Supervised Learning

Liyao Xiang
http://xiangliyao.cn/
Shanghai Jiao Tong University

Reference and Acknowledgement

- Most of the course materials are credited to Andrew Ng’s CS229 lecture notes.

Outline

- Linear Regression (线性回归)
- Classification and Logistic Regression (逻辑回归)
- Generalized Linear Models

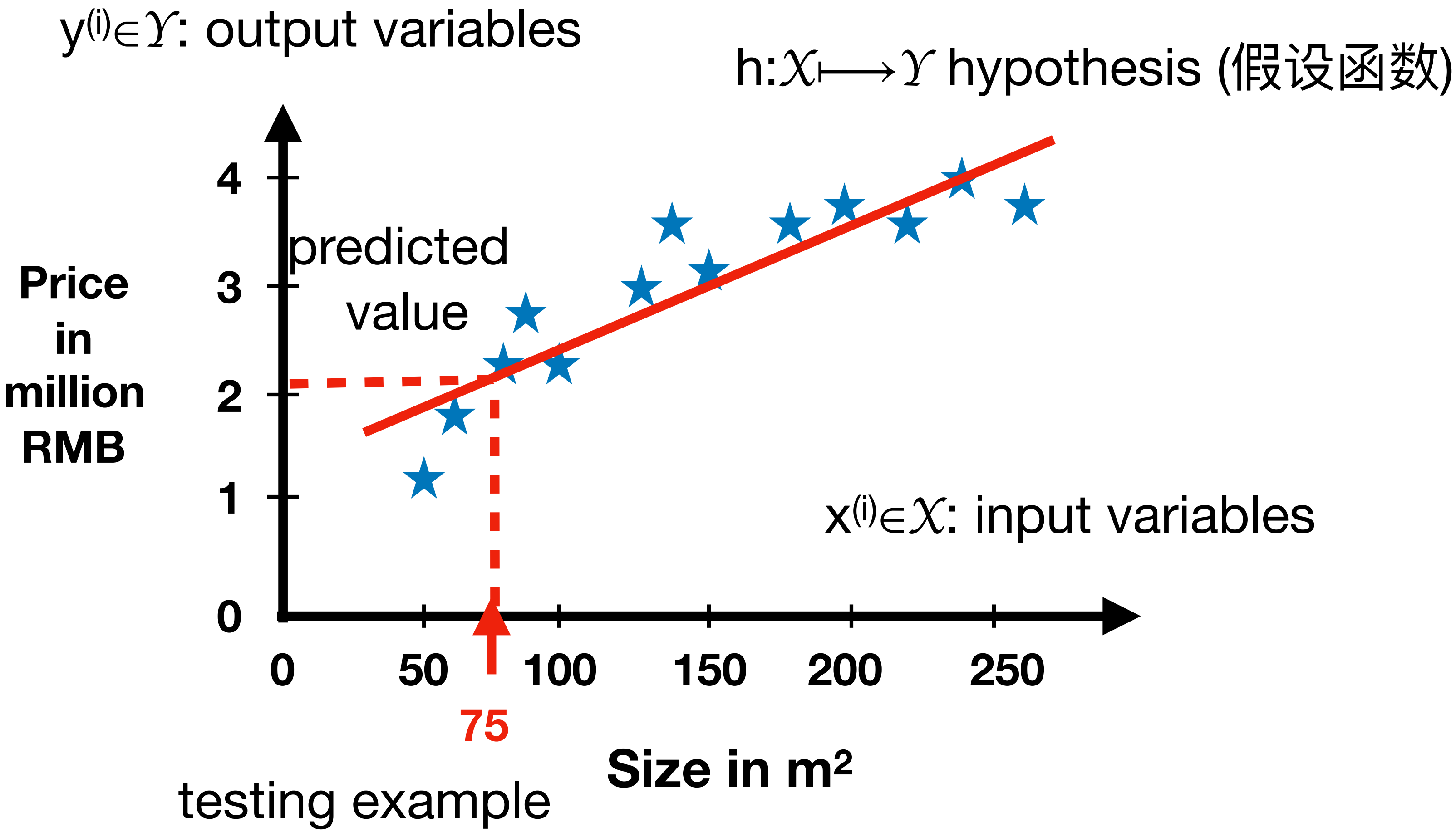
Outline

- Linear Regression (线性回归)
- Classification and Logistic Regression (逻辑回归)
- Generalized Linear Models

Supervised Learning Example Revisited

($x^{(i)}$, $y^{(i)}$): a training example

{($x^{(i)}$, $y^{(i)}$); i = 1,…, m}: training set



Supervised Learning Example Revisited

Let's consider a richer dataset in which we also know the number of bedrooms in each apartment

Size	#bedrooms	Price (million ¥)
40	0	1.2
65	1	1.9
80	2	2.2
89	2	3.3
120	3	5.3
…	…	…

- x: two-dimensional vectors in $\mathcal{R}^2$
- $x_1^{(i)}$: the size of the i-th apartment in the training set
- $x_2^{(i)}$: the number of bedrooms of the i-th apartment in the training set
- We decide hypothesis h as a linear function: $h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$
- θ_i: parameters/weights of h
- By letting $x_0 = 1$, we rewrite h as

Why a linear function? →

$$h(x) = \sum_{i=0}^{n} \theta_i x_i = \theta^T x$$

Supervised Learning Example Revisited

Let's consider a richer dataset in which we also know the number of bedrooms in each apartment

Size	#bedrooms	Price (million ¥)
40	0	1.2
65	1	1.9
80	2	2.2
89	2	3.3
120	3	5.3
…	…	…

- By letting $x_0 = 1$, we rewrite h as

$$h(x) = \sum_{i=0}^{n} \theta_i x_i = \theta^T x$$

- How can we learn θ? Making h(x) close to y for the training examples!
- cost function (损失函数):

$$J(\theta) = \frac{1}{2} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

Why a least-squares cost?

