机器学习的数学原理

线性代数基本运算

1、矩阵求导：

$\frac{\partial(X^TA)}{\partial X}=A$，$\frac{\partial(XA)}{\partial X}=A^T$，$\frac{\partial(AX)}{\partial X^T}=A$ $\frac{\partial(AX)}{\partial X}=A^T$

$\frac{\partial(X^TAX)}{\partial x}=AX+A^TX=(A+A^T)X$ $\frac{\partial w^Tw}{\partial w}=2w$

2、求逆矩阵

$A^{-1}=\frac{A^*}{|A|}$，$A^*=\begin{bmatrix} A_{11} & A_{21} & \cdots & A_{n1} \\ A_{12} & A_{22} & \cdots & A_{n2} \\ \vdots & \vdots & & \vdots \\ A_{1N} & A_{1n} & & A_{nn} \end{bmatrix}$

例：$A=\begin{bmatrix} 2 & 3 \\ 1 & 4 \end{bmatrix}$，$A^*=\begin{bmatrix} 4 & -3 \\ -1 & 2 \end{bmatrix}$

$\therefore A\cdot A^*=\begin{bmatrix} 2 & 3 \\ 1 & 4 \end{bmatrix}\begin{bmatrix} 4 & -3 \\ -1 & 2 \end{bmatrix}=\begin{bmatrix} 5 & 0 \\ 0 & 5 \end{bmatrix}$

3、求矩阵特征值与特征向量

$Ax=\lambda x \Rightarrow Ax=\lambda Ex \Rightarrow (A-\lambda E)x=0$

解 $|A-\lambda E|=0$

跟据得到的 λ，利用 $(\lambda E-A)x=0$

对 X 进行赋值得到 eigen value 与 eigen vector.

4、$cov(X,Y)=E(XY)-E(X)E(Y)$

Lecture 1. Linear Regression.

Given $D=\{(x_1,y_1)(x_2,y_2)\cdots(x_n,y_n)\}$, $x_i\in R^d$. $y_i\in R$

Linear Regression model: $f(x)=w^Tx+b$. $w\in R^d$.

✱ How to define $w.b \to$ by Empirical Risk Minimization

(w Loss function (squared Loss))

$$L(w,b)=\min_{w,b}\sum_{i\in[n]}(f(x_i)-y_i)^2=\min_{w,b}\left(\sum_{i\in[n]}(w^Tx_i+b-y_i)^2\right)$$

Note: 标量对向量求导是对每一维求导.

$$\begin{cases}\frac{\partial L}{\partial w}=2\sum_{i\in[n]}(w^Tx_i+b-y_i)x_i\\ \frac{\partial L}{\partial b}=2\sum_{i\in[n]}(w^Tx_i+b-y_i)\end{cases}\Longrightarrow \text{Here we can use gradient descent.}\quad \begin{matrix}w\leftarrow w-\alpha\cdot\frac{\partial L}{\partial w}\\ b\leftarrow b-\alpha\cdot\frac{\partial L}{\partial w}.\end{matrix}$$

For Linear Regression. we can use closed form solution.

(Least Squares)

$$X=\begin{bmatrix}x_1^T,1\\ x_2^T,1\\ \vdots\\ x_n^T,1\end{bmatrix}\in R^{n\times(d+1)},\quad \hat{w}=\begin{bmatrix}w\\ b\end{bmatrix}\in R^{d+1},\quad y=\begin{bmatrix}y_1\\ y_2\\ \vdots\\ y_n\end{bmatrix}\in R^n$$

$$L(\hat{w})=(y-X\hat{w})^T(y-X\hat{w})$$

$$\therefore\frac{\partial L}{\partial\hat{w}}=\frac{\partial(y^Ty-y^TX\hat{w}-\hat{w}^TX^Ty+\hat{w}^TX^TX\hat{w})}{\partial\hat{w}}$$

$$=-X^Ty-X^Ty+2X^TX\hat{w}$$

$$=2X^TX\hat{w}-2X^Ty$$

Let $\frac{\partial L}{\partial\hat{w}}=0\Rightarrow X^TX\hat{w}=X^Ty$

$$\therefore\hat{w}=(X^TX)^{-1}X^Ty.$$

Note that if we want to have this solution form, we need to X^TX no singular. Later we give how to define a matrix singular.

X^TX can be singular, when

① $d+1 > n$. $Rank(X^TX) \le Rank(X) \le \min(n, d+1) \le n < d+1$

then not full rank (parameter > samples)

② X has repeated columns/rows ?

$n \ge d+1$. $rank(X) < d+1$, not full rank.

Note: $X^TX = U\Lambda U^T = U\begin{bmatrix}\lambda_1 & & \\ & \ddots & \\ & & \lambda_n\end{bmatrix}U^T$. real-symmetric

$\Rightarrow \forall V \in R^{d+1}, \forall V \in R^{d+1}, V \neq 0, V^TX^TXV = \|XV\|^2 \ge 0$.

if λ_{d+1} is close to 0, X^TX close to singular $(X^TX)^{-1} = U\Lambda^{-1}U^T = U\begin{bmatrix}\frac{1}{\lambda_1} & & \\ & \ddots & \\ & & \underbrace{\frac{1}{\lambda_n}}_{\to +\infty}\end{bmatrix}U^T$

numerical instability.

We introduce L_2 Regularization. ← Ridge

$\min_{\hat{w}} L(\hat{w}) + \lambda\|\hat{w}\|^2$, where $\|\hat{w}\|^2 = w^Tw = \sum_{i\in[n]} w_i^2$

↳ constant parameter (超参数, $\lambda > 0$)

Let $J(\hat{w}) = (y - X\hat{w})^T(y - X\hat{w}) + \lambda\hat{w}^T\hat{w}$

$\frac{\partial J}{\partial \hat{w}} = 2X^TX\hat{w} - 2X^Ty + 2\lambda\hat{w} = 0$

$\Rightarrow (X^TX + \lambda)\hat{w} = X^Ty$

$\hat{w} = (X^TX + \lambda)^{-1}X^Ty$.

There is another version of Regularization L_1

L_1 regularization: $L(\hat{w}) + \lambda\|\underset{\sim}{w}\| \sum_{i\in[n]} |w_i|$

⇐

will cause many of parameter closed to 0.

Sparse.

Lasso Regression

Lecture2、logistic Regression

Apply to: Binary classification, $y \in \{0,1\}$, $x \in R^d$.

$f(x) = w^T x + b$, $w \in R^d$, $b \in R$. $f(x) \in R \to \{0,1\}$

Here we use sigmoid function.

$$\sigma(z) = \frac{1}{1+e^{-z}}, \quad \sigma(w^T x+b) = \frac{1}{1+e^{-(w^T x+b)}} \in (0,1)$$

$\because P(y=1|x) = \sigma(w^T x+b)$

$P(y=0|x) = 1-\sigma(w^T x+b)$

$\therefore MLE = \prod_{i\in[n]} P(y=y_i|x=x_i)$

$$= \prod_{i\in[n]} [\sigma(w^T x_i+b)]^{y_i} [1-\sigma(w^T x_i+b)]^{1-y_i}$$

$\therefore \log MLE = \sum_{i\in[n]} y_i \log(\sigma(w^T x_i+b)) + \sum_{i\in[n]} (1-y_i)\log(1-\sigma(w^T x_i+b))$

$\Leftrightarrow$ minimize $-\left(\sum_{i\in[n]} y_i \log(\sigma(w^T x_i+b)) + \sum_{i\in[n]} (1-y_i)\log(1-\sigma(w^T x_i+b))\right)$

Cross Entropy

Qustion is: How to optimize w, b?

$$\hat{x} = \begin{bmatrix} x \\ 1 \end{bmatrix} \in R^{d+1}, \quad \hat{w} = \begin{bmatrix} w \\ b \end{bmatrix} \in R^{d+1} \qquad 1-\sigma(z) = \sigma(-z)$$

$$CELoss = -\left(\sum_{i\in[n]} y_i \cdot \log \frac{1}{1+e^{-\hat{w}^T\hat{x}}} + \sum_{i\in[n]} (1-y_i)\cdot \log \frac{1}{1+e^{\hat{w}^T\hat{x}}}\right)$$

$$= -\left(\sum_{i\in[n]} y_i \log \frac{1+e^{\hat{w}^T\hat{x}}}{1+e^{-\hat{w}^T\hat{x}}} + \sum_{i\in[n]} (1-y_i)\log \frac{1}{1+e^{\hat{w}^T\hat{x}}}\right)$$

$$= -\left(\sum_{i\in[n]} \left(y_i \cdot \log \frac{1+e^{\hat{w}^T\hat{x}}}{1+e^{-\hat{w}^T\hat{x}}} - \log(1+e^{\hat{w}^T\hat{x}})\right)\right)$$

$$= -\left(\sum_{i\in[n]} (y_i \cdot \hat{w}^T\hat{x}_i - \log(1+e^{\hat{w}^T\hat{x}_i}))\right)$$

$$\frac{\partial L}{\partial \hat{w}} = -\left(\sum_{i\in[n]} y_i \cdot \hat{x}_i - \frac{\hat{x}_i e^{\hat{w}^T\hat{x}_i}}{1+e^{\hat{w}^T\hat{x}_i}}\right)$$

$$= -\left(\sum_{i\in[n]} (y_i - P(y=1|x_i))\hat{x}_i\right)$$

Let $\frac{\partial L}{\partial \hat{w}} = 0 \Rightarrow$ when $y_i = P(y=1|x_i)$

when $y \in \{-1, 1\}$,

(1) $y=1$ 时, $P(y=1|x_i) = \sigma(w^T x_i + b)$

此时 $l(x_i, y_i) = -\log \frac{1}{1+e^{-(w^T x_i + b)}} = \log(1+e^{-y_i(w^T x_i + b)})$

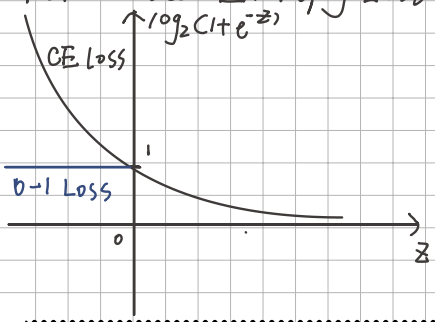
(2) $y=-1$ 时, $P(y=-1|x_i) = 1 - \sigma(w^T x_i + b)$

此时 $l(x_i, y_i) = -\log \frac{1}{1+e^{w^T x_i + b}} = \log(1+e^{-y_i(w^T x_i + b)})$

∴ 无论 y 取 1 还是 -1, $L(w,b) = \sum \log(1+e^{-y_i(w^T x_i + b)})$

∴ $L(w,b) = \sum_{i \in [n]} \log(1+e^{-y_i(w^T x_i + b)})$

For cross Entropy Loss



$z_i = y_i(w^T x_i + b)$ $\begin{cases} > 0, \text{分对了} \\ < 0, \text{分错了} \end{cases}$

CE Loss is an upper bound.

More common case

⟹ multi-class classification.

(1) Background: Assume $y \in \{1, 2, 3, \cdots, k\}$, $x \in R^d$.

