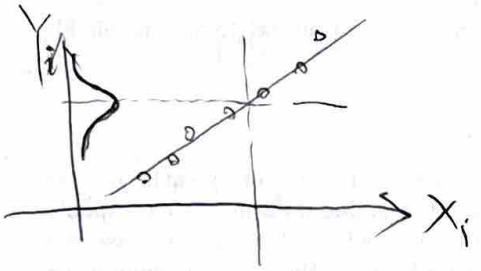
linear regression. $\theta = \begin{cases} w \\ \sigma \end{cases}$

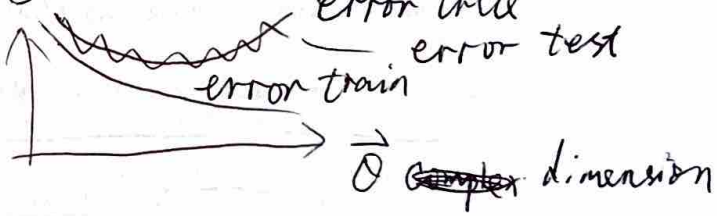
$$P(\vec{Y} \mid \vec{X}, \theta) = \prod_j \frac{1}{\sigma\sqrt{2\pi}} \exp\left\{ -\frac{1}{2\sigma^2} \left[Y_j - \sum_i w_i h_i(X_j) \right]^2 \right\}$$

MLE: $\frac{\partial \ln P(\vec{Y} \mid \vec{X}, \theta)}{\partial \theta} = 0$ $\Rightarrow$ closed form for $\begin{cases} w \\ \sigma \end{cases}$



Study on θ: θ dimension

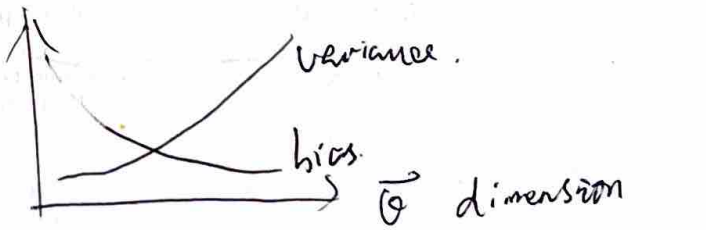
training error ↓ true error ↑



regularization

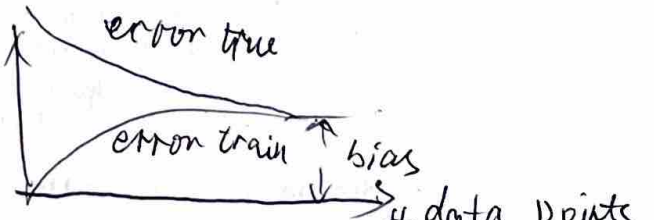
L_1 $\Rightarrow$ sparse. less variance.

L_2 $\Rightarrow$ same variance
more bias



error true = bias + variance.

cross-validation for λ
using validation data
or using training data
with LOOCV.
or
k<-fold



	ridge		LASSO	
	large λ	small λ	large λ	small λ
e_{train}	↑	↓	↑	↓
e_{test}	↑	↓	↑	↑
value of $\hat{w}$	↓	↑	↓	↑
# of nonzeros in $\hat{w}$	(0)	stay put	↓	↑

Consistent: $\lim_{\#data \to \infty}(bias) = 0$

$\Downarrow$

variance can be high

Logistic regression ⇒ classification.

$\theta = w$

$$P(Y=0 \mid \vec{X}, \theta) = \frac{1}{1+\exp(wx+w_0)}$$

$$P(Y=1 \mid \vec{X}, \theta) = \frac{\exp(wx+w_0)}{1+\exp(wx+w_0)}$$

$$\ln P(\vec{Y} \mid \vec{X}; \theta) = \ln \prod P(y^j \mid x^j, \theta) = \sum_j y^j (w_0 + x^j w) - \ln(1+\exp(w_0 + x^j w))$$

MLE: $\frac{\partial \ln P(\vec{Y} \mid \vec{X}, \theta)}{\partial \theta} = 0$. ⇒ no closed form ⇒ gradient descent.

~~Convex~~ Concave.

L_2 regularization

online ⇒ stochastic gradient descent.

adaboost.