Least-Mean Square Alg

- How to choose θ to minimize J(θ)? Let's start with some "initial guess" for θ, and use gradient descent (梯度下降) alg. repeatedly to make J(θ) smaller:

$$\theta_j := \theta_j - \alpha \frac{\partial J(\theta)}{\theta_j}$$

direction of steepest decrease of J

α: learning rate

- What is the partial derivative (偏导数) term?

$$\begin{aligned}
\frac{\partial}{\partial \theta_j} J(\theta) &= \frac{\partial}{\partial \theta_j} \frac{1}{2} \left(h_\theta(x) - y\right)^2 \\
&= 2 \cdot \frac{1}{2} \left(h_\theta(x) - y\right) \cdot \frac{\partial}{\partial \theta_j} \left(h_\theta(x) - y\right) \\
&= \left(h_\theta(x) - y\right) \cdot \frac{\partial}{\partial \theta_j} \left(\sum_{i=0}^{n} \theta_i x_i - y \right) \\
&= \left(h_\theta(x) - y\right) x_j
\end{aligned}$$

least mean square update rule:

$$\theta_j := \theta_j + \alpha \left(y^{(i)} - h_\theta(x^{(i)})\right) x_j^{(i)}$$

error term

Least-Mean Square Alg

- Two ways to modify the method:
  - batch gradient descent: scan through the entire training set before taking a single step

    Repeat until convergence {

    $$\theta_j := \theta_j + \alpha \sum_{i=1}^{m} \left(y^{(i)} - h_\theta(x^{(i)})\right) x_j^{(i)} \qquad \text{(for every } j\text{)}$$

    }

  - stochastic gradient descent: update parameters according to the gradient of the error w.r.t. a single training example

    Loop {

    for i=1 to m, {

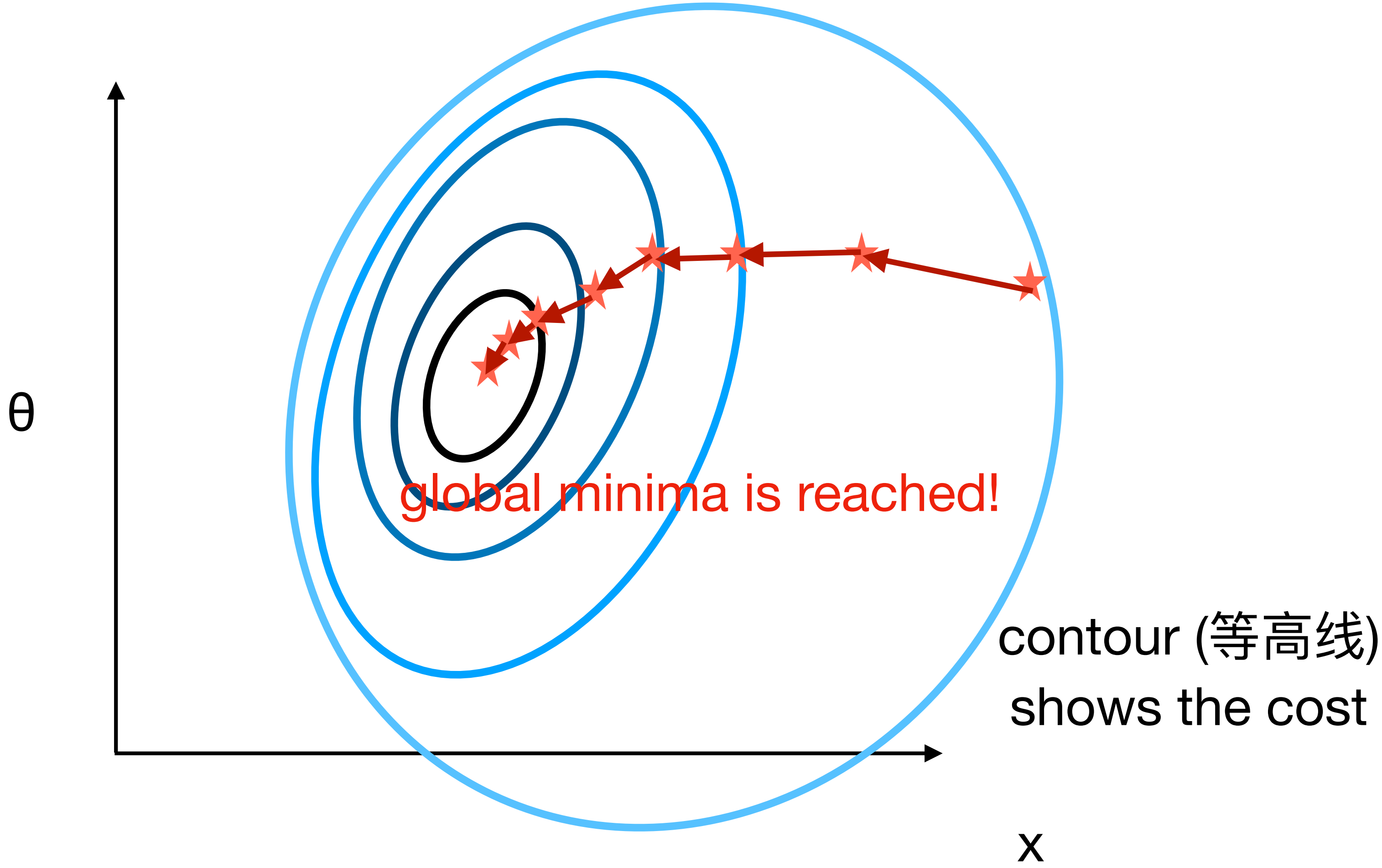
    $$\theta_j := \theta_j + \alpha \left(y^{(i)} - h_\theta(x^{(i)})\right) x_j^{(i)} \qquad \text{(for every } j\text{)}$$

