$\underline{A} = \begin{bmatrix} 3 & 6 \\ 2 & 5 \\ 1 & 4 \end{bmatrix}$ 3 rows, 2 cols

$\underline{A}$ is a 3 by 2 matrix

$[A]_{ij}$: element in row i col j

$L \times 1$ Let $\underline{a}_i = \begin{bmatrix} a_{1i} \\ a_{2i} \\ \vdots \\ a_{Li} \end{bmatrix}$, $i = 1, 2, \ldots M$

$L \times M$ $\underline{B} = \begin{bmatrix} \underline{a}_1 & \underline{a}_2 & \cdots & \underline{a}_M \end{bmatrix}$ a as cols

$M \times L$ $\underline{C} = \begin{bmatrix} \underline{a}_1^T \\ \underline{a}_2^T \\ \vdots \\ \underline{a}_M^T \end{bmatrix}$ a as rows

$\underline{C} = \underline{B}^T$: interchange rows and cols

$\underline{t}_i^T = [1 \;\; t_i \;\; t_i^2]$ => feature associated with i response/label.

s_i "label" (value goes along feature)

$$\Rightarrow \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_N \end{bmatrix} = \begin{bmatrix} \underline{t}_1^T \\ \underline{t}_2^T \\ \vdots \\ \underline{t}_N^T \end{bmatrix} \underline{W} \quad \text{or} \quad \underline{S} = \underline{T}\,\underline{W}$$

$N \times 1 \quad N \times 3 \quad 3 \times 1$

Matrix Multiplication

$\underline{A} : N \times M$ (i, j) $\quad \underline{B} : M \times L$ (k, l) $\quad \underline{C} : N \times L = \underline{A}\,\underline{B} : [C]_{mn}$ (m)

$$= \sum_{j=1} [\underline{A}]_{mj} [\underline{B}]_{jn}.$$

inner product of m^{th} row of $\underline{A}$ with n^{th} col of $\underline{B}$

$$\underline{A} = \begin{bmatrix} 3 & 4 \\ 2 & 5 \\ 1 & 6 \end{bmatrix}, \quad \underline{B} = \begin{bmatrix} -2 & 8 \\ 7 & -3 \end{bmatrix} \quad \underline{C} = \begin{bmatrix} 3(-2)+4\cdot 7 & 3\cdot 8 + 4(-3) \\ 2(-2)+5\cdot 7 & 2\cdot 8 + 5(-3) \\ 1(-2)+6\cdot 7 & 1\cdot 8 + 6(-3) \end{bmatrix}$$

Modeling multiple responses

r: $\begin{bmatrix} p_{r1} \\ p_{r2} \\ \vdots \\ p_{rL} \end{bmatrix} = \begin{bmatrix} \underline{t}_1^T \\ \underline{t}_2^T \\ \vdots \\ \underline{t}_L^T \end{bmatrix} \underline{W}_r$ $\quad b$: $\begin{bmatrix} p_{b1} \\ p_{b2} \\ \vdots \\ p_{bL} \end{bmatrix} = \begin{bmatrix} \underline{t}_1^T \\ \underline{t}_2^T \\ \vdots \\ \underline{t}_L^T \end{bmatrix}^T \underline{W}_b$

$r+b$ $\begin{bmatrix} p_{r1} & p_{b1} \\ \vdots & \\ p_{rL} & p_{bL} \end{bmatrix} = \underline{T} \; [\underline{W}_r : \underline{W}_b]$

$L \times 2$ $\quad L \times 3 \quad 3 \times 2$

Multiplication rules extended to block matrices

Generalizing $\underline{A} = \left[\begin{array}{c|c} \underline{A}_{11} & \underline{A}_{12} \\ \hline \underline{A}_{21} & \underline{A}_{22} \end{array}\right]$, $B = \left[\begin{array}{c|c} \underline{B}_{11} & \underline{B}_{12} \\ \hline \underline{B}_{21} & \underline{B}_{22} \end{array}\right]$

$$\underline{C} = \underline{A}\underline{B} = \left[\begin{array}{c|c} \underline{A}_{11}\underline{B}_{11} + \underline{A}_{12}\underline{B}_{21} & \underline{A}_{11}\underline{B}_{12} + \underline{A}_{12}\underline{B}_{22} \\ \hline \underline{A}_{21}\underline{B}_{11} + \underline{A}_{22}\underline{B}_{21} & \underline{A}_{21}\underline{B}_{12} + \underline{A}_{22}\underline{B}_{22} \end{array}\right]$$

All relevant submatrices must be conformable

Quiz: ① $A: 5\times3$ $B: 4\times3$ product AB is not defined.

② $A: 5\times3$ $B: 3\times1$ $C = Ab$: $C: 5\times1$

C is column vector

③ $y = Xw = \begin{bmatrix} 1\times1+2\times-1 \\ -1\times1+-1\times-1 \\ 2\times1+1\times-1 \end{bmatrix} = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$ 3×1

④ $X = TW = \begin{bmatrix} 1\times1+2(-1) & 1\times1+2\times2 \\ -1\times1+-1(-1) & -1\times1+-1\times2 \\ 2\times1+1(-1) & 2\times1+1\times2 \end{bmatrix} = \begin{bmatrix} -1 & 5 \\ 0 & -3 \\ 1 & 4 \end{bmatrix}$ 3×2

⑤ $C = \begin{bmatrix} A & b \end{bmatrix}$ 1×2 $F = \begin{bmatrix} D \\ e^T \end{bmatrix}$ 2×1

$CF = \begin{bmatrix} A\cdot D + b\cdot e^T \end{bmatrix}$ 1×1

$A: 3\times2$ $D: 2\times5$ → 3×5.

$b: 3\times1$ $e^T: 1\times5$ → 3×5

Classification is assigning a category to data

→ extract features based on a model

if we use a line to separate classes

$x_2 = mx_1 + b \Rightarrow x_2 - mx_1 - b = 0$

↳ rewrite as an inner product

$$\begin{bmatrix} x_2 & x_1 & 1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} = \underline{x}^T \underline{w} = 0$$

(x_2, x_1: features; w: classifier weights) ⇒ find $\underline{w}$

↳ curved decision boundaries:

$$\begin{bmatrix} x_2 & x_1^3 & x_1^2 & x_1 & 1 \end{bmatrix} \begin{bmatrix} w_1 \\ \vdots \\ w_5 \end{bmatrix} = \underline{x}^T \underline{w} = 0$$

A linear classifier is based on a weighted sum of features $\underline{x}^T \underline{w}$

Labels specify class associated with a feature binary classification (between 2)

Supervised learning: given features / labels.

$(\underline{x}_i, l_i)$ find $\underline{w}$ so $\underline{x}_i^T \underline{w} \approx l_i$

Training a linear classifier involves solving a system of linear equations.

$$\begin{bmatrix} \underline{x}_1^T \\ \underline{x}_2^T \\ \vdots \\ \underline{x}_N^T \end{bmatrix} \underline{w} \approx \begin{bmatrix} l_1 \\ l_2 \\ \vdots \\ l_N \end{bmatrix} \Rightarrow \underset{N \times M}{X} \underset{M \times 1}{\underline{w}} \approx \underset{N \times 1}{\underline{l}}$$

N training samples

M features

Classify candidate feature $\underline{x}$ using $\underline{w}$

↳ if $\underline{x}^T \underline{w} > 0 \Rightarrow$ label "1", if $\underline{x}^T \underline{w} < 0 \Rightarrow$ label "-1"

Patterns and Model Order

$\hat{p} = f(t) \rightarrow$ polynomial

"label" = $[\,1 \;\; t_i \;\; t_i^2\,]$ $\begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix}$ Quadratic

"feature" — "model"

building blocks - bases

$$\begin{bmatrix} \hat{p}_1 \\ \hat{p}_2 \\ \vdots \\ \hat{p}_{20} \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & t_1 & t_1^2 \\ 1 & t_2 & t_2^2 \\ \vdots & \vdots & \vdots \\ 1 & t_{20} & t_{20}^2 \end{bmatrix}}_{T} \begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} w_0 + \begin{bmatrix} t_1 \\ t_2 \\ \vdots \\ t_{20} \end{bmatrix} w_1 + \begin{bmatrix} t_1^2 \\ t_2^2 \\ \vdots \\ t_{20}^2 \end{bmatrix} w_{20} = \hat{p}$$

very simple model → might not good in data fit

complex + data fitting model ⇒ catching behavior not real / low generalization

- computing Tw involves inner product with rows of T
- Interpreting model with "bases" uses columns of T

Modeling Matrix Data

use a "taste profile" to model each user's pref

$\underline{t}_1 = \begin{bmatrix} 1 \\ 1 \\ -1 \\ -1 \end{bmatrix}$ $\sim -3\underline{t}_1 = \begin{bmatrix} -3 \\ -3 \\ 3 \\ 3 \end{bmatrix}$ $\sim 3\underline{t}_1 = \begin{bmatrix} 3 \\ 3 \\ -3 \\ -3 \end{bmatrix}$

$\hat{R}$ can be expressed as an "outer product"

$$= \begin{bmatrix} 1 \\ 1 \\ -1 \\ -1 \end{bmatrix} \underbrace{\begin{bmatrix} -3 & 3 & -1 & 3 \end{bmatrix}}_{\underline{s}_1^T} = \underline{t}_1 \underline{s}_1^T = \begin{bmatrix} -3 & 3 & -1 & 3 \\ -3 & 3 & -1 & 3 \\ 3 & -3 & 1 & -3 \\ 3 & -3 & 1 & -3 \end{bmatrix}$$

$\underline{t}_2 = \begin{bmatrix} 1 \\ -1 \\ 1 \\ -1 \end{bmatrix}$ $\hat{R} = \underline{t}_1 \underline{s}_1^T + \underline{t}_2 \underline{s}_2^T$

Interpret matrix multiplication as sum of outer products.

$$\hat{R} = [\underline{t}_1 \vdots \underline{t}_2] \begin{bmatrix} \underline{s}_1^T \\ \underline{s}_2^T \end{bmatrix} = \underline{t}_1 \underline{s}_1^T + \underline{t}_2 \underline{s}_2^T$$

$$R = TS = [\underline{t}_1 \vdots \underline{t}_2 \vdots \cdots \underline{t}_L] \begin{bmatrix} \underline{s}_1^T \\ \underline{s}_2^T \\ \vdots \\ \underline{s}_L^T \end{bmatrix} = \sum_{l=1}^{L} \underbrace{\underline{t}_l \underline{s}_l^T}_{N \times K}$$

$N \times K$ $N \times L$ $L \times K$

- choose L to trade model fit and generalization
- Inner product for computation, outer for interpretation

1. $c = Ab$ $\quad k \times 1$

$$\begin{bmatrix} a_1 & a_2 & \cdots & a_m \\ a_2 & & & \\ \vdots & & & \\ a_k & & & a_{mk} \end{bmatrix} \begin{bmatrix} b \\ \vdots \\ b_m \end{bmatrix}$$

$k \times m$. $\quad m \times 1 \rightarrow k \times 1$

2. 500×10000

3. $X = TW$

$3 \times 2 \cdot 2 \times 4 \rightarrow 3 \times 4$

$$\begin{bmatrix} 2\times1+2(-1) & 2\times1+2\times2 & 2\times3+2\times2 & 2\times2+2\times1 \\ -1(1)+1(-1) & -1(1)+1\times2 & -1\times3+1\times2 & -1\times2+1\times1 \\ 2\times1+1(-1) & 2\times1+2\times1 & 2\times3+1\times2 & 2\times2+1\times1 \end{bmatrix}$$

4. $\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \times \begin{bmatrix} 2 & -3 & -2 \end{bmatrix}$ $\quad 3 \times 3$