(2) Method: Softmax Regression

(3) Mechanism:

① Define k models. $f_k(x) = w_k^T x + b_k$, $\forall k \in [K]$

② Find $f_k(x_i) > f_j(x_i)$, $\forall j \neq k$, predict $y_i = k$.

(4) Details:

$P(y=k|x) = \frac{e^{f_k(x)}}{\sum_{j \in [K]} e^{f_j(x)}} \geq 0$ → $\sum P(y=k|x) = 1$

→ when $f_k(x) \gg f_j(x)$, $P(y=k|x) \to 1$, $P(y=j|x) \to 0$

MLE: maxmize $\sum_{i \in [n]} \log P(y=y_i|x=x_i) = \sum_{i \in [n]} \log \frac{e^{w_i^T x_i + b_i}}{\sum_{j \in [K]} e^{w_j^T x_j + b_j}}$

when $k=2$, it seems like the logistic Regression.

And it doesn't matter what the label look like.

Assume $y \in \{1, 2\}$

$$P(y=1|x) = \frac{e^{w_1^T x + b_1}}{e^{w_1^T x + b_1} + e^{w_2^T x + b_2}} = \frac{1}{1 + e^{-[(w_1^T - w_2^T)x + (b_1 - b_2)]}}$$

let $w^T = w_1^T - w_2^T$, $b = b_1 - b_2$

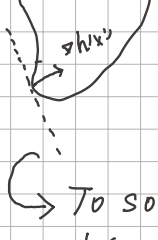
∴ 此时, $P(y=1|x) = \phi(wx+b)$

Lecture 3 Support Vector Machine.

Background: Constrained Optimization

(1) Equality Constraint

Goal: $\min_x f(x)$, subject $h(x)=0$



① $\forall$ point X on the constrait surface $h(x)=0$ we have $\nabla h(x)$ orthogonal to the surface

② For a local minimum X^*, $\nabla f(X^*)$ must be orthogonal to the surface.

↳ To solve this problem, we introduce Lagrange function

$L(x,\lambda) = f(x) + \lambda h(x)$ (λ is the Lagrange multiplier)

when we have more constraints. (such as k)

$$L(x,\lambda) = f(x) + \sum_{i\in[k]} \lambda_i h_i(x)$$

if x^* is the local minimum.

we have $\begin{cases} \nabla_x L(x^*,\lambda) = 0 \text{ (because } f(x^*) + \lambda h(x^*) = 0 \\ \nabla_\lambda L(x^*,\lambda) = 0 \Rightarrow \text{because } h(x) = 0 \end{cases}$

(2) Inequality Constraints.

Goal: $\min_x f(x)$, Subject $h(x) \le 0$

① $\forall$ point X on the surface, $\nabla g(x)$ orthogonal to the surface.

② For a local minimum. X^*

(i) X^* is on the surface, $-\nabla f(x^*)$, must be the same direction to the $\nabla g(x)$

$\exists \mu > 0$. s.t. $\nabla f(x^*) + \mu \cdot \nabla g(x^*) = 0$

(ii) X^* is within the surface, that $g(x^*) < 0$

we only need $f(x^*) = 0$.

$\exists \mu = 0$ s.t $\nabla f(x^*) + \mu \cdot \nabla g(x^*) = 0$

Lagrange function: $L(x, u) = f(x) + u \cdot g(x)$

if x^* is local minimum $\Rightarrow$ $\begin{cases} \nabla_x L(x^*, u) = 0 \\ u \geq 0 \\ g(x^*) \leq 0 \\ u \cdot g(x^*) = 0 \end{cases}$ → kkt conditions.

(3) The most common cases:

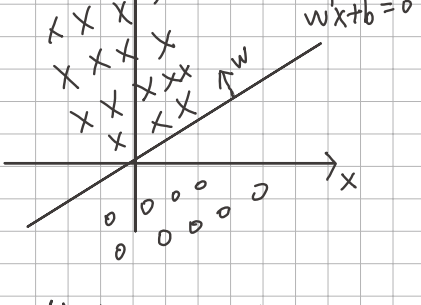
$\min_x f(x)$. s.t. $h_i(x) = 0$. $i = 1, 2, \cdots, k$

$g_j(x) \leq 0$. $j = 1, 2, \cdots, l$

$L(x, \lambda, u) = f(x) + \sum_{i \in [k]} \lambda_i h_i(x) + \sum_{j \in [l]} u_j g_j(x)$

x^* is local minimum $\begin{cases} \nabla_x L = 0 \\ h_i(x^*) = 0 \text{. } g_j(x^*) \leq 0 \\ u_j \geq 0 \\ \sum u_j g_j(x^*) = 0 \end{cases}$

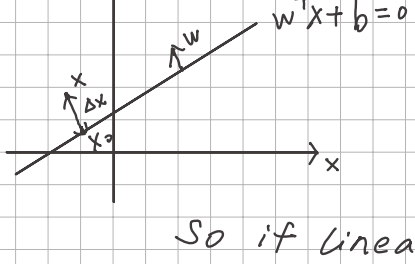
SVM Introduction



(1) Goal: maximize margin Criterion

(2) maximize margin (which refers to the minimum distance to the hyperplace $w^T x + b = 0$ over all x_i, $i \in [n]$

distance calculation



$\begin{cases} x_0 + \Delta x \cdot \frac{w}{\|w\|} = x \quad ① \\ w^T x_0 + b = 0 \quad ② \end{cases}$

$w^T \cdot ① \Rightarrow w^T x_0 + \Delta x \|w\| = w^T x$

$\therefore \Delta x \cdot \|w\| = w^T x + b$

$\Rightarrow \Delta x = \frac{w^T x + b}{\|w\|}$

So if linear Separable,

we have $r_i = y_i \cdot \Delta x_i = \frac{y_i (w^T x_i + b)}{\|w\|} \geq 0$.

So we just need to minimize $\gamma = \min_{i\in[n]} r_i = \min_{i\in[n]} \frac{y_i(w^T x_i + b)}{\|w\|}$

which means that. $\max_{w,b,\gamma}$, s.t. $\frac{y_i(w_i^T x + b)}{\|w\|} \geq \lambda$

Assume there is (x_0, y_0). $\lambda = \frac{y_0(w_i^T x_0 + b)}{\|w\|}$, geometric margin

约分 $\|w\|$. we can get that $y_i(w_i^T x + b) \geq y_0(w_i^T x_0 + b)$

let 1 ← functional margin

So we can have:

$\max_{w,b} \frac{1}{\|w\|}$, s.t. $y_i(w_i^T x + b) \geq 1$

$\Leftrightarrow \min_{w,b} \frac{1}{2}\|w\|^2$, s.t. $y_i(w_i^T x + b) \geq 1$. $\forall i \in [n]$

Here we can use Lagrange Function to solve this prob.

$L(w, b, \alpha) = \frac{1}{2}\|w\|^2 + \sum_{i\in[n]} \alpha_i (1 - y_i(w_i^T x_i + b))$ $(\alpha_i \geq 0)$

So $p^* = \min_{w,b}\left(\max_{\alpha\geq 0} \frac{1}{2}\|w\|^2 + \sum_{i\in[n]} \alpha_i (1 - y_i(w_i^T x_i + b))\right)$

Note that if $1 - y_i(w_i^T x_i + b) > 0$

$\alpha \to +\infty$. $\max_{\alpha} L(w, b, \alpha) \to +\infty$

So if p^* exists, $1 - y_i(w_i^T x_i + b) < 0$.

we can use KKT to solve this problem

$$\begin{cases} \frac{\partial L}{\partial x} = 0 \\ 1 - y_i(w^T x_i + b) \leq 0 \\ \alpha_i \geq 0 \\ \sum \alpha_i g(x_i) = 0 \end{cases}$$

⇒ But α_i is dependent on w, b so it's hard to determine α with w, b, we can try to solve the dual problem.

⇒ Dual Problem (对偶问题)

In this problem, we can ensure $d^* = p^*$

① $p^* = \min_{w,b} \max_{\alpha} \left(\frac{1}{2}\|w\|^2 + \sum_{i\in[n]} (1 - y_i(w_i^T x_i + b))\right)$

② $d^* = \max_{\alpha} \min_{w,b} \left(\frac{1}{2}\|w\|^2 + \sum_{i\in[n]} (1 - y_i(w_i^T x_i + b))\right)$

$d^* = p^*$ Here. Next we try to solve this problem.

$$d^* = \max_{\alpha} \cdot \min_{w,b} \frac{1}{2}\|w\|^2 + \sum_{i\in[n]} \alpha_i (1 - y_i(w_i^T x_i + b))$$

$$= \max_{\alpha} \min_{w,b} \left(\frac{1}{2}\|w\|^2 + \sum_{i\in[n]} \alpha_i - \sum_{i\in[n]} \alpha_i y_i (w_i^T x_i + b)\right)$$

$$\frac{\partial L}{\partial w} = w - \sum_{i\in[n]} \alpha_i y_i x_i = 0 \Rightarrow w = \sum_{i\in[n]} \alpha_i y_i x_i$$

$$\frac{\partial L}{\partial b} = -\sum_{i\in[n]} \alpha_i y_i = 0 \Rightarrow \sum_{i\in[n]} \alpha_i y_i = 0.$$

Then substitute w into L,

$$L(w, b, \alpha) = \frac{1}{2}\left[\sum_{i\in[n]} \alpha_i y_i x_i^T\right]\left[\sum_{j\in[n]} \alpha_j y_j x_j^T\right] + \sum_{i\in[n]} \alpha_i - \sum_{i\in[n]} \alpha_i y_i \sim,$$

$$= \sum_{i\in[n]} \alpha_i - \frac{1}{2} \sum_{i\in[n]} \sum_{j\in[n]} \alpha_i \alpha_j y_i y_j x_i x_j^T$$

外层: $\max_{\alpha \geq 0} \left(\sum_{i\in[n]} \alpha_i - \frac{1}{2} \sum_{i\in[n]} \sum_{j\in[n]} \alpha_i \alpha_j y_i y_j x_i x_j^T \right.$

$$\text{s.t.} \sum_{i\in[n]} \alpha_i y_i = 0$$

when we get α^* for d^*, we can then get w^*

$$\therefore w^* = \sum_{i\in[n]} \alpha_i^* y_i x_i$$

$$b^* = \frac{1}{y_0} - w^{*T} x_j. \quad \forall (x_j, y_j) \text{ is a support vector.}$$

Questions is that:

How to determine support vector:

$\forall i$, $1 - y_i(w_i^T x_i + b) = 0 \Rightarrow \alpha_i > 0$ (active)

$1 - y_i(w_i^T x_i + b) < 0 \Rightarrow \alpha_i = 0$ (inactive)

So. if $\alpha^* = 0$. not support vector

if $\alpha^* > 0$, then support vector.

when we encounter X.

$$f(x) = \sum_i \alpha_i^* y_i x_i^T x + b$$

we can use SMO get the optimized α^*

In some cases, the points may be not linear separable, we may use kernel trick at this time.