    }

    }

Convergence

- In most cases, gradient descent converges to local minima. Linear regression only has one global minima, which the gradient descent always converges to. This is because the cost function J is a convex quadratic function (二次凸函数).



Normal Equations (标准方程)

- Gradient descent gives one way of minimizing J. How about others?
- We minimize J by explicitly taking derivatives w.r.t. θ and setting them to 0s. And solve the equations!

f: $\mathcal{R}^{m\times n} \longmapsto \mathcal{R}$

A: m x n matrix

$$\nabla_A f(A) = \begin{bmatrix} \frac{\partial f}{\partial A_{11}} & \cdots & \frac{\partial f}{\partial A_{1n}} \\ \vdots & \ddots & \vdots \\ \frac{\partial f}{\partial A_{m1}} & \cdots & \frac{\partial f}{\partial A_{mn}} \end{bmatrix}$$

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$$

$$f(A) = \frac{3}{2}A_{11} + 5A_{12}^2 + A_{21}A_{22}$$

$$\Rightarrow \quad \nabla_A f(A) = \begin{bmatrix} \frac{3}{2} & 10A_{12} \\ A_{22} & A_{21} \end{bmatrix}$$

Normal Equations (标准方程)

1. trace (迹): $\mathrm{tr}A = \sum_{i=1}^{n} A_{ii}$, the trace of a real number is itself

2. trace of a matrix = trace of its transpose (转置矩阵) $\mathrm{tr}A = \mathrm{tr}A^T$

3. $\mathrm{tr}(A+B) = \mathrm{tr}A + \mathrm{tr}B$, $\mathrm{tr}AB = \mathrm{tr}BA$

4. $\nabla_A \mathrm{tr}AB = B^T$

5. $\nabla_{A^T} \mathrm{tr}ABA^TC = B^TA^TC^T + BA^TC$

$$X = \begin{bmatrix} — (x^{(1)})^T — \\ — (x^{(2)})^T — \\ \vdots \\ — (x^{(m)})^T — \end{bmatrix} \qquad \vec{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix}$$

$$X\theta - \vec{y} = \begin{bmatrix} (x^{(1)})^T\theta \\ \vdots \\ (x^{(m)})^T\theta \end{bmatrix} - \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(m)} \end{bmatrix} = \begin{bmatrix} h_\theta(x^{(1)}) - y^{(1)} \\ \vdots \\ h_\theta(x^{(m)}) - y^{(m)} \end{bmatrix}.$$

Normal Equations (标准方程)

$$
\begin{aligned}
\nabla_\theta J(\theta) &= \nabla_\theta \frac{1}{2}(X\theta - \vec{y})^T(X\theta - \vec{y}) \\
&= \frac{1}{2}\nabla_\theta \left(\theta^T X^T X\theta - \theta^T X^T \vec{y} - \vec{y}^T X\theta + \vec{y}^T \vec{y}\right) \\
\text{Property 1} \quad &= \frac{1}{2}\nabla_\theta \operatorname{tr}\left(\theta^T X^T X\theta - \theta^T X^T \vec{y} - \vec{y}^T X\theta + \vec{y}^T \vec{y}\right) \\
\text{Property 2, 3} \quad &= \frac{1}{2}\nabla_\theta \left(\operatorname{tr}\theta^T X^T X\theta - 2\operatorname{tr}\vec{y}^T X\theta\right) \\
\text{Property 4, 5} \quad &= \frac{1}{2}\left(X^T X\theta + X^T X\theta - 2X^T\vec{y}\right) \\
&= X^T X\theta - X^T\vec{y} \quad = 0
\end{aligned}
$$

$$\Rightarrow \quad \theta = (X^T X)^{-1} X^T \vec{y}.$$

Probabilistic View

- The target variables and the inputs are related by

$$y^{(i)} = \theta^T x^{(i)} + \epsilon^{(i)} \quad \text{error term}$$

- Assume $\epsilon^{(i)}$ are distributed IID (independently and identically distributed 独立同分布) and $\epsilon^{(i)} \sim \mathcal{N}(0, \sigma^2)$

- Implies

$$p(y^{(i)}|x^{(i)};\theta) = \frac{1}{\sqrt{2\pi}\sigma} \exp\Big(-\frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}\Big)$$

- Given X and θ, what is the distribution of $y^{(i)}$'s? Likelihood function:

$$\begin{aligned} L(\theta) &= \prod_{i=1}^{m} p(y^{(i)} \mid x^{(i)};\theta) \\ &= \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}\right) \end{aligned}$$

Probabilistic View

- Maximum likelihood: we should choose θ to make the data as high probability as possible
- Equivalently, we maximize the log likelihood:

