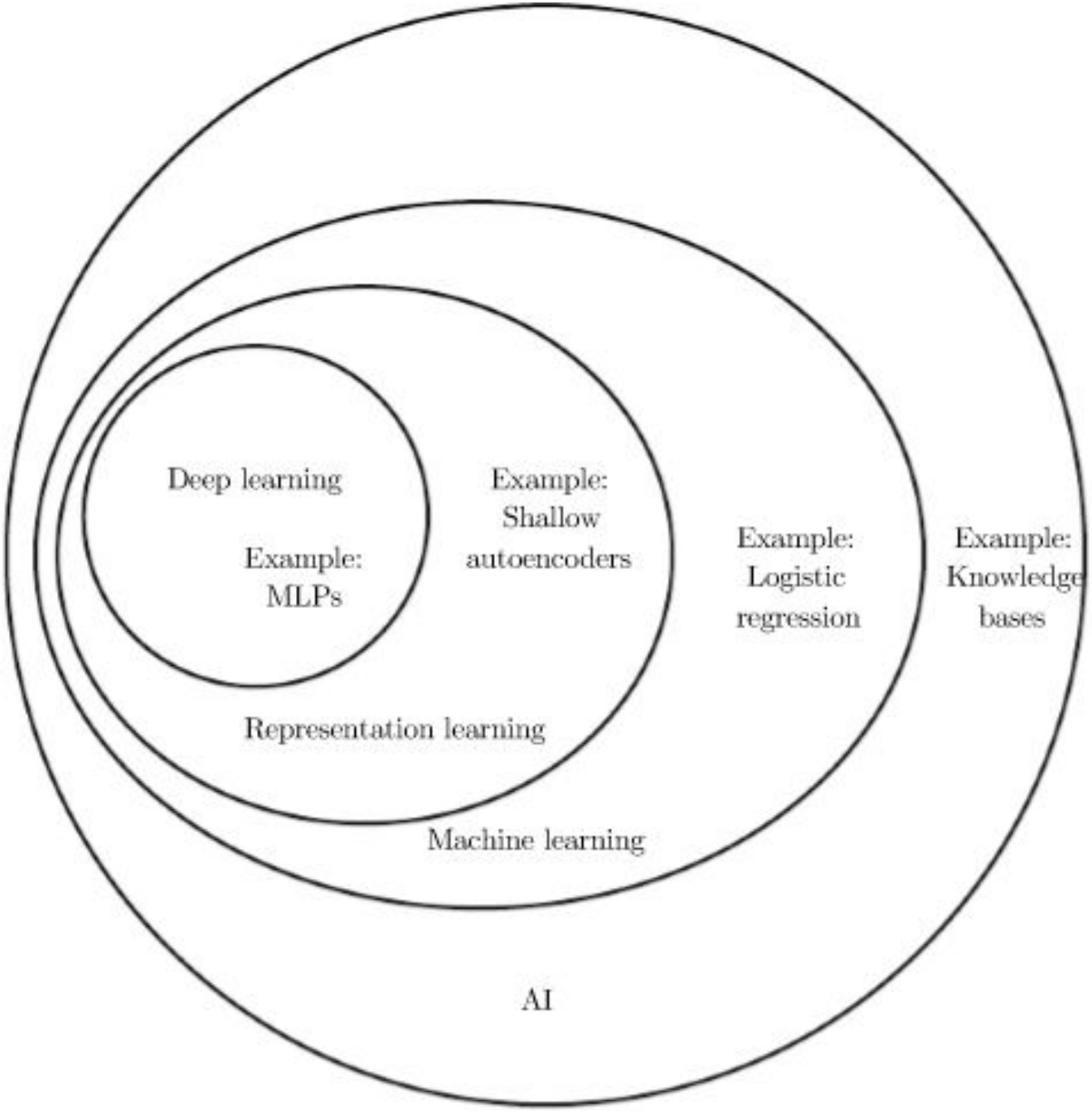
Math In DL

Zhimin Cao – Face++

Face++ 旷视

What is Deep Learning



Face++ 旷视

Linear Algebra

Vector, Matrix (Real-valued)

$$\boldsymbol{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \qquad A = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{pmatrix}$$

Let $A = (a_{i,j})_{m\times n}$, $B = (b_{i,j})_{n\times p}$, define $C = AB = (c_{i,j})_{m\times p}$ with

$c_{i,j} = \sum_{k=1}^{n} a_{i,k} b_{k,j}$

Special matrix $0_{m,n}$, $I_{m,n}$

Square Matrix

$$A = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{pmatrix}$$

$$\operatorname{tr}(A) = \sum_{i=1}^{n} a_{i,i}$$

$$\det(A) = \sum_{\sigma \in S_n} \operatorname{sgn}(\sigma) \prod_{i=1}^{n} a_{i,\sigma_i}$$

Square Matrix – Real Orthogonal Matrix

- What can you say about $AA^T = A^TA = I_n$?

Real Orthogonal Matrix (Unitary Matrix)

Whose rows & columns forms a orthogonal basis (orthogonal unit vectors)

In geometry view, orthogonal matrix defines an orthogonal transformation (rotation, reflection, etc).

Square Matrix – Eigenvalue & eigenvector

View square matrix A as a linear transformation from a vector space into itself.
Non-zero vector v is a eigenvector of A if there is a scalar λ (eigenvalue) so that

$$Av = \lambda v$$

Given an eigenvalue λ, to find all the eigenvectors

$$(\lambda I_n - A)v = 0_n$$

To find all eigenvalues, note the above equation has solution iff.

$$\det(\lambda I - A) = 0$$

Face++ 旷视

Square Matrix – Characteristic Polynomial

$$\det(\lambda I - A) = \lambda^n - \mathrm{tr}(A)\lambda^{n-1} + ... + (-1)^n \det(A)$$

$$\det(\lambda I - A) = \Pi_{i=1}^{n}(\lambda - \lambda_i)$$

比较上下两个式子，利用 Vieta's theorem，我们得到

$$\mathrm{tr}(A) = \Sigma_{i=1}^{n}\lambda_i$$

$$\det(A) = \Pi_{i=1}^{n}\lambda_i$$

$$A \text{ has inverse} \Leftrightarrow \det(A) \neq 0 \Leftrightarrow \forall i \ \lambda_i \neq 0$$

Real Symmetric Matrix (RSM)

Square matrix $A = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{pmatrix}$ where $a_{i,j}=a_{j,i}$

Quadratic form

$$q_A(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{i,j} x_i x_j = x^T A x$$

RSM - Eigendecomposition

RSM has a very important property with respect to its eigenvalues and eigenvectors. The result is summarized as the following "eigen-decomposition" of RSM.

