L4

Recap: GD.

Today: SGD Model Regularization

Gradient Descent:

$\{x_i, y_i\}$ $i = 1 \cdots n$

Find linear function f that $y_i \approx f(x_i)$

Alg 1: Matrix Inversion

$y = Xw$ $w = \underbrace{(X^T X)^{-1}}_{\text{slow}} X^T y$

Alg 2: Gradient Descent

Iterate:

$$w \leftarrow w - \alpha \nabla \underline{L(w)}$$ MSE

$$L = \frac{1}{2} \|y - Xw\|_2^2$$

(Constant) assume $n = 2$

$$w_{k+1} \leftarrow w_k + \alpha_k X^T (y - X w_k)$$

Pros: ① efficient learning time $O(ndT)$ $T = \log_{\frac{1}{\rho}} \frac{\|w^*\|_2}{\varepsilon}$

$\rho = \frac{1 - \frac{1}{L}}{1 + \frac{1}{L}}$ $(0 \sim 1)$

② Simple to implement

Cons: ① possible it stuck in local minimum

② Need to choose α, T

↳ can be set via "line-search"

③ Requires multiple passes over data (Not Memory efficient)

④ $O(nd)$ can be impacted by large n, d

What if dataset is too big that can't fit in Mem.

$\Downarrow$

SGD

Stochastic Gradient Descent (SGD)

"Back Propagation"

① How to speed up GD ?

GD: $w_{k+1} = w_k + \alpha_k X^T(y - Kw_k)$

$$= w_k + \alpha_k \sum_{i=1}^{n} (y_i - \langle w_{ki}, x_i \rangle) x_i$$

$$= w_k + n\alpha_k \underbrace{\frac{1}{n} \sum_{i=1}^{n} (y_i - \langle w_{ki}, x_i \rangle) x_i}$$

Average of $(y - \langle w_{ki}, x_i \rangle) x_i$

$\|$

idea: instead of computing average over all data points, pull a subset S uniformly at random.

(SGD) $W_{k+1} = W_k + \alpha_k \sum_{i \in S} (y_i - \langle W_k, x_i \rangle) x_i$

"Stochastic Gradient Descent"

size of S $\longrightarrow$ whatever you like (even a single point)

more points, more variance, less bias.

trade-off

SGD (single data point) 单点

① choose $i \in \{1, 2, \dots, n\}$

(x_i, y_i)

② $W_{k+1} = W_k + \alpha_k x_i (y_i - \langle W_k, x_i \rangle)$

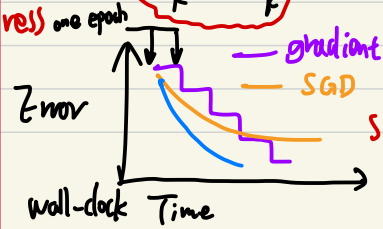
Running Time: $O(d, \#epochs)$

$\frac{1}{k}$ is choose good.

it won't be too small that we can't make progress

if learning rate $\alpha_k \propto \frac{1}{k}$

如果选择learning rate, then #epochs $\propto \frac{1}{\varepsilon}$



— sth better than SGD (SVRG, SAG ...)

SGD其实也不是平滑的，只是相比更平滑，每个epoch时间更小

Model Selection

Is linear Model the right thing to do?

problems:

① Data X (Nonlinearity), label y is a nonlinear function of X

Solution: preprocess X to $[x^1, x^2, x^3, \dots, x^d, \dots, x^{-1}, \dots \exp(-x)]$ 一次方，二次方一处理

在比较这些，看哪个更好

d选得太复杂 ⇒ New problem: Overfitting 过拟合

不好控制（次数d太大 --）

② Challenge in very large dataset

not all feature are relevant 不是所有data都有关.

eg. glucose prediction, we only need a subset of

{height, age, sugar level, ——}

③ insufficient data（不可逆）

Linear Regression $W = (X^TX)^{-1}X^TY$

$d \times d$ matrix

invertible when $\text{rank}(X^TX) = d$

$$\begin{bmatrix} | & | & & | \\ x_1 & x_2 & \cdots & x_n \\ | & | & & | \end{bmatrix} \begin{bmatrix} - & x_1 & - \\ - & x_2 & - \\ & \vdots & \\ - & x_n & - \end{bmatrix} \qquad X^TX = \sum_{i=1}^{n} x_i x_i^T$$

invertible only when $n \geq d$

？怎么解决？

#

Solution to overfitting : model selection

training dataset

$(x_1, y_1) \ldots (x_n, y_n)$

pretend data unknown but follow relationship

假定存在个真实模型：

$y = t(x) + \underset{\text{noise}}{\varepsilon}$ "True model" real function

Really, we care about

$$\underline{\text{TEST MSE}} = E_D\left(y - f(x)^2\right)$$

model we learn

distribution of data

simulated by hold-out set of training datapoints

已有所有数据的一部分

避免人为选择的影响

To avoid artifact in constructing hold-out set, repeat simulation k times "k-fold cross validation" 见表

$$\underline{\text{TEST MSE}} = E(y - f(x))^2 = E(t(x) + \varepsilon - f(x))^2$$

$$= E(\varepsilon^2) + E[(t(x) - f(x))^2] + 2\underbrace{E(\varepsilon)}_{0} E\, t(x) - f(x)$$

typically zero mean

不能用训练的数据去算 MSE，一部分训练、一部分 test.

真实模型 $t(x) + \varepsilon$

$$= E(\varepsilon^2) + E[(t(x) - f(x))^2]$$

$$E(z^2) = (E(z))^2 + Var(z)$$

$$E(\varepsilon^2) + [E(t(x) - f(x))]^2 + Var(t(x) - f(x))$$

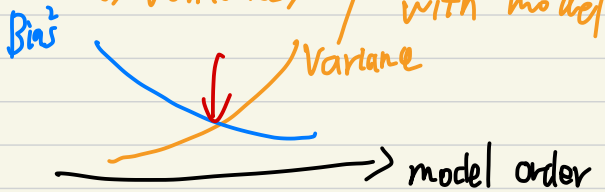
TEST MSE > $E(\varepsilon^2) + \underbrace{Bias^2 + Variance}_{ERROR}$

1) Noise

填充模型的方差

2) $Bias^2$, decrease with model complexity

3) Variance, ↑ with model complexity



k-fold:

将 dataset 分为 K 份，每次循环挑一个份为 test，剩下 k-1 份为 train，然后挑出结果最好的 1 份为模型

对数据为一化处理，得到更好结果，因为每个参数的大小不一样会导致权重差别很大，实际上它们一样重要。