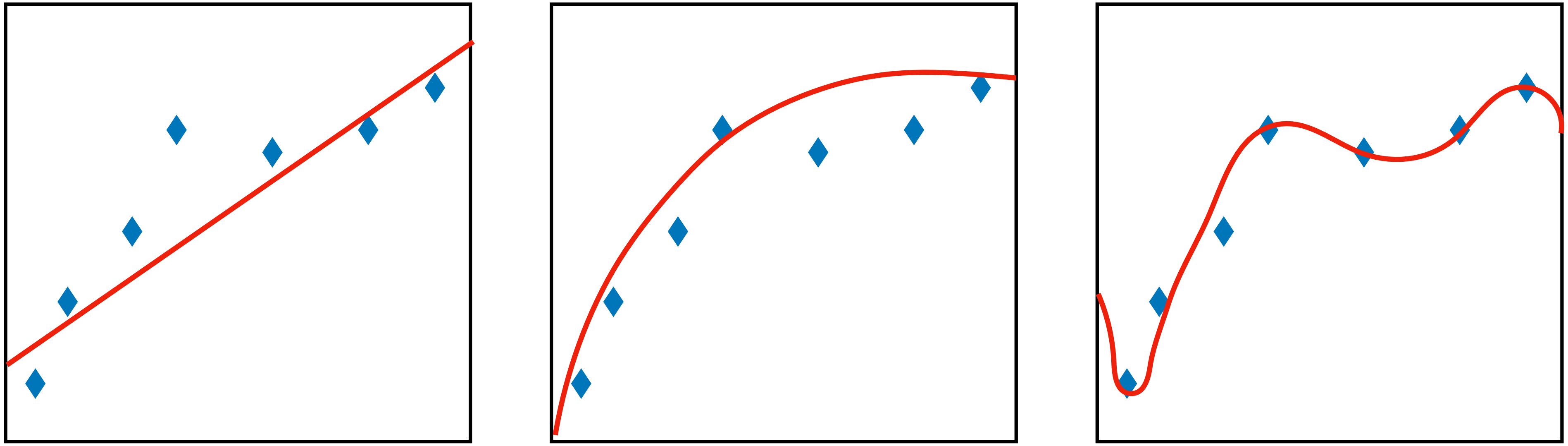
$$
\begin{aligned}
\ell(\theta) &= \log L(\theta) \\
&= \log \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}\right) \\
&= \sum_{i=1}^{m} \log \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}\right) \\
&= m \log \frac{1}{\sqrt{2\pi}\sigma} - \frac{1}{\sigma^2} \cdot \boxed{\frac{1}{2} \sum_{i=1}^{m} (y^{(i)} - \theta^T x^{(i)})^2}
\end{aligned}
$$

minimizing this term instead!
original least-squares cost

Underfitting & Overfitting

- Fitting to different hypotheses:

$$y = \theta_0 + \theta_1 x \qquad y = \theta_0 + \theta_1 x + \theta_2 x^2 \qquad y = \sum_{j=0}^{5} \theta_j x^j$$



underfitting   overfitting

The more features we add, the better. However, there is also a risk in adding too many features.

Locally Weighted Linear Regression

- The choice of features is important to learning performance!
- Locally weighted linear regression
  1. Fit θ to minimize $\sum_i w^{(i)}(y^{(i)} - \theta^T x^{(i)})^2$
  2. Output $\theta^T x$
- larger $w^{(i)}$ -> try harder to make $(y^{(i)} - \theta^T x^{(i)})^2$ small; otherwise, ignore the corresponding error term
- Standard choice for the weight:

$$w^{(i)} = \exp\left(-\frac{(x^{(i)} - x)^2}{2\tau^2}\right)$$

Non-parametric Alg: keep the entire training dataset when making predictions

θ is giving a higher weight to the training examples close to the testing data x

Summary

- Linear regression
  - Linear hypothesis class $h(x) = \sum_{i=0}^{n} \theta_i x_i = \theta^T x$
  - Cost function $J(\theta) = \frac{1}{2} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$
  - Least mean square algorithm: $\theta_j := \theta_j + \alpha \left(y^{(i)} - h_\theta(x^{(i)}) \right) x_j^{(i)}$
    - Batch/stochastic gradient descent
  - Probabilistic view:
    - Errors ~ I.I.D. Gaussian distribution
    - Maximum likelihood
  - Overfitting & Underfitting
  - Locally weighted linear regression

Outline

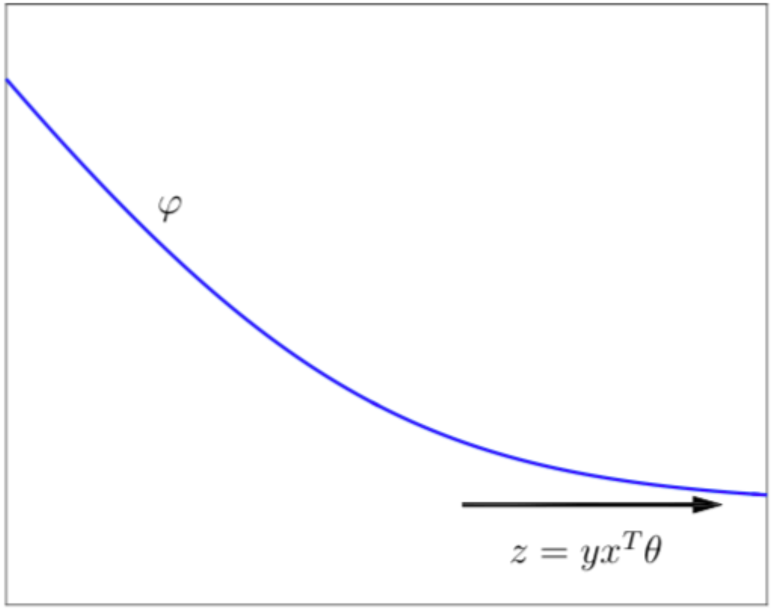
- Linear Regression (线性回归)
- Classification and Logistic Regression (逻辑回归)
- Generalized Linear Models