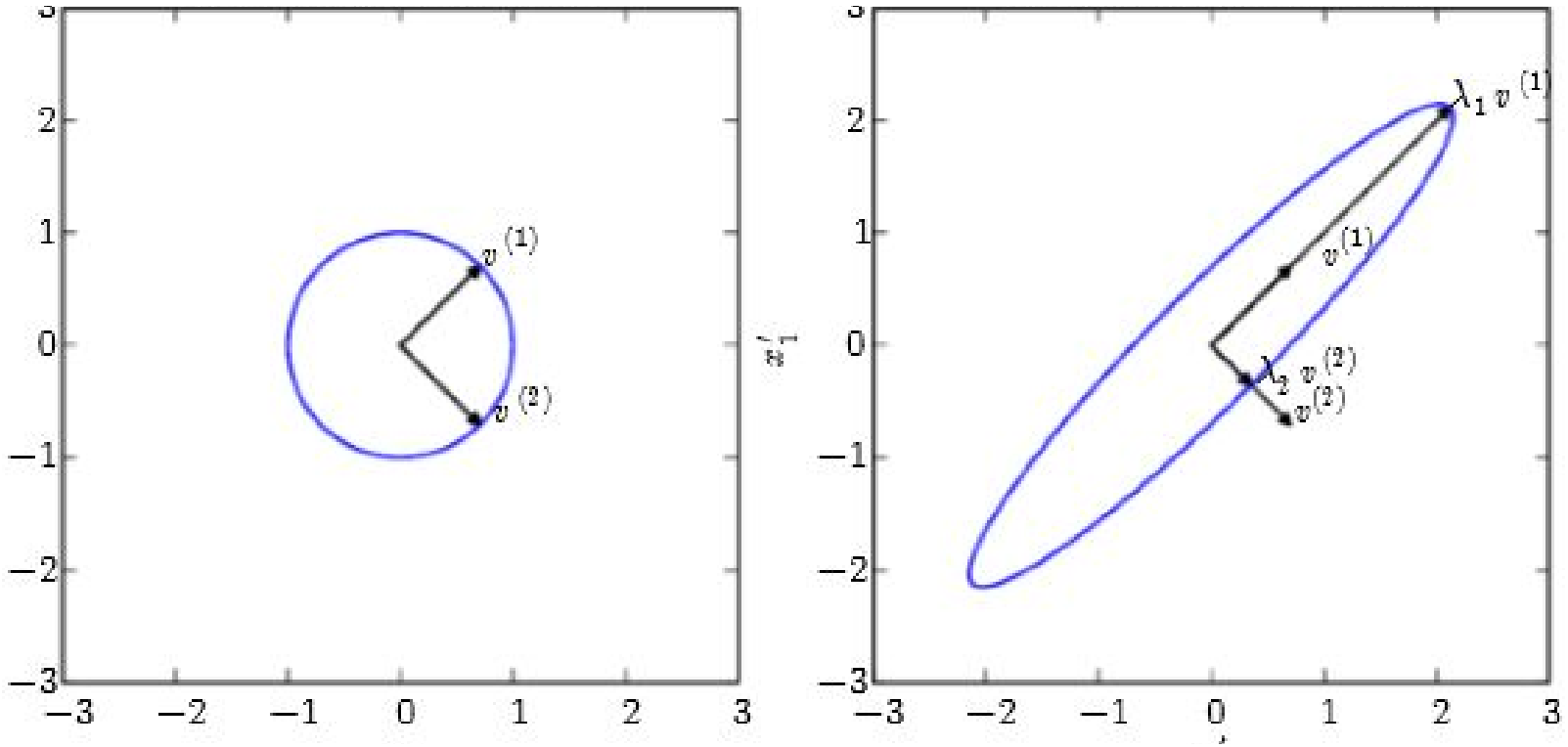
$$A = V\Lambda V^T \text{where } VV^T = I_n \text{and } \Lambda = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$$

$$\text{It is obvious that each column of } V \text{ is an eigenvector of } A \ (Av_i = \lambda_i v_i)$$

RSM – Geometry of Eigendecomposition

$$x^T Ax = x^T V \Lambda V^T x = (V^T x)^T \Lambda (V^T x) = z^T \Lambda z \text{ where } z = V^T x$$

$$(\min_i \lambda_i)\|z\|_2^2 \leq x^T Ax = \Sigma_{i=1}^n \lambda_i z_i^2 \leq (\max_i \lambda_i)\|z\|_2^2$$



Face++ 旷视

RSM – Positive Definite Matrix

If $\lambda_i > 0$, A is called positive definite (PD)
If $\lambda_i \geq 0$, A is called positive semidefinite (PSD)

$$A \text{ is SPD} \Leftrightarrow \forall x, x^T A x \geq 0 \Leftrightarrow \forall i, \lambda_i \geq 0$$

For any matrix L, $L^T L$ and $L L^T$ are PSD (WHY?).

Singular Value Decomposition

$A \in \mathbb{R}^{m \times n} (m \leq n)$. Now $AA^T \succeq 0 \Rightarrow AA^T = U \Lambda U^T$

Let $B = U^T A, BB^T = \Lambda$, so B's rows are orthogonal

$\Rightarrow B = SV^T$ where $S \in \mathbb{R}^{m \times n}$ is diagonal and $VV^T = I_n$

$\Rightarrow A = UB = USV^T$, so called the singular value decomp of A

Probability

R.V., PMF, PDF

R.V. 刻画一个样本空间中随机事件所产生的结果 (outcome) 的映射

Discrete Random Variable

Assume X's outcome consists of a countable Set $\{x_1, x_2, \ldots\}$

Probability mass function (PMF) $P(X = x_i) = p_i >= 0$

Note $\sum_i p_i = 1$

Continuous Random Variable

X takes all the values in R

Probability density function (PDF) $p(x)>=0$ and $\int p(x)\,dx = 1$

Multiple R.V.

X, Y are R.V.

Joint Distribution $p_{X,Y}(X = x, Y = y)$

Marginal Distribution

$$p_X(X = x) = \int p_{X,Y}(X = x, Y = y)\,\mathrm{d}y$$

$$p_Y(Y = y) = \int p_{X,Y}(X = x, Y = y)\,\mathrm{d}x$$

Conditional Distribution

$$p_{X|Y}(X = x|Y = y) = \frac{p_{X,Y}(X = x, Y = y)}{p_Y(Y = y)}$$

Independence

X, Y are R.V.

$$p_{X,Y}(X = x, Y = y) = p_X(X = x)p_Y(Y = y)$$

$$p_{X|Y}(X = x|Y = y) = p_X(X = x)$$
$$p_{Y|X}(Y = y|X = x) = p_Y(Y = y)$$

Bayes' Rule

- $$p_{X|Y}(X = x|Y = y) = \frac{p_{X,Y}(X = x, Y = y)}{p_Y(Y = y)}$$

$$= \frac{p_{Y|X}(Y = y|X = x)p_X(X = x)}{p_Y(Y = y)}$$

$$= \frac{p_{Y|X}(Y = y|X = x)p_X(X = x)}{\int p_{Y|X}(Y = y|X = x)p_X(X = x)\,\mathrm{d}x}$$

Expectation, Variance

Let X, Y be a continuous R.V. with pdf $p(x), p(y)$

$$\mathbb{E}[X] = \int xp(x)\,\mathrm{d}x$$

$$\mathrm{Var}[X] = \mathbb{E}[(X - \mathbb{E}[X])^2] = \mathbb{E}[X^2] - \mathbb{E}[X]^2$$

$$\mathrm{Cov}(X, Y) = \mathbb{E}[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])] = \mathbb{E}[XY] - \mathbb{E}[X]\mathbb{E}[Y]$$

Expectation, Variance - Properties

X, Y be R.V.

$$\mathbb{E}[\alpha X + \beta Y] = \alpha \mathbb{E}[X] + \beta \mathbb{E}[Y]$$

$$\mathrm{Cov}(X, X) = \mathrm{Var}[X]$$

$$\begin{aligned}\mathrm{Var}[\alpha X + \beta Y] &= \alpha^2 \mathrm{Var}[X] + 2\alpha\beta \mathrm{Cov}(X, Y) + \beta^2 \mathrm{Var}[Y] \\ &= [\alpha, \beta] \begin{bmatrix} \mathrm{Var}[X] & \mathrm{Cov}(X, Y) \\ \mathrm{Cov}(X, Y) & \mathrm{Var}[Y] \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix}\end{aligned}$$

Covariance Matrix

$X_1, X_2, \ldots, X_n$ are R.V.

定义他们的协方差矩阵 (Covariance Matrix)

$$\mathrm{Cov}(X_1, X_2, \ldots, X_n) = \begin{bmatrix} \mathrm{Var}[X_1] & \mathrm{Cov}(X_1, X_2) & \mathrm{Cov}(X_1, X_3) & \ldots & \mathrm{Cov}(X_1, X_n) \\ \mathrm{Cov}(X_2, X_1) & \mathrm{Var}[X_2] & \mathrm{Cov}(X_2, X_3) & \ldots & \mathrm{Cov}(X_2, X_n) \\ \mathrm{Cov}(X_3, X_1) & \mathrm{Cov}(X_3, X_2) & \mathrm{Var}[X_3] & \ldots & \mathrm{Cov}(X_3, X_n) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathrm{Cov}(X_n, X_1) & \mathrm{Cov}(X_n, X_2) & \mathrm{Cov}(X_n, X_3) & \ldots & \mathrm{Var}[X_n] \end{bmatrix}$$

Covariance Matrix Cont'd

$$\forall \alpha_i, \text{Let } \alpha = [\alpha_1, \alpha_2, \ldots, \alpha_n]^T$$
$$X = [X_1, X_2, \ldots, X_n]^T$$
$$\text{Var}[\alpha^T X] = \alpha^T \text{Cov}(X) \alpha \geq 0$$

It is concluded Covariance Matrix is **positive semidefinite.**

Face++ 旷视

Common Distributions

Bernoulli / Binomial Distribution

Toss a coin with head up probability p. The outcome follows Bernoulli distribution.

$$P(X = 0) = p, P(X = 1) = 1 - p$$

Toss the coin n times, the outcome follows Binomial distribution.

$$P(X = 0 \text{ for } k \text{ times}) = \binom{n}{k} p^k (1-p)^{n-k}$$

Multinoulli / Multinomial Distribution

Generalize the two-value coin to k value case.