kernel Trick

⇒ Define the similarity of 2 points X, Z

① Linear kernel: $K(X,Z) = X^T Z$

② Polynominal kernel: $k(X,Z) = (X^T Z + 1)^p$

③ RBF kernel: $k(X,Z) = \exp\left(-\frac{\|X-Z\|^2}{2\sigma^2}\right)$

kernel: Maps X to high-dimension

such as $X = (x_1, x_2)^T \in R^2$

Define $\varphi(X) = (1, x_1, x_2, x_1^2, x_2^2, x_1 x_2) \in R^6$

Dual Problem: 替换 $\varphi^T(x_i)\varphi(x_j)$

$f(x) = \sum_{i \in [m]} \alpha_i^* y_i \varphi^T(x_i)\varphi(x) + b^*$

e.g. $x \in R^2$, $X = (x_1, x_2)^T$, $Z = (z_1, z_2)^T$

$$(X^T Z + 1)^2 = (x_1 z_1 + x_2 z_2 + 1)^2 = x_1^2 z_1^2 + x_2^2 z_2^2 + \cdots$$

$$= (1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2, \sqrt{2}x_1 x_2)^T$$

$$(1, \sqrt{2}z_1, \sqrt{2}z_2, z_1^2, z_2^2, \sqrt{2}z_1 z_2)^T$$

Gaussian Kernel:

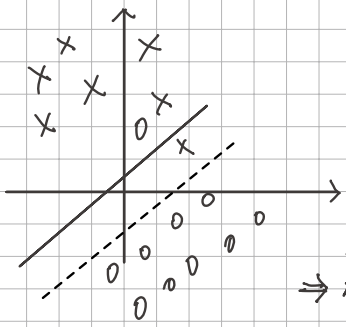
$$k(X,Z) = \exp\left(-\frac{\|X-Z\|^2}{2\sigma^2}\right) = \exp\left(-\frac{\|X\|^2}{2\sigma^2}\right) \cdot \exp\left(-\frac{\|Z\|^2}{2\sigma^2}\right) \cdot \exp\left(\frac{X^T Z}{\sigma^2}\right)$$

最后一项、Taylor 展开

$$1 + \frac{1}{1!}\cdot\left(\frac{X^T Z}{\sigma^2}\right)^1 + \cdots + \frac{1}{n!}\left(\frac{X^T Z}{\sigma^2}\right)^n = \sum_{p=0}^{\infty} \frac{1}{p!}\left(\frac{X^T Z}{\sigma^2}\right)^p$$

σ: length parameter

☆ Slack variables to handle outliners.



Soft-margin SVM: Previously we need $y_i(w_i^T x_i + b) \geq 1$

Now we introduce the slack variable $\xi_i \geq 0$, require $y_i(w_i^T x_i + b) \geq 1 - \xi_i$

Here we require ξ_i to be small

$$\Rightarrow \min_{w,b,\xi} \frac{1}{2}\|w\|^2 + C \cdot \sum_{i\in[n]} \xi_i$$

$$s.t.\ y_i(w^T x_i + b) \geq 1 - \xi_i, \quad \forall i \in [n]$$

$$\xi_i \geq 0, \forall i \in [n]$$

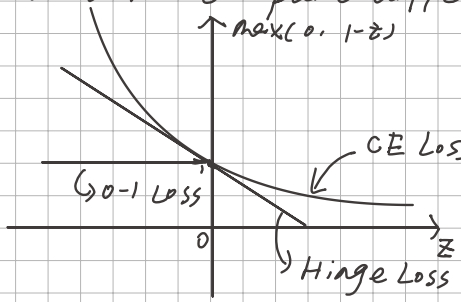
we need that: $\xi_i \geq 0$, $\xi_i \geq 1 - y_i(w^T x_i + b)$

$\Rightarrow \xi_i \geq \max(0, 1 - y_i(w^T x_i + b))$ ("=" when $1 - \sim = 0$)

Goal: $\min_{w,b} \frac{1}{2}\|w\|^2 + C \cdot \sum_{i\in[n]} \underbrace{\max(0, 1 - y_i(w^T x_i + b))}_{\text{Hinge Loss}}$.

Here we transfer it to a constrained optimization.

Next we compare different Loss:



When categorize correctly the penality of Hinge Loss is less.

Dual of soft SVM

$$L(w, b, \xi, \alpha, \beta) = \frac{1}{2}\|w\|^2 + C \cdot \sum_{i\in[n]} \xi_i + \sum_{i\in[n]} \alpha_i [1 - \xi_i - y_i(w_i^T x_i + b)] - \sum_{i\in[n]} \beta_i \xi_i$$

$\min\max L$: Dual $\max_{\alpha\geq 0, \beta \geq 0} \min_{w,b,\xi} L$.

$$\begin{cases} \frac{\partial L}{\partial w} = w - \sum_{i\in[n]} \alpha_i y_i x_i = 0 \\ \frac{\partial L}{\partial b} = -\sum_{i\in[n]} \alpha_i y_i = 0 \\ \frac{\partial L}{\partial \xi_i} = C - \alpha_i - \beta_i = 0 \end{cases}$$

we can get that $\begin{cases} w = \sum_{i \in [n]} \alpha_i y_i x_i \\ \sum \alpha_i y_i = 0 \\ \alpha_i + \beta_i = C \end{cases}$

we can substitute w into L. and we can get

$$\begin{cases} \max_{\alpha} -\sum_{i \in [n]} \alpha_i - \frac{1}{2} \sum_{i \in [n]} \sum_{j \in [n]} \alpha_i \alpha_j y_i y_j x_i^T x_j \\ \text{s.t.} \quad 0 \le \alpha_i \le C \\ \quad \sum_{i \in [n]} \alpha_i y_i = 0 \end{cases}$$. if C is small, robust to outlier.

if $\alpha_i = 0$, 分类正确

$0 < \alpha_i < C$, support vector

$\alpha_i = C$, 分类错误（好像也是支持向量）

Lecture 4. Representer Theorem (表示定理)

By default, we use $\varphi(x)$ to replace X, $\varphi(x)=[x^T, 1]^T$

$f(x)=w^T\varphi(x)$ can recover w^Tx+b

1. Linear Regression

$$Loss=\min_w \sum_{i\in[n]} (w^T\varphi(x_i)-y_i)^2+\lambda\|w\|^2$$

2. Logistic Regression

$$\min_w \sum_{i\in[n]} \log(1+\exp(-y_i w^T\varphi(x_i)))$$

3. SVM (soft-SVM)

$$\Leftrightarrow \min_w \sum_{i\in[n]} \max(0, 1-y_i(w^T\varphi(x_i)+b))+\lambda\|w\|^2$$

About these Resolution:

We Introduce ξ_i, $\xi_i\geq 0$, $\xi_i\geq 1-y_i w^T\varphi(x_i)$

At Last we can get $\begin{cases} w=\frac{1}{\lambda}\sum_{i\in[n]}\alpha_i y_i\varphi(x_i)=\sum_{i\in[n]}\alpha_i\varphi(x_i) \\ \alpha_i+\beta_i=1 \end{cases}$

$$\Rightarrow f(x)=w^T\varphi(x)=\sum_{i\in[n]}\varphi(x_i)\varphi(x_i)^T=\sum_{i\in[n]}\alpha_i k(x_i, x)$$

For linear regression.

$$\frac{\partial L}{\partial w}=2\cdot\sum_{i\in[n]}(w^T\varphi(x_i)-y_i)\cdot\varphi(x_i)+2\lambda w=0$$

$$\Rightarrow w=\frac{1}{\lambda}\sum_{i\in[n]}(y_i-w^T\varphi(x_i))\varphi(x_i)$$

$$=\sum_{i\in[n]}\alpha_i f(x_i),\ \alpha_i=\frac{1}{\lambda}(y_i-w^T\varphi(x_i))\quad !\ ?$$

$$\therefore f(x_i)=\sum_{i\in[n]}\alpha_i\varphi(x_i)^T\varphi(x)=\sum_{i\in[n]}\alpha_i k(x_i, x)$$

Here we introduce Hilbert Space.

(1) Attribute:

① $\langle f, g \rangle_H = \langle g, f \rangle_H$

② $\langle a_1 f + a_2 f, g \rangle = a_1 \langle f, g \rangle + a_2 \langle f, g \rangle$

③ $\langle f, f \rangle_H \geq 0$, "=" when $f = 0$

$\Rightarrow$ Loss is independent of regularization minizes at $\nu=0$.

$\Rightarrow f(x) = \sum_{i\in[n]} \alpha_i k(x_i, x)$

Significance:

(1) Tunes a potentially infinite-dim optimization of f into a search of $\alpha_1, \alpha_2, \cdots \alpha_n$

(2) shows that a wide range of learning algorithms have solutions expressed as weighted sum of kernel functions on finite traing data. $f(\cdot) = \sum_{i\in[n]} \alpha_i k(x_i, \cdot)$

Here we show some example.

Ridge Regression

$$\min_w J(w) = \frac{1}{2} \cdot \sum_{i\in[n]} (w^T \varphi(x_i) - y_i)^2 + \frac{1}{2} \lambda w^T w$$

$$let \ \frac{\partial J}{\partial w} = \sum_{i\in[n]} (w^T \varphi(x_i) - y_i) \cdot \varphi(x_i) + \lambda w = 0$$

$$\Rightarrow w = \sum_{i\in[n]} \alpha_i \varphi(x_i) \ (where, \ \alpha_i = \frac{1}{\lambda}(y_i - w^T \varphi(x_i)))$$

$$\therefore f(x) = \sum_{i\in[n]} \alpha_i \varphi(x_i) \cdot \varphi(x) = \sum_{i\in[n]} \alpha_i k(x_i, x)$$

$$= w^T \varphi(x)$$

$$\therefore w = \sum_{i\in[n]} \alpha_i \varphi(x_i)$$

$$let \ \Phi = \begin{pmatrix} \varphi(x_1)^T \\ \vdots \\ \varphi(x_n)^T \end{pmatrix} \in R^{n\times d} \Rightarrow W = \Phi^T \cdot \alpha \in R^d.$$

$$J(w) = \frac{1}{2}(\Phi w - Y)^T (\Phi w - Y) + \frac{\lambda}{2} w^T w \ (w = \Phi^T \alpha)$$

$$= \frac{1}{2}(w^T \Phi^T \Phi w + Y^T Y - 2 w^T \Phi^T Y) + \frac{\lambda}{2} w^T w$$

代入 $w = \Phi^T \alpha \Rightarrow J(w) = \frac{1}{2} \alpha^T \Phi \Phi^T \Phi \Phi^T \alpha + \frac{1}{2} Y^T Y - \alpha^T \Phi \Phi^T Y + \frac{\lambda}{2} \alpha^T \Phi \Phi^T \alpha$

而 $\phi(x)\phi(x)^T$

$= \begin{pmatrix} \phi(x_1) \\ \vdots \\ \phi(x_n) \end{pmatrix} (\phi(x_1) \cdots \phi(x_n)) = \begin{pmatrix} \phi(x_1)^T\phi(x_1) & \phi(x_1)^T\phi(x_2) & \\ & \ddots & \\ & & \phi(x_n)^T\phi(x_n) \end{pmatrix}$

$= \begin{pmatrix} k(x_1,x_1) & k(x_1,x_2) \cdots \\ & \ddots \\ & k(x_n,x_n) \end{pmatrix} = K$ Gram Matrix

代入 $J(w)$ 有

$J = \frac{1}{2}\alpha^T K \cdot K\alpha + \frac{1}{2}y^T y - \alpha^T K y + \frac{\lambda}{2}\alpha^T K \alpha = J(\alpha)$,

$\frac{\partial J}{\partial \alpha} = 0 \Rightarrow KK\alpha - KY + \lambda K\alpha = 0$ ①

K^{-1} ① $\Rightarrow K\alpha - Y + \lambda I\alpha = 0$

$\therefore \alpha = (K+\lambda I)^{-1} Y$

而 $f(x) = \sum_{i\in[n]} \alpha_i k(x_i, x) = k_{(x)}^T \alpha$ $\left(k(x) = \begin{pmatrix} k(x_1,x) \\ k(x_2,x) \\ \vdots \\ k(x_n,x) \end{pmatrix} \right)$

$= \underbrace{k_{(x)}^T (K+\lambda I)^{-1}} Y$

思考题

$w = (X^TX + \lambda I)^{-1} X^T y$ ①

$= (\phi_{(X)}^T \phi(X) + \lambda I)^{-1} \phi^T y$ ②

Anthony Zador

而 $f(x) = w^T\phi(x) = \phi_{(x)}^T (\phi^T\phi + \lambda I)\phi^T y$

与 $k_{(x)}^T (K+\lambda I)^{-1} Y$ 是否等价.

