Notes 1.

1. LMS algorithm

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

$\Rightarrow$ gradient descent: $\theta_j := \theta_j - \alpha \cdot \frac{\partial}{\partial \theta_j} J(\theta)$.

(interpretation: ① simultaneously performed for all values of j
② α: learning rate
③ "$-$": we are minimizing.

Further, solve the partial derivative part:

Repeat until convergence {

$$\theta_j := \theta_j + \alpha \underbrace{\sum_{i=1}^{m}\boxed{(y^{(i)} - h_\theta(x^{(i)}))}\, x_j^{(i)}}_{\downarrow} \quad \text{(for every } j\text{)}.$$

intuition: larger error $\Rightarrow$ larger change

look at every example in the entire training set on every step
$\Rightarrow$ is called "batch gradient descent"

}

An alternative:

Loop {

for $i = 1$ to m, {

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

}

}

"stochastic gradient descent"

do not have to scan through the entire training set before taking a single step

2. The normal equations

2.1 Matrix derivatives.

① $f: \mathbb{R}^{m\times n} \mapsto \mathbb{R}$

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

② tr: trace

(i) if AB is square $\quad trAB = trBA$

(ii) $trABC = trCAB = trBCA$

(iii) $trABCD = \cdots$

(iv) A & B are square matrices

$trA = trA^T$

$tr(A+B) = trA + trB$

$tr aA = a\,trA$

⇒ Some facts of matrix derivatives.

$\nabla_A trAB = B^T$

$\nabla_{A^T} f(A) = (\nabla_A f(A))^T$

$\nabla_A trABA^TC = CAB + C^TBA^T$

$\nabla_A |A| = |A|(A^{-1})^T$.

2.2 Least squares revisited.

design Matrix: $X = \begin{bmatrix} (x^{(1)})^T \\ \vdots \\ (x^{(m)})^T \end{bmatrix}$ m samples & n features

$\vec{y} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(m)} \end{bmatrix}$

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

$\Rightarrow \frac{1}{2}(X\theta - \vec{y})^T(X\theta - \vec{y})$

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

$= J(\theta)$

$\Rightarrow \nabla_\theta J(\theta) = \cdots = X^TX\theta - X^T\vec{y} \overset{\text{to minimize}}{=} 0$

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

3. Probabilistic interpretation. → solve: why is J (cost function) a reasonable choice.

Assume $y^{(i)} = \theta^T x^{(i)} + \varepsilon^{(i)}$

($\varepsilon^{(i)}$ is an error term that captures either unmodeled effects or random noise.)

$\varepsilon^{(i)} \overset{i.i.d.}{\sim} N(0, \sigma^2)$

$\Rightarrow P(y^{(i)} | x^{(i)}; \theta) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}\right)$.

Namely, $y^{(i)} | x^{(i)}; \theta \sim N(\theta^T x^{(i)}, \sigma^2)$

likelihood function:

$L(\theta) = L(\theta; X, \vec{y}) = P(\vec{y} | X; \theta)$

$\overset{\text{independence assumption}}{=} \prod_{i=1}^{m} P(y^{(i)} | x^{(i)}; \theta)$

$= \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}\right)$.

Principal of maximum likelihood: $\ell(\theta) = \log L(\theta) = m \log \frac{1}{\sqrt{2\pi}\sigma} - \frac{1}{\sigma^2} \cdot \frac{1}{2} \sum_{i=1}^{m} (y^{(i)} - \theta^T x^{(i)})^2$

$\Leftrightarrow$ minimize $\frac{1}{2} \sum_{i=1}^{m} (y^{(i)} - \theta^T x^{(i)})^2$.

Interesting fact: our final choice of θ did not depend on σ^2

4. Locally weighted linear regression. —— non-parametric algorithm.

(LWR)

Fit θ to minimize $\sum_i w^{(i)} (y^{(i)} - \theta^T x^{(i)})^2$

A fairly standard choice:

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

(x: the particular point at which we're trying to evaluate x)

interpretation: ① give a much higher weight to the training examples close to the query point x

② τ: bandwidth parameter

controls how quickly the weight of a training example falls off with distance of its $x^{(i)}$ from the query point x.

Part II. Classification and logistic regression.

5. Logistic Regression.

– use linear regression.

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

where

$$g(z) = \frac{1}{1+e^{-z}}$$

logistic function / sigmoid function.

A useful property: $g'(z) = g(z)(1-g(z))$

Assumptions: $P(y=1 \mid x;\theta) = h_\theta(x)$

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

$\Rightarrow P(y \mid x;\theta) = (h_\theta(x))^y (1-h_\theta(x))^{1-y}$

$$\Rightarrow L(\theta) = p(\vec{y} \mid X;\theta)$$

$$= \prod_{i=1}^{m} P(y^{(i)} \mid x^{(i)};\theta)$$

$$= \prod_{i=1}^{m} (h_\theta(x^{(i)}))^{y^{(i)}} (1-h_\theta(x^{(i)}))^{1-y^{(i)}}$$

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

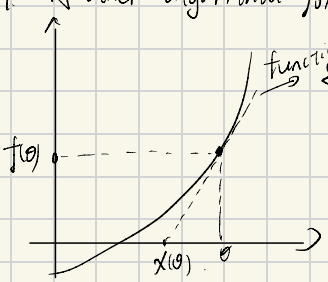
$\Rightarrow$ gradient descent: $\theta_j := \theta_j + \alpha (y^{(i)} - h_\theta(x^{(i)})) x_j^{(i)}$.