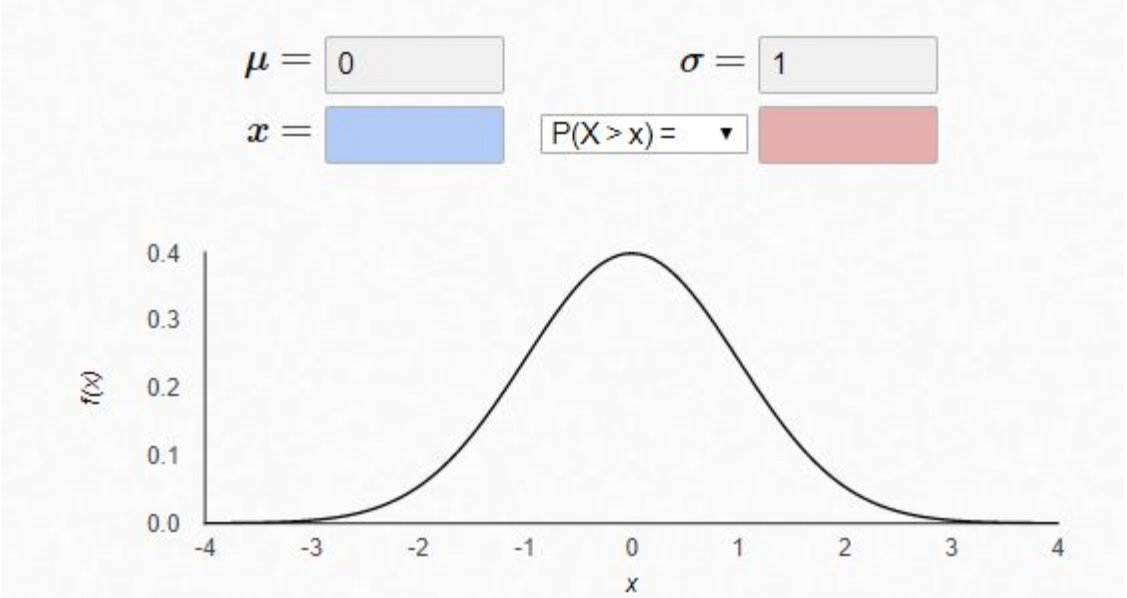
$$P(X = x_1 \text{ for } k_1 \text{ times, } = x_2 \text{ for } k_2 \text{ times, } \ldots) = \frac{n!}{\prod k_i!} \prod p_i^{k_i} \text{ here } \sum k_i = n$$

Common Distributions – Normal Distribution

$$\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sqrt{\frac{1}{(2\pi)^n \det(\boldsymbol{\Sigma})}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right)$$

μ is the expectation, Σ is the covariance matrix.

http://homepage.stat.uiowa.edu/~mbognar/applets/normal.html



Face++ 旷视

Information Entropy

不常见的事件更有信息量
对于 R.V. X , 随机抽样一次样本，如果取到的值是发生几率较低的，则说明这次抽样包含了较大的信息量，使用$-\log p(x)$ 来刻画，则平均来说，这个随机变量的信息量 (熵) 是

$$\mathbb{E}_{X\sim P}[-\log P(x)] = -\int p(x)\log p(x)\,\mathrm{d}x$$

Cross Entropy and KL-divergence

两个不同的R.V..，衡量他们分布之间的差距

$$D_{KL}(P \parallel Q) = \mathbb{E}_{X\sim P}[\log \frac{P(x)}{Q(x)}] = \mathbb{E}_{X\sim P}[-\log Q(x)] - H(x)$$

引入交叉熵的概念

$$H(P,Q) = \mathbb{E}_{X\sim P}[-\log Q(x)]$$

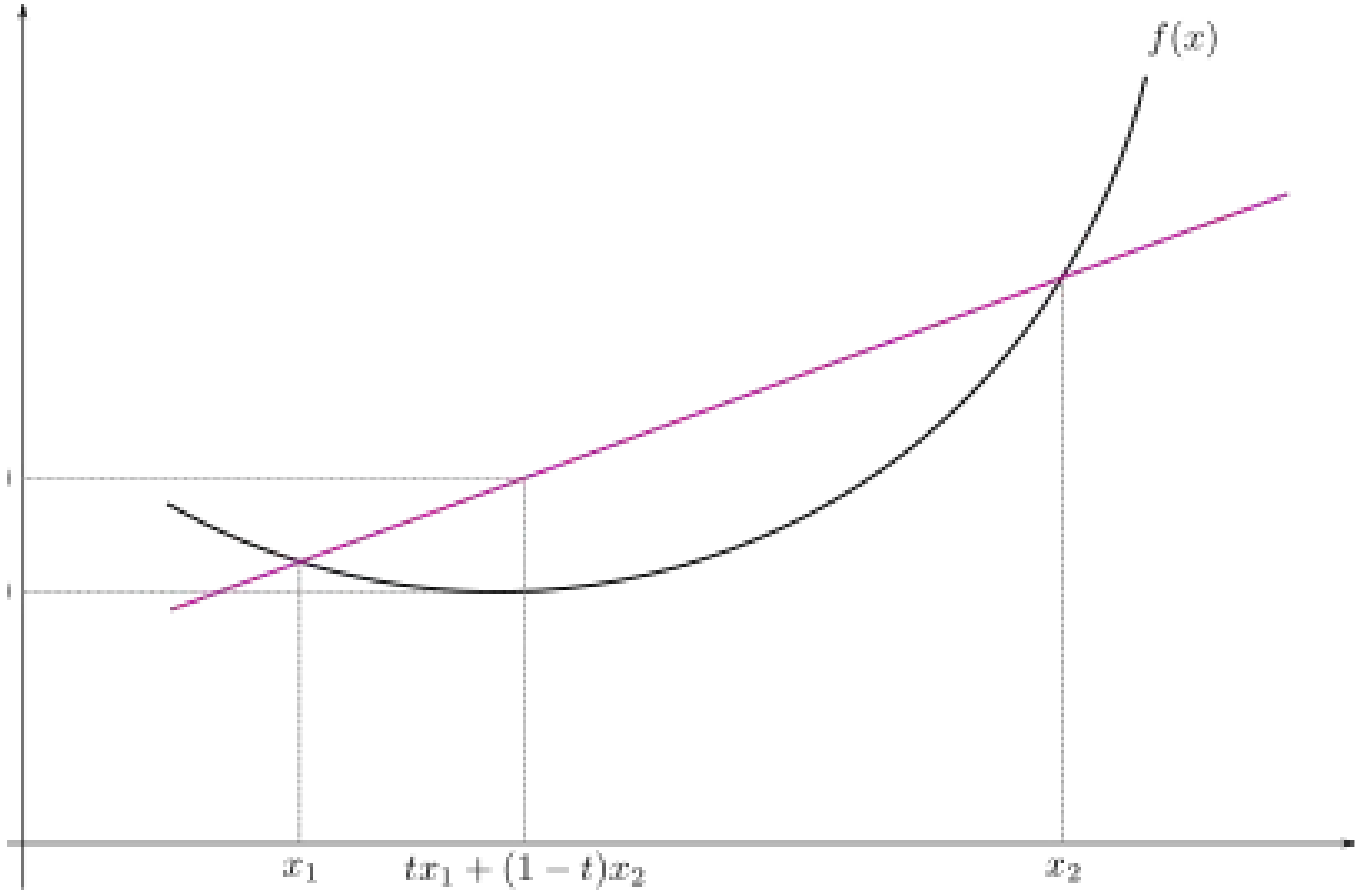
当给定一个P，优化Q使得尽量接近P的时候，减小交叉熵等同于减小KL-divergence

注意 $D_{KL}(P \parallel Q) \neq D_{KL}(Q \parallel P)$

KL-divergence - Properties

Recall Convex function and
Jensen's inequality

For convex function φ,
$\varphi(\mathbb{E}_{X\sim P}[X]) \leq \mathbb{E}_{X\sim P}[\varphi(X)]$.