$3 \times 1 \quad 1 \times 3$

$$\hookrightarrow \begin{bmatrix} 1\times2 & 1\times(-3) & 1\times(-2) \\ 2\times2 & 2\times(-3) & 2\times(-2) \\ 3\times2 & 3\times(-3) & 3\times(-2) \end{bmatrix}$$

5. $X = TW \quad T = \begin{bmatrix} t_1 & t_2 \end{bmatrix} \quad W = \begin{bmatrix} w_1^T \\ w_2^T \end{bmatrix}$

$1 \times 2 \quad 2 \times 1$

$t_1 w_1^T + t_2 w_2^T$

2.1. Linear dependence and Ranking in Learning

Model fitting: $\begin{bmatrix} \hat{p}_1 \\ \hat{p}_2 \\ \vdots \\ \hat{p}_{20} \end{bmatrix} = \begin{bmatrix} 1 & t_1 & t_1^2 \\ 1 & t_2 & t_2^2 \\ \cdots & \cdots & \cdots \\ 1 & t_{20} & t_{20}^2 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix}$

$\underline{A}\,\underline{w} = \underline{d}$ ($N\times M$, $M\times 1$, $N\times 1$)

Classifier design: $\begin{bmatrix} l_1 \\ l_2 \\ \vdots \\ l_n \end{bmatrix} = \begin{bmatrix} \underline{x}_1^T \\ \vdots \\ \underline{x}_N^T \end{bmatrix} \underline{w}$

To solve $\underline{A}\underline{w} = \underline{d}$, $\underline{A}$ can be represented as column vector $[\underline{a}_1 \dots \underline{a}_n]$ ($N\times 1$)

$\underline{A}\underline{w} = \underline{d} \Rightarrow \underline{d} = \sum_{i=1}^{N} \underline{a}_i w_i$ (weighted sum of $\underline{a}_i$)

in graph $\underline{d}$ is a set $\hat{\underline{d}} \neq \underline{a}_1 w_1 + \underline{a}_2 w_2$

↳ $\underline{d} - \sum_{i=1}^{M} \underline{a}_i w_i = 0$

Linear independence: A set of M vectors $\underline{v}_1, \underline{v}_2, \dots \underline{v}_m \in \mathbb{R}^N$ is linearly independent iff $\sum_{i=1}^{M} \underline{v}_i \alpha_i = 0 \Leftrightarrow \alpha_i = 0, i = 1,2,\dots M$ (if weights are all 0)

otherwise it is linearly dependent ← if there is non-zero weights that

Rank of matrix: number of linearly independent columns (or rows) (row rank = column rank)

Example: $\underline{A} = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & -1 \\ \underline{a}_1 & \underline{a}_2 & \underline{a}_3 \end{bmatrix}$

$\underline{a}_3 = \underline{a}_2 - \underline{a}_1$

so $\alpha_1 \underline{a}_1 + \alpha_2 \underline{a}_2 + \alpha_3 \underline{a}_3 = 0$

for $\alpha_1 = 1 \quad \alpha_2 = -1 \quad \alpha_3 = 1 \neq 0$

a_1, a_2, a_3 are linearly dependent

while a_1, a_2 | a_2, a_3 | a_1, a_3 are independent ← can only describe in 2-D dimension

↳ rank of $\underline{A}$ is 2

solution: $\underline{A}\underline{w} - \underline{d} = \underline{0}$ or

$\sum_{i=1}^{M} \underline{a}_i w_i + (-1)\underline{d} = 0 \Rightarrow \underline{a}_1, \underline{a}_2 \dots \underline{a}_m, \underline{d}$ are linearly dependent

1) if $\underline{d}$ is a linear combination of $\underline{a}_1, \underline{a}_2, \dots \underline{a}_m$ there is a solution

$\text{rank}(A) = \text{rank}([\underline{A} \vdots \underline{d}])$

↳ A with d add as an extra column (d is dependent)

2) if $\underline{a}_1, \underline{a}_2 \dots \underline{a}_m, \underline{d}$ are linearly independent. there is no solution

$\text{rank}(A) < \text{rank}([\underline{A} \vdots \underline{d}])$

Solution to $\underline{A}\underline{w} = \underline{d}$ may not be unique

↳ Suppose $\underline{A}\underline{w} = \underline{d}$, does $\underline{f} \neq 0$ exist so that $\tilde{\underline{w}} = \underline{w} + \underline{f}$ also satisfy $\underline{A}\tilde{\underline{w}} = \underline{d}$?

$$\underline{A}\tilde{\underline{w}} = \underline{d} \Rightarrow \underline{A}\underline{w} + \underline{A}\underline{f} = \underline{d}$$

$$\Rightarrow (\underline{A}\underline{w} - \underline{d}) + \underline{A}\underline{f} = 0$$

$$\Rightarrow \underline{A}\underline{f} = 0$$

↳ $\sum_{i=1}^{M} a_i f_i = 0$ for $\underline{f} \neq 0$ ← linearly dependent

⇒ infinite

we got non unique solution iff cols. in $\underline{A}$ are linearly dependent

Solution to $\underline{A}\underline{w} = \underline{d}$:

① $\text{rank}\{\underline{A}\} < \text{rank}\{[\underline{A} : \underline{d}]\}$ no solution

② $\text{rank}\{\underline{A}\} = \text{rank}\{[\underline{A} : \underline{d}]\}$

1' $\text{rank}\{\underline{A}\} = \dim\{\underline{w}\}$ unique solution

2' $<$ non-unique solution

Quiz: 1' $A : 10 \times 4$ $w : 4 \times 1$ $d : 10 \times 1$

Subspace in ML

A subspace $S \subseteq \mathbb{R}^N$ (set of N-dim points) satisfy

1. $\underline{0} \in S$ (contain origin)
2. if $\underline{f}, \underline{g} \in S$, then $\underline{f} + \underline{g} \in S$ (close under addition)
3. if $\underline{f} \in S$, then $\alpha \underline{f} \in S$ (close under scalar multi.)

Consider $S \subseteq \mathbb{R}^N$, $\{[\underline{v}_1 \underline{v}_2 \dots \underline{v}_m] \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_m \end{bmatrix} = \underline{V}\underline{w}$ (conv) for $\underline{w} \in \mathbb{R}^m\}$ ×.

Dimension of $\{\underline{V}\underline{w}\}$ is rank $(\underline{V})$

1) if $\underline{w} = 0$, $\underline{V}\underline{w}$ is $\underline{0} \in S$

2) $f = \underline{V}\underline{w}_f$ $g = \underline{V}\underline{w}_g$ $f + g = \underline{V}(\underline{w}_f + \underline{w}_g) \in S$

3) $f = \underline{V}\underline{w}_f$ $\alpha f = \underline{V}(\alpha \underline{w}_f) \in S$

In general $S = \{\underline{V}\underline{w}\}$ is a $k = \text{rank}\,\underline{V}$ dim hyperplane (origin) in $\mathbb{R}^N$

$\hat{\underline{R}} = \underline{T}\underline{S}$

↳ $\text{rank}(\underline{T}\underline{S}) \leq \min\{\text{rank}(\underline{T}), \text{rank}(\underline{S})\}$

- special case: $\underline{T} = N \times M$, $\text{rank}(T) = M$ iff $\text{rank}(\hat{\underline{R}}) = M$

$\underline{S} : M \times k$, $\text{rank}(S) = M$

$\underline{R} \approx \underline{\hat{R}}$, $\underline{\hat{R}} = \underline{T}\,\underline{S} = \sum_{i=1}^{m} \underline{t}_i\,\underline{s}_i^T$ rank-M approximation to R

$\underline{\hat{R}} = [\underline{\hat{r}}_1, \underline{\hat{r}}_2, \ldots \underline{\hat{r}}_k]$ $\underline{\hat{r}}_i$ lie in M-dimensional subspace

Bases

$S = \text{span}\{\underline{v}_i\} \quad \sum_{i=1}^{M} \underline{v}_i w_i$, $w_i \in \mathbb{R}$

1) $\underline{v}_i$ arbitrary. – hard computing if linear dep.
2) $\underline{v}_i$ linearly independent
3) $\underline{v}_i$ orthonormal – easiest computing

} Basis → unique relationship between $\underline{x}$ and w_i in 2) 3)

Orthonormal: $\underline{v}_i$ are orthogonal to each other and unit length

$\underline{v}_i^T \underline{v}_i = 1$, $\underline{v}_i^T \underline{v}_j = 0 \;\forall\; i \neq j$

Examples. $\underline{v}_1 = \begin{bmatrix}1\\1\\0\end{bmatrix}$ $\underline{v}_2 = \begin{bmatrix}0\\1\\0\end{bmatrix}$ basis for x-y plane 2D subspace in $\mathbb{R}^3$

$\underline{f} = \underline{v}_1 w_1 + \underline{v}_2 w_2$

↳ $= \frac{1}{\sqrt{2}}\begin{bmatrix}1\\1\\0\end{bmatrix} = \frac{1}{\sqrt{2}}\begin{bmatrix}-1\\1\\0\end{bmatrix}$ $\underline{v}_1^T \underline{v}_1 = 1$ $\underline{v}_1^T \underline{v}_2 = 0$

↳ thonormal basis for x-y plane

Approximate solutions, Norms, and the least-squares Problem

Solve exact solution for $\underline{A}\,\underline{w} = \underline{d}$

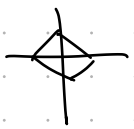
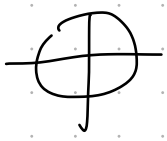
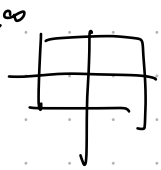
↳ $\sum_i \underline{a}_i w_i = \underline{d}$ $\underline{d}$ must lie in the subspace spanned by the column of $\underline{A}$

hard to satisfy.

↳ can we find $\underline{w}$ so $\underline{A}\,\underline{w} \approx \underline{d}$? $\underline{e} = \underline{A}\,\underline{w} - \underline{d}$, want $\underline{e}$ small

↳ To define "small", a vector "norm" measures the size of vector

- $\|\underline{e}\|_1 = \sum_i |e_i| = a+b$ → Taxi-cab norm l_1
- $\|\underline{e}\|_2 = \left(\sum_i |e_i|^2\right)^{1/2}$ Euclidean norm l_2
- $\|\underline{e}\|_\infty = \max_i |e_i|$ l_∞ max norm
- $\|\underline{e}\|_q = \left(\sum_i |e_i|^q\right)^{1/q}$ l_q norm

unit ball : $\{x : \|x\| = 1\}$ l_1  l_2  l_∞ 

- A vector norm $\|\cdot\|$ maps from $\mathbb{R}^N \to \mathbb{R}$ and satisfies:

① $\|\underline{x}\| \geq 0$ for all $\underline{x}$

② $\|\underline{x}\| = 0$ iff $\underline{x} = 0$

③ $\|b\underline{x}\| = |b|\,\|\underline{x}\|$ for all $b \in \mathbb{R}$, $\underline{x} \in \mathbb{R}^N$

④ triangle inequality $\|\underline{x} + \underline{y}\| \leq \|\underline{x}\| + \|\underline{y}\|$

l_p norms : $1 \leq p \leq q \leq \infty$ $\|\underline{x}\|_q \leq \|\underline{x}\|_p$

↳ $\|\underline{x}\|_\infty \leq \|\underline{x}\|_1$.

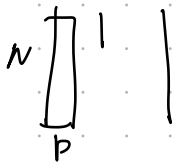
$\|\underline{x}\|_1$ → sparse solution.