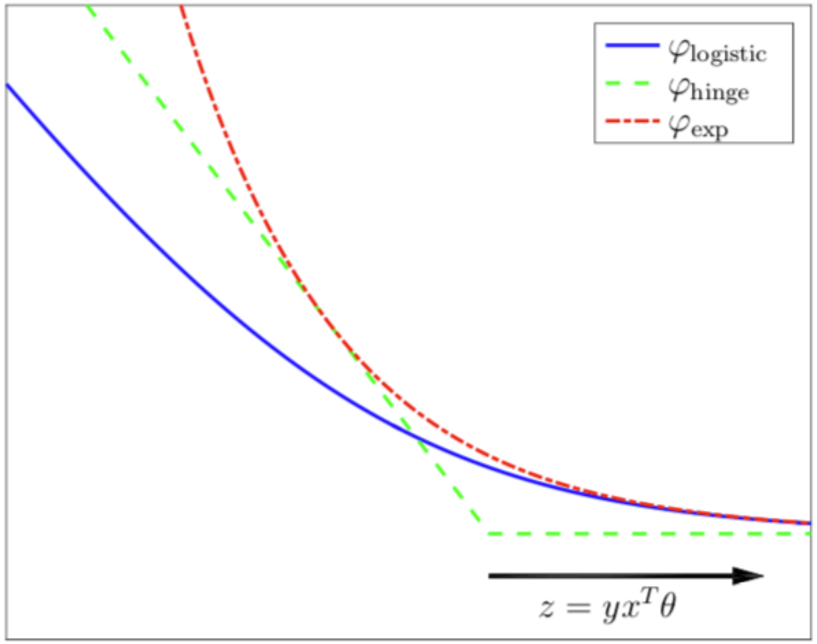
Binary Classification

- The target y can only take two values: $y \in \{-1, +1\}$. $y = 1$ if the example belongs to the positive class, otherwise, it is a member of the negative class
- Hypothesis: $h(x) = \theta^T x$. Given x, we classify it as positive or negative depending on the sign of $\theta^T x$, i.e., $\text{sign}(\theta^T x) = y \iff y\theta^T x > 0$
- Margin for the example (x, y): $y\theta^T x$ — the more $\theta^T x$ is negative (or positive), the stronger the belief that y is negative (or positive)
- loss function: should penalize the θ for which $y(i)\theta^T x(i) < 0$ frequently in the training data. Loss value should be small if $y(i)\theta^T x(i) > 0$ and large if $y(i)\theta^T x(i) < 0$
- We expect the loss function to be continuous and convex (easy to converge to the global minima!)

Binary Classification

- Expect the loss to satisfy: Loss_func (y(i)θ^Tx(i)) → 0 as y(i)θ^Tx(i) →∞ and Loss_func (y(i)θ^Tx(i)) → ∞ as y(i)θ^Tx(i) →-∞





$Loss_{logistic}(z) = \log(1 + e^{-z})$ logistic regression

$Loss_{hinge} = \max\{1 - z, 0\}$ support vector machines

$Loss_{exp} = e^{-z}$ boosting

Logistic Regression

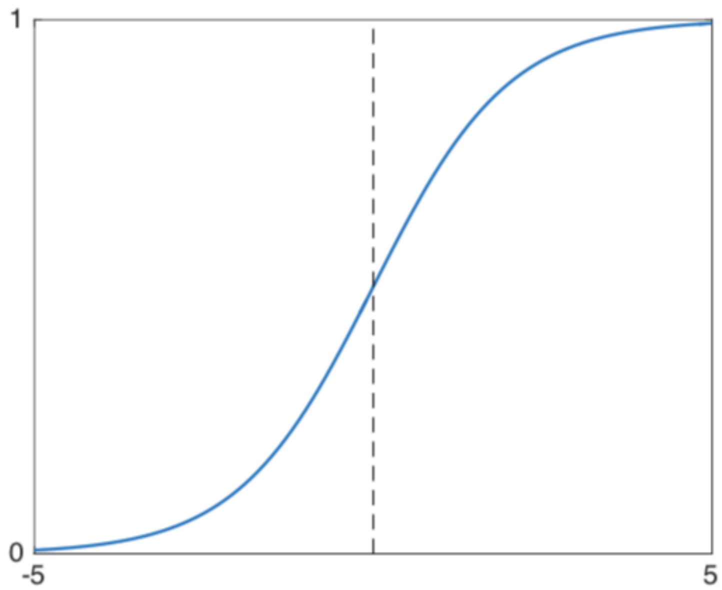
- Choose θ to minimize

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} Loss_{logistic}(y^{(i)}\theta^T x^{(i)}) = \frac{1}{m}\sum_{i=1}^{m} \log(1 + \exp(-y^{(i)}\theta^T x^{(i)}))$$

which hopefully yields θ that $y(i)\theta^T x(i) > 0$ for most training examples

- **Alternative view:** Logistic (Sigmoid) function $$g(z) = \frac{1}{1+e^{-z}}$$
$\rightarrow 1$ as $z \rightarrow \infty$ and $g(z) \rightarrow 0$ as $z \rightarrow -\infty$

- g(z) + g(-z) = 1 we could use it to define the probability model for binary classification.