Lecture 5. Learning Theory

Lecture 6. Tree Model And Ensemble Learning

Previous Model: $f(x) = w^T x + b$. $y = \text{sign}(w^T x + b)$

$\Leftrightarrow$ Decision logic $\begin{cases} ① \ w^T x + b \geq 0, +1 \\ ② \ w^T x + b < 0, -1 \end{cases}$

Defilics: Unable to do classification in unlinear space.

Here we introduce the Decision Trees,

$\Rightarrow$ Definition: A decision Tree is a tree that contains a Root, internal nodes, connected by directed edges. Each non-leaf node partions the data by some features.

Then the question is that:

✡ How to select a good feature to partition?

The Answer is: Purity!

Here we use the information theory:

(1) $H(x) = -\sum p(x) \cdot \log_2 p(x) = \sum p(x) \cdot \log_2 \frac{1}{p(x)}$

$= E\left(\log_2 \frac{1}{p(x)}\right)$

$H(x) \leq \log_2 n$ (Jensen's 不等式)

(2) Cross Entropy:

$H(p, q) = \sum p(x_i) \cdot \log_2 q(x_i)$.

where $P(x)$ measures the observed data.

$q(x)$ measures the probability of y_i

$E(x_i, y_i) = -\sum_{y_i \in \{0,1\}} P(y_i) \cdot \log q(y_i)$

In the next part, we will learn to how to measure the purity of branch.

How to measure purity?

1. Information Gain.

$g(D,A) = H(D) - H(D|A)$ ⇐ select A to maximize.

D: training set; A: a feature/Attribute

suppose $A \in \{a_1, \dots, a_m\}$ m discrete volumes

$y \in \{1, 2, \dots, k\}$

$H(D) = -\sum_{k\in[k]} \frac{|C_k|}{|D|} \cdot \log \frac{|C_k|}{|D|}$, approximate $P(y=k)$ $H(y)$

$H(D|A) = \sum_{i\in[m]} \frac{|D_i|}{|D|} \cdot H(D|A=a_i)$

$= \sum_{i\in[m]} \frac{|D_i|}{|D|} \left(-\sum_{k\in[k]} \frac{|D_i \cap C_k|}{|D_i|} \cdot \log \frac{|D_i \cap C_k|}{|D_i|} \right)$ $H(y|A=a_i)$

So we just need the A to maximize $H(D) - H(D|A)$

---

2. Information Gain Ratio

$g_R(D,A) = \frac{g(D,A)}{H_A(D)}$, where $H_A(D) = -\sum_{i\in[m]} \frac{|D_i|}{|D|} \cdot \log_2 \frac{|D_i|}{|D|}$

when each $A=a_i$ has equal prob. $H_A(D) = \log_2 m$

Penalize A with a large m.

---

3. Gini Index

$Gini(D) = \sum_{i\in[k]} \frac{|C_k|}{|D|} \left(1 - \frac{|C_k|}{|D|}\right)$. approximates $\sum_{i\in[k]} P(y=k)(1-P(y=k))$

$= 1 - \sum_{i\in[k]} \left(\frac{|C_k|}{|D|}\right)^2$ $\begin{cases} \max = 1 - (\frac{1}{k})^2 \cdot k = 1 - \frac{1}{k} \\ \min = 0 \Leftarrow \text{when purity.} \end{cases}$

---

4. L_2 Loss

For Regression, just use L_2 Loss to measure purity

$\bar{y}_{D_i} = \frac{1}{|D_i|} \sum_{j\in D_i} y_j$

$L(D,A) = \sum_{i\in[m]} \left[\sum_{j\in D_i} (y_j - \bar{y}_{D_i})^2 \right]$

$\operatorname{argmin} L(D,A)$

Some bagging Methods.

1. Bootstrap

① Sample n points from D w/ replacement

② Repeat ① for T times, get $\hat{D}_1, \hat{D}_2, \cdots, \hat{D}_T$

③ $\hat{f}(x) = \frac{1}{T}\sum_{t\in[T]} f(x, \hat{D}_t)$ as the bagged model

✡ Bootstrap has no theoretical guarantee! Empirically can reduce the test error greatly.

⇓

Random Forest (the most successful bagging model)

Algorithm:

- for t in $[T]$:
  - (1) sample n points $\hat{D}_t$ from D w/ replacement.
  - (2) Sample $d' < d$ features F' from F without replacement.
  - (3) Build a full decision tree on $\hat{D}_t, F$ (can do some pruning to minimize out-of-bag error)
- End for

Then we can average all trees.

(1) If Regression, we use the mean as result.

(2) If categorization, we use the mode as result.

Boosting (Reduce Bias)

Application: Adapt to low variance high bias models.

✡ AdaBoost: $D = \{(x_1, y_1), \cdots, (x_n, y_n)\}$, $y \in \{-1, 1\}$, $x \in R^d$

Input: D, a weak learning algorithm A.

Algorithm:

(1) Initialize sample weights, $w_i^{(0)} = \bar{w}_i^{(0)} = \frac{1}{n}$, $\forall i \in [n]$

(2) For t in [T]:

a) use D and $\{w_i^{(t)}\}$ to train A. get a model $f_t(x): R^d \to \{-1, 1\}$

b) Evalute $f_t(x)$ weighted classification error on D

$$e_t = \sum_{i \in [n]} w_i^{(t)} \cdot 1(f_t(x) \neq y_i)$$

c) Compute a weight α_t for $f_t(x)$. $\alpha_t = \frac{1}{2} \cdot \log \frac{1 - e_t}{e_t}$

d) Update sample weights: $\bar{w}_i^{(t+1)} = \bar{w}_i^{(t)} \cdot \exp(-\alpha_t \cdot y_i \cdot f_t(x_i))$

Normalization: $w_i^{(t+1)} = \bar{w}_i^{(t+1)} / \sum_{j \in [n]} \bar{w}_j^{(t+1)}$

(3) Combine T classifiers linearly.

$$g(x) = \text{sign}\left(\sum_{t \in [T]} \alpha_t \cdot f_t(x)\right)$$

Now let's focus on $\alpha_t = \frac{1}{2} \log \frac{1 - e_t}{e_t}$

First, ① $e_t \downarrow$, $\alpha_t \uparrow$

② when $e_t > 0.5$, $\alpha_t < 0$

$e_t < 0.5$, $\alpha_t > 0$

(2) let's focus on $\bar{w}_i^{(t+1)} = \bar{w}_i^{(t)} \cdot \exp(-\alpha_t \cdot y_i \cdot f_t(x_i))$

$y_i \neq \text{sign}(\sum_{t \in [T]} \alpha_t \cdot f_t(x_i))$, get larger weights.

(3) Because $w_i^{(t+1)} = \bar{w}_i^{(t+1)} / \sum_{j=1}^{n} \bar{w}_j^{(t+1)} \Rightarrow \sum_{i \in [n]} w_i^{(t+1)} = 1$

Next we will discuss about More common perspective of boosting, which is called Additive Model.

Additive Model (A general paradigm for boosting)

$g(x) = \sum_{t \in [T]} \alpha_t \cdot f_t(x)$. Define $g_t(x) = \sum_{j \in [t]} \alpha_j \cdot f_j(x)$

At step t, keep $g_{t-1}(x)$ fixed. Then Learn $\alpha_t, f_t(x)$ by minimizing $\sum_{i \in [n]} L(y_i, g_{t-1}(x_i) + \alpha_t \cdot f_t(x_i))$

Finally, $g(x) = g_T(x)$

AdaBoost is an additive model using exponentially loss, $L(y, f(x)) = \exp(-y f(x))$. $y f(x)$ only takes $1/-1$

So, $L(y, f(x))$ only takes $e / \frac{1}{e}$

AdaBoost Derivation

At step t, we already have $g_{t-1} = \sum_{j \in [t-1]} \alpha_j f_j(x)$

we need to optimize: $\min_{\alpha_t, f_t} \sum_{i \in [n]} \exp(-y_i (\underbrace{g_{t-1}(x) + \alpha_t f_t(x_i)}_{g_t(x_i)}))$

Here we define $\exp(-y_i g_{t-1}(x_i))$ to be $\bar{w}_i^{(t)}$

$= \prod_{j \in [t-1]} \exp(-y_i \cdot \alpha_j f_j(x_i))$

$= \bar{w}_i^{(t-1)} \cdot \exp(-y_i \cdot \alpha_t f_t(x_i))$

$\therefore$ Goal $\Leftrightarrow \min_{\alpha_t} \cdot \min_{f_t} \sum_{i \in [n]} w_i^{(t)} \cdot \exp(-y_i \alpha_t \cdot f_t(x_i))$

Fix α_t, optimize f_t. In AdaBoost, we just train f_t using its original loss and simple weights $w_i^{(t)}$

After Solving f_t, we optimize

$\min_{\alpha_t} \sum_{i \in [n]} w_i^{(t)} \exp(-y_i \alpha_t \cdot f_t(x_i))$

Next we try to solve this problem and get the Related Results.

$$\min_{\alpha_t} \sum_{i\in[n]} w_i^{(t)} \exp(-\alpha_t \cdot y_i f_t(x_i))$$

$$= \min_{\alpha_t} \sum_{y_i = f_t(x_i)} w_i^{(t)} \exp(-\alpha_t \cdot \underbrace{y_i f_t(x_i)}_{1}) + \sum_{y_i \neq f_t(x_i)} w_i^{(t)} \exp(-\alpha_t \cdot \underbrace{y_i f_t(x_i)}_{-1})$$

$$= \min_{\alpha_t} \Big(\sum_{y_i = f_t(x_i)} w_i^{(t)} \exp(-\alpha_t) + \sum_{y_i \neq f_t(x_i)} w_i^{(t)} \exp(-\alpha_t) \Big)$$

$$+ \Big(\sum_{y_i \neq f_t(x_i)} w_i^{(t)} \exp(\alpha_t) - \sum_{y_i \neq f_t(x_i)} w_i^{(t)} \exp(-\alpha_t) \Big)$$

$$= \min_{\alpha_t} \Big(\underbrace{\Big(\sum_{i\in[n]} w_i^{(t)} \Big)}_{1} \exp(-\alpha_t) \Big) + \Big[\Big(\sum_{y_i \neq f_t(x_i)} w_i^{(t)} \Big) \big(\exp(\alpha_t) - \exp(-\alpha_t) \big) \Big]$$

$$\Leftrightarrow \min_{\alpha_t} \underbrace{L =}\ \exp(-\alpha_t) + e_t \cdot \big[\exp(\alpha_t) - \exp(-\alpha_t) \big]$$

$$\frac{\partial L}{\partial \alpha_t} = -\exp(-\alpha_t) + e_t \big[\exp(\alpha_t) + \underbrace{\exp(-\alpha_t)}_{\beta} \big] = 0$$

$$\therefore \Big(\beta + \frac{1}{\beta} \Big) e_t = \beta$$

$$\Rightarrow (1 - e_t) \beta^2 = e_t \Rightarrow \beta = \sqrt{\frac{e_t}{1 - e_t}} = e^{-\alpha_t}$$

$$\therefore \alpha_t = \frac{1}{2} \log \frac{1 - e_t}{e_t}$$

Lecture 7 Gaussian Process

Previously, we treat w as fixed (constant) parameters. In Bayesian view, even the parameters w are R.