$\|\underline{x}\|_\infty$ → constant magnitude solution

$\|\underline{x}\|_2$ → minimize squared error (Least-squares problem)

The least square problem

$\underline{A}\underline{w} = \underline{d}$ $\quad N \geq P$ $\quad \text{rank}(\underline{A}) = P$ $\quad \underline{\min \|A w - d\|_2^2}$ (minimize)



Solution: $\hat{\underline{d}} - \underline{d} \perp A$

$\perp$ span (A)

"orthogonality condition"

$A^T(Aw - d) = 0 \Rightarrow \underline{A}^T\underline{A}\,\underline{w} = \underline{A}^T\underline{d}$

$(A^TA)^{-1}(A^TA)w = (A^TA)^{-1}A^Td$ matrix inverse

$w = (A^TA)^{-1}A^Td$

Matrix inversion

Let $\underline{B}$ be a $P \times P$ invertible matrix. B^{-1} satisfies:

$B^{-1}B = I = \begin{bmatrix} 1 & & \\ & \ddots & \\ & & 1 \end{bmatrix}$ (identity matrix $\underline{I}_v = \underline{v}$)

$B\,B^{-1} = I$

$\underline{B} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$ $\quad B^{-1} = \frac{1}{ad - bc}\begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$

Not all matrix have inverses. Full rank square matrix are invertible

- A^TA is invertable iff $\underline{A}$ $(N \times P,\ P \leq N)$ is rank P
- positive definite $\Rightarrow$ invertable. $\underline{Q}$ is positive definite $(\underline{Q} > 0)$ iff $\underline{v}^T\underline{Q}\underline{v} > 0 \quad \forall\, \underline{v} \neq 0$

A^TA is positive definite:

Let $\underline{y} = \underline{A}\underline{v}$ $\quad \text{rank}(\underline{A}) = P \Rightarrow y \neq 0$ for $v \neq 0$

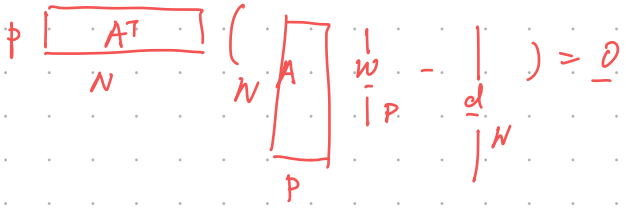
$(Av)^TAv = v^TA^TAv = y^Ty = \sum_i y_i^2 > 0$ for $(v \neq 0)$

$\Rightarrow (A^TA)^{-1}$ exists

Summary

$\min_{\underline{w}} \|\underline{A}\underline{w} - \underline{d}\|_2^2 \Rightarrow \min_{\underline{w}} \|\underline{e}\|_2^2$

$\Rightarrow \underline{e} \perp \text{span}\{\underline{A}\}$ $\quad \underline{A}^T\underline{e} = \underline{0}$



$\underline{w} = (A^TA)^{-1}A^Td$

$\|e\|_2^2 = \|d\|_2^2 - \|\hat{d}\|_2^2$

$= \underline{d}^T(I - \underline{P}_A)\underline{d}$

projection $\quad \underline{P}_A = \underline{A}(\underline{A}^T\underline{A})^{-1}\underline{A}^T$ $\quad \hat{d} = \underline{P}_A\underline{d}$

$\underline{P}_A^n = \underline{P}_A$ $\quad e = \underline{P}_{A^\perp}\underline{d}$

$\underline{P}_{A^\perp} = \underline{I} - \underline{P}_A$ projections onto space $\perp$ to span $\{\underline{A}\}$

w is min when $\Sigma^T(w)A^TA\Sigma(w) = 0$

$\min(f(w))$ is $w = w_0$

$= d^Td - d^TA(A^TA)^{-1}A^Td$

$= \|Aw - d\|_2^a$

N feature — parameter — N labels

notes: (ignore if grading)

$Ax = b$ if $b \in$ { $[A:b]$ rank $= [A]$; span(cols(A)) }

↳ A is linear ind → unique sol

dep → inf

column space

↳ not in → no solution

$$\begin{bmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_N^T \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_p \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_N \end{bmatrix}$$

feature vector | model parameter | labels

$\underline{A}\,\underline{w} = \underline{d}$ $\quad N \geq p$

$N \times p \quad p \times 1 \quad N \times 1$ $\quad \text{rank}(\underline{A}) = p$

full rank

$\min_{\underline{w}} \|\underline{A}\underline{w} - \underline{d}\|_2^2$ Let $\hat{\underline{d}} = A\underline{w}$, $\underline{d}$ lies in p dim subspace spanned by col of A

Solution $\hat{\underline{d}} - \underline{d} \perp$ span{col of A} $\Rightarrow \underline{a}_i^T(\hat{\underline{d}} - \underline{d}) = 0$..

$\underline{A}^T(\hat{\underline{d}} - \underline{d}) = 0$ orthogonality condition

$\underline{A}^T(\underline{A}\underline{w} - \underline{d}) = 0 \Rightarrow A^TAw = A^Td$ → square matrix, & full rank, have inverse.

↳ $(A^TA)^{-1}(A^TA)w = (A^TA)^{-1}A^Td$

$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$ $A^{-1} = \frac{1}{ad-bc}\begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$

Invertable condition:

- A^TA is invertable iff $\underline{A}$ ($N \times p$, $p \leq N$) is rank p
- Positive definite ⇒ invertable. Q is positive definite ($Q > 0$) iff

$\underline{v}^T\underline{Q}\,\underline{v} > 0 \quad \forall\, v \neq 0$

A^TA is positive since:

Let $y = Av$ since A is full rank, and $v \neq 0$, $Av \neq 0$, therefore $y \neq 0$

$(Av)^TAv = v^TA^TAv = y^Ty = \sum_i y_i^2 > 0 \quad (v \neq 0)$

Q is positive semidefinite iff $v^TQv \geq 0 \quad \forall\, v \neq 0$

$\|\underline{z}\|_2^2 = \underline{z}^T\underline{z}$

$\|Aw - d\|_2^2 = (Aw - d)^T(Aw - d)$

$= \underbrace{w^TA^TAw - w^TA^Td - d^TAw + d^Td}_{f(w)}$ ← we want to minimize

Scalar problem: ∪ set $\frac{d}{dw}g(w) = 0$ $\quad a^2 > 0 \rightarrow$ concave up

Gradients: differentiate $f(w)$ with respect to vector $\underline{w}$ $\quad \frac{d}{d\underline{w}}f(w) = 0$

↳ $\nabla_w f(w) = \left[\frac{d}{dw_1}f(w) \;\; \frac{d}{dw_2}f(w) \;\ldots\; \frac{d}{dw_p}f(w)\right]^T$

Suppose $f(w) = w^Th = h^Tw = \sum_{i=1}^{p} w_ih_i$ $\quad \frac{d}{dw_j}f(w) = h_j$

$\nabla_{\underline{w}} f(w) = [h_1\; h_2 \ldots h_p]^T = h$

Suppose $f(w) = w^TQw$ can show $\nabla_w f(w) = Q^Tw + Qw$

$Q = Q^T \Rightarrow \nabla_w f(w) = 2Qw$ if Q is symmetric

and $Q = A^TA$ $\quad (A^TA)^T = A^TA$ ⇐ symmetric.

$\nabla_w(w^TA^TAw - w^TA^Td - d^TAw + d^Td) = 0$

$2A^TAw - A^Td - A^Td = 0$

$$2A^TAw = 2A^Td.$$
$$w = (A^TA)^{-1}A^Td$$

$A^TA > 0 \Rightarrow \min f(w)$ when $w = w_0$

$$\min f(w) = d^Td - d^TA(A^TA)^{-1}A^Td$$

Projection and the Pythagorean Theorem

$\hat{d} = A(A^TA)^{-1}A^Td = P_Ad$. $P_A = A(A^TA)^{-1}A^T$ projection matrix.

↳ project vector into span plane onto A

$P_A^2 = P_A$ $P_A^n = P_n$

$e = d - \hat{d} = (I - P_A)d = P_{A^\perp}d$.

$P_{A^\perp} = I - P_A$ projects onto the space $\perp$ to span $\{A\}$

↳ also projection vector

$$\|d\|_2^2 = \|e\|_2^2 + \|\hat{d}\|_2^2$$
$$\|e\|_2^2 = \|d\|_2^2 - \|\hat{d}\|_2^2.$$
$$= d^Td - d^TP_Ad.$$
$$= d^T(I - P_A)d$$

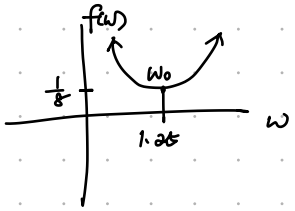
Geometry of $f(w)$ in P-dim space.

$f(w) = (w - w_0)^TA^TA(w - w_0) + d^TP_{A^\perp}d$ $\quad w_0 = (A^TA)^{-1}A^Td$

$A^TA > 0$ $\quad f(w) \geq f(w_0) = d^TP_{A^\perp}d$

$A = \begin{bmatrix}1\\1\end{bmatrix}$ $d = \begin{bmatrix}1\\1.5\end{bmatrix}$ $\quad w_0 = (A^TA)^{-1}A^Td = \frac{1}{2}(2.5) = 1.25$

$$f(w) = (w - w_0)^TA^TA(w - w_0) + d^TP_{A^\perp}d$$
$$= 2(w - w_0)^2 + \frac{1}{8}$$



Bowl shaped surface

- $f(w) = \text{const}$ are circles
- parabolic in each coordinate (w)

if A^TA not diagonal. $A^TA = U\Lambda^2U^T$, $\Lambda^2 = \text{diag}\{\lambda_1^2, \lambda_2^2 \ldots \lambda_p^2\}$

col of U are orthonormal $U = [u_1, u_2 \ldots u_p]$

$$u_l^Tu_k = \begin{cases}1 & l = k\\ 0 & l \neq k\end{cases}$$

$$U^TU = I$$

$$(w - w_0)^TA^TA(w - w_0) = (w - w_0)^TU\Lambda^2\underbrace{U^T(w - w_0)}_{z} = z^T\Lambda^2z$$

$$w - w_0 = Uz = u_1z_1 + u_2z_2$$

Orthonormal basis, projections

An orthonormal basis for a set of vectors x_1, x_2 is another set of vectors $u_1, u_2 \dots$ such that

① $u_i^T u_j = 0$ for all $i \neq j$

② $u_i^T u_i = 1$ for all i

③ span $\{x_1 \dots x_n\}$ = span $\{u_1, \dots u_m\}$

subspace

span $\{x_1 \dots x_n\} = \{x : x = \sum_{i=1}^{n} w_i x_i \quad w_i \in R \quad i = 1 \dots n\}$

m is dim(subspace)

$U = [u_1 \dots u_m]$

properties of orthonormal basis

1. $m \leq n$ $\quad m = n$ if full rank
2. $m \leq \dim(x_i)$
3. $U^T U = I_{m \times m}$
4. if U square $U^{-1} = U^T$
5. if U square $UU^T = I_{n \times n}$

Projections:

$\text{proj}_u d = u(u^T d)$

proj d onto u — direction — amount of d in direction of u

$$\text{Proj}_u d = \sum_{i=1}^{m} u_i (u_i^T d)$$

$$= \begin{bmatrix} | & & | \\ u_1 & \cdots & u_m \\ | & & | \end{bmatrix} \begin{bmatrix} u_1^T d \\ \vdots \\ u_m^T d \end{bmatrix}$$

$= UU^T d$ → proj onto space span by x

$\text{proj}_x x = UU^T x = x$

$\Downarrow$

$UU^T = x(x^T x)^{-1} x^T$ → projection vector

$\text{proj}_x d = x(x^T x)^{-1} x^T d$

- Gram-Schmidt orthogonalization
- orth(X)
- SVD

Low-rank decomposition emphasize patterns.

