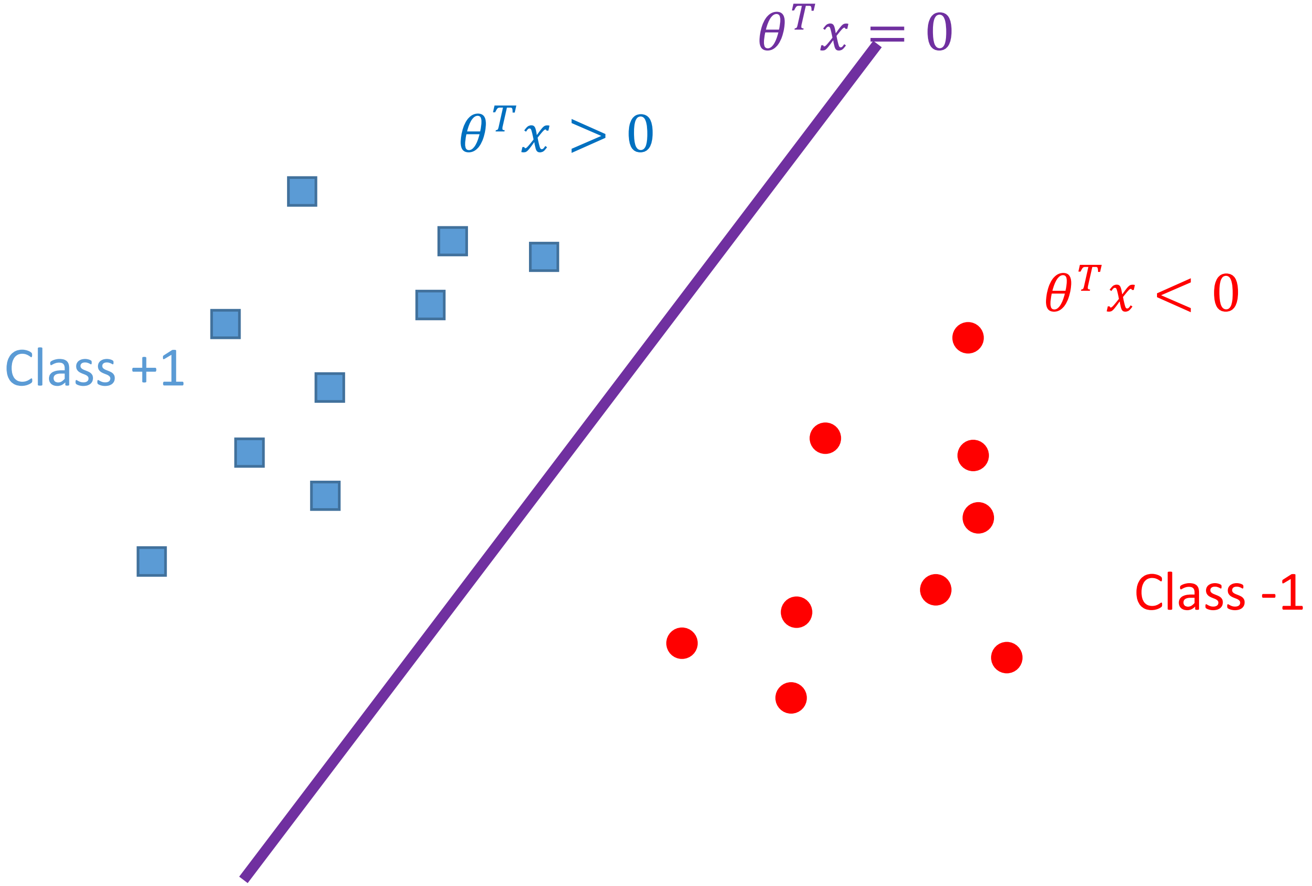
- Linear regression only needs to be linear in β
- Can use nonlinear features of the input
- Polynomial regression on input z
  - let $x = (1, z, z^2, z^3, \dots, z^p)$
  - Function $y = \beta^T x = \beta_0 + \beta_1 z + \beta_2 z^2 + \dots + \beta_p z^p$
- Higher order regression on input $z = (z_1, z_2)$
  - let $x = (1, z_1, z_2, z_1 z_2, z_1^2, z_2^2)$
  - Function $y = \beta^T x = \beta_0 + \beta_1 z_1 + \beta_2 z_2 + \beta_3 z_1 z_2 + \beta_4 z_1^2 + \beta_5 z_2^2$
- In general, for input z
  - Let $x = (1, \phi_1(z), \phi_2(z), \phi_3(z), \dots, \phi_p(z))$ for functions ϕ_j
  - Function $y = \beta_0 + \sum_j \beta_j \phi_j(z)$