Probabilistic View

- For y ∈ {-1, +1}, we define the logistic model as

$$p(Y = y|x;\theta) = g(yx^T\theta) = \frac{1}{1+e^{-yx^T\theta}}$$, & refine hypothesis class as

$$h_\theta(x) = \frac{1}{1+e^{-x^T\theta}}$$

- The likelihood of the training data is

$$L(\theta) = \prod_{i=1}^{m} p(Y = y^{(i)} \mid x^{(i)}; \theta) = \prod_{i=1}^{m} h_\theta(y^{(i)} x^{(i)})$$

- The log-likelihood is

$$\ell(\theta) = \sum_{i=1}^{m} \log h_\theta(y^{(i)} x^{(i)}) = -\sum_{i=1}^{m} \log\left(1 + e^{-y^{(i)}\theta^T x^{(i)}}\right) = -mJ(\theta)$$

- maximizing likelihood in the logistic model = minimizing the average logistic loss

Gradient Descent

- For the $Loss_{logistic}(z) = \log(1 + e^{-z})$, the derivative is

Sigmoid function

$$\frac{\mathrm{d}}{\mathrm{d}z} Loss_{logistic}(z) = \frac{1}{1+e^{-z}} \cdot \frac{\mathrm{d}}{\mathrm{d}z} e^{-z} = -\frac{e^{-z}}{1+e^{-z}} = -g(-z)$$

- For a single training example (x, y):

$$\frac{\partial}{\partial \theta_k} Loss_{logistic}(yx^T\theta) = -g(-yx^T\theta)\frac{\partial}{\partial \theta_k}(yx^T\theta) = -g(-yx^T\theta)yx_k$$

- Update rule for stochastic gradient descent:

$$\theta^{t+1} = \theta^t - \alpha_t \cdot \nabla_\theta Loss_{logistic}(-y^{(i)}x^{(i)T}\theta^t)$$

incorrect label

$$= \theta^{(t)} + \alpha_t g(-y^{(i)}x^{(i)^T}\theta^{(t)})y^{(i)}x^{(i)} = \theta^{(t)} + \alpha_t h_{\theta^{(t)}}(-y^{(i)}x^{(i)})\underline{y^{(i)}x^{(i)}}$$

Update Rule when y∈{0,1}

$$P(y=1|x;\theta) = h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$$

$$P(y=0|x;\theta) = 1 - h_\theta(x)$$

$$\Rightarrow \quad p(y|x;\theta) = (h_\theta(x))^y (1-h_\theta(x))^{1-y}$$

$$\begin{aligned} L(\theta) &= p(\vec{y} \mid X;\theta) \\ &= \prod_{i=1}^{m} p(y^{(i)} \mid x^{(i)};\theta) \\ &= \prod_{i=1}^{m} \left(h_\theta(x^{(i)})\right)^{y^{(i)}} \left(1-h_\theta(x^{(i)})\right)^{1-y^{(i)}} \end{aligned}$$

$$\begin{aligned} \ell(\theta) &= \log L(\theta) \\ &= \sum_{i=1}^{m} y^{(i)} \log h(x^{(i)}) + (1-y^{(i)}) \log(1-h(x^{(i)})) \end{aligned}$$

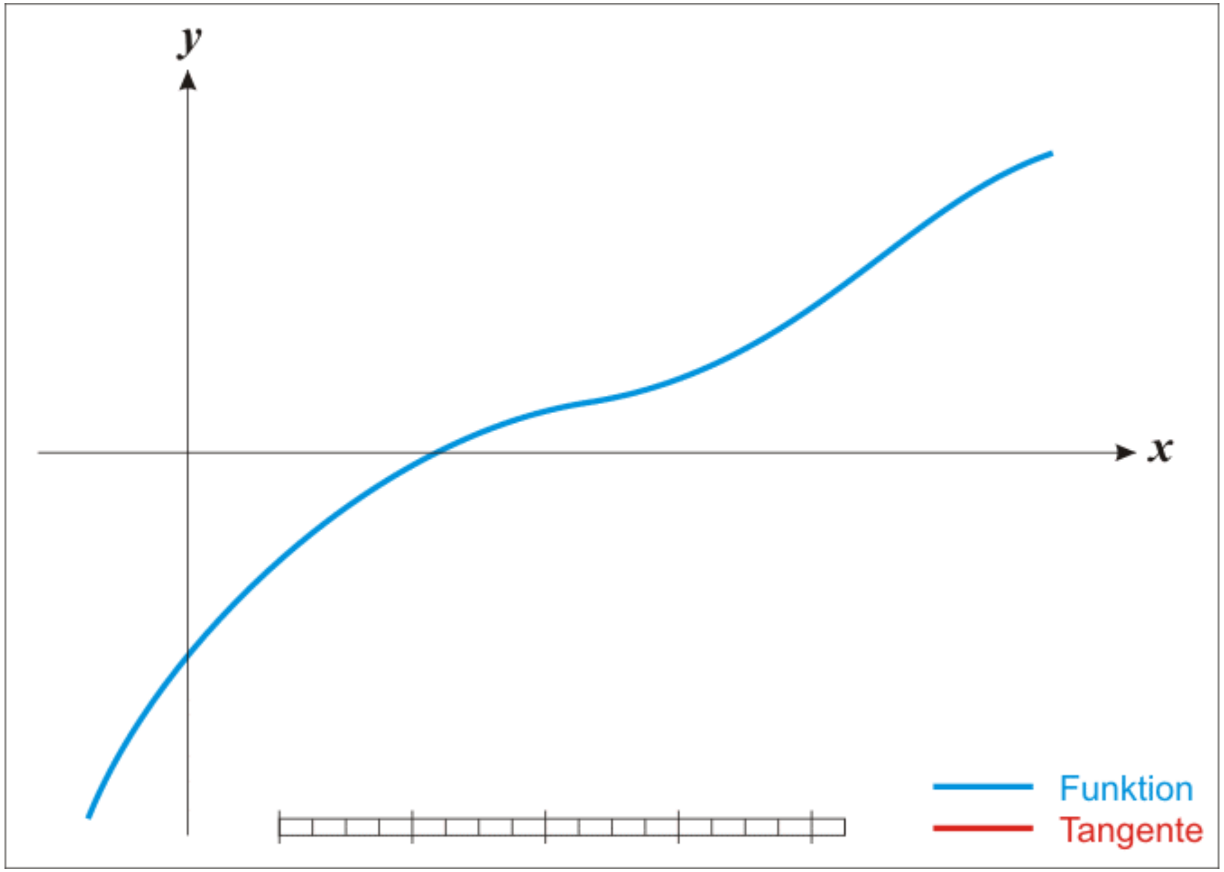
similar to least mean square update rule, but h is non-linear!