(stochastic)

6. Digression: The perceptron learning algorithm.

$$g(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{if } z < 0 \end{cases}$$

7. Another algorithm for maximizing $\ell(\theta)$.



function: $y - f(\theta) = f'(\theta)(x-\theta)$

let $y=0$

$f'(\theta)(x-\theta) = -f(\theta)$

$x-\theta = -\frac{f(\theta)}{f'(\theta)}$

$\Rightarrow x = \theta - \frac{f(\theta)}{f'(\theta)}$.

⇒ It performs the following update:

$$\theta := \theta - \frac{f(\theta)}{f'(\theta)}$$

interpretation: ① approximate the function f via a linear function that is tangent to f at the current guess θ

② let the next guess for θ be where that linear function is zero

⇒ Application: maximize ℓ

find point where $\ell'(\theta) = 0$

let $f(\theta) = \ell'(\theta)$

update rule:

$$\theta := \theta - \frac{\ell'(\theta)}{\ell''(\theta)}$$

⇒ generalize for vector-valued θ

$$\vec{\theta} := \vec{\theta} - H^{-1} \nabla_\theta \ell(\vec{\theta})$$

(H: (Hessian) $H_{ij} = \frac{\partial^2 \ell(\theta)}{\partial \theta_i \partial \theta_j}$)

Result: faster convergence than batch gradient descent

Called: Fisher scoring

Part III. Generalized Linear Models

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

In classification: $y | x; \theta \sim \text{Bernoulli}(\theta)$

8. The exponential family (canonical parameter)

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

($T(y)$: sufficient statistic; $a(\eta)$: log partition function (make sure integral equals to 1))

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

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

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

$\eta = \log\frac{\phi}{1-\phi}$

$\Rightarrow T(y) = y$

$a(\eta) = \log(1+e^{\eta})$

$b(y) = 1$

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

$= \frac{1}{\sqrt{2\pi}} \exp(-\frac{1}{2}y^2) \cdot \exp(\mu y - \frac{1}{2}\mu^2)$

$\Rightarrow \eta = \mu$

$T(y) = y$

$a(\eta) = \mu^2/2 = \eta^2/2$

$b(y) = (1/\sqrt{2\pi}) \exp(-y^2/2)$

9. Constructing GLMs.

To derive a GLM, have to make assumptions.

① $y|x;\theta \sim$ ExponentialFamily(η)

② the prediction $h(x)$ output to satisfy $h(x) = E[y|x]$

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

9.1 Ordinary Least Squares

$h_\theta(x) = E[y|x;\theta]$

Assumption 2 $\Rightarrow = \mu$

Assumption 1 $\Rightarrow = \eta$

$= \theta^T x$

↑ Assumption 3.

9.2 Logistic Regression.

$h_\theta(x) = E[y|x;\theta]$

$= \phi$

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

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

Once we assume that y conditioned on x is Bernoulli,
it arises as a consequence of the definition of GLMs and exponential family distributions.

q.3 Softmax Regression.

$y \in \{1, 2, \cdots, k\}$. ($y$ is response variable)

↓

Multinominal distribution.

⇒ parameterize $\phi_1 \cdots \phi_{k-1}$ (since $\sum_{i=1}^{k} \phi_i = 1$).

define $T(y) \in R^{k-1}$ as follows:

$$T(1) = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad T(2) = \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad T(3) = \begin{bmatrix} 0 \\ 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}, \quad \cdots, \quad T(k-1) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}, \quad T(k) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Notation. $(T(y))_i$ denotes the i-th element of the vector $T(y)$.

⇒ $T(y)_i = 1\{y = i\}$

$E[(T(y))_i] = P(y = i) = \phi_i$

⇒ $P(y; \phi) = \phi_1^{1\{y=1\}} \phi_2^{1\{y=2\}} \cdots \phi_k^{1\{y=k\}}$

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

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

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

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

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

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

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

$b(y) = 1$.

Φ link function: $\eta_i = \log \frac{\phi_i}{\phi_k}$

define: $\eta_k = \log(\phi_k / \phi_k) = 0$

$e^{\eta_i} = \phi_i / \phi_k \Rightarrow \phi_k e^{\eta_i} = \phi_i$

$\Rightarrow \phi_k \sum_{i=1}^{k} e^{\eta_i} = \sum_{i=1}^{k} \phi_i = 1$

$\Rightarrow \phi_k = \frac{1}{\sum_{i=1}^{k} e^{\eta_i}}$

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

Assumption 3. $\eta_i = \theta_i^T x$

$\Rightarrow P(y=i \mid x; \theta) = \phi_i$

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

$= \frac{e^{\theta_i^T x}}{\sum_{j=1}^{k} e^{\theta_j^T x}}$ softmax regression

Output: $h_\theta(x) = E[T(y) \mid x; \theta]$

$= E\left[\begin{bmatrix} 1\{y=1\} \\ 1\{y=2\} \\ \vdots \\ 1\{y=k-1\} \end{bmatrix} \middle| x; \theta \right]$

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

parameter fitting (choose θ)

$l(\theta) = \sum_{i=1}^{m} \log P(y^{(i)} \mid x^{(i)}; \theta)$

$= \sum_{i=1}^{m} \log \prod_{l=1}^{k} \left(\frac{e^{\theta_l^T x^{(i)}}}{\sum_{j=1}^{k} e^{\theta_j^T x^{(i)}}} \right)^{1\{y^{(i)}=l\}}$