$A(N\times M) \approx T(N\times P), W^T(P\times M)$ $P<N$ $P<M$ $\text{rank}(T)=\text{rank}(W)=P$

$\text{rank}(TW^T)=P$

$$TW^T = \begin{bmatrix} | & | & & | \\ t_1 & t_2 & \cdots & t_P \\ | & | & & | \end{bmatrix} \begin{bmatrix} \cdots & w_1^T & \cdots \\ \cdots & w_2^T & \cdots \\ & \vdots & \\ \cdots & w_P^T & \cdots \end{bmatrix}$$

$$= \sum_{i=1}^{P} \underbrace{t_i w_i^T}_{\text{rank 1 patterns}} = \sum_{i=1}^{P} \underbrace{\begin{bmatrix} | \\ t_i \\ | \end{bmatrix}}_{N} \underbrace{\begin{bmatrix} - & w_i^T & - \end{bmatrix}}_{M} = \sum_{i=1}^{P} \boxed{N\times M}$$

Ex finding patterns

1) $\min_{T,W} \|A - TW^T\|$ singular value decomposition

2) $A \approx TW^T$ $T, W \geq 0$ non-negative matrix factorization

3) $A \approx TW^T$ each col W_a^T ~~[illegible]~~ all 0 and single 1 $\Rightarrow$ clustering

Clustering groups similar columns.

$$[a_1, a_2, a_3, a_4 \cdots a_m] \approx [t_1\ t_2\ t_3] \begin{bmatrix} 0 & 1 & 1 & & \\ 1 & 0 & 0 & \cdots & 0 \\ 0 & & & & \end{bmatrix}$$

$\Rightarrow a_1 \approx t_2$, $a_m \approx t_2$ $a_2 \approx t_1$ .... groups ones that are similar.

Algorithms: K-means

or Low rank models "complete" missing data.

Suppose $a = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} w_1 + \begin{bmatrix} -1 \\ \vdots \\ -1 \\ 1 \end{bmatrix} w_a$ $\begin{bmatrix} 1 & -1 \\ \vdots & \vdots \\ 1 & -1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} J\cdot J \\ a \end{bmatrix}$

average preference. $\hat{a} = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} J\cdot J + \begin{bmatrix} -1 \\ * \\ -1 \end{bmatrix} a$ = predicted value

↓ mitigate noise

Noisy data: $A_m = A_t + E$ (strong patterns) $\Rightarrow$ no dominant pattern.

low rank model: $\hat{A}_m = TW^T$ can be closer to A_t than A_m

Low rank classifier / model fit $Tw' = d$. T: transformed features. $x_i'^T$

$\hat{A}_m w = d$ $\begin{bmatrix} T \end{bmatrix} \begin{bmatrix} W^T \end{bmatrix} \underline{w} = \begin{bmatrix} | \\ d \\ | \end{bmatrix}$ w' $P\times 1$ $\quad x_i'^T = x_i^T W (W^T W)^{-1}$

new feature x'^T: $x'^T = x^T W (W^T W)^{-1}$ $\Rightarrow \hat{d} = \text{sign}(x'^T w')$

k-means

Clustering: organizing data into groups: given $a_i \in \mathbb{R}^N$, $i = 1, 2, \dots m$. find centroids $\mu_j, j = 1, \dots k$ and clusters $S_j = \{ i \mid a_i \text{ belongs to cluster } j \}$

unsupervised learning: data w/o labels, # clusters unknown

Matrix factorization: $A \approx T W^T$ $T = [\mu_1 \mu_2 \dots \mu_k]$

$T = [\mu_1 \mu_2]$ $W^T = \begin{bmatrix} 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$ $[W^T]_{lm} = \begin{cases} 1 & m \in S_l \\ 0 & m \notin S_l \end{cases}$

k-means algorithm → k clusters

clusters: $S_j = \{ i \mid a_i \in \text{cluster } j \}$, $|S_j| = \#\, a_i$ in S_j

centroids: $\mu_j = \frac{1}{|S_j|} \sum_{i \in S_j} a_i$ coherence: $c_j = \sum_{i \in S_j} \| a_i - \mu_j \|_2^2$

↳ average of data vector in cluster ↳ error between data from centroids

overall coherence: $c = \sum_{j=1}^{k} c_j = \sum_{j=1}^{k} \sum_{i \in S_j} \| a_i - \mu_j \|_2^2 = \| A - T W^T \|_F^2$

① initialize: choose $\mu_j^0, j = 1, 2, \dots k$ randomly from a_{ij} set $l = 0$

② Assignment: put $a_i \in S_j^l$ if a_i is closest to μ_j^l

③ update centroids $\mu_j^{l+1} = \frac{1}{|S_j^{l+1}|} \sum_{i \in S_j^l} a_i$

④ if converaged → stop

else → $l = l+1$, go to 2

converge: not changing.

options: initization.

fixed iteration.

overall / cluster.

use different norms

↳ challenge: convergence to local maxima

unknown k

SVD matrix decomposition that lead to good low rank approximations

Any $N \times M$ matrix A can be written as $A = U \Sigma V^T$

U: $N \times N$ orthonormal
V: $M \times M$ orthonormal
Σ: $N \times M$ diagonal $\Sigma_{ii} \geq 0$

$N > M$

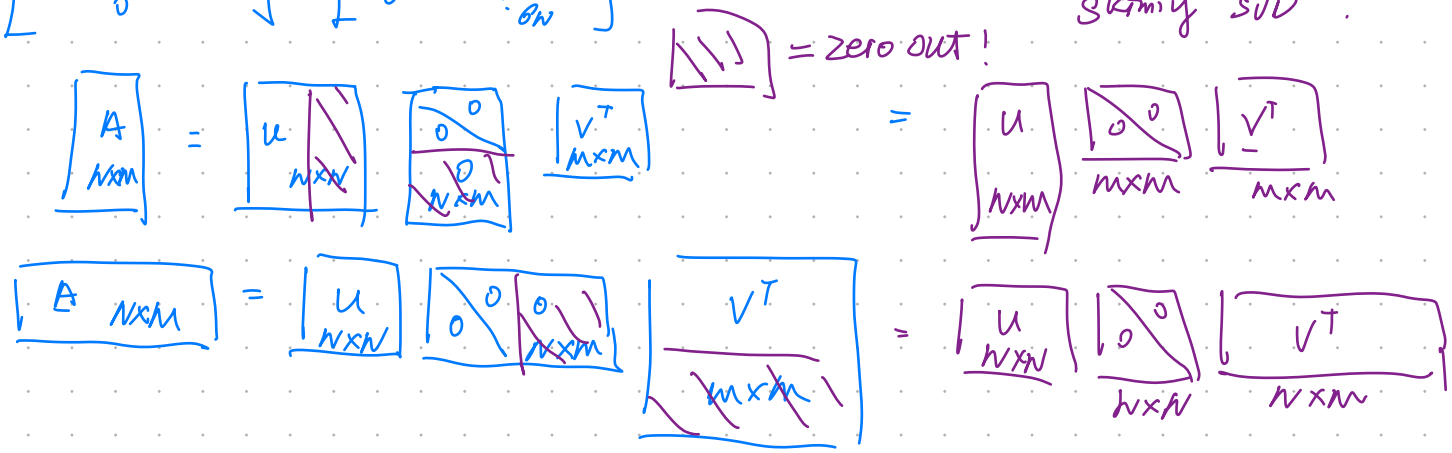
$$\begin{bmatrix} \sigma_1 & & 0 \\ & \sigma_2 & \\ & & \ddots \\ 0 & & \sigma_M \end{bmatrix}$$

$M > N$

$$\begin{bmatrix} \sigma_1 & & & 0 \\ & \sigma_2 & & \\ & & \ddots & \\ 0 & & & \sigma_N \end{bmatrix}$$

$\sigma_1 \geq \sigma_2 \geq \sigma_{N/M} \geq 0$ largest to smallest.

Skinny SVD.



Sum of outer products form:

$$A = \begin{bmatrix} | & | & & | \\ u_1 & u_2 & \cdots & u_M \\ | & | & & | \end{bmatrix} \begin{bmatrix} \sigma_1 & & 0 \\ & \sigma_2 & \\ 0 & & \sigma_M \end{bmatrix} \begin{bmatrix} - & v_1^T & - \\ & \vdots & \\ - & v_M^T & - \end{bmatrix} = \sum_{i=1}^{M} \sigma_i u_i v_i^T = \sum_{i=1}^{m} \underbrace{\sigma_i u_i v_i^T}_{N \times M}$$

$u_i v_i^T$: rank 1 matrix

"rank 1"

看 rank-k approximation

Frobenius norm: $\|A\|_F^2 = \sum_{i=1}^{N} \sum_{j=1}^{M} ([A]_{ij})^2 = \|\text{vec}(A)\|_2^2$

Eckart-Young Theorem: Let $\text{rank}(A) = r$ and $k < r$:

$$= \sum_{i=k+1}^{r} \sigma_i^2 \quad \text{for } B = \sum_{i=1}^{k} \sigma_i u_i v_i^T \text{ where } A = U \Sigma V^T \text{ is SVD}$$

closest ↑

$$\min_{\text{rank}(B) \leq k} \|A - B\|_F^2$$

B is best rank-k approximation to A

σ_i provide ordered ranking of components

$$A \approx \sigma_1 \underbrace{u_1 v_1^T}_{\text{most important}} + \sigma_2 \cdots \quad \text{(2-nd)} \quad \cdots \quad \text{(k-th)}$$

patterns

SVD describes matrix as an operator

$A : N \times M$ $\quad x : M \times 1$ $\quad y : N \times 1$

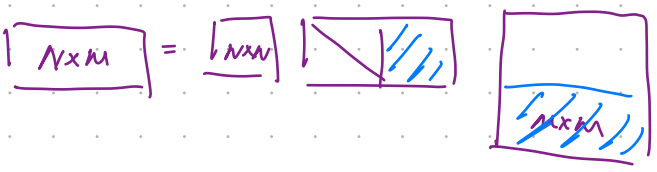
$$y = Ax = U \Sigma V^T x = U[\Sigma (V^T x)]$$

Orthonormality

$U^T U = I$; $V^T V = I$ → full / econ

$UU^T = I_n$; $VV^T = I_m$ → full only

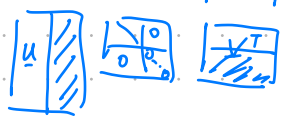
↳ in econ : U is ▯ ▭ ⇒ rank(M)× rank(N) cannot be N dim



Rank

$\text{rank}(A) = P \iff \sigma_1 \geq \cdots \geq \sigma_P > \sigma_{P+1} = \cdots = \sigma_{\min(M,N)} = 0$

$A = \sum_{i=1}^{P} \sigma_i u_i v_i^T$



20x10 10x10 10x10

20x26

$$\begin{bmatrix} | & | & & | \\ a_1 & a_2 & \cdots & a_m \\ | & | & & | \end{bmatrix} = \begin{bmatrix} | & | & & | \\ u_1 & u_2 & \cdots & u_p \\ | & | & & | \end{bmatrix} \begin{bmatrix} | & | & & | \\ c_1 & c_2 & \cdots & c_m \\ | & | & & | \end{bmatrix} \Rightarrow a_i = \sum_{j=1}^{P} u_j [c_i]_j$$

X, $C = \Sigma V^T$

a_i is built from $[c_i]_j$ in basis of u

left singular vector u: form an orthonormal basis from the columns of A

j^{th} coord $\sim \sigma_j [c_i]_j = \sigma_j [V^T]_{j,i}$ (≤ 1 since orthonormal)

the coordinate associated with j^{th} column of u is proportional to size of singular value.

right singular vector V is orthonormal basis for rows of A $\quad D = U\Sigma$

j coord $\sim \sigma_j [d_i]_j = \sigma_j [U]_{i,j}$ → rows !