Prior Prob. $P(w) = N(w|0, \delta_w^2 I)$

$$P(y|x, w; \delta^2, \delta_w^2) = N(y|w^T\phi(x), \delta^2)$$

$$P(w|y, x) = \frac{P(y|x, w)\cdot P(w|x)}{P(y|x)} \to Z$$

$$\downarrow \prod_{i\in[n]} P(y_i|x_i, w)$$

$$= \frac{1}{Z}\cdot\left[\left(\frac{1}{\sqrt{2\pi\delta^2}}\right)^n \exp\left(-\frac{\sum_{i\in[n]}(y_i - w^T\phi(x_i))^2}{2\delta^2}\right)\right]\cdot\frac{1}{(\sqrt{2\pi\delta_w^2})^d}\exp\left(-\frac{w^Tw}{2\delta_w^2}\right)$$

Hence it selects the mode of the posterior distribution as a point estimation.

Goal: $\max(\log P(w|y, x)) = \log(\text{const})$

$$\Leftrightarrow \max_w -\frac{1}{2\delta^2}\sum_{i\in[n]}(y_i - w^T\phi(x_i))^2 - \frac{1}{2\delta_w^2}w^Tw$$

$$\Leftrightarrow \min_w \sum_{i\in[n]}(y_i - w^T\phi(x_i))^2 + \underbrace{\frac{\delta^2}{\delta_w^2}}_{\lambda}\|w\|^2$$

Stochastic Process

(1) Definition: A collection of (infinite/many) R.Vs along an index set.

Gaussian Process

(1) Definition: specify the joint distribution over any finite collection of R.V.s and require the joint distribution to be Gaussian.

$\{x_1, \cdots, x_n\}$ is any set of n points in the index set. Then GP $\Leftrightarrow y_1, \cdots, y_n$ have a multivariate Gaussian Distribution.

Multivariate Gaussian: $X \in R^d$

$$N(X|\mu,\Sigma) = \frac{1}{(2\pi)^{\frac{d}{2}}} \cdot \frac{1}{|\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{(x-\mu)^T\Sigma^{-1}(x-\mu)}{2}\right)$$

parameters: $d + \frac{d(d+1)}{2}$

we can get that $\begin{cases} E(x) = \mu \\ Cov(x) = E[(x-\mu)(x-\mu)^T] = \Sigma \end{cases}$

$$\Sigma_{ij} = E[(x_i-\mu_i)(x_j-\mu_j)] = cov(x_i, x_j)$$

Lamma

Any linear distribution of Gaussian Distribution R.V.s follows a Gaussian. Concatentation of Gaussian R.V.s results in a multivariate Gaussian distributed R.V.

$$a \sim N(\mu_a, \sigma_a^2) \quad b \sim N(\mu_b, \sigma_b^2) \quad \begin{bmatrix} a \\ b \end{bmatrix} \sim N(\mu, \Sigma)$$

Question: How to capture the joint distribution, need to only specify $\mu_i = m(x_i)$ $\Sigma_{ij} = k(x_i, x_j)$

Suppose n training points $\{x_1, \dots, x_n\}$. let k be Gram matrix. $K_{ij} = k(x_i, x_j) \longrightarrow$ valid kernel if $k \geq 0$ for any $\{x_1, \dots, x_n\}$

$GP(m(\ ), k(\cdot,\cdot))$. $P(y) = N(y|0, K)$

Now given a new test point X_*, we can compute joint distribution of $\begin{bmatrix} y_* \\ y \end{bmatrix} \in R^{n+1}$

$$P\left(\begin{bmatrix} y_* \\ y \end{bmatrix}\right) = N\left(\begin{bmatrix} y_* \\ y \end{bmatrix} \middle| 0, \begin{pmatrix} k(x^*, x^*) & k(x^*, x) \\ k(x^*, x) & k(x, x) \end{pmatrix}\right)$$

$P(y^*|y) = N(y_*|\mu^*, \Sigma^*)$, ⇐ How to compute this thing is important.

Given $\begin{cases} X_a \sim \mathcal{N}(\mu_a, \Sigma_{aa}) \\ X_b \sim \mathcal{N}(\mu_b, \Sigma_{bb}) \end{cases} \Rightarrow \begin{cases} \mu_{a|b} = \mu_a + \Sigma_{ab}\cdot\Sigma_{bb}^{-1}(X_b - \mu_b) \\ \Sigma_{a|b} = \Sigma_{aa} - \Sigma_{ab}\cdot\Sigma_{bb}^{-1}\Sigma_{ba} \end{cases}$

$\therefore \mu^* = k(x^*, x)^T \cdot K^{-1} y$

$\Sigma^* = k(x^*, x^*) - k(x^*)^T K^{-1} k(x^*)$

In a more realistic setting, we can only observe $\hat{y} = y + \varepsilon,\ \varepsilon \sim \mathcal{N}(0, \sigma^2 I)$

$\Rightarrow \begin{cases} \mu^* = k(x^*)^T (K + \lambda I)^{-1} y \\ \Sigma^* = k(x^*, x^*) - k^T(x^*)(K + \lambda I)^{-1} k(X_*, X) \end{cases}$

For MAP,

Goal: $\min_w \sum_{i\in[n]} (y_i - w^T\phi(x_i))^2 + \lambda w^T w$

Bayesian View MAP Function:

$y_i = w^T\phi(x_i) + \varepsilon_i,\ \varepsilon_i \sim \mathcal{N}(0, \sigma^2)$

$w \sim \mathcal{N}(0, \sigma_w^2 I)$

$Y = w^T\phi + \varepsilon \sim \mathcal{N}(\mu, \Sigma)$

$cov(y_i, y_j) = E(y_i y_j) - E(y_i)E(y_j)$

$= E((w^T\phi(x_i) + \varepsilon_i)(w^T\phi(x_j) + \varepsilon_j))$

$= E(\phi(x_i)^T w \cdot w^T \phi(x_j)) + E(\varepsilon_i \varepsilon_j)$

$= \phi(x_i)^T \underbrace{E(w w^T)}_{\sigma_w^2 I} \phi(x_j) + \sigma^2 I$

$\Rightarrow \sigma_w^2 \phi(x_i)^T \phi(x_j) + \sigma^2 I$

$\Sigma = \sigma_w^2 K + \sigma^2 I$

同理 $\mu_* = \sigma_w^2 k(x_*)^T [\sigma_w^2 K + \sigma^2 I]^{-1} y$

$= k(x_*)^T [K + \underbrace{\frac{\sigma^2}{\sigma_w^2}}_{\text{let be } \lambda} I]^{-1} y$

Bayesian Optimization (BO)

1. For black-box functions

Steps: ① Randomly sample n points $x_1, \cdots, x_n$

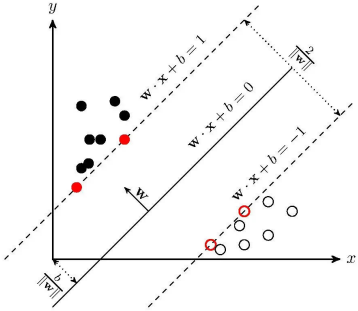
② fit a GP → Lower confidence bound

③ Use some aquisition function to select next point to evaluate

④ 迭代上述过程: use all x, y to fit a new GP

⑤ Repeat until reaching a budget.

SVM数学原理



Train Data: $T=\{(x_1,y_1)\cdots(x_n,y_n)\}$
$x_i\in R^n$, $y_i\in\{+1,-1\}$

For any datapoint:
the geometric distance $r_i=y_i\left(\frac{w}{\|w\|}\cdot x_i+\frac{b}{\|w\|}\right)$

Let $r=\min r_i$

So our Goal is that:
$\max_{w,b} r$ s.t. $y_i\left(\frac{w}{\|w\|}\cdot x_i+\frac{b}{\|w\|}\right)\geq r$.
$\Leftrightarrow y_i\left(\frac{w}{\|w\|\cdot r}\cdot x_i+\frac{b}{\|w\|\cdot r}\right)\geq 1$
$\Leftrightarrow y_i(wx_i+b)\geq 1$, $\left(w=\frac{w}{\|w\|\cdot r}\right)$

In the picture above, $\max r \Leftrightarrow \max\frac{1}{\|w\|} \Leftrightarrow \min\frac{1}{2}\|w\|^2$
$\Rightarrow$ Goal Function: $\min_{w,b}\frac{1}{2}\|w\|^2$. s.t. $y_i(wx_i+b)\geq 1$

(1) How to solve this problem?
拉格朗日乘子: $L(w,b,\alpha)=\frac{1}{2}\|w\|^2-\sum_{i=1}^{n}\alpha_i(y_i(wx_i+b)-1)$ $(\alpha_i\geq 0)$ ①
Let $\theta(w)=\max_{\alpha_i\geq 0} L(w,b,\alpha)$

Here're different conditions:
① $y_i(wx_i+b)-1<0\Rightarrow\alpha_i\to+\infty$, $\theta(w)\to+\infty$
② $y_i(wx_i+b)-1\geq 0$, >0, $\alpha_i=0$ / $=0$, α_i 任意.