gradient ascent:

$$\theta := \theta + \alpha \nabla_\theta \ell(\theta) \quad \Rightarrow \quad \theta_j := \theta_j + \alpha \left(y^{(i)} - h_\theta(x^{(i)})\right) x_j^{(i)}$$

Another Update Rule to Maximize $l(\theta)$

- Newton's method for finding a zero of a function: $f(\theta) = 0$
- Update rule: $\theta := \theta - f(\theta)/f'(\theta)$



Another Update Rule to Maximize $l(\theta)$

- Newton's method for finding a zero of a function: f(θ) = 0
- What if we want to maximize some loss function? The maxima of the loss corresponds to points where its first derivative is 0
- Update rule: $\theta := \theta - \frac{l'(\theta)}{l''(\theta)}$
- Multidimensional setting: $\theta := \theta - H^{-1}\nabla_\theta l(\theta)$ Hessian matrix
- Advantage: Newton's method typically enjoys faster convergence than gradient descent, and requires many fewer iterations to get very close to the minimum.
- Disadvantage: more expensive in one iteration

Summary

- Logistic regression
  - Hypothesis $h(x) = \theta^T x$
  - Cost function $Loss_{logistic}(z) = \log(1 + e^{-z})$
  - Update rule: $\theta^{t+1} = \theta^t - \alpha_t \cdot \nabla_\theta Loss_{logistic}(-y^{(i)} x^{(i)T} \theta^t)$
    - Newton's method $\theta := \theta - \dfrac{l'(\theta)}{l''(\theta)}$
  - Probabilistic view:
    - maximizing likelihood in the logistic model = minimizing the average logistic loss

Outline

- Linear Regression (线性回归)
- Classification and Logistic Regression (逻辑回归)
- Generalized Linear Models

Generalized Linear Models

- Given the distributions of y | x, how do we come up with the hypothesis?
  - linear regression: $y|x;\theta \sim \mathcal{N}(\mu, \sigma^2)$, hypothesis: $h_\theta(x) = \theta^T x$
  - logistic regression: $y|x;\theta \sim \text{Bernoulli}(\phi)$, hypothesis:
    $$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$
  - We show both of the methods are special cases of generalized linear models

Generalized Linear Models

- Probabilistic view of regression:

$$y|x;\theta \sim \mathcal{N}(\mu, \sigma^2)$$

$$p(y;\mu) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}(y-\mu)^2\right)$$

$$= \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}y^2\right) \times$$

b(y)

$$\exp\left(\mu y - \frac{1}{2}\mu^2\right)$$

-a(η)

η^T T(y)

Gaussian, Bernoulli, …

exponential family distributions

- Probabilistic view of classification:

$$y|x;\theta \sim \text{Bernoulli}(\phi)$$

$$p(y;\phi) = \phi^y (1-\phi)^{1-y}$$

$$= \exp(y \log \phi + (1-y)\log(1-\phi))$$

$$= \exp\left(\left(\log\left(\frac{\phi}{1-\phi}\right)\right)y + \log(1-\phi)\right)$$

η^T T(y)

-a(η)

$$p(y;\eta) = b(y)\exp(\eta^T T(y) - a(\eta))$$

A fixed choice of T, a, and b defines a family of distributions parameterized by η

Construct GLMs

- Knowing the distribution, how to construct GLMs?
- Assumptions about P(y | x) and hypothesis:

1. $y|x;\theta$ follows a distribution that belongs to exponential family
2. $h(x) = \mathbb{E}[T(y)|x]$. E.g., in logistic regression,

$$h_\theta(x) = p(y = 1|x;\theta) = 0 \cdot p(y = 0|x;\theta) + 1 \cdot p(y = 1|x;\theta) = \mathbb{E}[y|x]$$