$N = M$ $\quad A = U\Sigma V^T$ $\quad U, \Sigma, U$: NxN all invertible

Non invertible (singular): rank(A) < N (not full rank)

$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_P > \sigma_{P+1} = \sigma_N = 0$

Invertible : rank(A) = N $\quad A^{-1} = V\Sigma^{-T}U^T$

$A \cdot A^{-1} = U\Sigma V^T V \Sigma^{-T} U^T = U \Sigma \Sigma^{-1} U^T$

Σ: NxN diagnol, $\Sigma^T = \Sigma$

$= U I U^T = I$. (no econ SVD for full rank square)

$A = \sum_{i=1}^{N} \sigma_i u_i v_i^T$, $A^{-1} = \sum_{i=1}^{N} \frac{1}{\sigma_i} v_i u_i^T$ ($\frac{1}{\sigma_i} \to \Sigma^{-1}$) $\quad$ SVD of A → SVD of A^{-1}

SVD vs LSP

NxP Nx(N-P)

$U = [\tilde{U} : U_1]$ NxN $\quad$ $\tilde{U}$ span A

$A = \tilde{U} \Sigma V^T$

$(EFG)^{-1} = G^{-1}F^{-1}E^{-1}$ $\quad V^{-1} = V^T$

$w = (A^TA)^{-1}A^Td = (V\Sigma \tilde{U}^T \tilde{U} \Sigma V^T)^{-1} V \Sigma \tilde{U}^T d$

$= (V\Sigma^2 V^T)^{-1} V \Sigma \tilde{U}^T d$

$= V^{-T} \Sigma^{-2} V^{-1} V \Sigma \tilde{U}^T d$

$= V^{-T} \Sigma^{-2} V^T V \Sigma \tilde{U}^T d$

$= V \Sigma^{-1} \tilde{U}^T d = \sum_{i=1}^{P} \frac{1}{\sigma_i} v_i (\tilde{u}_i^T d)$

$w = V \Sigma^{-1} U^T d$

pseudo inverse of A

LS error and projection

$$\hat{d} = A[(A^TA)^{-1}A^T]d = \tilde{U}\Sigma V^T[V\Sigma^{-1}\tilde{U}^T]d$$

$$= \tilde{U}\tilde{U}^Td \qquad P_A = \tilde{U}\tilde{U}^T \rightarrow \text{span}\{A\} = \text{span}\{\tilde{U}\}$$

$$P_{A\perp} = I - \tilde{U}\tilde{U}^T$$

$$e = d - \hat{d} = (I - \tilde{U}\tilde{U}^T)d = P_{A\perp}d$$

$$U^TU = I = [\tilde{U} : U_\perp]\begin{bmatrix}\tilde{U}^T \\ U_\perp^T\end{bmatrix}$$

$$I = \tilde{U}\tilde{U}^T + U_\perp U_\perp^T$$

$$P_{A\perp} = U_\perp U_\perp^T$$

Classification → all same norm, tell importance, can look at relative size of w'

$A = \tilde{U}\Sigma V^T$ $\quad \tilde{U}w' = d \rightarrow w' = \tilde{U}^Td$ $\quad |w_i'| \rightarrow$ importance of i^{th} ortho feature

$w = V\Sigma^{-1}w'$

orthobases prediction

$\tilde{y} = \text{sign}(\tilde{x}^Tw) = \text{sign}(\tilde{x}^TV\Sigma^{-1}w')$

$\tilde{x}'^T = \tilde{x}^TV\Sigma^{-1}$ $\quad \tilde{y} = \text{sign}(\tilde{x}'^Tw')$

transformed feature $\quad$ orthobasis classifier

$A = UG \quad G = \Sigma V^T \quad x'^T = x^TG^{-1}$

$$x'^T = x^TV^{-T}\Sigma^{-1}$$

$$= x^TV\Sigma^{-1}$$

$$x' = xV^T\Sigma^{-T}$$

$$= xV^{-1}\Sigma^{-1}$$

LSP $A = U\Sigma V^T$ $\quad w = V\Sigma^{-1}U^T d = \sum_{i=1}^{p} \frac{1}{\sigma_i} v_i (u_i^T d)$

↳ $\|w\|_2^2 = \sum_{i=1}^{p} \left(\frac{1}{\sigma_i}\right)^2 (u_i^T d)^2$ small $\sigma_i \Rightarrow$ large $\|w\|_2$.

$\tilde{y} = (\bar{x} + \varepsilon)^T w = x^T w + \varepsilon^T w$

$|\varepsilon^T w|^2 = \|w\|_2^2 \|\varepsilon\|_2^2 \cos^2\theta$ → proportional to $\|w\|$ large $\|w\|_2^2$ → amplify error

if rank $(A) < p$ → $\sigma_p = 0$ → no unique solution.

Regularized LS via truncated SVD

replace $\sum_{i=1}^{p} \frac{1}{\sigma_i} v_i (u_i^T d)$ with $\sum_{i=1}^{r} \frac{1}{\sigma_i} v_i (u_i^T d)$ where $r < p$

avoid small singular value.

↳ rank-r approximation (replace $A = \sum_{i=1}^{p} \sigma_i u_i v_i^T$ with $A_r = \sum_{i=1}^{r} \sigma_i u_i v_i^T$

increase $\min_w \|Aw - d\|_2^2$

ridge regression

$\min_w \|Aw - d\|_2^2 + \lambda \|w\|_2^2 \Rightarrow w = (A^TA + \lambda I)^{-1} A^T d$. control size of norm

use SVD $A^TA = V\Sigma^2 V^T$ $\quad \lambda I = V \lambda I V^T$

$w = (V(\Sigma^2 + \lambda I)V^T)^{-1} V\Sigma U^T d$

$= V \underbrace{(\Sigma^2 + \lambda I)^{-1}}_{D} V^{-1} V \Sigma U^T d$.

$D = \begin{bmatrix} \frac{1}{\sigma_1^2+\lambda} & & 0 \\ & \ddots & \\ 0 & & \frac{1}{\sigma_r^2+\lambda} \end{bmatrix} \begin{bmatrix} \sigma_1 & & 0 \\ & \ddots & \\ 0 & & \sigma_r \end{bmatrix} = \begin{bmatrix} \frac{\sigma_1}{\sigma_1^2+\lambda} & & \\ & \ddots & \end{bmatrix}$

$\sigma_i \to 0$ $\frac{\sigma_i}{\sigma_i^2+\lambda} \to \frac{\sigma_i}{\lambda}$

$w = \sum_{i=1}^{p} \frac{\sigma_i}{\sigma_i^2+\lambda} v_i (u_i^T d)$ $\quad$ increased value $\|Aw - d\|_2^2$

Principle component analysis. PCA ⇒ maximum "variance"

Data : a_i, $i = 1, 2, \dots M$ $(N \times 1)$ vectors, $A = [a_1\ a_2 \dots a_M]$

① assumes zero mean, center the data

$a_i = \tilde{a}_i - \frac{1}{M} \sum_{j=1}^{M} \tilde{a}_j$

PC 1 : direction f accounting for maximum variance in data

$\|f\|_2^2 = 1$ ← unit norm $\quad \max_{\|f\|_2^2=1} \{\frac{1}{M} \sum_{i=1}^{M} \|a_i f\|_2^2\}$ best line

also $\min_{\alpha_i} \|a_i - \alpha_i f\|_2^2 \Rightarrow \max_{\|f\|=1} \{\frac{1}{M} \sum_{i=1}^{M} |f^T a_i|^2\}$

$\alpha_i = f^T a_i$

$\|f^T A\|_2^2 = \|A^T f\|_2^2$

PC are singular vectors

$\max_{\|f\|_2^2=1} \frac{1}{M} f^T A A^T f \Rightarrow A = U\Sigma V^T$ $\quad AA^T = U\Sigma V^T V \Sigma U^T = U\Sigma^2 U^T$

$\max_{\|f\|_2^2=1} \frac{1}{M} f^T U \Sigma^2 U^T f \Rightarrow f = u_1$ will maximize

variance with 1 PC $\frac{1}{M} u_1^T A A^T u_1 = \frac{\sigma_1^2}{M}$

Coord of data $\alpha_i = u_1^T a_i$ $\quad \alpha^T = [\alpha_1\ \alpha_2 \dots \alpha_M]$

$\alpha^T = u_1^T A = u_1^T U \Sigma V^T = \sigma_1 v_1^T$

right singular of v_1

root mean square coord.

$$\left(\frac{1}{M}\sum_{i=1}^{M}|\alpha_i|^2\right)^{1/2} = \frac{1}{M^{1/2}}\|\alpha\|_2 = \frac{\sigma_1}{M^{1/2}}$$

2th PC: $g^T u_1 = 0 \quad \max_{\|g\|_2=1} \frac{1}{M}\sum_{i=1}^{M}|g^T a_i|^2 \quad g = u_2$ (left singular vector)

k^{th} PC: left singular u_k

variance $\frac{\sigma_k^2}{M}$

coord RMS $\frac{\sigma_k}{M^{1/2}}$

rows $\Rightarrow$ right singular v_i PC

SVD give best rank r approximation to A $\quad A \approx u_1 (\sigma_1 v_1^T) \quad r=1$

$$w = V\Sigma^{-1}U^T d$$

Data is often contaminated by noise

$N \times M$ $A = S + G$

$M > N$ measured clean noise.

A: diffuse.

S: structured.

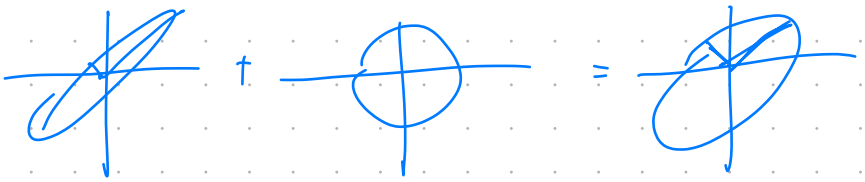
G: white noise – no preferred direction.

sum of squared errors: $\|G\|_F^2$

$$\|G\|_F^2 = \sum_{i=1}^{N} M\left(\frac{1}{M}\sum_{j=1}^{M} g_{ij}^2\right) = M\sum_{i=1}^{N} \text{var}_i \approx NM\sigma_g^2 \quad \text{(same for all } i\text{)}$$

isotropic

singular vectors are invariant (approx) to isotropic noise.



principal comp directions are unchanged.

variance along each component (singular vector vals) changes ($M \to \infty$)

$$U\Sigma_A V^T \approx U\Sigma_S V^T + U\Sigma_G V^T$$

$$\Sigma_A = \begin{bmatrix} \sigma_{A_1} & & \\ & \sigma_{A_2} & \\ & & \sigma_{A_N} \end{bmatrix} \quad \Sigma_S = \begin{bmatrix} \sigma_{S_1} & & \\ & \ddots & \\ & & \sigma_{S_N} \end{bmatrix} \quad \Sigma \approx \begin{bmatrix} M^{1/2}\sigma_g & & \\ & \ddots & \end{bmatrix} \quad \sigma_{g_i} = M^{1/2}\cdot \text{RMS}.$$

trade bias for variance error: $A - S \quad \|G\|_F^2 = NM\sigma_g^2$.

low rank: $\hat{A}_r = \sum_{i=1}^{r} \sigma_{A_i} u_i v_i^T \approx \hat{S}_r + \hat{G}_r$

$$\hat{S}_r = \sum_{i=1}^{r} \sigma_{S_i} u_i v_i^T \qquad \hat{G}_r \approx \sum_{i=1}^{r} M^{1/2}\sigma_g u_i u_i^T$$

bias^2: $b^2(r) = \|S - \hat{S}_r\|_F^2$

$$= \|\sum_{i=r+1}^{N} \sigma_{S_i} u_i v_i^T\|_F^2$$

sum of squared "tail" singular values

decrease as r increase

variance: $V(r) = \|\hat{G}_r\|_F^2$

$$V(r) = \|\sum_{i=1}^{r} M^{1/2}\sigma_g u_i v_i^T\|_F^2 = rM\sigma_g^2$$

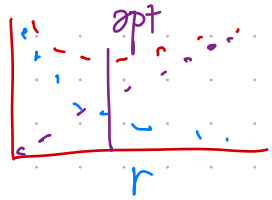
dimensions; variance per dimension

increase as r increase

S: highly structured $\sigma_1 \gg \sigma_2$.

weakly structured.