Face++ 旷视

KL-divergence – Properties Cont'd

KL Divergence is **non-negative**.

$$\varphi = -\log(x), X(s) = \frac{Q(s)}{P(s)}$$

$$\mathbb{E}_{X\sim P}[X] = \int P(s)X(s)\,\mathrm{d}s = \int P(s)\frac{Q(s)}{P(s)}\,\mathrm{d}s = \int Q(s)\,\mathrm{d}s = 1$$

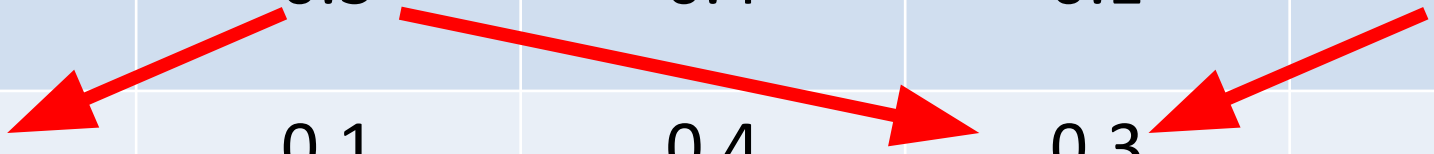
$$\varphi(\mathbb{E}_{X\sim P}[X]) = -\log(1) = 0 \leq \mathbb{E}_{X\sim P}[\varphi(X)] = -\int P(s)\log\frac{Q(s)}{P(s)}\,\mathrm{d}s = D_{KL}(P \parallel Q)$$

Face++ 旷视

Wasserstein Distance

Check discrete distribution first, try to move the “weight” assigned by P to the ones by Q via shortest distance.

value	0	1	2	3	4
P	0.1	0.3	0.4	0.1	0.1
Q	0.2	0.1	0.4	0.3	0.0



Wasserstein Distance – Earth Mover

	0.1	0.3	0.4	0.1	0.1
0.2	0.1	0.1			
0.1		0.1			
0.4			0.4		
0.3		0.1		0.1	0.1
0.0					

$$\min_{f_{i,j}} \Sigma_{i=0,j=0}^{n-1,n-1} f_{i,j}|i-j|$$

$$s.t.\ \forall j, \Sigma_{i=0}^{n-1} f_{i,j} = P_j$$

$$\forall i, \Sigma_{j=0}^{n-1} f_{i,j} = Q_i$$

$$\forall i, j, f_{i,j} \geq 0$$

Face++ 旷视

Wasserstein Distance – Definition

$$W(P,Q) = \inf_{\gamma \in \Pi(P,Q)} \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\gamma}[\| \boldsymbol{x} - \boldsymbol{y} \|]$$

$$\Pi(P,Q) = \{\gamma : \int \gamma(\boldsymbol{x},\boldsymbol{y})\,\mathrm{d}\boldsymbol{x} = Q(\boldsymbol{y}) \text{ and } \int \gamma(\boldsymbol{x},\boldsymbol{y})\,\mathrm{d}\boldsymbol{y} = P(\boldsymbol{x})\}$$

You call it distance, is it correct?

$$W(P,R) \leq W(P,Q) + W(Q,R)$$

Face++ 旷视

Wasserstein Distance

$$\forall \gamma \in \Pi(P, Q), \delta \in \Pi(Q, R)$$

$$\text{let } \varphi(x, z) = \int \frac{\gamma(x, y)\delta(y, z)}{Q(y)} \, \mathrm{d}y$$

$$\int \varphi(x, z) \, \mathrm{d}z = \int \mathrm{d}z \int \frac{\gamma(x, y)\delta(y, z)}{Q(y)} \, \mathrm{d}y = \int \frac{\gamma(x, y)}{Q(y)} \, \mathrm{d}y \int \delta(y, z) \, \mathrm{d}z$$

$$= \int \frac{\gamma(x, y)}{Q(y)} Q(y) \, \mathrm{d}y = \int \gamma(x, y) \, \mathrm{d}y = P(x)$$

$$\int \varphi(x, z) \, \mathrm{d}x = \int \mathrm{d}x \int \frac{\gamma(x, y)\delta(y, z)}{Q(y)} \, \mathrm{d}y = \int \frac{\delta(y, z)}{Q(y)} \, \mathrm{d}y \int \gamma(x, y) \, \mathrm{d}x$$

$$= \int \frac{\delta(y, z)}{Q(y)} Q(y) \, \mathrm{d}y = \int \delta(y, z) \, \mathrm{d}y = R(z)$$

Face++ 旷视

Wasserstein Distance

$$
\begin{aligned}
\iint \varphi(x,z)\|x-z\|\,\mathrm{d}x\,\mathrm{d}z &= \iint \mathrm{d}x\,\mathrm{d}z \int \frac{\gamma(x,y)\delta(y,z)\|x-z\|}{Q(y)}\,\mathrm{d}y \\
&\le \iint \mathrm{d}x\,\mathrm{d}z \int \frac{\gamma(x,y)\delta(y,z)(\|x-y\|+\|y-z\|)}{Q(y)}\,\mathrm{d}y \\
&= \iint \mathrm{d}x\,\mathrm{d}y \frac{\gamma(x,y)\|x-y\|}{Q(y)} \int \delta(y,z)\,\mathrm{d}z + \iint \mathrm{d}y\,\mathrm{d}z \frac{\delta(y,z)\|y-z\|}{Q(y)} \int \gamma(x,y)\,\mathrm{d}x \\
&= \iint \gamma(x,y)\|x-y\|\,\mathrm{d}x\,\mathrm{d}y + \iint \delta(y,z)\|y-z\|\,\mathrm{d}y\,\mathrm{d}z
\end{aligned}
$$

Face++ 旷视

Optimization

Minimization

- Machine Learning usually involves some optimization problem.

$$x^* = \text{argmin}\, f(x)\ x \in X$$

We are interested in "continuously differentiable" $f(x)$.

Now if we are standing at x_0, how to find another point $x'$where $f(x')$ evaluates smaller than x_0 ?

Minimization – Gradient Descent

Taylor's expansion tells

$$f(\mathbf{x_0} + \alpha\boldsymbol{\mu}) \approx f(\mathbf{x_0}) + \alpha\boldsymbol{\mu}^T\nabla f(x)$$

So with same step size α , we should choose direction $\boldsymbol{\mu} = -\nabla f(x)$ to make the "descent" steepest.