$w_0^j = \frac{1}{N}$

$\sum_j w_t^j = 1$

$$w_{t+1}^j = \frac{w_t^j \exp(-\alpha_t y^j h_t(x^j))}{\sum_j w_t^j \exp(-\alpha_t y^j h_t(x^j))} = \frac{w_t^j g^j}{\sum_{j=1}^N w_t^j g_t^j} = \frac{w_t^j g_t^j}{Z_t}$$

$\mathcal{E}(\cdot)$ $g(\cdot)$ $e^{y^j h_t(x^j)}$

$$\alpha_t = \frac{1}{2} \ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$$

$$\varepsilon_t = \sum_{j=1}^N w_t^j \mathbb{1}[h_t(x^j) \neq y^j]$$

$y^j = h(x^j)$

$$\begin{cases} Z_t = \sum_{i=1}^N \prod_{t=1}^T g_i^{(t)} \\ w_j^{(t)} = \frac{\prod_{t=1}^T g_j^{(t)}}{\sum_{i=1}^N \prod_{t=1}^T g_i^{(t)}} \end{cases}$$

closed form!

$$\sum_{j=1}^{N} \mathbb{1}\left[H(x^j) \neq y^j\right]$$

$$\sum_{j=1}^{N} \mathbb{1}\left[\operatorname{sign}\left(\sum_t \alpha_t h_t(x^j)\right) \neq y^j\right]$$

$$\frac{1}{N}\sum_{j=1}^{N} \exp\left[-y^j\left(\sum_m \alpha_m h_m(x^j)\right)\right]$$

(MLE could be.)

$$H(x) = \operatorname{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right) \Rightarrow \text{ensemble} \rightarrow \text{include} \{\text{bagging, boosting, random forest}$$

vote

① train h_t over $D_t \cdot X$

② $Y = h(x)$

③ $D_{t+1}(j) = \dfrac{D_t(j)\exp(-\alpha_t y^j h_t(x^j))}{\sum_{j=1}^{N} D_t(j)\exp(-\alpha_t y^j h_t(x^j))}$ $\left(\sum_j D(j) = 1\right)$

$\left(\alpha_t = \frac{1}{2}\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)\right)$

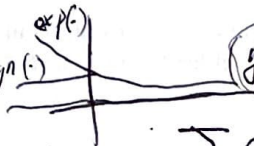
④ go to ①.

$\left(\varepsilon_t = \sum_{j=1}^{N} D_t(j)\,\mathbb{1}(h_t(x^j) \neq y^j)\right)$

remark: $D_0(j) = \frac{1}{N} \quad | \; j = 1 \rightarrow N$

$$\begin{cases} Z(t) = \sum_{j=1}^{N} \prod_{t=1}^{T-1} \exp(\alpha_t y^j h_t(x^j)) \cdot g_j^{(t)} \\ D_j^{(t)} = \dfrac{\prod_{t=1}^{T-1} g_j^{(t)}}{\sum_{j=1}^{N} \prod_{t=1}^{T-1} g_j^{(t)}} \end{cases}$$

mathematical induction



$$\text{training error} \quad \frac{1}{N}\sum_{j=1}^{N} \mathbb{1}[H(x^j) \neq y^j] \leq \frac{1}{N}\sum_{j=1}^{N} \exp\left(-y^j \sum_t \alpha_t h_t(x^j)\right) = \prod_t Z(t)$$

$$\leq \prod_t 2\sqrt{\varepsilon_t(1-\varepsilon_t)} \leq \exp\left(-2\left(\sum_t \gamma_t^2\right)\right) \quad \left(\varepsilon_t = \frac{1}{2} - \gamma_t\right) \quad ②$$

log

There are ways to explain Boosting using KL divergence
- statistical

Decision trees

$$IG(X) = H(Y) - H(Y|X)$$

- $H(Y)$: before branching
- $H(Y|X)$: after branching
- feature selected

overfit
⇓ variance! (no bias)
⇓ pruning

$$\sum_i -P(X_i)\log(P(X_i))$$

$$\sum_i \frac{|S(X_i)|}{|S|} \sum_j -P(Y_j|X_i)\log(P(Y_j|X_i))$$

(Not MLE)

Non-parametric models / instance-based learning

(nearest neighbors) ⇒ curse of dimensionality (must store and retrieve all data)

1-NN ~ distance metrics $D(x, x^*) = \sqrt{(x-x^*)^T \Sigma (x-x^*)}$

$$\Sigma = \begin{bmatrix} \sigma_1^2 & & 0 \\ & \ddots & \\ 0 & & \sigma_N^2 \end{bmatrix}$$

k-NN ~ classification ⇒ majority vote

Weighted k-NN: $\hat{y} = \frac{\sum_i^k \pi_i y_i}{\sum_i^k \pi_i}$

Kernel: $\pi(x^i, x^*) = \exp\left\{\frac{-D^2(x^i, x^*)}{\rho^2}\right\}$

get ρ with gradient descent

locally weighted regression: $w = ((\pi X)^T \pi X)^{-1}(\pi X)^T \pi y$

~~ApplicationController < ActionController::Base~~

- ~~returning User.~~

- ~~Compute Cn~~ ~~1250 / 10 = 125~~

(Not MLE)

online learning ⇒ **perceptron** (report averaged weights at the end).