- $b^2(r) + V(r)$



- bias
- noise

most bias.

Eigendecomposition applies to square matrices

Eigenvector e_i, eigenvalue λ, B $(k \times k)$

$B e_i = \lambda_i e_i$ matrix mult $\Leftrightarrow$ scalar mult

$e_i \rightarrow \boxed{B} \rightarrow \lambda_i e_i$

- k eigenvalues, possibly complex valued
- distinct $\lambda_i \Rightarrow$ linearly independent e_i
- symmetric $B \Rightarrow$ k orthonormal e_i ($E E^T = E^T E = I$)

$\hookrightarrow B e_i = \lambda_i e_i \Rightarrow B [e_1\ e_2 \dots e_k] = [e_1 e_2 \dots e_k] \begin{bmatrix} \lambda_1 & & \\ & \lambda_2 & \\ & & \ddots \\ & & & \lambda_k \end{bmatrix}$

$BE = E\Lambda \Rightarrow B = E \Lambda E^T = \sum_{i=1}^{k} \lambda_i e_i e_i^T$

Symmetric PSD matrices and SVD

$A = [a_1 \dots a_M] = U \Sigma V^T$ full SVD

$\hookrightarrow B = AA^T = \sum_{i=1}^{M} a_i a_i^T$

$= U \Sigma V^T V \Sigma^T U^T = U \Sigma^2 U^T = U \begin{bmatrix} \sigma_1^2 & & & \\ & \sigma_2^2 & & \\ & & 0 & \\ & & & \ddots \end{bmatrix} U^T$

Network graph

Unit I

Gradient Descent for Least Square Problems

- Intuition: $\arg\min_w \|Aw-d\|_2^2 + \lambda r(w)$
  1. Computation Cost for $w=(A^TA)^{-1}A^Td$.
  2. Closed solution might not be available
  3. Adapt w to new feature / labels

↳ Iterative approach

- $f(w) = \|Aw-d\|_2^2$ $\quad \tau' > 0$ step size

$w^{k+1} = w^k - \tau' \nabla_w f(w)$ as $\nabla_w f(w) \downarrow$, step $\downarrow$

$f(w) = (Aw-d)^T(Aw-d)$

$= w^TA^TAw - 2w^TA^Td + d^Td$

$\nabla_w f(w) = 2A^TAw - 2A^Td = 2A^T(Aw-d)$

$w^{k+1} = w^k - \tau A^T(Aw^k - d)$

τ: convergence behavior

τ too small : slow convergence too big : might no convergence

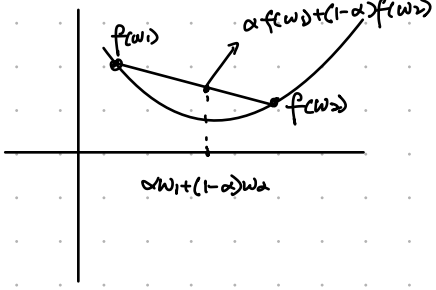
for convergence : $f(w^{k+1}) < f(w^k)$ ⇒ cost decrease

$\|Aw^{k+1}-d\|_2^2 < \|Aw^k-d\|_2^2$

$0 < \tau < 2/\|A\|_{op}^2 = 2/\sigma_{max}^2$

→ guaranteed convergence to $(A^TA)^{-1}A^Td$

Gradient descent is effective for convex cost functions



⇒ convex = have local minima

$f(\alpha w_1 + (1-\alpha)w_2) \le \alpha f(w_1) + (1-\alpha) f(w_2)$ for $0 \le \alpha \le 1$

Multidimensional case

$H(w) \succcurlyeq 0 \quad [H(w)]_{ij} = \frac{\partial^2}{\partial w_i \partial w_j} f(w)$

$\|X\|_{op}$: largest norm X could have ⇒ $\|Xw\|_2 \le \|X\|_{op}\|w\|_2$

Graph of contour

- σ_1/σ_2 ratio ↑ the ratio between major & minor ↑
- right singular vectors define orientation of ellipse.
- if w start at major / minor axis, path is perpendicular to the w_{LS}
- steeper graph (ratio higher) ⇒ more steps.

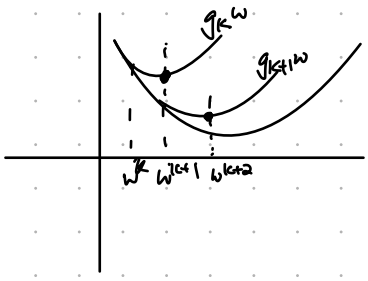
Alternative way for LSP with regularizer

$A^T(AA^T+\lambda I) = (A^TA+\lambda I)A^T$

$w = (A^TA+\lambda I)^{-1}A^Ty = A^T(AA^T+\lambda I)y$

↳ computation cost for $A = n \times m$

Proximal Gradient Descent for regularized LSP



- solve sequence for simpler problems
- simple for separatable function $r(w) = \sum_i h(w_i)$ → no $w_i w_j$ terms

find $g_k(w)$ so $f(w) \le g_k(w)$, $g_k(w^k) = f(w^k)$

minimize $g_k(w)$ ⇒ $f(w)$ decrease

$$f(w) = \|d - Aw\|_2^2 + \lambda r(w)$$
$$= \|(d - Aw^k) + (Aw^k - Aw)\|_2^2 + \lambda r(w)$$
$$= \underbrace{\|d - Aw^k\|_2^2}_{c_k} + \underbrace{\|A(w^k - w)\|_2^2}_{\le \|A\|_{op}^2 \|w^k - w\|_2^2} + \underbrace{2(d - Aw^k)^T A(w^k - w)}_{v_k^T} + \lambda r(w)$$

step size $0 < \tau < 1/\|A\|_{op}^2$

$\frac{1}{\tau} < \|A\|_{op}^2$

$$f(w) \le g_k(w) = c_k + \frac{1}{\tau}\|w^k - w\|_2^2 + 2v_k^T(w^k - w) + \lambda r(w)$$

if r is separable function then $g_k(w)$ is also

$$g_k(w) = c_k + \sum_{i=1}^{M} q_i(w_i)$$

↳ find $\operatorname{argmin}_w g_k(w) = w^{k+1}$

$$g_k(w) = c_k + \frac{1}{\tau}\|w^k - w\|_2^2 + 2v_k^T(w^k - w) + \lambda r(w)$$

$$\tau g_k(w) = \tau c_k + (w^k - w)^T(w^k - w) + 2\tau v_k^T(w^k - w) + \lambda\tau r(w)$$
$$= \tau c_k - \tau^2 v_k^T v_k + \underbrace{(\tau v_k + w^k - w)^T(\tau v_k + w^k - w)}_{\downarrow z^{(k)}} + \lambda\tau r(w)$$

$$w^{k+1} = \operatorname{argmin}_w \|z^k - w\|_2^2 + \lambda\tau r(w)$$

$$z^k = w^k + \tau v_k$$
$$= w^k + \tau A^T(d - Aw^k)$$
$$= w^k - \tau A^T(Aw^k - d)$$

If $r(w)$ separable ⇒ $r(w) = \sum_{i=1}^{M} h_i(w_i)$ $\quad w^{k+1} = \operatorname{argmin}_{w_1,\dots,w_M} \sum_{i=1}^{M} (z_i^k - w_i)^2 + \lambda\tau h_i(w_i)$ ⇒ M x scalar

Alternate LS Descent & reg.

$w^0 = 0$, $0 < \tau < \frac{1}{\|A\|_{op}^2}$ init

→ $z^k = w^k - \tau A^T(Aw^k - d)$ LS GD

$w^{k+1} = \operatorname{argmin} \|z^k - w\|_2^2 + \lambda\tau r(w)$ reg.

if $\|w^{k+1} - w^k\| < \varepsilon$ stop converge.

On Ridge Regression

$$f(w) = \|d - Aw\|_2^2 + \lambda\|w\|_2^2$$

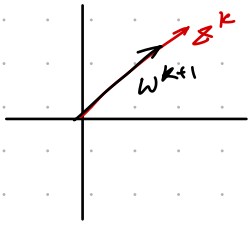
LSGD: $z^k = w^k - \tau A^T(Aw^k - d)$

Regularization:

$$w^{k+1} = \operatorname{argmin}_{w_1,\dots,w_M} \sum (z_i^k - w_i)^2 + \lambda\tau w_i^2$$

$$\Rightarrow w_i^{k+1} = \frac{1}{1+\lambda\tau} z_i^k$$

↳ shrink toward origin



Graph: decrease λ ⇒ less shrinkage

RR: shrinkage toward origin.

LASSO: shrinkage with offset of $\frac{\lambda\tau}{2}$

On LASSO

$f(w) = \|Aw - d\|_2^2 + \lambda\|w\|_1$ → no closed form solution

LSGD: $z^k = w^k - \tau A^T(Aw^k - d)$

Regularization:

$$w^{k+1} = \operatorname{argmin}_w \|z^k - w\|_2^2 + \tau\lambda\|w\|_1$$
$$= \min_{w_1,\dots,w_M} \sum (z_i^k - w_i)^2 + \lambda\tau|w_i|$$

① when $w_i \ge 0$, find derivative = 0

$\min (z_i - w_i)^2 + \lambda\tau w_i$

↳ $\frac{d}{dw_i}\{(z_i - w_i)^2 + \lambda\tau w_i\} = 0$

$2(z_i - w_i) + \lambda\tau = 0$

$w_i = z_i - \frac{\lambda\tau}{2}$, $w_i \ge 0$

$w_i = (z_i - \frac{\lambda\tau}{2})_+$

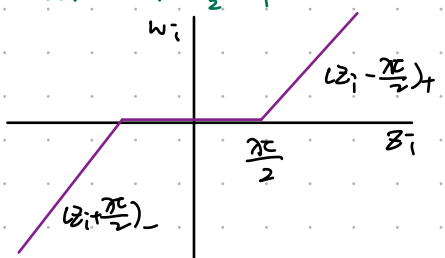
② when $w_i \le 0$

$-2(z_i - w_i) - \lambda\tau = 0$ $\quad w_i \le 0$

$w_i = z_i + \frac{\lambda\tau}{2}$, $w_i \le 0$

$w_i = (z_i + \frac{\lambda\tau}{2})_-$

$$w_i = \begin{cases} 0, & -\frac{\lambda\tau}{2} < z_i < \frac{\lambda\tau}{2} \\ z_i - \frac{\lambda\tau}{2}, & z_i > \frac{\lambda\tau}{2} \\ z_i + \frac{\lambda\tau}{2}, & z_i < -\frac{\lambda\tau}{2} \end{cases}$$