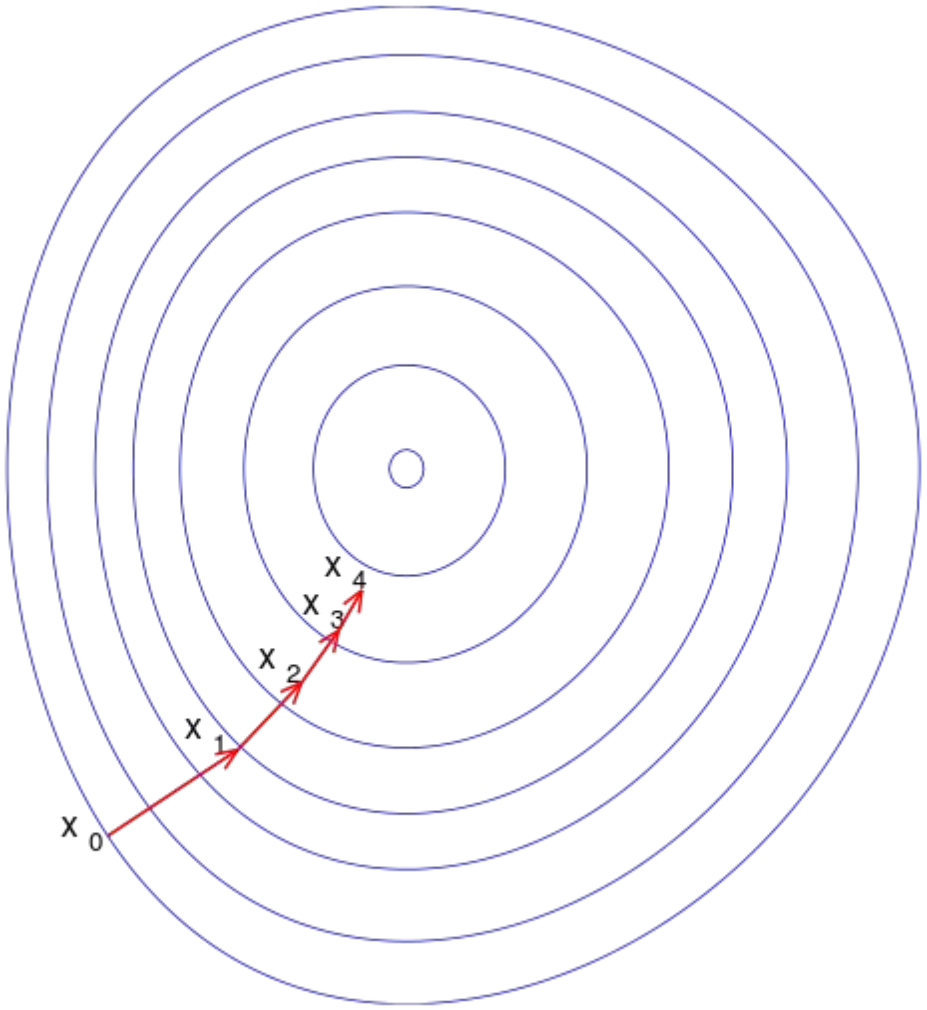
$$x' = x_0 - \epsilon\nabla f(x)$$

Here ϵ is called learning rate, a positive scalar determining the step size.

The optimal step size $\epsilon^\star$ could be obtained by **line search**.

Face++ 旷视

Minimization – Gradient Descent



Face++ 旷视

When $f(x)$ is computationally expensive to evaluate – Stochastic Gradient Descent

A common example encountered in machine learning is the “training set” is too large.

$$f(\boldsymbol{x}) = \Sigma_{\boldsymbol{c_i} \in \boldsymbol{D}} g(x, \boldsymbol{c_i}) \quad \nabla_{\boldsymbol{x}} f(\boldsymbol{x}) = \Sigma_{\boldsymbol{c_i} \in \boldsymbol{D}} \nabla_{\boldsymbol{x}} g(x, \boldsymbol{c_i})$$

$$\nabla_{\boldsymbol{x}} f(\boldsymbol{x}) \approx \Sigma_{\boldsymbol{c_i} \in \boldsymbol{D^\star}} \nabla_{\boldsymbol{x}} g(x, \boldsymbol{c_i})$$

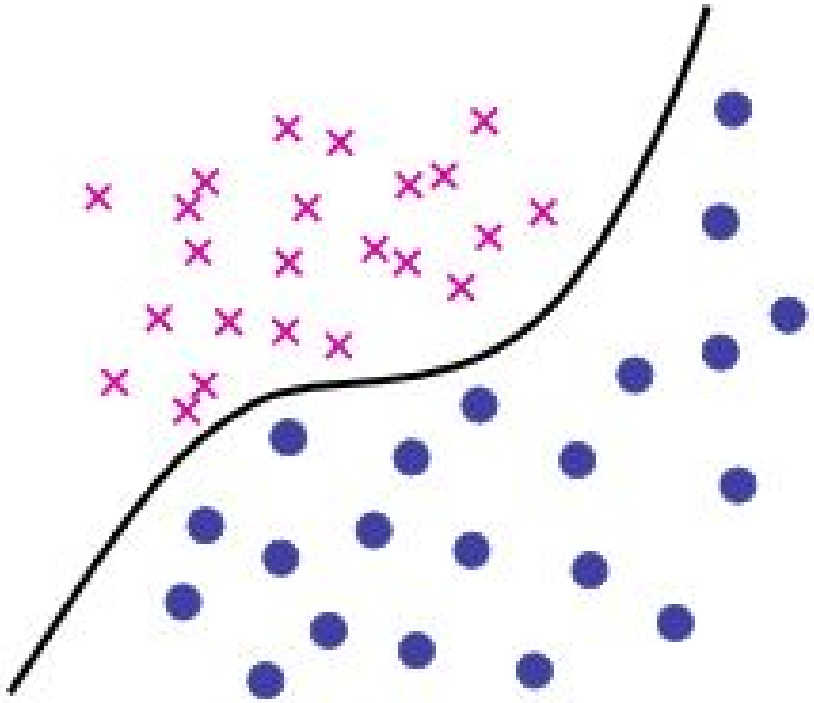
with $\boldsymbol{D^\star}$ is a much smaller subset uniformly sampled from $\boldsymbol{D}$

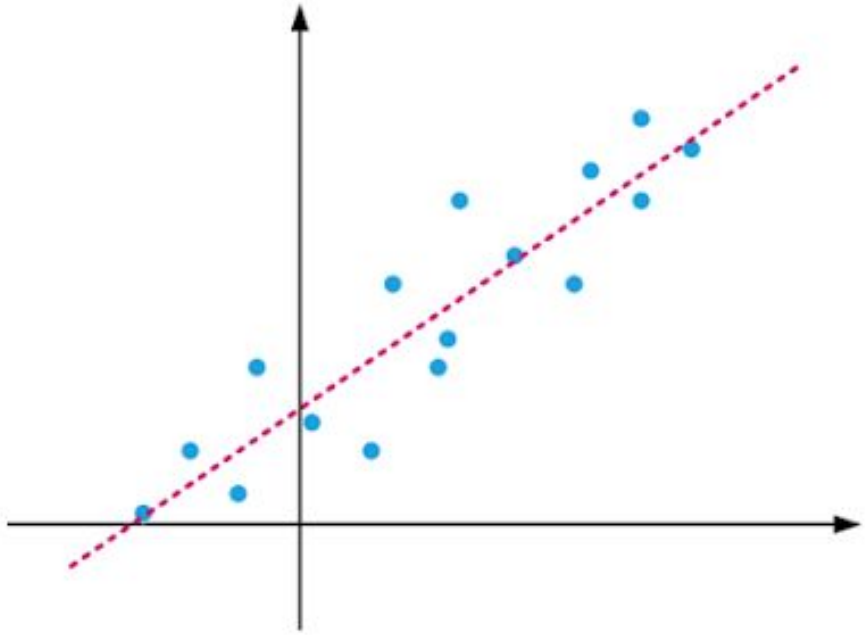
Face++ 旷视

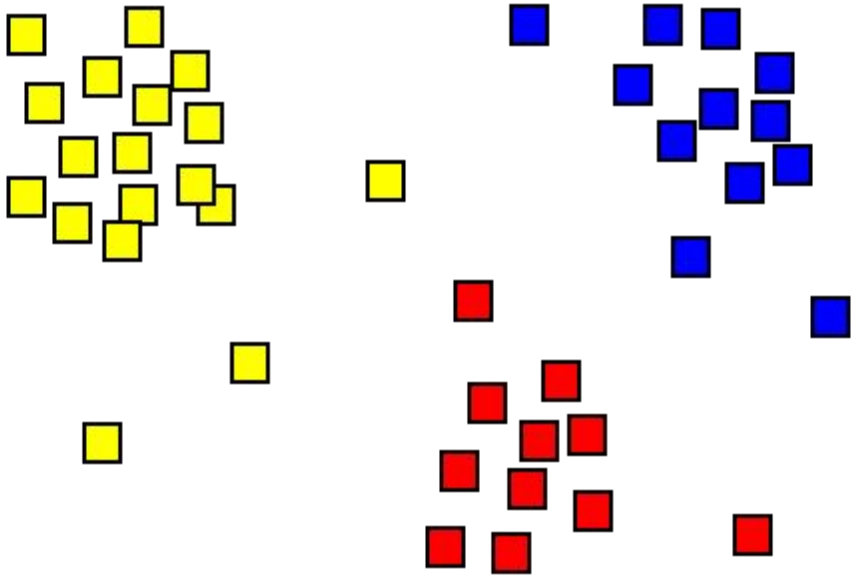
Machine Learning Basics

Face++ 旷视

Examples







Informal Definition

Task T

Classification, regression, clustering, ranking, etc.

Dataset $\boldsymbol{D}$

$\boldsymbol{D}$ consists of "samples", usually denoted by a vector $\boldsymbol{x} \in \mathbb{R}^n$. Each dimension of x is called a "feature".

From a probabilistic view, $\boldsymbol{x}$ are sampled from a underlying distribution $p(\boldsymbol{x})$

Usually we have training set $\boldsymbol{D}_{train}$ and testing set $\boldsymbol{D}_{test}$.