(classification) $\hat{y} = \text{sign}(w, x)$. ~~1250 × 3 / 10 = 375 = 37.5~~

if $\hat{y} = y$ $w^{t+1} = w^t$

otherwise $w^{t+1} = w^t + \Delta$ ⇒ ? Cin perceptron: $\Delta = y^i x^i$ $= \frac{\nabla l(w)}{\nabla w}$.

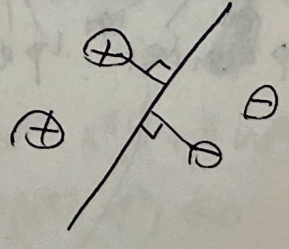
linearly separable case: $y = +1$ or -1

$[(x^1, y^1) \dots (x^T, y^T)]$

$\begin{cases} y^i = +1 & w^* x^i \geq \gamma \\ y^i = -1 & w^* x^i \leq -\gamma \end{cases}$ ⇒ $\forall t \quad y^t (w^* x^t) \geq \gamma$

$$l = \begin{cases} 0 & \text{if } y(w \cdot x) > 0 \\ -y(w \cdot x) & \text{if } yw \cdot x \leq 0 \end{cases} \quad \text{hinge loss}$$

$$\frac{\partial l}{\partial w} = \mathbb{1}(y(w \cdot x) \leq 0)(-y \cdot x)$$



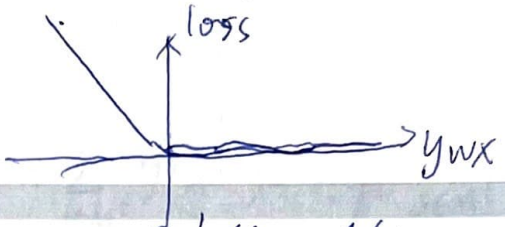
Batch hinge minimization ~~algorithm~~:

$$\Delta = \eta \frac{1}{N} \sum_{i=1}^{N} \{ \mathbb{1}[y^{(i)}(w^{(t)} \cdot x^{(i)}) \leq 0] \, y^{(i)} x^{(i)} \}$$

perceptron is the hinge loss minimization using SGD and $\eta = 1$.

make a mistake when $ywx \le 0$

$$\text{loss} = \begin{cases} 0 & \text{if } ywx > 0 \\ -ywx & \text{if } ywx < 0 \end{cases}$$



$$\frac{\partial \text{loss}}{\partial w} = \mathbb{1}(ywx \le 0)(-yx)$$

(hinge loss minimization using SGD) ($\eta = 1$).

① perceptron: (not MLE).

if $\hat{y} = y^t$ $w^{t+1} \leftarrow w^t$

else $w^{t+1} \leftarrow w^t + y^t x^t$ (mistake when $y^t w^t x^t \le 0$).

online learning: report average $\hat{w} = \frac{1}{T+1} \sum_{t=0}^{T} w^t$

(1) linearly separable ($\|w^*\| \triangleq 1$).

$$\left.\begin{array}{l} \forall t \text{ if } y^t = +1 \quad w^* x^t \ge \gamma \\ \quad\; y^t = -1 \quad w^* x^t \le -\gamma \end{array}\right\} \Rightarrow \forall t \; y^t w^* x^t \ge \gamma$$

mistake at time t: $w^{(t+1)} = w^{(t)} + y^{(t)} x^{(t)}$.

△ taking dot product with w^*: $w^* w^{(t+1)} = w^* w^{(t)} + y^{(t)} w^* x^t$

$\therefore w^* w^{t+1} - w^* w^t \ge \gamma$

$\therefore w^* w^{t+1} - w^* w^0 \ge m\gamma$ (after m mistakes) $(m \le t+1)$.

$\therefore w^* w^{t+1} \ge m\gamma$

△ taking square: $\|w^{(t+1)}\|^2 = \|w^{(t)}\|^2 + 2y^{(t)}(w^{(t)} \cdot x^{(t)}) + \|w^{(t)}\|^2$.

$\because y^{(t)}(w^{(t)} x^{(t)})$ is mistaken, $\therefore 2y^{(t)}(x^{(t)} \cdot x^{(t)}) \le 0$.