$w_i = (|z_i| - \frac{\lambda\tau}{2})_+ \operatorname{sign}(z_i)$

Sparse Solutions to Least Square Problems Using LASSO

$$Aw = [a_1\ a_2\ \dots\ a_m] \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_m \end{bmatrix} = \sum_{i=1}^{m} w_i a_i$$

suppose $w_i \approx 0 \Rightarrow a_i$ is unimportant

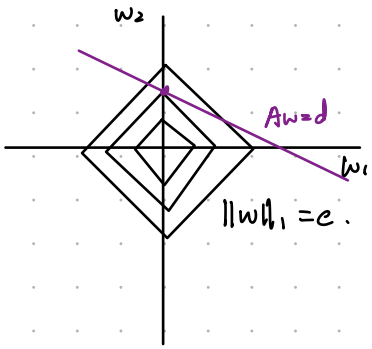
If a small number of w_i are nonzero $\Rightarrow$ w is sparse

l_0 norm : $\|\alpha w\|_0 \neq \alpha \|w\|_0$ (not convex!)

$$\|w_0\| = \sum_{i=1}^{m} \mathbb{1}_{\{w_i \neq 0\}} \quad (\#\text{ of non-zero elements})$$

Least Absolute Selection + Shrinkage Operator is convex

↳ $\min_w \|w\|_1$ s.t. $\|Aw - d\|_2^2 < \varepsilon$



$c = \sum_{i=1}^{m} |w_i| : |w_1| + |w_2| = c$

$\min \|w\|_1$ s.t. $Aw = d$

"corners" on $\|w\|_1$ $\Rightarrow$ sparse solution

if $\|w\|_2^2$ $\Rightarrow$ circular, non sparse

LASSO is a regularized LSP

$$\min_w \|w\|_1 \text{ s.t. } \|Aw - d\|_2^2 < \varepsilon \quad = \quad \min_w \|Aw - d\|_2^2 + \lambda \|w\|_1 \quad \text{for some } \lambda, \varepsilon$$

LASSO	vs	Ridge Regression
sparse w_L		non sparse w_R
can have small model error $w_{opt} - w_L$		can have great prediction error $\|Aw_{opt} - Aw_R\|_2^2$
iterative solution		can be solved in closed form

LASSO in model selection

$w_L = \arg\min_w \|Aw - d\|_2^2 + \lambda \|w\|_1$

$S_L = \{i : [w_L]_i \neq 0\}$ selected features

$$Aw_L = \sum_{i=1}^{m} a_i [w_L]_i = \sum_{i \in S_L} a_i [w_L]_i$$

Debiasing

$A_L = \{a_i : i \in S_L\}$

$\hat{w}_L = \arg\min_w \|A_L w - d\|_2^2$

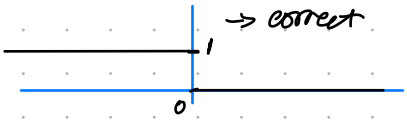
$= (A_L^T A_L)^{-1} A_L^T d$

↳ avoid shrinkage due to $\|w\|_1$

Hinge Loss & SVM

Intuition: squared error loss is sensitive to "easy to classify" data
↳ result in poor classification.

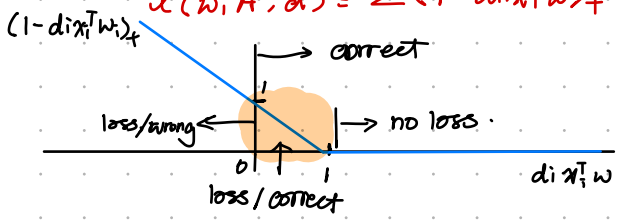
Ideal loss:

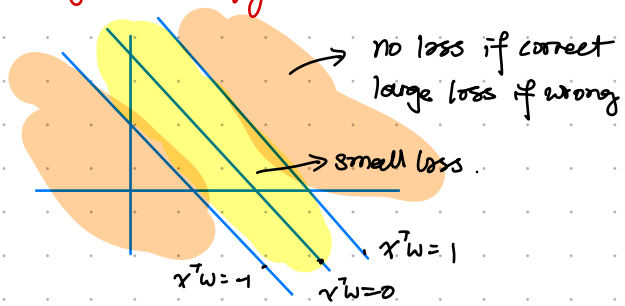


→ nonconvex function. → Hinge loss

Hinge loss is convex and has no loss for easy to classify data.

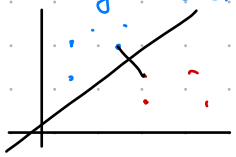
$$\ell(w; A, d) = \sum^{N} (1 - d_i x_i^T w)_+$$





Maximizing margin for separable training data.

margin: distance from boundary to nearest sample



Decision: $\hat{d} = \text{sign}(x^T w)$

$$\hat{d} = \begin{cases} 1 & x^T w + w_0 > 0 \\ -1 & x^T w + w_0 < 0 \end{cases}$$

boundary: $x^T w + w_0 = 0$

Margin determine by $\|w\|_2^{-1}$ ($\because$ distance)

unit normal to boundry plane $v = \frac{\tilde{w}}{\|\tilde{w}\|_2}$

Margin $= m = \frac{1}{2}\|\tilde{x}_1 - \tilde{x}_0\|_2$ $\quad \tilde{x}_1 = \tilde{x}_0 + 2mv$

$$1 = \tilde{x}_1^T \tilde{w} + w_0 = \tilde{x}_0^T \tilde{w} + 2m \frac{\tilde{w}^T}{\|\tilde{w}\|_2} \tilde{w} + w_0$$

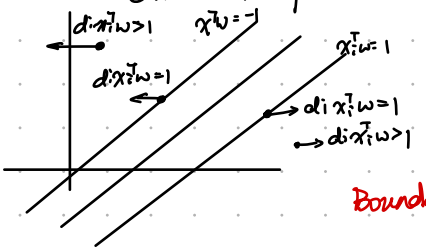
since $\tilde{x}_0^T \tilde{w} + w_0 = -1$

$$2m \frac{\tilde{w}^T}{\|\tilde{w}\|_2} \tilde{w} = 2 \rightarrow 2m\|\tilde{w}\|_2 = 2$$

$$m = \|\tilde{w}\|_2^{-1}$$

Support vector maximize margin

∵ correct classification $d_i(x_i^T \tilde{w} + w_0) \geq 1$



$$\min \|\tilde{w}\|_2^2 \text{ s.t. } d_i(x_i^T \tilde{w} + w_0) \geq 1$$

max margin $\quad$ perfect classification ⇒ there is a unique solution

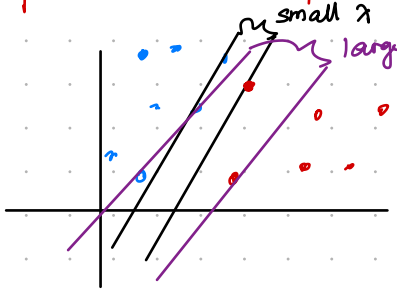
Boundary defined by x_i for which $d_i x_i^T w = 1$ ⇒ Support vectors

SVM for nonseparable data uses hinge loss

$$\min_w \sum^{N} (1 - d_i x_i^T w)_+ + \lambda \|\tilde{w}\|_2^2$$

loss $\quad$ margin norm

Only points on boundary matters.



trade off
↳ accuracy
margin length

(? future classification)
↳ when it is separatable
↳ ideally, no misclassification

Gradient Descent for Support Vector Machines & Subgradients

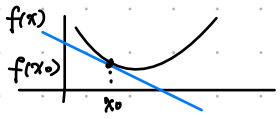
Support Vector Machine need iterative algorithms $\min_w \sum^N (1 - d_i x_i^T w)_+ + \lambda \|w\|_2^2$

↳ no closed form solution, but convex ⇒ gradient descent while hinge loss is not differentiable

Subderivatives generalize derivatives – convex, but not differentiable $f(x)$

Normal Derivative.

$$d(x_0) = \lim_{x \to x_0} \frac{f(x) - f(x_0)}{x - x_0}$$



convex: $f(x) \geq f(x_0) + d(x_0)(x - x_0)$

Subderivative (convex)

Any $d_s(x_0)$: $f(x) \geq f(x_0) + d_s(x_0)(x - x_0)$

⇒ blow $f(x)$

Example: $f(x) = \begin{cases} \pm x^2 & x < 1 \\ 4x & x > 1 \end{cases}$

$$d_s(x) = \begin{cases} x & x < 1 \\ [1,4] & x = 1 \\ 4 & x > 1 \end{cases}$$

- Gradient is generalized by Subgradients
- Gradient Descent optimization: replace gradient with subgradient

Gradient for SVMs

$$l(w) = \sum^N (1 - d_i x_i^T w)_+$$

$$l_i(w) = \begin{cases} 1 - d_i x_i^T w & d_i x_i^T w < 1 \\ 0 & d_i x_i^T w > 1 \end{cases}$$

Subgradient:

$$V_i(w) = \begin{cases} -d_i x_i^T & d_i x_i^T w < 1 \\ 0 & d_i x_i^T w \geq 1 \end{cases}$$

$$= -d_i x_i \left[\{ d_i x_i^T w < 1 \} \right]$$

Cost:

$$f(w) = l(w) + \lambda \|w\|_2^2$$

$$\nabla f(w)|_{w^k} = \sum^N \left(-d_i x_i \left[\{ d_i x_i^T w^k < 1 \} \right] \right) + 2\lambda w^k$$

Gradient Descent:

$$w^{k+1} = w^k - \tau \nabla f(w)|_{w^k}$$

Stochastic Gradient Descent

Stochastic gradient descent use part of the training data instead of all update w

Recall Squared Error

$$l(w) = \sum_{i}^{N} (d_i - x_i^T w)^2$$

$$\nabla_w l(w) = -2 \sum_{i}^{N} (d_i - x_i^T w) x_i$$

Hinge loss

$$l(w) = \sum_{i}^{N} (1 - d_i x_i^T w)_+$$

$$\nabla_w l(w) = -\sum_{i}^{N} I\{d_i x_i^T w < 1\} x_i$$

⇒ depend on all data

SGD $f(w) = \sum_{i}^{N} f_i(w)$ Define i_k, $k = 1, 2, \ldots$

$$w^{k+1} = w^k - \frac{\tau}{2} \nabla_w f_{i_k}(w^k)$$

SGD cycle through the training data

1) Cyclical : (incremental gradient descent)

$i_k = k \mod N$ $\quad i_k = 1, 2, 3, 4, 1, 2, 3, 4 \ldots$

2) Random permutation (reshuffle every N rounds)

$i_k = 2, 4, 1, 3 \mid 2, 1, 4, 3 \mid 4, 3 \ldots$

3) Stochastic gradient descent (uniformly at random) ⇒ most used

Update by $-\frac{\tau}{2} \nabla_w f_{i_k}(w)$ at each iteration

on average give $E\{\nabla_w f_{i_k}(w)\} \approx \frac{\nabla_w f(w)}{N}$

SGD has computational benefits

1) Computing $\nabla_w f_{i_k}(w^k)$ is easier/faster than $\nabla_w f(w^k)$
2) May not be able to store $x_i \ldots i-1 \ldots N$ in memory.
3) Noisy gradient introduces added regularization

On Ridge Regression ↗ $f_i(w)$

$$f(w) = \sum_{i}^{N} (d_i - x_i^T w)^2 + \lambda \|w\|_2^2 = \sum_{i}^{N} \{(d_i - x_i^T w)^2 - \frac{\lambda}{N} \|w\|_2^2\}$$

$$\nabla_w f_i(w) = \nabla_w \left[(d_i - x_i^T w)^2 + \frac{\lambda}{N} w^T w\right]$$

$$= -2(d_i - x_i^T w) x_i + \frac{2\lambda}{N} w$$

$$w^{(k+1)} = w^k - \frac{\tau}{2} \nabla_{w^k} f_{i_k}(w^k)$$

$$= w^k + \tau(d_i - x_i^T w^k) x_{i_k} - \frac{\tau\lambda}{N} w^k$$

vs $w^{k+1} = w^k + \tau A^T(Aw^k - d) - \lambda\tau w^k$

$A : N \times M$

On LASSO ↗ $f_i(w)$

$$f(w) = \sum_{i}^{N} (d_i - x_i^T w)^2 + \lambda \|w\|_1 = \sum_{i}^{N} \{(d_i - x_i^T w)^2 + \frac{\lambda}{N} \|w\|_1\}$$

use subgradient

Consider $\nabla_w \sum^{M} |w_e|$

$$\frac{d}{dw_e} |w_e| = \begin{cases} \text{sign}(w_e) & w_e \neq 0 \\ [-1, 1] \text{ / } 0 & w_e = 0 \end{cases}$$