Informal Definition – Cont'd

Model M

Given $\boldsymbol{D}_{train}$, the learning algorithm tries to select a model from the model spaces to perform the task.

Performance measure P

The way to evaluate M's performance on tasking T with respect to $\boldsymbol{D}_{test}$.

Supervised & Unsupervised Learning

Given data points $\boldsymbol{x}$ sampling from a distribution, sometimes we also have "target" or "label" y associated with the samples.

- Learning $p(\boldsymbol{x})$ - Density Estimation, Dimension Reduction, Clustering

  CV/DL examples – Autoencoder, GAN (Generative Adversarial Networks)

- Learning $p(y|\boldsymbol{x})$ or $p(\boldsymbol{x}, y)$ – Classification, Regression

  Discriminative models vs. Generative models

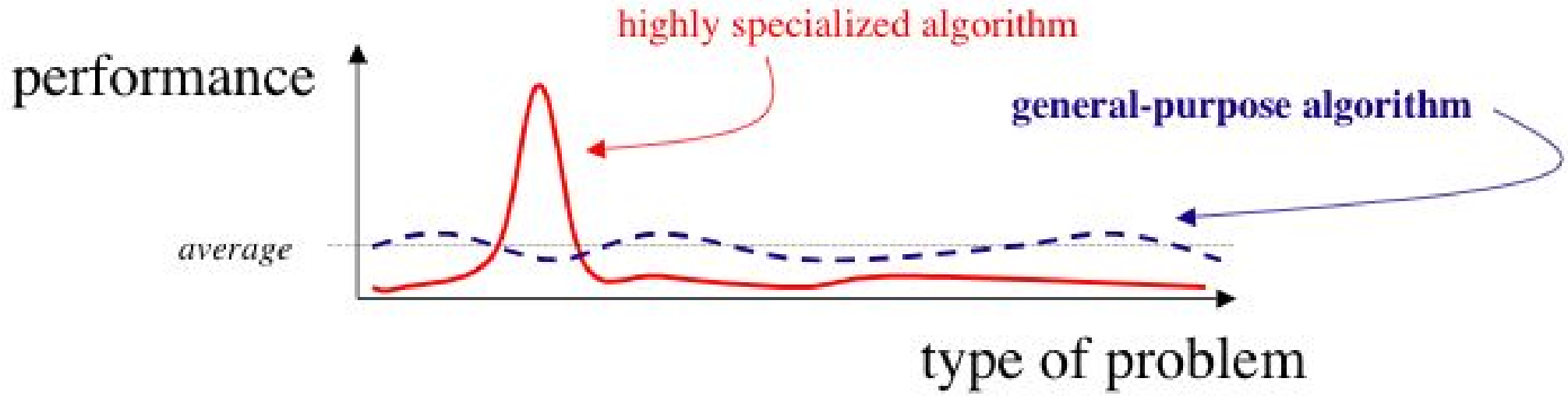
  CV/DL examples – image classification, semantic segmentation, object detection

Not a strict categorization, and other forms of ML exists (RL).

Face++ 旷视

No Free Lunch Theorem

All learning algorithms are equal!
But some algorithms are more equal than others.



Face++ 旷视

A formal view of Supervised Learning

- Given training set $\boldsymbol{D}_{train} = \{(\boldsymbol{x_1}, y_1), (\boldsymbol{x_2}, y_2), (\boldsymbol{x_3}, y_3) \dots\}$, we want to learn a model with parameters θ, let it be $f(\boldsymbol{x}; \theta)$ to predict the y.
- Define performance metric between $f(\boldsymbol{x}; \theta)$ and y. For example, we usually take L^2 metric in $\mathbb{R}$ for regression problems. The following performance measure is called "generalization error" or "test error". And our task is to minimize it!

$$P(f) = \iint \Omega(y, f(\boldsymbol{x}; \theta)) p(\boldsymbol{x}, y)\, \mathrm{d}\boldsymbol{x}\mathrm{d}y = \int p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} \int \Omega(y, f(\boldsymbol{x}; \theta)) p(y|\boldsymbol{x})\, \mathrm{d}y$$

$$\approx \frac{1}{|\boldsymbol{D}_{test}|} \Sigma_{(\boldsymbol{x},y)\in \boldsymbol{D}_{test}} \Omega(y, f(\boldsymbol{x}; \theta))$$

Face++ 旷视

We do not have access to $\boldsymbol{D}_{test}$!

We generally do not access to $\boldsymbol{D}_{test}$! So we have no way to optimize test error directly!
A workaround is to minimize the performance measure with respect to training set $\boldsymbol{D}_{train}$, called “training error”.

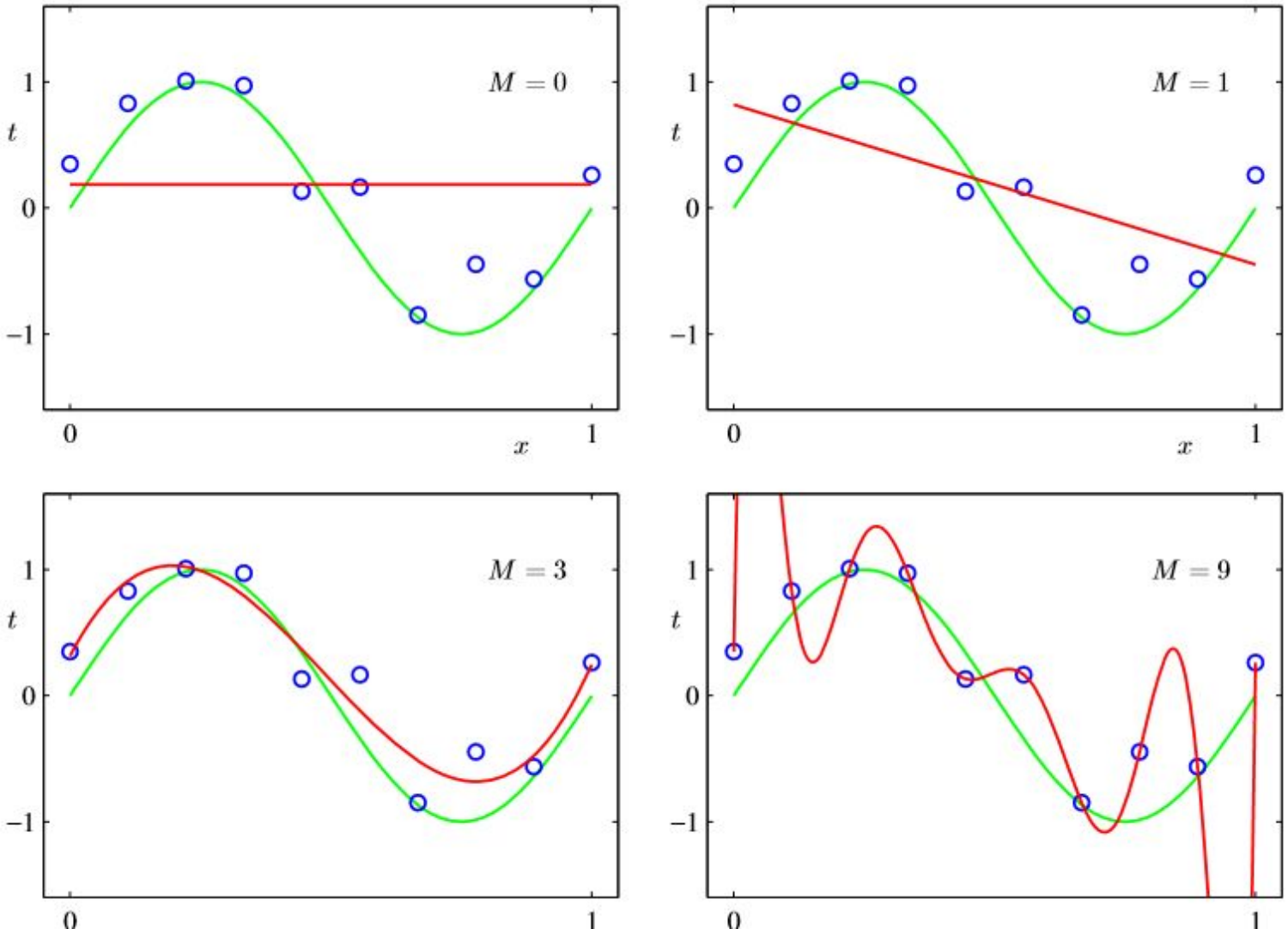
$$\frac{1}{|\boldsymbol{D}_{train}|}\Sigma_{(\boldsymbol{x},y)\in\boldsymbol{D}_{train}}\Omega(y, f(\boldsymbol{x};\theta))$$