上述问题 $\Leftrightarrow \min_{w,b}\theta(w)=\min_{w,b}\max_{\alpha_i\geq 0}L(w,b,\alpha)=p^*$

dual form: change min and max
$\rightarrow \max_{\alpha_i\geq 0}\min_{w,b}L(w,b,\alpha)=d^*$

$p^*=d^*$ when {① 凸优化; ② KKT条件 → $\alpha_i\geq 0$, $y_i(wx_i+b)-1\geq 0$, $\alpha_i(y_i(wx_i+b)-1)=0$}

而 Let $\frac{\partial\theta}{\partial w}=0\Rightarrow w=\sum_{i=1}^{N}\alpha_i y_i x_i$, $\sum_{i=1}^{N}\alpha_i y_i=0$

代入①有 $L=-\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j(x_i\cdot x_j)+\sum_{i=1}^{N}\alpha_i$

即 $\min_{w,b}L(w,b,\alpha)=-\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j(x_i\cdot x_j)+\sum_{i=1}^{N}\alpha_i$

求 $\min_{w,b} L(w,b,\alpha)$ 对 α 即大，即是一个对偶问题

$$\max_{\alpha} -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j (x_i \cdot x_j) + \sum_{i=1}^{N}\alpha_i$$

$$s.t.\ \sum_{i=1}^{n}\alpha_i y_i = 0.\quad \alpha_i \geq 0.$$

使用SMO解得 α^*

$$\begin{cases} w^* = \sum_{i=1}^{n}\alpha_i^* y_i x_i \\ b^* = y_j - \sum_{i=1}^{N}\alpha_i^* y_i (x_i \cdot x_j) \end{cases}$$

(2) For none-linear separable, we use soft-margin.

$$\min_{w,b,s} \frac{1}{2}\|w\|^2 + C\cdot\sum_{i=1}^{m}\xi_m$$

$$s.t.\ \begin{cases} y_i(wx_i+b) \geq 1-\xi_i \\ \xi_i \geq 0, \end{cases}$$ Actually, $\xi_i = \max(0, 1-y_i(wx_i+b))$

此时 $\begin{cases} 0<\alpha_i<C \cdot y_i(wx_i+b)=1 \\ \alpha_i = 0,\ y_i(wx_i+b)>1 \\ \alpha_i = C,\ y_i(wx_i+b)<1 \end{cases}$

(3) For Kernel Trick

$$\begin{cases} f(x) = sign\left(\sum_{i=1}^{N}\alpha_i^* y_i k(x_i, x) + b^*\right) \\ b^* = y_i - \sum_{i=1}^{N}\alpha_i^* y_i k(x_i, x_j) \end{cases}$$

How to get the α^* ⇒ SMO Algorithm.

$$W = \min_{\alpha} \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j k(x_i, x_j) - \sum_{i=1}^{N}\alpha_i$$

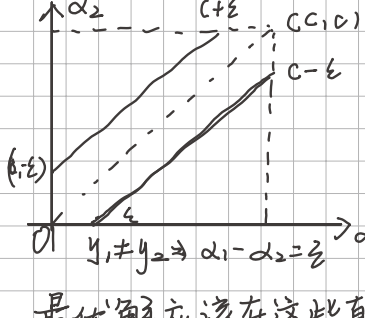
$$s.t.\ \sum_{i=1}^{n}\alpha_i y_i = 0,\ 0 \leq \alpha_i \leq C$$

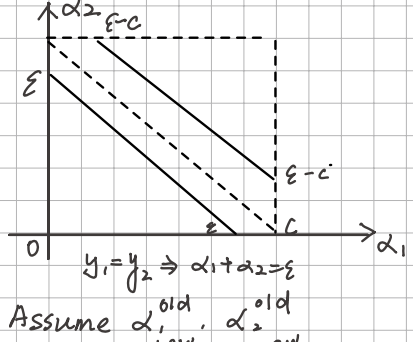
基本思想：每次只优化两个 α，假设其为 α_1, α_2

$$\therefore W = \frac{1}{2}k_{11}\alpha_1^2 + \frac{1}{2}k_{22}\alpha_2^2 + y_1y_2k_{12}\alpha_1\alpha_2 - (\alpha_1+\alpha_2) + y_1\alpha_1\sum_{i=3}^{N}y_i\alpha_i k_{i1} + y_2\alpha_2\sum_{i=3}^{N}y_i\alpha_i k_{i2} + t$$

$$s.t.\ \alpha_1 y_1 + \alpha_2 y_2 = -\sum_{i=3}^{N}\alpha_i y_i = \xi$$

这是一个两个变量二次规化问题.





最优解应该在这些直线上. Assume $\alpha_1^{old}, \alpha_2^{old}$ $\alpha_1^{new}, \alpha_2^{new}$

有 $L \le \alpha_2^{new} \le H$

而 $\begin{cases} L = \max(0, \alpha_2^{old} - \alpha_1^{old}), H = \min(C, C + \alpha_2^{old} - \alpha_1^{old}), & y_1 \neq y_2 \\ L = \max(0, \alpha_1^{old} + \alpha_2^{old} - C), H = \min(C, \alpha_2^{old} + \alpha_1^{old}) & y_1 = y_2 \end{cases}$

这里这样写是因为就是 $\alpha_i + \alpha_j > 0$. 其可以大于C, 也可以小于C

$\alpha_i - \alpha_j$ 可以大于0也可以小于0.

Define Prediction Error:

$$E_i = \left(\sum_{j=1}^{N} \alpha_j y_j K(x_j, x_i) + b\right) - y_i, \quad i = 1, 2$$

$$\eta = K_{11} + K_{22} - 2K_{12}$$

剪辑解: $\alpha_2^{new,unc} = \alpha_2^{old} + \frac{y_2(E_1 - E_2)}{\eta}$

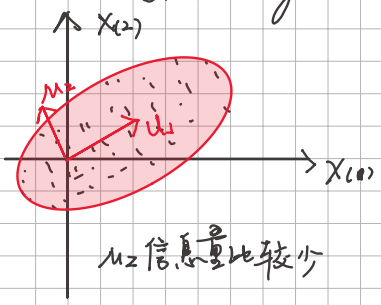
考虑边界有: $\alpha_2^{new} = \begin{cases} H, & \text{if } \alpha_2^{new,unc} > H \\ \alpha_2^{new,unc}, & \text{if } L \le \alpha_2^{new,unc} \le H \\ L, & \text{if } \alpha_2^{new,unc} < L \end{cases}$

进一步有 $\alpha_1^{new} = \alpha_1^{old} + y_1 y_2 (\alpha_2^{old} - \alpha_2^{new})$

Lecture 8 Unsupervised Learning: X only, no y.

Tasks
① Dimension Reduction: 10000 dim → 100 dim. $X \xrightarrow{x} X' \xrightarrow{x'} y$
② Clustering (聚类) group similar data together.
③ Generative models: Sample new data from P(x)

(1) Dimensionality Reduction → Principal Component Analysis (PCA)



$X^{(1)}, X^{(2)}$ highly correlated.

Principals: find a new basis u_1, u_2 s.t. $u_1 \perp u_2$, U_1 keeps the most information (data variance on u_1 is the largest). Then throw away U_2. Represent a simple x by u_1

Here are algorithm details:

$X = \begin{pmatrix} X_1^T \\ i \\ X_n^T \end{pmatrix} \in R^{n \times d}$, calculate $\bar{x} = \frac{1}{n} \sum_{i \in [n]} X_i$

Data covariance matrix: $\frac{1}{n} \sum_{i \in [n]} (X_i - \bar{x}) \cdot (X_i - \bar{x})^T$

Now compute $U_1 \in R^d$, need project each X_i to u_1 and maximize variance.

Loss Function $= \frac{1}{n} \sum_{i \in [n]} (\underbrace{u_1^T X_i}_{\text{project}} - u_1^T \bar{x})^2 = \frac{1}{n} \sum_{i \in [n]} u_1^T (X_i - \bar{x})(X_i - \bar{x})^T u_1$

$= u_1^T \left(\frac{1}{n} \sum_{i \in [n]} (X_i - \bar{x})(X_i - \bar{x})^T \right) u_1$

$= u_1^T \Sigma u_1 \leftarrow \max$

Goal: $\max u_1^T \Sigma u_1$, s.t. $u_1^T u_1 = 1$ if not 1, can ↑

带约束的优化问题 ⇒ 拉格朗日数乘

$L(u_1, \lambda) = u_1^T \Sigma u_1 + \lambda(1 - u_1^T u_1)$

$\frac{\partial L}{\partial u_1} = \Sigma u_1 + \Sigma^T u_1 - 2\lambda u_1 = 0 \Rightarrow \Sigma u_1 = \lambda u_1$

$\Rightarrow u_1$ is an eigenvector of Σ corresponding to eigenvalue λ

while, $u_1^T \Sigma u_1 = \lambda u_1^T u_1 = \lambda_1 \Rightarrow$ Find maximum eigenvalue

Next, find u_2 the next direction that preserves the most variance

$\max_{u_2} u_2^T \Sigma u_2$, s.t. $u_2^T u_2 = 1$, $u_2^T u_1 = 0$

$L(u_2, \lambda_1, \alpha_1) = u_2^T \Sigma u_2 + \lambda(1 - u_2^T u_2) + \alpha u_2^T u_1$

$\Rightarrow \frac{\partial L}{\partial u_2} = 2\Sigma u_2 - 2\lambda_2 u_2 + \alpha u_1 = 0$ ①

$\therefore 2\underbrace{u_1^T \Sigma u_2}_{u_2^T \Sigma u_1 = \lambda u_2^T u_1 = 0} - 2\lambda_2 \underbrace{u_1^T u_2}_{0} + \alpha \underbrace{u_1^T u_1}_{1} = 0 \Rightarrow \alpha = 0$

代入①中 $\Rightarrow \Sigma u_2 = \lambda_2 u_2$

$\frac{\partial L}{\partial \lambda} = 0 \Rightarrow u_2^T \Sigma u_2 = \lambda u_2^T u_2 = \lambda_2$

By Induction, when need K prineipal components, just compute k largest eigenvalues and eigenvectors of Σ.

$[u_1, u_2, \cdots, u_K]$ project X by $X U_{1K} \in R^{n \times k}$, $k \ll d$.

$U_{1K} \in R^{d \times k}$

$\Sigma = U \Lambda U^T$ Λ diagonal, U orthogoti

$= [u_1, \cdots, u_n] \cdot \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix} [u_1, u_2, \cdots, u_n]^T$

$\hat{X} = \begin{pmatrix} x_1^T - \bar{x}^T \\ \vdots \\ x_n^T - \bar{x}^T \end{pmatrix} \in R^{n \times d}$ then $\Sigma = \frac{1}{n} \hat{X}^T \cdot \hat{X}$. we can directly do eigendecomposition for $\hat{X}^T \cdot \hat{X}$

Alternately, we can perform Singular

Value Decomposition (SVD) 奇异值分解，适用于矩阵

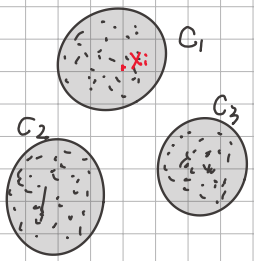
for $\hat{X}^T = U\hat{\Sigma}V^T$, U, V orthogonal

$$\hat{\Sigma} = \underbrace{\begin{bmatrix} \sigma_1 & & 0 \\ & \sigma_2 & \\ 0 & & \sigma_d \end{bmatrix}}_{n} \; 0 \Big]\, d$$

$$\hat{X}^T\hat{X} = U\hat{\Sigma}\underbrace{V^T V}_{I}\hat{\Sigma}^T U^T = U(\hat{\Sigma}\hat{\Sigma}^T)U^T \quad \begin{bmatrix} \sigma_1^2 & & \\ & \ddots & \\ & & \sigma_n^2 \end{bmatrix} = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix}$$

(2) Clustering. Given $D = \{x_1, \cdots, x_n\} \in R^d$

We want to partition D into k clusters.



① $u_k \in R^d$, the center of k

② Let $r_{ik} \in \{0, 1\}$ denote that x_i is closest to u_k. $\sum_{k \in [n]} r_{ik} = 1 \;\; \forall i$.

k-means: Find $\{u_k \mid k \in [k]\}, \{r_{ik}\}$ s.t. the sum of squared distances of each x_i to its assigned cluster center u_k.