$\because \|w^{(t)}\|^2 \le R^2$ $\therefore \|w^{t+1}\|^2 - \|w^t\|^2 \le R^2$ $\therefore \|w^{t+1}\|^2 - \underbrace{\|w^0\|^2}_{=0} \le mR^2$.

△ putting things together

$m\gamma \le w^* w^{t+1} \le \|w^*\| \|w^{t+1}\| \le \sqrt{m} R$ $\therefore m \le \left(\frac{R}{\gamma}\right)^2$

(2). not linearly separable.

Let $M^{(t)}$ be time steps up to t when mistakes are made:

$$w^{(t)} = \sum_{j \in M^{(t)}} y^j x^j \quad \left(\sum \frac{\partial loss}{\partial w}\right)$$

$(1 \times k)(k \times 1)$

$$\therefore \hat{y} = sign(w^t \cdot x) = sign\left(x \cdot \sum_{j \in M^t} y^j x^j\right) = sign\left(\sum_{j \in M^t} y^j x \cdot x^j\right)$$

$$\therefore \hat{y} = sign\left(\sum_{j \in M^t} y^j \phi(x) \cdot \phi(x^j)\right) \text{ when using high dimensional features}$$

polynomials of degree exactly d: $\phi(u) \cdot \phi(v) = (\vec{u} \cdot \vec{v})^d$

polynomials of degree up to d: $\phi(u) \cdot \phi(v) = (\vec{u} \cdot \vec{v} + 1)^d$.

Gaussian kernel: $\phi(u)\,\phi(v) = \exp\left(-\frac{\|\vec{u} - \vec{v}\|^2}{2\sigma^2}\right)$

Sigmoid: $\phi(u) \cdot \phi(v) = \tanh(\eta \vec{u} \cdot \vec{v} + \theta)$

(3) Hinge regularization $\Rightarrow$ SVM

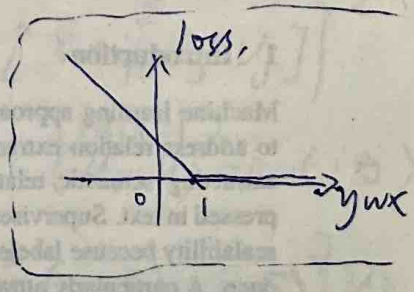
$$\Delta \begin{cases} w x^+ = 1 \\ w x^- = -1 \\ x^+ = x^- + 2\gamma \frac{w}{\|w\|} \end{cases} \xrightarrow{\text{convention}} \Rightarrow \gamma = \frac{1}{\|w\|} \Rightarrow$$

maximizing margin γ

$\Updownarrow$

minimizing $\|w\|$

$$\Delta\ loss = \begin{cases} 0 & \text{if } y w x \geq 1 \\ 1 - ywx & \text{if } ywx < 1. \end{cases}$$



Δ trade-off margin violation with $\|w\|$ $\Rightarrow$

$$loss = \|w\|^2 + C \sum_{j=1}^{N} (1 - y^j w x^j)$$

SGD for SVM: $\frac{\partial loss}{\partial w} = +2w^t + C\,\mathbb{1}(y w^t x \leq 1)(-y x)$.

(4) neural network: SGD, forward propagation (predicting)

backward propagation (learning)

clustering. ⑧.

(1) hard: k-means ⇒ Coordinate descent ⇒ MLE !!

{ Classify:
  recenter:

(2) soft: EM (density estimation). ⇒ MLE !!

Coordinate descent; local optima

$\theta = \arg\max_\theta$ [log-likelihood] $= \ell(\theta) = \log \prod_{j=1}^{m} p(x_j|\theta)$

$= \sum_{j=1}^{m} \log \sum_z p(x_j, z|\theta) = \sum_{j=1}^{m} \log \sum_z p(z|x_j,\theta)\, p(x_j|\theta)$

$= \sum_{j=1}^{m} \log \sum_z p(z|x_j,\hat\theta) \frac{p(x_j, z|\theta)}{p(z|x_j,\hat\theta)}$

$\geq \sum_{j=1}^{m} \sum_{z=1}^{k} p(z|x_j,\hat\theta) \log \frac{p(x_j, z|\theta)}{p(z|x_j,\hat\theta)}$ ⇒ KL-divergence

{ M step: fix $p(z|x_j,\theta)$, max θ.
  E step: compute $p(z|x_j,\theta)$ using θ. Using Bayesian rule.

(3) PCA using SVD to scale up.