$\nabla_w \|w\|_1 = \text{sign}(w)$

$$\nabla_w f_i(w) = -2(d_i - x_i^T w) x_i + \frac{\lambda}{N} \text{sign}(w)$$

$$w^{k+1} = w^k + \tau(d_{i_k} - x_i^T w) x_i - \frac{\lambda\tau}{2N} \text{sign}(w^k)$$

↳ for each term of w^k take $\text{sign}(w_i^k) \cdot 1$

Kernel Regression

Let $x = [x_1\ x_2 \dots x_m]^T \in \mathbb{R}^m$ for example: $x = [x_1\ x_2]^T$

$\phi^T(x) = [x_1^2\ x_2^2\ \sqrt{2}x_1x_2\ x_1\ 1]$

consider $d(x) = \phi^T(x)w$, $\phi(x) \in \mathbb{R}^p$ $p > m$

Problem: finding w "training" data (Ridge)

$$\min_w \sum^N (d^i - \phi^T(x^i)w)^2 + \lambda \|w\|_2^2$$

for $d = [d^1\ d^2 \dots d^N]^T$ $\Phi = [\phi(x^1)\ \phi(x^2) \cdots \phi(x^N)]^T$ $N \times p$

$$w = (\Phi^T\Phi + \lambda I)^{-1}\Phi^T d$$

$$d(x) = \phi^T(x)w = \phi^T(x)(\Phi^T\Phi + \lambda I)^{-1}\Phi^T d \quad (p \times p)$$

since $(\Phi^T\Phi + \lambda I)^{-1}\Phi^T = \Phi^T(\Phi\Phi^T + \lambda I)^{-1}$

$$= \phi^T(x)\Phi^T(\Phi\Phi^T + \lambda I)^{-1}d \quad (N \times N)$$

Note: $[\Phi\Phi^T]_{ij} = \phi^T(x^i)\phi(x^j)$, $[\phi^T(x)\Phi^T]_j = \phi^T(x)\phi(x^j)$ $\Big\}$ $K(u,v) = \phi^T(u)\phi(v)$

Let $\alpha = [\alpha_1 \dots \alpha_N]^T$ $d(x) = \sum_{i=1}^N \alpha_i \phi^T(x)\phi(x^i) = \sum_{i=1}^N \alpha_i K(x, x^i)$

$= (\Phi\Phi^T + \lambda I)^{-1}d$

The Kernel to find $d(x)$ without computing $\phi(x)$

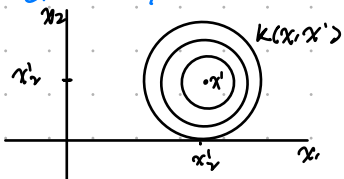
$u^Tv = \|u\|_2 \|v\|_2 \cos\theta$

linear: $k(u,v) = (u^Tv)$

Monomials of degree q: $\phi(x) = x_1^q, x_1^{q-1}x_2 \dots$ $p = \frac{(q+m-1)!}{q!(m-1)!}$ $K(u,v) = (u^Tv)^q$.

polynomials up to degree q: $k(u,v) = (u^Tv + 1)^q$

Gaussian / radical kernel: $k(u,v) = \exp\left\{\frac{-\|u-v\|_2^2}{2\sigma^2}\right\}$



- no explicit $\phi(x)$
- all polynomial orders
- smoothness controlled by σ

Kernel regression considerations

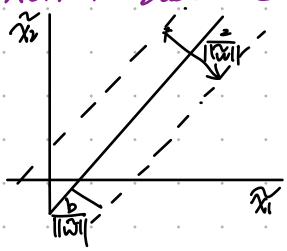
$d(x) = \phi^T(x)w$ vs $d(x) = \sum^N \alpha_i K(x, x^i)$

① store and compute α $(N \times 1)$ vs w $(P \times 1)$

② Binary classification $\text{sign}(d(x))$

③ Avoid "overfitting" with high dim feature space ⇒ use cross validation.

Kernel Based SVM



$\tilde{x}^T\tilde{w}+b=0$

$x^Tw=0$

$w=[\tilde{w}^T\ b]$, depends only on support vectors

$$d(x)=\phi^T(x)w=\sum^N \alpha_j k(x,x^j)$$

Let $\phi(x)=x \Rightarrow k(x,x^j)=x^Tx^j$

$$d(x)=\sum^N \alpha_j x^Tx^j = x^T\sum^N \alpha_j x^j$$

$w=\sum^N \alpha_j x^j \Rightarrow$ all $\alpha_j=0$ except support vectors

High-dimentional feature space

$$\hat{d}(x)=\text{sign}(\phi^T(x)w)$$

Hinge loss with ridge regression

$$\min_w \sum^N (1-d^i\phi^T(x^i)w)_+ + \lambda\|w\|_2^2 \Rightarrow w=\sum^N \phi(x^i)\alpha_j$$

Kernel trick replace $\phi^T(x^i)\phi(x^j)$ with $K(x^i,x^j)$

$$\min_\alpha \sum_i^N (1-d^i\phi^T(x^i)\underbrace{\sum_j^N \alpha_j\phi(x^j)}_{w})_+ + \lambda\sum_i^N \alpha_i\phi^T(x^i)\underbrace{\sum_j^N \alpha_j\phi(x^j)}_{\|w\|_2^2}$$

$$= \min_\alpha \sum_i^N (1-d^i\sum_j^N \alpha_j K(x^i,x^j))_+ + \lambda\sum_i^N\sum_j^N \alpha_i\alpha_j K(x^i,x^j)$$

SVM: $\min_\alpha \sum_i^N (1-d^i\sum_j^N \alpha_j K(x^i,x^j))_+ + \lambda\sum_i^N\sum_j^N \alpha_i\alpha_j K(x^i,x^j)$

- a kernel have maximum value when $x=x^i$ for $K(x,x^i)$
  x^i is usually center of kernel

Support Vector Machine has sparse α

decision boundary: $d(x)=0=\phi^T(x)w=\sum_{j=1}^N \alpha_j K(x,x^j)$

- Boundary only depends on support vectors
- $K(u,v)$ measures the similiarity of u,v
- solve for α use gradient descent.

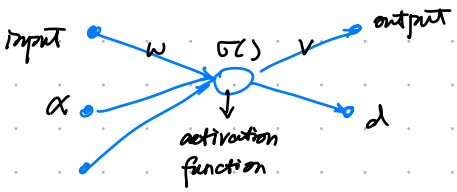
Graphing:

- larger $\lambda \Rightarrow$ smaller kernel norm (larger penalty) $\Rightarrow$ smooth, underestimation
- larger $\sigma \Rightarrow$ wider kernel, more smooth,
  really small $\sigma \Rightarrow$ reply to noise,

↳ if on extreme case width $\Rightarrow 0$, prediction only depends on closet sample.

- when kernel width grow $\Rightarrow$ account for more training samples for prediction

Neural Network



↓ hidden layer

all ● : neuron

$\hat{d} = \sigma(x^T w)$

Common $\sigma(z)$

ReLU: $\sigma(z) = \max(0, z)$

logistic: $\sigma(z) = \frac{1}{1+e^{-z}}$

Sign: $\sigma(z) = \text{sign}(z)$

• multiple outputs ⇒ solve multiple problems

$h_l = \sigma(\sum_j^M w_{lj} x_j)$

$\hat{d}_m = \sigma(\sum_n^P v_{mn} h_n)$

Pending Problems

• Network structure: # of layer, # of hidden nodes

• Choose the weights (Nonconvex??)

↳ SGD and backpropagation

Backpropagation uses SGD to train weights.

• N training samples $(x_1^i, x_2^i \dots x_M^i, d^i, i=1,2\dots N)$

1) Initialize $w_{n,l}$, $v_{q,n}$

2) for $t = 1, 2, 3 \dots$

pick $i_t \in \{1,2,3\dots,N\}$ randomly, compute $h_j^{i_t}$, $\hat{d}_k^{i_t}$,

$v_{k,l}^{t+1} = v_{k,l}^t - \alpha_t \frac{\partial f^{i_t}}{\partial v_{k,l}}$

$w_{m,j}^{t+1} = w_{m,j}^t - \alpha_t \frac{\partial f^{i_t}}{\partial w_{m,j}}$

Chain rule

for $\frac{\partial f^i}{\partial v_{k,l}}$: $\left(\frac{\partial f^i}{\partial \hat{d}_k^i}\right)\frac{\partial \hat{d}_k^i}{\partial v_{k,l}}$ $\quad f(v,w) = \sum^N \pm \sum^Q (\hat{d}_q^i - d_q^i)^2$

$\frac{\partial f^i}{\partial v_{k,l}} = (\hat{d}_k^i - d_k^i)\cdot \sigma'(\sum_n^P h_n^i v_{k,n}) \cdot \frac{\partial}{\partial v_{k,l}}(\sum_n^P h_n^i v_{k,n})$

$= \underbrace{(\hat{d}_k^i - d_k^i)\hat{d}_k^i(1-\hat{d}_k^i)}_{\delta_k^i} h_l^i$

$= \delta_k^i h_l^i$

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

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

$\sigma(\sum_n^P (h_n^i v_{k,n})(1-\sigma(\sum_n^P h_n^i v_{k,n}))$

$= \hat{d}_k^i(1-\hat{d}_k^i)$

layer input

$v_{k,l}^{t+1} = v_{k,l}^t - \alpha_t \delta_k^i h_l^i$

"error" at output

for $\frac{\partial f^i}{\partial w_{mj}} = \sum_q^Q \frac{\partial f^i}{\partial \hat{d}_q^i}\cdot\frac{\partial \hat{d}_q^i}{\partial h_m^i}\cdot\frac{\partial h_m^i}{\partial w_{mj}}$ $\quad \frac{\partial f^i}{\partial \hat{d}_q^i} = (\hat{d}_q^i - d_q^i)$

$= \sum_{q=1}^Q (\hat{d}_q^i - d_q^i)\hat{d}_q^i(1-\hat{d}_q^i) v_{q,m}^i h_m^i (1-h_m^i) x_j^i = \gamma_m^i x_j^i$

$w_{m,j}^{t+1} = w_{m,j}^t - \alpha^t \gamma_m^i x_j^i$

Summary

Initialize: $w_{m,j}^0$, $v_{k,l}^0$

Iterate: for $t = 0, 1, 2 \dots$

SGD: choose $i_t \in \{1,2,\dots N\}$ at random

Forward Net: compute $h_m^{i_t}$, $\hat{d}_q^{i_t}$ from $x_l^{i_t}$, $w_{m,j}^t$, $v_{k,l}^t$

Backward update:

$v_{k,l}^{t+1} = v_{k,l}^t - \alpha_t \delta_k^{i_t} h_l^{i_t}$

$\gamma_m^{i_t} = \sum_q^Q \delta_q^{i_t} v_{q,m}^t h_m^{i_t}(1-h_m^{i_t})$

$w_{m,j}^{t+1} = w_{m,j}^t - \alpha_t \gamma_m^{i_t} x_j^{i_t}$