Objective: $\min_{r,u} \sum_{i \in [n]} \sum_{k \in [k]} r_{ik} \|x_i - u_k\|^2$

Idea: Alternately optimize u and r

① Given r fixed, u can be optimized very easily.

$$\frac{\partial L}{\partial u_k} = -\sum_{i \in [n]} 2 r_{ik}(x_i - u_k) = 0 \Rightarrow u_k = \frac{\sum_{i \in [n]} r_{ik} \cdot x_i}{\sum_{i \in [n]} r_{ik}}$$

② Given u fixed, $r_{ik} = \begin{cases} 1, & \text{if } k = \arg\min_{j \in [k]} \|x_i - u_j\|^2 \\ 0, & \text{otherwise} \end{cases}$

k-means: Randomly initialize u_k and repeat ① and ② until convergence.

why convergence? Both ①, ② reduce obj.

K means only focus on local minimum, not global.

In practice. we often run k-means for multiple times, each of different initialization.

✡ Mixture of Gaussians (MoG)

Models $P(x)$ ⇒ Assumes points belonging to cluster k follow a Gaussian Distribution. $N(\mu_k, \Sigma_k)$

A generative model. $P(x) = \sum_z P(x|z) \cdot P(z)$

For R.V. X, let Z_k be another R.V. $\in \{0, 1\}$, $z_{ik}=1$ means X generated from cluster k $\sum_{k \in [n]} z_{ik} = 1$.

$$\begin{cases} P(x|Z_k=1) = N(x|\mu_k, \Sigma_k) \\ P(Z_k=1) = \pi_k, \ 0 \le \pi_k \le 1, \ \sum_{k \in [k]} \pi_k = 1 \end{cases}$$

MoG STEPs:

① Randomly initialize $r_{ik} \in [0, 1]$, soft assignment of x_i to k. $\sum_{k \in [k]} r_{ik} = 1$

② Compute weighted sampled mean of covariance,

$$\mu_k = \frac{\sum r_{ik} \cdot x_i}{\sum_{i \in [n]} r_{ik}}, \quad \Sigma_k = \frac{\sum_{i \in [n]} r_{ik} (x_i - \mu_k)(x_i - \mu_k)^T}{\sum_{i \in [n]} r_{ik}}$$

Update π, by $\pi_k = \frac{\sum_{i \in [n]} r_{ik}}{n}$.

③ Update r_{ik}, by $r_{ik} = \frac{\pi_k N(x_i | \mu_k, \Sigma_k)}{\sum_{j \in [k]} \pi_j N(x_i | \mu_j, \Sigma_j)}$

Iterate ② and ③ until convergence.

Lecture 9. latent variable Model

Expectation-Maximization In general.

Suppose θ contains all parameters to estimate

We directly optimize $P(x,\theta)$ is difficult but optimizing $P(x,z;\theta)$ is easy.

$$P(x,z) = P(x|z)\cdot P(z)$$

$$P(z|x) = \frac{P(x|z)\cdot P(z)}{\sum_z P(x|z)\cdot P(z)}$$ → Intractable.

How to solve this problem?

Find a variational distributional $q(z|x)$ to approximate $P(z|x)$!

Now we introduce $q(z|x)$ to approximate $P(z|x;\theta)$

We have the following alway holds.

$$\log P(x;\theta) = \sum_z q(z|x)\cdot \log P(x;\theta)$$

$$= \sum_z q(z|x)\left[\log P(x,z;\theta) - \log P(z|x;\theta)\right]$$

$$= \sum_z q(z|x)\Big(\left[\log P(x,z;\theta) - \log q(z|x)\right] - \left[\log P(z|x;\theta) - \log q(z|x)\right]$$

$$= \underbrace{\sum_z q(z|x)\cdot \log \frac{P(x,z;\theta)}{q(z|x)}}_{\text{Evidence lower Bound}} - \underbrace{\sum_z q(z|x)\cdot \log \frac{P(z|x;\theta)}{q(z|x)}}_{\text{KL divergence.}}$$

we can use Jensen's Ineq to prove KL divergence ≥ 0

EM is maxmizing ELBO! Maxmizing the lower bound also ↑ evidence. Next page we will introduce the EM Algorithm.

EM Algorithm:

A two step iterative algorithm to maximize ELBO. $\mathcal{L}(q,\theta) = \sum_z q(z|x) \cdot \log \frac{p(x,z;\theta)}{q(z|x)}$

① E-step: Given θ fixed. optimize q. Let's assume current $\theta = \theta^{old}$. ELBO $\Rightarrow \mathcal{L}(q, \theta^{old})$ is a functional of q. Also when θ^{old} is fixed. $\log p(x, \theta^{old})$ is a constant.

To maxmize ELBO $\Leftrightarrow$ minimize $KL(q\|p)$:

$\Rightarrow q(z|x) = p(z|x, \theta^{old})$

That is, E step just let $q(z|x)$ take posterior $p(z|x,\theta^{old})$

② M-step. Given q fixed and optimize θ.

Now we have $q(z|x) = p(z|x, \theta^{old})$

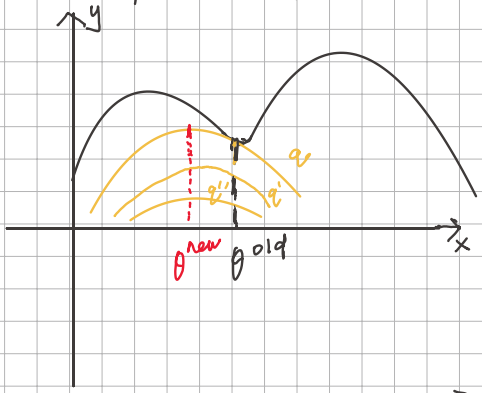
$$\mathcal{L}(q,\theta) = \sum_z p(z|x,\theta^{old}) \log \frac{p(x,z;\theta)}{p(z|x,\theta^{old})}$$

$$= \sum_z p(z|x,\theta^{old}) \cdot \log p(x,z;\theta) - \text{const}$$

$\Rightarrow \max E_{z \sim p(z|x,\theta^{old})} \log p(x,z;\theta)$ $\Rightarrow$ maximize the expectation

③ Repeat until convergence.

EM 算法示意图



choice: q

- E step: choose ELBO $= \log P(X;\theta)$
- M step: maximize ELBO

choice θ.

For many circumstances it has the closed solution.

$z_i = [0, 0, \cdots, \underset{k}{1}, 0]$

$z_{ik} = 1$

EM Algorithm for MoG (n observation; k cluster)

$$\log P(X,\theta) = \sum_{i\in[n]} \log\Big(\sum_{k\in[k]} \pi_k \cdot N(x_i|\mu_k,\Sigma_k)\Big)$$

$\theta = \pi, \mu, \Sigma$

$P(x) = \sum P(x|z)\cdot P(z)$

θ^{old}

(1) E-step. Compute $q(z|X) = P(z|X,\mu,\Sigma,\pi)$

$$P(z,X,\mu,\Sigma,\pi) = \prod_{i\in[n]} \underbrace{P(x_i|z_i,\mu,\Sigma)}\cdot P(z_i,\pi)$$

$$\prod_{k\in[k]} N(x_i|\mu_k,\Sigma_k)^{z_{ik}} \qquad \prod_{k\in[k]} \pi_k^{z_{ik}}$$

$$= \prod_{i\in[n]}\prod_{k\in[k]} \Big\{N(x_i|\mu_k,\Sigma_k)^{z_{ik}}\cdot \pi_k\Big\}^{z_{ik}}$$

$$\therefore P(z|X,\mu,\Sigma,\pi) = \frac{1}{C}\prod_{i\in[n]}\prod_{k\in[k]}\Big\{\pi_k N(x_i|\mu_k,\Sigma_k)\Big\}^{z_{ik}}$$

$$= \prod_{i\in[n]}\Big\{\frac{1}{C_i}\prod_{k\in[k]}\Big\{\pi_k\cdot N(x_i|\mu_k,\Sigma_k)\Big\}^{z_{ik}}$$

$C = \prod_{i\in[n]} C_i$, $C_i = P(x_i)$ $\downarrow$ $P(z_i|x_i,\theta^{old})$

$$P(z|x) = \prod_{i\in[n]} P(z_i|x_i)$$

M step: $\max_\theta E_{\vec{z}}[\log P(X, \vec{z}; \mu, \Sigma, \pi)]$, $\vec{z} \sim P(\vec{z} \mid X, \theta^{old})$

$= E_{\vec{z}}\left[\sum_{i\in[n]} \sum_{k\in[k]} z_{ik} \cdot \log(\pi_k \cdot N(x_i \mid \mu_k, \Sigma_k))\right]$

$= \sum_{i\in[n]} \sum_{k\in[k]} E_{\vec{z}}[z_{ik}] \log(\pi_k \cdot N(x_i \mid \mu_k, \Sigma_k))$

$E_{\vec{z}}(z_{ik}) = 1 \cdot P(z_{ik}=1 \mid x_i) + 0 \cdot \sim = \frac{P(z_{ik}=1, x_i)}{P(x_i)}$

$= \frac{\pi_k \cdot N(x_i \mid \mu_i, \Sigma_k)}{\sum_{j\in[k]} \pi_j N(x_i \mid \mu_j, \Sigma_j)} = r_{ik}$ ~ soft assignment to k.

$\Rightarrow \max_\theta \sum_{i\in[n]} \cdot \sum_{k\in[k]} r_{ik} \cdot \log(\pi_k N(x_i \mid \mu_k, \Sigma_k))$

Language $L(\mu, \Sigma, \pi, \lambda) = \sum_{i\in[n]} \sum_{k\in[k]} r_{ik}(\pi_k N(x_i \mid \mu_k, \Sigma_k) + \lambda(1 - \sum_{k\in[k]} \pi_k)$

$\frac{\partial L}{\partial \pi_k} = \sum_{i\in[n]} r_{ik} \cdot \frac{1}{\pi_k} - \lambda = 0$

$\Rightarrow \pi_k \cdot \lambda = \sum_{i\in[n]} r_{ik} \Rightarrow \lambda \sum_{k\in[k]} \pi_k = \sum_{i\in[n]} \sum_{k\in[k]} r_{ik} = n$

$\Rightarrow \lambda = n$

$\therefore \pi_k = \frac{\sum_{i\in[n]} r_{ik}}{n}$

For μ_k, $N(x_i \mid \mu_k, \Sigma_k) = \frac{1}{(2\pi)^{\frac{1}{2}} |\Sigma_k|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x_i - \mu_k)^T \Sigma_k^{-1} (x_i - \mu_k)\right)$

$\Leftrightarrow L = \sum_{i\in[n]} \sum_{i\in[k]} r_{ik}\left(\log \pi_k - \frac{1}{2}\log 2\pi - \frac{1}{2}\log|\Sigma_k| - \frac{1}{2} \sim\right)$

$\frac{\partial L}{\partial \mu_k} = -\sum_{i\in[n]} r_{ik} \Sigma_k^{-1}(x_i - \mu_k) = 0$