However, a model performs well on $\boldsymbol{D}_{train}$ does not guarantee doing well on $\boldsymbol{D}_{test}$

Overfitting & Underfitting

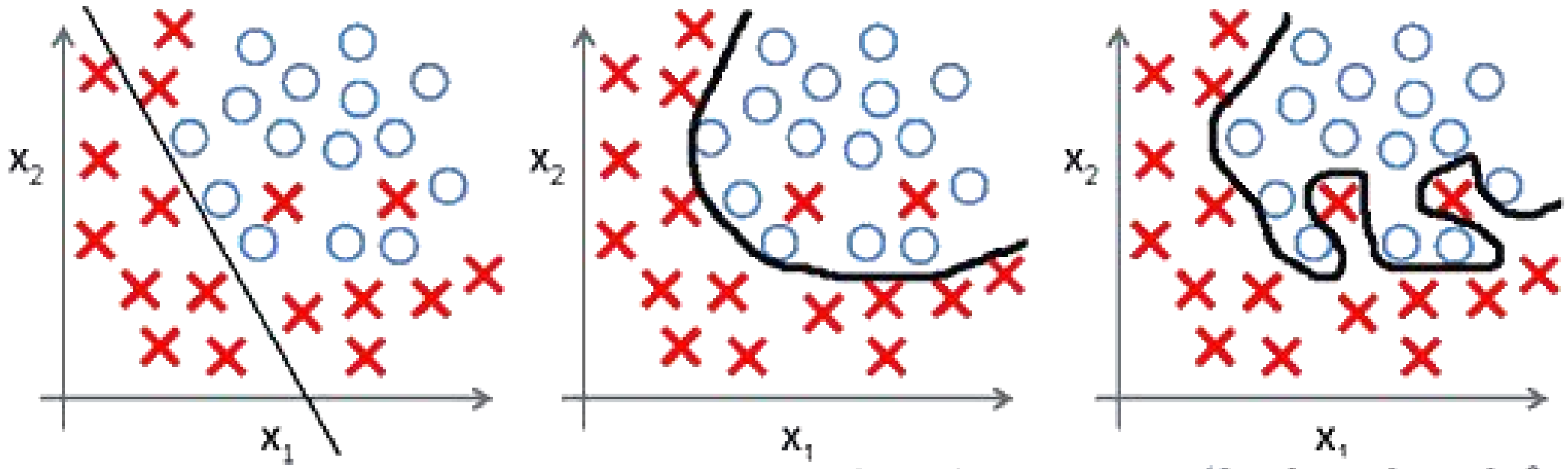
$$f(\boldsymbol{x};\boldsymbol{\theta}) = \Sigma_d \theta_d x^d$$

$$\min_{\boldsymbol{\theta}} P(f) = \Sigma_i (y_i - (\Sigma_d \theta_d x_i^d))^2$$