Logistic Regression

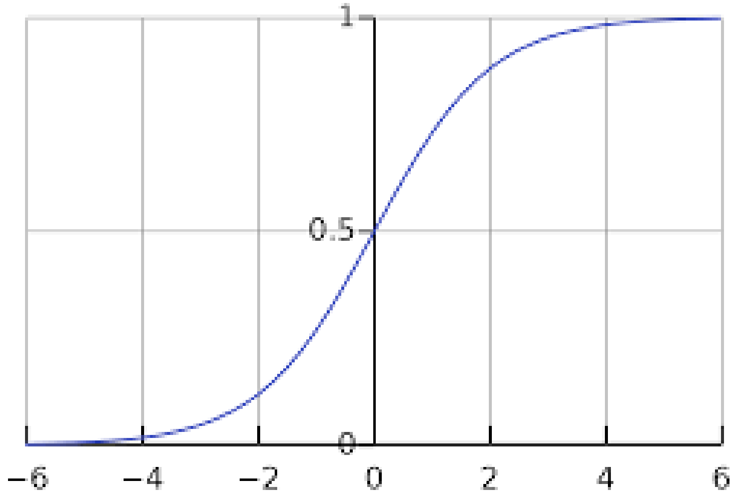
Linear classification

- $x \in R^{p+1}, y \in \{-1, +1\}$

- Intuition: use $\theta^T x$
  - $y = +1$ if $\theta^T x$ positive
  - $y = -1$ if $\theta^T x$ negative



Logistic regression



- $x \in R^{p+1}, y \in \{-1, +1\}$
- Idea: squash $\theta^T x$ to $[0,1]$ so that it represents the probability $y = +1$

$$p(y = +1|x) = \sigma(\theta^T x) = \frac{1}{1 + \exp(-\theta^T x)}$$

$$p(y = -1|x) = 1 - \sigma(\theta^T x) = \frac{\exp(-\theta^T x)}{1 + \exp(-\theta^T x)} = \frac{1}{1 + \exp(\theta^T x)}$$

where $\sigma(z) = \frac{1}{1+\exp(-z)}$ is the logistic function

Logistic regression

- Training: Maximum Likelihood Estimate (on the conditional likelihood)

$$\max_{\boldsymbol{\theta}} \sum_{\boldsymbol{i}} \log p(y_i | x_i, \theta)$$

$$\min_{\boldsymbol{\theta}} \sum_{\boldsymbol{i}} \log\left(1 + \exp(-y_i \theta^T x_i)\right)$$

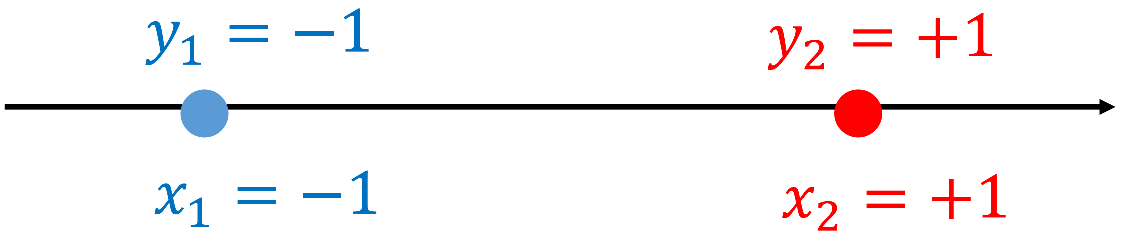
- When training data is linearly separable, MLE is bad
    1. $\|\theta\|_2$ goes to infinity
    2. There can be many solutions

Logistic regression

- Training: Maximum Likelihood Estimate (on the conditional likelihood)

$$\min_{\boldsymbol{\theta}} \sum_{\boldsymbol{i}} \log\left(1 + \exp(-y_i \theta^T x_i)\right)$$

- When training data is linearly separable, MLE is bad
  1. $\|\theta\|_2$ goes to infinity
  2. There can be many solutions
- To see 1, consider the simple example below

