Ch 5 Meeting

Thursday, June 14, 2018 10:05 AM

Xiao and Suya

task T performance measure P experience (data) E

linear regression:

task - predict y output, given X input

$$\hat{y} = w^T x$$

Performance measure (P) - mean square error (on test data)

MSE_{test}

Experience (E): training data set (X, Y)

split into $(X, Y)_{train}$ $(X, Y)_{test}$

Cost function

minimize MSE_{train}: $w = \arg\min_w MSE_{train}(w)$

logistic regression: classification task - (binary) output is probability of class

(X, y): $P(y=1 \mid x, \theta) = \sigma(\theta^T x)$

$P(y=0 \mid x, \theta) = 1 - \sigma(\theta^T x)$

sigmoid $\sigma = \frac{1}{1+e^{-x}}$

typically, predict output class by max probability

decision boundary still linear -

$$\frac{1}{1+e^{-\theta^T x}} > 0.5$$

$$e^{-\theta^T x} < 1$$

$$\theta^T x > 0$$

training points $\{x^{(1)}, \ldots, x^{(m)}\}$ — this requires i.i.d. assumption

likelihood $L(\theta) = \prod_{i=1}^{m} p(y_i \mid x_i, \theta)$

$$= \prod_{i=1}^{m} (\sigma(\theta^T x_i))^{y_i} \cdot [1 - \sigma(\theta^T x_i)]^{1-y_i}$$

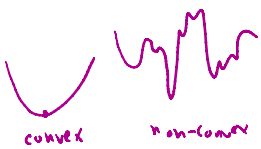
MLE: $\arg\max_\theta L(\theta)$

use log likelihood: $\arg\min_\theta -\log(L(\theta))$ — avoid underflow problems with product

why is MLE better than MSE?

1. consistency, no bias
2. always can find global minimum (convex)

MSE is non-convex



gradient descent to minimize:

$$\theta_{ML} = \arg\min_\theta -\log L(\theta)$$

$$= \arg\min_\theta \sum_{i=1}^{m} -\log\left(-y_i \log\underbrace{(\sigma(\theta^T x))}_{J_i(\theta)} - (1-y_i)\log(1-\sigma(\theta^T x_i))\right)$$

$$\frac{\partial J_i(\theta)}{\partial \theta} = \begin{bmatrix} \frac{\partial J_i(\theta)}{\partial \theta_1} \\ \frac{\partial J_i(\theta)}{\partial \theta_n} \end{bmatrix}$$

$\left[\frac{\partial J_i(\theta)}{\partial \theta_p}\right]$

using chain rule

$$\frac{\partial J_i(\theta)}{\partial \theta} = -y_i \frac{1}{\sigma(\theta^T x)} \cdot \frac{\partial(\sigma(\theta^T x))}{\partial \theta} - (1-y_i)\frac{-1}{1-\sigma(\theta^T x)} \cdot \left(\frac{-\partial(\sigma(\theta^T x))}{\partial \theta}\right)$$

need to compute gradient

$$\sigma'(z) = \frac{e^{-z}}{(1+e^{-z})^2} \quad \text{derivative of } \sigma(z) = \frac{1}{1+e^{-z}}$$

$$= \sigma(z)(1-\sigma(z))$$

$$\frac{\partial(\sigma(\theta^T x))}{\partial \theta} = \sigma(\theta^T x)(1-\sigma(\theta^T x))$$

$$= -y_i(1-\sigma(\theta^T x_i))x_i - (1-y_i)\sigma(\theta^T x_i)(-x_i)$$

$$= -y_i x_i + \sigma(\theta^T x_i)x_i$$

$$= \sigma(\theta^T x_i - y_i)x_i$$

So, $J(\theta) = -\log L(\theta)$

$$\frac{\partial J(\theta)}{\partial \theta} = \sum_{i=1}^{n}\left[\sigma(\theta^T x_i - y_i)x_i\right]$$

initialize $\theta = 0$
for $t = 1 \ldots T$ $\theta = \theta - \sum \frac{\partial J(\theta)}{\partial \theta}$ gradient descent
output θ

guarantees?
if convex, with low enough ϵ, always converges
if non-convex, no guarantees

regularization: add term to avoid overfitting

L^2 norm regularization $\lambda \|\theta\|_2^2$

$$\operatorname{argmin}_\theta J(\theta) + \lambda\|\theta\|_2^2$$

or

$$\operatorname{argmin}_\theta J(\theta) \quad \text{subject to constraint } \|\theta\|_2 \leq C$$

constrained optimization

rewrite as unconstrained:

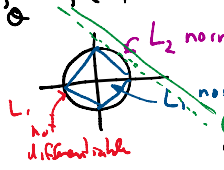
$$\operatorname{argmin}_\theta J(\theta) + \lambda\|\theta\|_2^2 - \lambda C$$

constant

L^1 vs L^2 regularization

$\operatorname{argmin}_\theta f(\theta) + \lambda\|\theta\|_2^2 \rightarrow \operatorname{argmin}_\theta f(\theta)$ with constraint $\|\theta\|_2^2 \leq C$

$\operatorname{argmin}_\theta f(\theta) + \lambda\|\theta\|_1 \rightarrow \operatorname{argmin}_\theta f(\theta)$ with constraint $\|\theta\|_1 \leq C$



constraint vs. unconstrained
L_1 pushes to corner → sparse (some 0s)
L_2 preserves high dimensionality