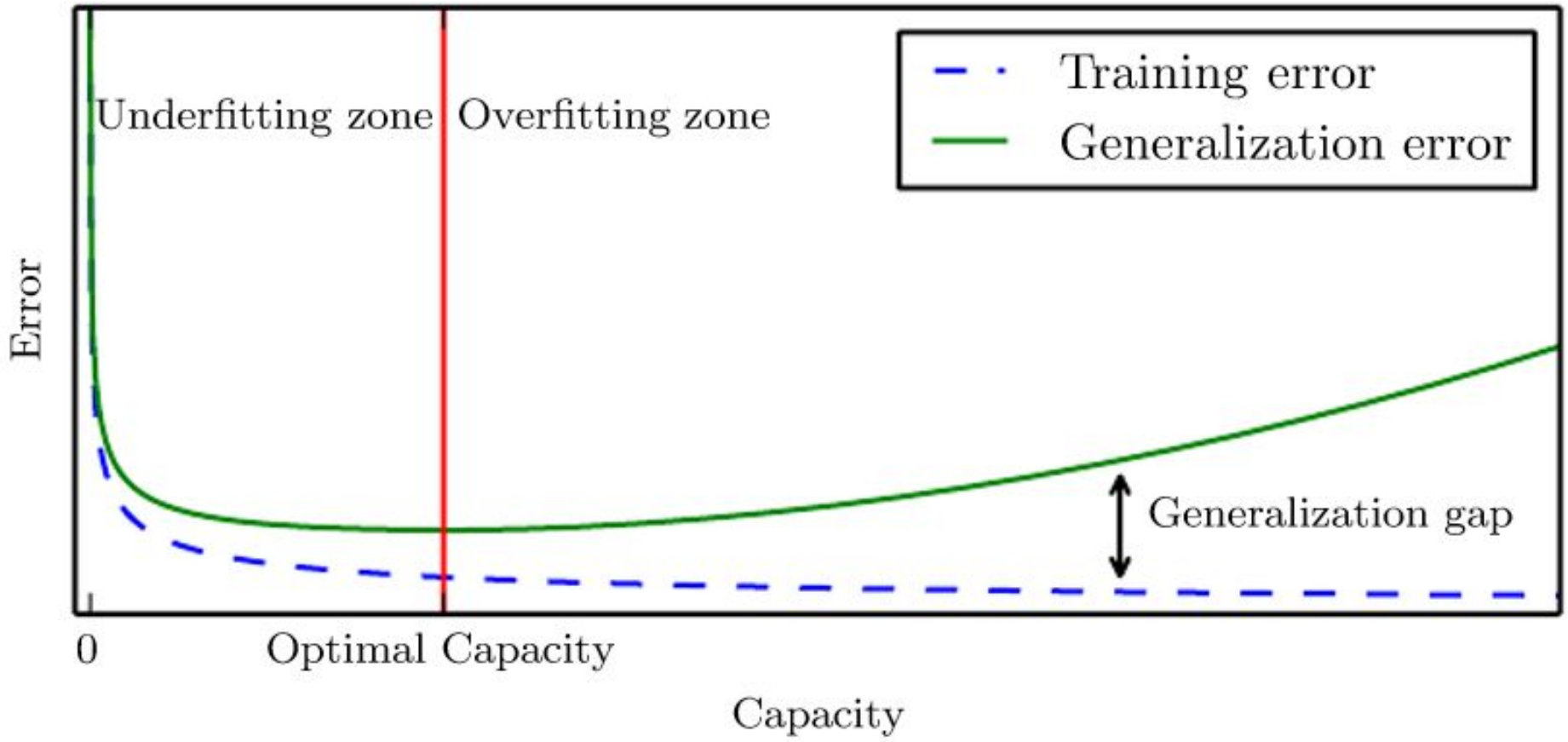


Face++ 旷视

Overfitting & Underfitting



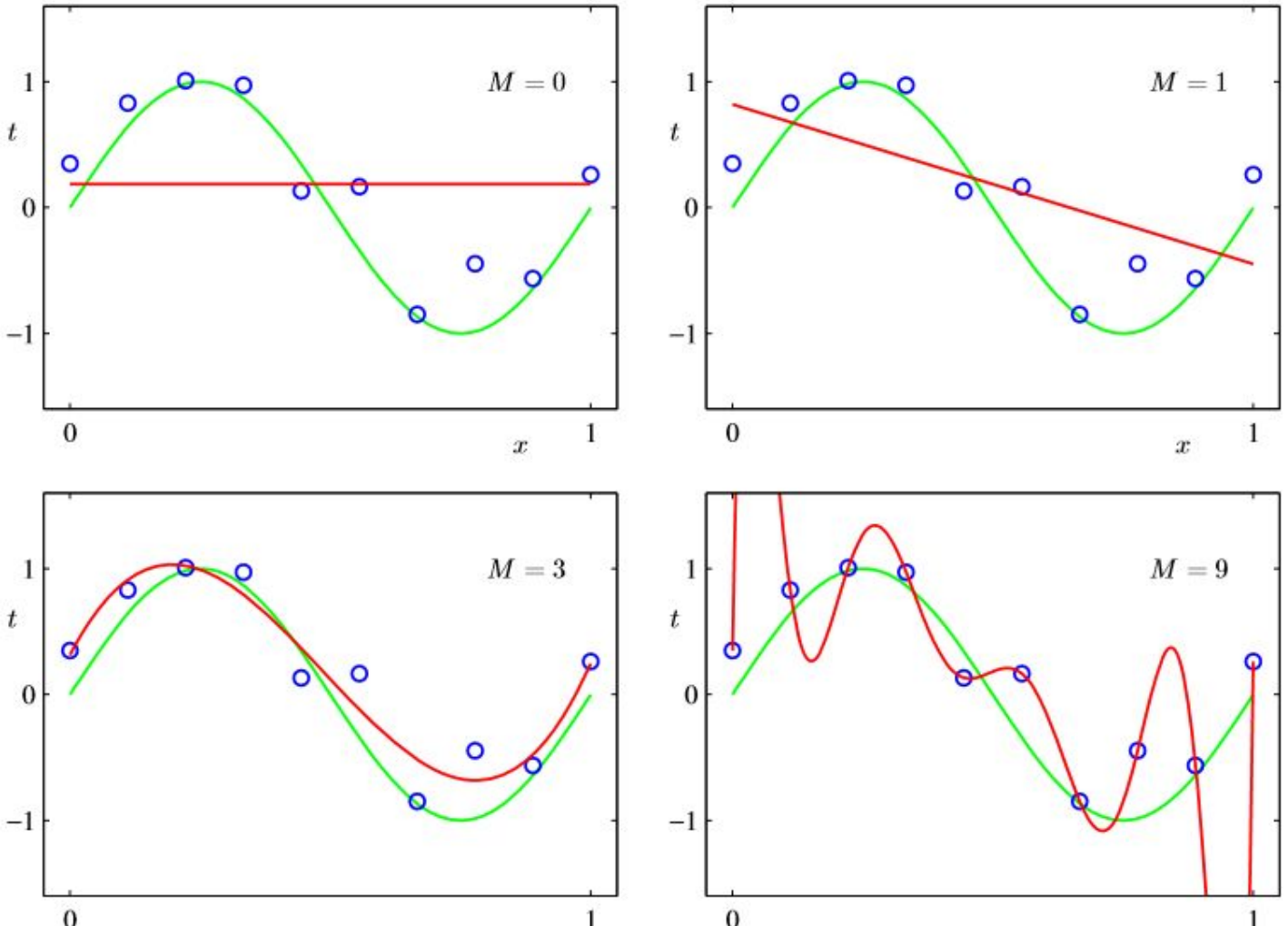
Model (Space) Capacity vs. Generalization Error



Regularization as Reducing Generalization Error

Capacity Control - Give preference to simple models

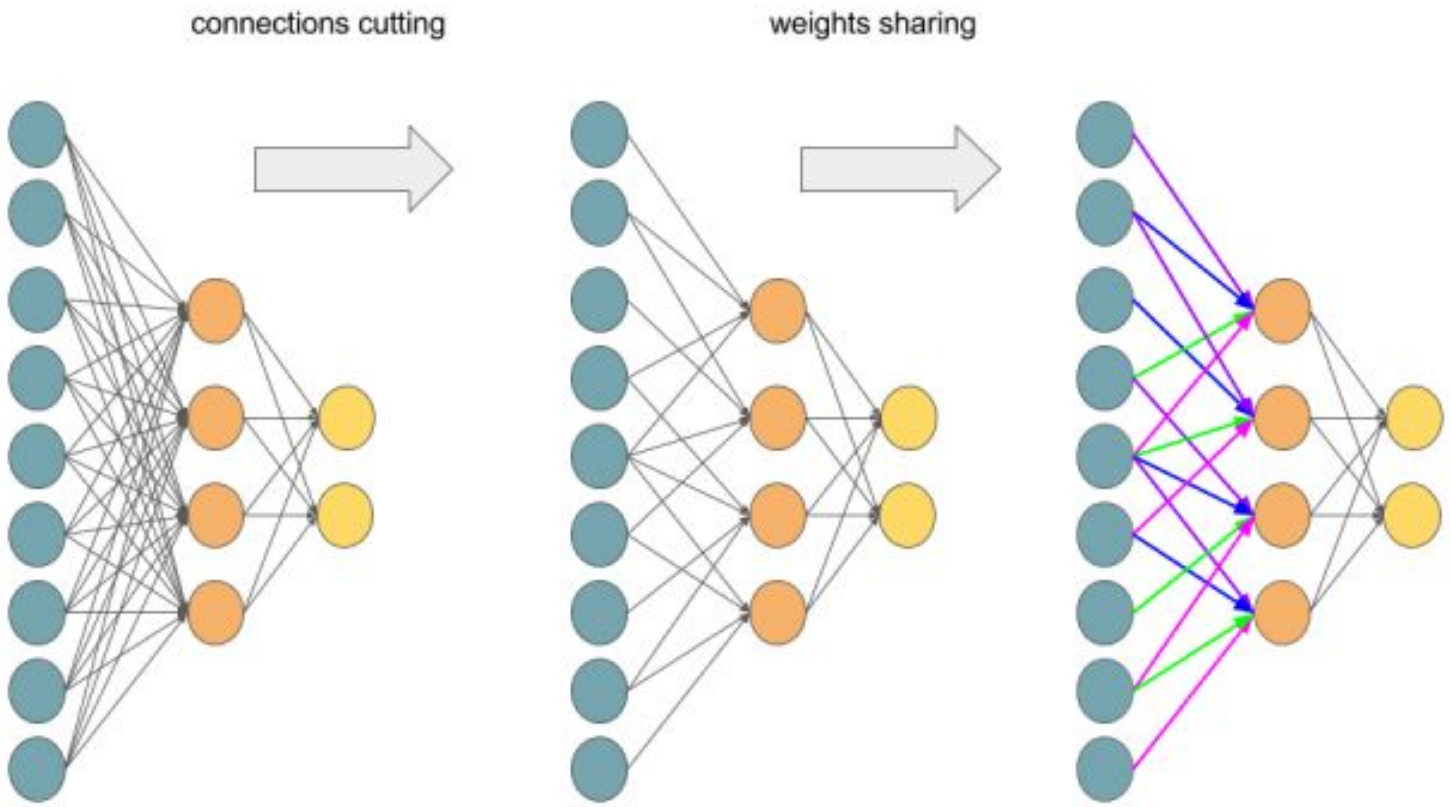
$$\min_{\boldsymbol{\theta}} \Sigma_i (y_i - (\Sigma_d \theta_d x_i^d))^2 + \lambda(\Sigma_d \theta_d^2)$$



Face++ 旷视

Regularization – Data Augmentation

Regularization – Parameter Reduce and Tying



With four parameters I can fit an elephant, and with five I can make him wiggle his trunk.
- John von Neumann

Face++ 旷视

Thanks！

Face++ 旷视

Matrix Multiplication - Why

Why matrix multiplication is defined as the way it is?

View it as the linear transformation mapping from a vector space to another. Compose two such linear mappings will lead to the matrix multiplication.

$$(AB)(x) = (A \circ B)(x) = A(B(x))$$

https://www.quora.com/Linear-Algebra-Why-is-matrix-multiplication-defined-the-way-it-is-1

Vector, Matrix - Norm

$$\|x\|_p = (\Sigma_{i=1}^{n} x_n^p)^{1/p}$$

$$\|A\|_F = \sqrt{\Sigma_{i=1,j=1}^{m,n} |a_{i,j}|^2}$$

Square Matrix – Matrix Inverse

Square matrix A is called invertible if there is a square matrix B (called the inverse matrix) so that

$$AB = BA = I_n$$

A is invertible if $\det(\mathrm{A}) \neq 0$.

Denote the inverse of A as A^{-1}

How to calculate the A^{-1} (see Cramer's Rule)

Square Matrix - properties

$$\mathrm{tr}(AB) = \Sigma_{i=1}^{m}\Sigma_{j=1}^{n} a_{i,j}b_{j,i} = \Sigma_{j=1}^{n}\Sigma_{i=1}^{m} b_{j,i}a_{i,j} = \mathrm{tr}(BA)$$

$$\|A\|_F = \sqrt{\Sigma_{i=1,j=1}^{m,n} |a_{i,j}|^2} = \sqrt{\mathrm{tr}(AA^T)} = \sqrt{\mathrm{tr}(A^T A)}$$