$\Rightarrow \sum_{i\in[n]} r_{ik}(x_i - \mu_k) = 0$

$\Rightarrow \mu_k = \frac{\sum_{i\in[n]} r_{ik} x_i}{\sum_{i\in[n]} r_{ik}}$

$\frac{\partial |X^{-1}|}{\partial x} = -|X^{-1}|(X^{-1})^T$

$\to -|\Sigma_k| \Sigma_k$

Finally, $\frac{\partial L}{\partial \Sigma^{-1}} = 0 \Rightarrow \sum_{i\in[n]} r_{ik} \frac{1}{|\Sigma_k|} \cdot \frac{\partial |\Sigma_k|}{\partial \Sigma_k^{-1}} + r_{ik}(x_i - \mu_k)(x_i - \mu_k)^T = 0$

$$\Rightarrow -\sum_{i\in[n]} r_{ik}\Sigma_k + \sum_{i\in[n]} r_{ik}(x_i-\mu_k)(x_i-\mu_k)^T = 0$$

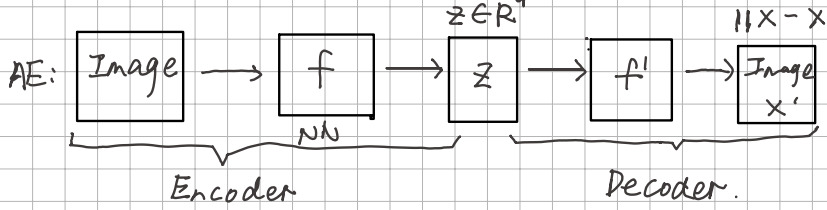
$$\Rightarrow \Sigma_k = \frac{\sum_{i\in[n]} r_{ik}(x_i-\mu_k)(x_i-\mu_k)^T}{\sum_{i\in[n]} r_{ik}}$$

when $\Sigma_k = \delta^2 I$, $\delta \to 0$

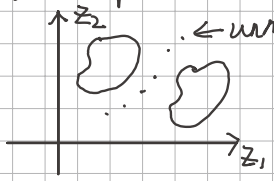
MoG $\to$ K means

Lecture 10. Variational AutoEncoder (VAE)

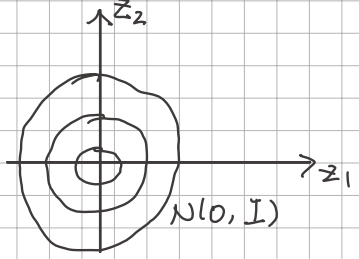
Auto Encoder:



It's z space



You will find hard to sample z



VAE: regularize $Z \sim N(0, I)$

VAE Algorithm:

Given a set of training examples. $D = \{x_1, x_2, \cdots x_n\}$ (such as images.)
we want to generate new data similar to D.

Some points:

① In general, sampling from a distribution $X \sim P(x)$ is difficult! (MCMC) Actually some sample distributions are easy to sample from: $U[a,b]$, $N(u, I)$, $N(0, I)$

⇒ what if we sample z from a simple distribution. then use deterministic $f(z;\theta)$ to map $Z \to X$ (output)

$Z \to$ R.V. $\Rightarrow X \to$ R.V. . $P(X) = \int P(x|z) \cdot P(z)\, dz$

$P(x|z)$: $N(x | f(z;\theta), \sigma^2 I)$; $P(z)$: $N(0, I)$

MoG

Different of z discrete in MoG, Here we use z as one continuous distribution.

we want to optimize. θ.

objective: MLE of $P(x)$ on D to learn θ.

Can we sample a large amount of z from $P(z)$ to approximate $P(x)$? $P(x) \approx \frac{1}{N}\sum_{z_i \sim P(z)} P(x|z_i)$, it cannot work because $P(z)$ is high-dimensional need a super large # of samples z_i to actually approximate $P(x)$!

Second thought: MLE of $\log P(x)$ w/ hidden variables z (intractable)

Reminds us of EM!

Here. E-step: Compute $P(z_i|x_i;\theta^{old}) = \frac{P(z_i|x_i;\theta^{old}) \cdot P(x_i)}{\int P(x_i|z_i) \cdot P(z_i)\, dz}$

M-step: $\theta = \arg\max \int_z \log P(x,z;\theta) \cdot P(z|x_i;\theta^{old})\, dz$

EM not work.

Essential Reason: why GD/EM not work, is $P(z|x,\theta)$ not intractable. Find a tractable variational distribution $q_\phi(z|x;\theta)$ to approximate $P(z|x;\theta)$!

Goal:

$$\log P(x;\theta) = \underbrace{\sum_z q(z|x) \cdot \log \frac{P(x,z;\theta)}{q(z|x)}}_{\text{ELBO}} - \underbrace{\sum_z q(z|x) \cdot \log \frac{P(z|x;\theta)}{q(z|x)}}_{\text{KL divergence}}$$

连续后: $\log P(x;\theta) = \underbrace{\int_z q(z|x) \cdot \log \frac{P(x,z;\theta)}{q(z|x)}\, dz}_{\text{ELBO}} - \underbrace{\int_z q(z|x) \cdot \log \frac{P(z|x;\theta)}{q(z|x)}\, dz}_{\text{KL divergence}}$.

VAE ignores KL divergence, maximize ELBO!

Hope KL divergence is not large, maximize ELBO

Now Let's rewrite ELBO

$$\text{ELBO} = \underbrace{\int_z q(z|x;\theta') \log P(x|z;\theta)\, dz}_{\text{Reconstruction quality}} + \underbrace{\int_z q(z|x;\theta) \log \frac{P(z)}{q(z|x,\theta')}\, dz}_{= -KL(q(z|x;\theta)\,||\,P(z))}$$

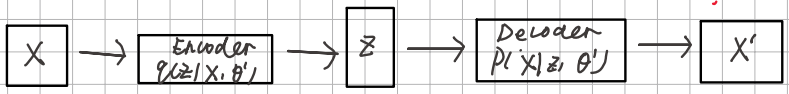
regularize $q(z|x;\theta)$ to approximate $P(z)$!

$E_{z\sim q(z|x;\theta')} \log P(x|z;\theta)$

↓

Encoder Network

Decoder Network.



$p(x|z,\theta) = N(x|f(z,\theta), \sigma^2 I)$

$\Rightarrow \log p(x|z,\theta) = \log \frac{1}{(2\pi\sigma)^{\frac{d}{2}}} \exp\left(-\frac{\|x - f(z,\theta)\|^2}{2\sigma^2}\right)$

$= C - \frac{1}{2\sigma^2}\|x - x'\|^2$

$\min_{\theta,\theta'} E_{z\sim q(z|x,\theta')} \|x - x'\|^2 \approx \|x - x'\|^2$ decoded from 1 z encoded by x

VAE Loss: $\min_{\theta,\theta'} \frac{1}{n}\sum_{i\in[n]} \left[\|x_i - x_i'\|^2 + KL(q(z|x_i;\theta')\|p(z))\right]$

λ β

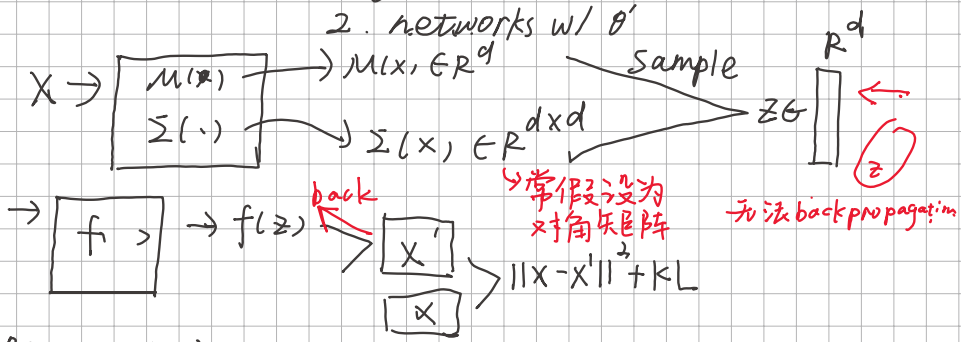
Usually, we need to balance the interaction of KL, $\|x - x_i\|^2$

So we introduce β, VAE Loss = $\|x_i - x_i'\|^2 + \beta \cdot KL$ divergence.

$\beta \to 0 \Rightarrow$ AE

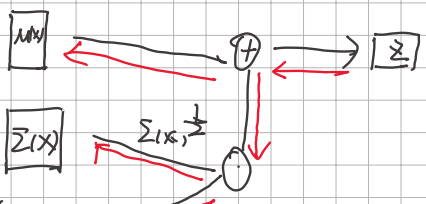
$q(z|x,\theta') = N(z|\mu(x), \Sigma(x))$

2 networks w/ θ'



Reparameterization Trick.

$\varepsilon \sim N(0, I)$. $z = \Sigma(x)^{\frac{1}{2}} \cdot \varepsilon + \mu(x) \sim N(\mu(x), \Sigma(x))$



IDC. No parameter.

KL divergence in VAE

$$KL\ \text{divergence} = KL(q(z|x,\theta') \,||\, p(z)) = KL(N(z|\mu(x),\Sigma(x)) \,||\, N(0,I))$$

$$= \int_z N(z|\mu,\Sigma)\cdot \log \frac{N(z|0,\Sigma)}{N(0,I)}\, dz$$

$$= \frac{1}{(2\pi)^{\frac{d}{2}}|\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(z-\mu)^T\Sigma^{-1}(z-\mu)\right)\cdot \log\left[|\Sigma|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(z-\mu)^T\Sigma^{-1}(z-\mu) + \frac{1}{2}z^Tz\right)\right]$$

$$= \sim \cdot\left[-\frac{1}{2}\log|\Sigma| - \frac{1}{2}(z-\mu)^T\Sigma^{-1}(z-\mu) + \frac{1}{2}z^Tz\right]$$

$$= -\frac{1}{2}\log|\Sigma| - \frac{1}{2}E_z[(z-\mu)^T\Sigma^{-1}(z-\mu)] + \frac{1}{2}E_z(z^Tz) - \frac{1}{2}E_z[(z-\mu)^T\Sigma^{-1}(z-\mu)]$$

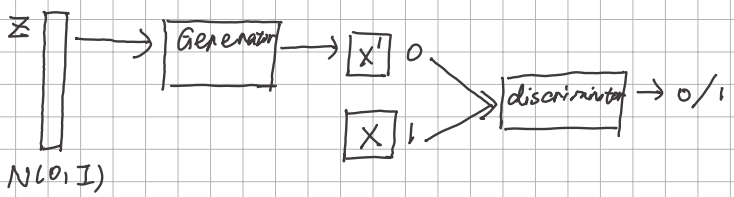
$E_z(z^Tz) \rightarrow E_z[(z-\mu)^T(z-\mu)] + E_z(\mu^T\mu) + 2E_z[(z-\mu)^T\mu] = tr(\Sigma) + \mu^T\mu$

$tr(ABC) = tr(CBCA) = tr(CAB)$

$\therefore E_z(tr(\Sigma^{-1}(z-\mu)(z-\mu)^T)) = tr(\Sigma^{-1}E_z(z-\mu)(z-\mu)^T) = d$

$$= \frac{1}{2}[tr(\Sigma) + \mu^T\mu - d - \log|\Sigma|)$$

GAN



Diffusion models