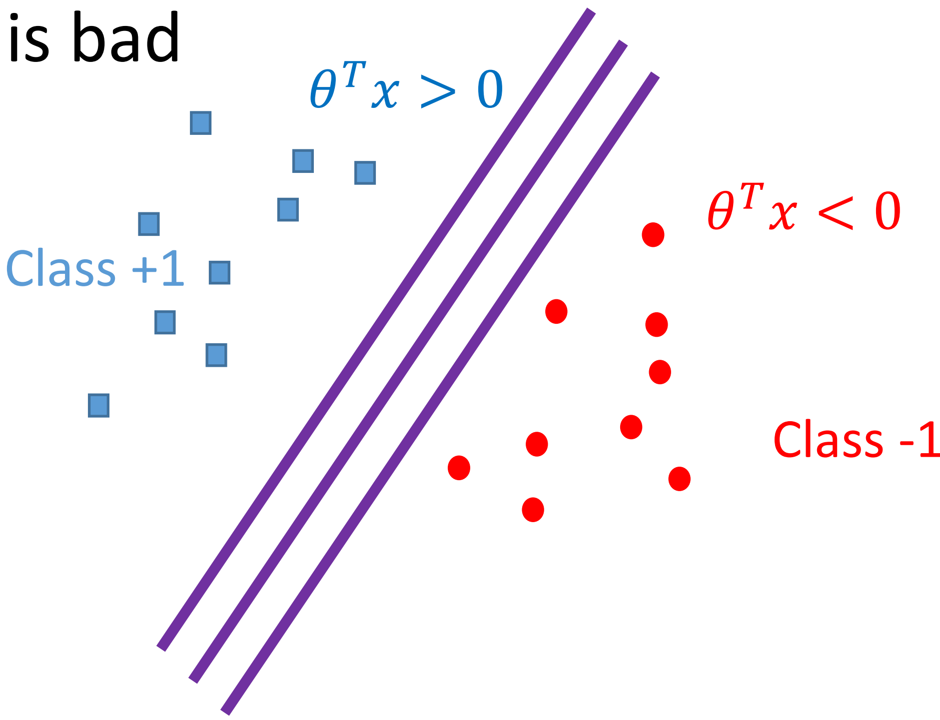


Logistic regression

- Training: Maximum Likelihood Estimate (on the conditional likelihood)

$$\min_{\boldsymbol{\theta}} \sum_{\boldsymbol{i}} \log\left(1 + \exp(-y_i \theta^T x_i)\right)$$

- When training data is linearly separable, MLE is bad
  1. $\|\theta\|_2$ goes to infinity
  2. There can be many solutions
- To see 2, consider the figure



Logistic regression

- Training: Maximum A Posteriori
- Assume prior $\beta \sim N(0, \frac{1}{\lambda} I)$ for some $\lambda > 0$
- MAP leads to

$$\min_{\boldsymbol{\theta}} \sum_{\boldsymbol{i}} \log\left(1 + \exp(-y_i \theta^T x_i)\right) + \frac{\lambda}{2} \|\theta\|_2^2$$

- Convex optimization
- But no closed form solution; solve via (stochastic) gradient descent

Gradient descent

- Suppose we have optimization

$$\min_{\boldsymbol{\theta}} \sum_i l(x_i , y_i, \theta)$$

  - Regularized logistic regression: $l(x_i, y_i, \theta) = \log(1 + \exp(-y_i \theta^T x_i)) + \frac{\lambda}{2n} \|\theta\|_2^2$
  - Ridge regression: $l(x_i, y_i, \beta) = (\beta^T x_i - y_i)^2 + \frac{\lambda}{n} \|\beta\|_2^2$

- Gradient descent (GD) with step size $\eta > 0$:
  - Initialize $\theta^{(0)}$
  - For $t = 1,2, \ldots ,$ set $\theta^{(t)} = \theta^{(t-1)} - \eta \sum_i \nabla_\theta l(x_i, y_i, \theta^{(t-1)})$

Stochastic gradient descent

- Suppose we have optimization

$$\min_{\boldsymbol{\theta}} \sum_i l(x_i, y_i, \theta)$$

  - Regularized logistic regression: $l(x_i, y_i, \theta) = \log(1 + \exp(-y_i \theta^T x_i)) + \frac{\lambda}{2n} \|\theta\|_2^2$
  - Ridge regression: $l(x_i, y_i, \beta) = (\beta^T x_i - y_i)^2 + \frac{\lambda}{n} \|\beta\|_2^2$

- Stochastic gradient descent (SGD) with step size $\eta > 0$:
  - Initialize $\theta^{(0)}$
  - For $t = 1, 2, \ldots,$ randomly sample an i, and $\theta^{(t)} = \theta^{(t-1)} - \eta n \nabla_\theta l(x_i, y_i, \theta^{(t-1)})$

Multi-class logistic regression

- $x \in R^{p+1}, y \in \{1, 2, \dots, K\}$
- Each class has parameter $\theta_j \in R^{p+1}$
- Use softmax to squash $(\theta_1^T x, \theta_2^T x, \dots, \theta_K^T x)$ into probabilities:

$$p(y = j | x, \{\theta_k\}) = \frac{\exp(\theta_j^T x)}{\sum_k \exp(\theta_k^T x)}$$

- Training by MLE:

$$\max_{\boldsymbol{\theta}} \sum_{\boldsymbol{i}} \log p(y_i | x_i, \{\theta_k\})$$

Multi-class logistic regression

- $x \in R^{p+1}, y \in \{1, 2, \ldots, K\}$
- Each class has parameter $\theta_j \in R^{p+1}$
- Use softmax to squash $(\theta_1^T x, \theta_2^T x, \ldots, \theta_K^T x)$ into probabilities:

$$p(y = j | x, \{\theta_k\}) = \frac{\exp(\theta_j^T x)}{\sum_k \exp(\theta_k^T x)}$$

- Training by MAP:

$$\min_{\boldsymbol{\theta}} \sum_{\boldsymbol{i}} \log p(y_i | x_i, \{\theta_k\}) + \frac{\lambda}{2} \sum_{\boldsymbol{k}} \|\theta_k\|_2^2$$