3. parameter η and inputs x are linearly related: $\eta = \theta^T x$

Construct Linear Regression Model

- Target variable follows Gaussian distribution: $y|x;\theta \sim \mathcal{N}(\mu, \sigma^2)$

$$h_\theta(x) = \mathbb{E}[y|x;\theta]$$ Assumption 2

$$= \mu$$ $y|x;\theta \sim \mathcal{N}(\mu, \sigma^2)$

$$= \eta$$ Assumption 1: Write the Gaussian distribution in the form of exponential family distribution

$$p(y;\mu) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}y^2\right) \times \exp\left(\mu y - \frac{1}{2}\mu^2\right)$$

$$= \theta^T x$$ Assumption 3

Construct Logistic Regression Model

- Target variable is binary-valued. Thus we choose the Bernoulli family distributions to model the conditional distribution:

$$y|x;\theta \sim \text{Bernoulli}(\phi)$$

$$h_\theta(x) = \mathbb{E}[y|x;\theta]$$ Assumption 2

$$= \phi$$ Bernoulli distribution

$$= 1/(1+e^{-\eta})$$ Assumption 1

$$p(y;\phi) = \exp\left(\Bigg(\underbrace{\log\left(\frac{\phi}{1-\phi}\right)}_{=\eta}\Bigg)y + \log(1-\phi)\right)$$

$$= 1/(1+e^{-\theta^T x})$$ Assumption 3

k-Classification

- Target variable takes on any of k values: $y \in \{1, 2, \ldots, k\}$
- We choose multinomial distribution to model: k parameters $\phi_1, \ldots, \phi_k$ denoting the probability of each outcome
  - $\sum_{i=1}^{k} \phi_i = 1$, k - 1 parameters
  - $$T(1) = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, T(2) = \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, T(3) = \begin{bmatrix} 0 \\ 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}, \cdots, T(k-1) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}, T(k) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

$$(T(y))_i = 1\{y = i\} \Rightarrow \mathbb{E}[(T(y))_i] = P(y = i) = \phi_i$$

k-Classification

- Multinomial is a member of the exponential family.

$$p(y;\phi) = \phi_1^{1\{y=1\}}\phi_2^{1\{y=2\}}\dots\phi_k^{1\{y=k\}}$$

$$= \phi_1^{1\{y=1\}}\phi_2^{1\{y=2\}}\dots\phi_k^{1-\sum_{i=1}^{k-1}1\{y=i\}}$$

$$= \phi_1^{(T(y))_1}\phi_2^{(T(y))_2}\dots\phi_k^{1-\sum_{i=1}^{k-1}(T(y))_i}$$

$$= \exp((T(y))_1\log(\phi_1) + (T(y))_2\log(\phi_2) + \cdots + \left(1-\textstyle\sum_{i=1}^{k-1}(T(y))_i\right)\log(\phi_k))$$

$$= \exp((T(y))_1\log(\phi_1/\phi_k) + (T(y))_2\log(\phi_2/\phi_k) + \cdots + (T(y))_{k-1}\log(\phi_{k-1}/\phi_k) + \log(\phi_k))$$

$$= b(y)\exp(\eta^T T(y) - a(\eta))$$

$$\eta = \begin{bmatrix} \log(\phi_1/\phi_k) \\ \log(\phi_2/\phi_k) \\ \vdots \\ \log(\phi_{k-1}/\phi_k) \end{bmatrix}$$

$$a(\eta) = -\log(\phi_k)$$

$$b(y) = 1.$$

$$\sum_{i=1}^{k}\phi_i = 1$$

$$\phi_i = \frac{e^{\eta_i}}{\sum_{j=1}^{k} e^{\eta_j}}$$

probability of class i!
known as softmax function

Softmax Regression Model

$$h_\theta(x) = \mathrm{E}[T(y)|x;\theta]$$

Assumption 2

$$= \mathrm{E}\left[\left.\begin{array}{c} 1\{y=1\} \\ 1\{y=2\} \\ \vdots \\ 1\{y=k-1\} \end{array}\right| x;\theta \right]$$

Our hypothesis outputs the estimated probability that p(y=i | x; θ) for every i∈{1,…k}.

$$= \begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_{k-1} \end{bmatrix} = \begin{bmatrix} \frac{\exp(\theta_1^T x)}{\sum_{j=1}^k \exp(\theta_j^T x)} \\ \frac{\exp(\theta_2^T x)}{\sum_{j=1}^k \exp(\theta_j^T x)} \\ \vdots \\ \frac{\exp(\theta_{k-1}^T x)}{\sum_{j=1}^k \exp(\theta_j^T x)} \end{bmatrix}$$

Multinomial distribution

Express the multinomial distribution in the form of Exponential family & Assumption 3

Softmax Regression

- Training by maximizing the log-likelihood by gradient ascent or Newton's method

$$
\begin{aligned}
\ell(\theta) &= \sum_{i=1}^{m} \log p(y^{(i)}|x^{(i)};\theta) \\
&= \sum_{i=1}^{m} \log \prod_{l=1}^{k} \left(\frac{e^{\theta_l^T x^{(i)}}}{\sum_{j=1}^{k} e^{\theta_j^T x^{(i)}}} \right)^{1\{y^{(i)}=l\}}
\end{aligned}
$$

Summary

- Generalized Linear Models
  - distribution of the target variable —> hypothesis
    1. rewrite the distribution in the form of exponential family distributions
    2. find the relation between the expected value of the target variable and the natural parameter η
    3. express the natural parameter η in terms of inputs x (linear in most cases)