$\text{error} = \sum_{i=1}^{N} (x^i - \hat{x}^i)^2 = \sum_{i=1}^{N} \left[\left(\bar{x} + \sum_{j=1}^{d} z_j^i u_j \right) - \left[\bar{x} + \sum_{j=1}^{k} z_j^i u_j \right] \right]^2$

$= \sum_{i=1}^{N} \left[\sum_{j=k+1}^{d} z_j^i u_j \right]^2 = \sum_{i=1}^{N} \sum_{j=k+1}^{d} (z_j^i)^2$ $\quad (\|u_j\| = 1)$ $\quad (u_i \cdot u_j = 0 \text{ if } i \neq j)$

$= \sum_{i=1}^{N} \sum_{j=k+1}^{d} \left[u_j (x^i - \bar{x}) \right]^2 = \sum_{j=k+1}^{d} u_j^T \left[\sum_{i=1}^{N} (x^i - \bar{x})(x^i - \bar{x}) \right] u_j$

$= N \sum_{j=k+1}^{d} u_j^T \begin{bmatrix} \sigma_1^2 & \sigma_{12} & \cdots \\ \sigma_{21} & \sigma_2^2 & \ddots \\ \vdots & & \end{bmatrix} u_j$

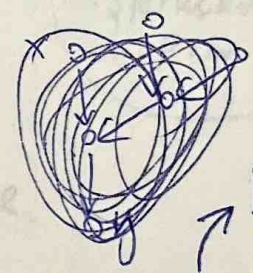
where $\sigma_{rs} = \frac{1}{N} \sum_{i=1}^{N} (x_r^i - \bar{x}_r)(x_s^i - \bar{x}_s)$.

minimize reconstruction error equals to
picking orthonormal basis with smallest eigen values.

1. recenter $X_c \leftarrow X - \bar{X}$
2. covar $\Sigma \leftarrow \frac{1}{N} X_c^T X_c$
3. eigen vectors & values of Σ.
4. principal components: k eigen vectors with highest eigen values

SVD: $X = W S V^T$

weight / Coordinate — S — eigenvectors.



④ Bayesian Net (MLE!) (Conditional probability table : CPT)

e.g.

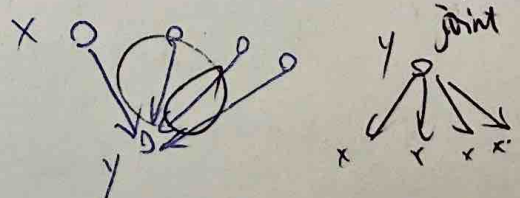
X_1	X_2	....	Y
0	0	....	1
0	1	....	0
1	0	....	0
1	1	....	4

m classes
k binary features
then size of CPT = 2^k. (row).
need $\sum_1^m (2^k - 1)$ params. (every node counts).

e.g. Naive bayes: $\sum_k m(2^1 - 1) = km$ params.

$$y = \arg\max_y P(y) \prod_{i=1}^{k} P(x_i|y). \quad \text{(MLE!)}$$

joint likelihood data likelihood.



e.g. edgeless graph: least parameters ⇒ high bias.
e.g. fully-connected graph: most parameters ⇒ high variance.

Learning the structure of a Bayesian Net. ⇒ MLE!

structure + parameters ($P(X \mid Parents(X))$)

($I(X_i, X_j) = \sum_{x_i, x_j} \hat{P}(x_i, x_j) \log \frac{\hat{P}(x_i, x_j)}{\hat{P}(x_i)\hat{P}(x_j)}$ where $\hat{P}(x_i, x_j) = \frac{count(x_i, x_j)}{m}$)

1. compute weight $I(X_i, X_j)$ of each possible edge (X_i, X_j)
2. find a maximum weight spanning tree (MST): greatest total weight.

$$\max \sum_{(X_i, X_j) \in E} I(X_i, X_j)$$

⇓ greedy algo: {Kruskal, Prim

3. Give directions to edges in MST
   (pick any node as root, do Breadth-first-search)

NP-hard to learn structures with #parents > 1

(start from Chow-Liu tree,
local search: add / delete / invert edge.
score .....).

(1) PAC bound.

$$P(error_{true}(h) > \varepsilon) \le |H| e^{-N\varepsilon} \le \delta$$

H: hypothesis space.

N: i.i.d samples.

h: any learned hypothesis that is consistent on the training data.

ε: $0 < \varepsilon < 1$.

union bound $\le P(\exists h: error_{train}(h) = 0 \ \& \ error_{true}(h) \ge \varepsilon) \le k(1-\varepsilon)^N$

(k hypothesis). $\le |H|(1-\varepsilon)^N \le |H| e^{-N\varepsilon}$.

total # of hypothesis

$$\begin{cases} N \ge \dfrac{\ln|H| + \ln\frac{1}{\delta}}{\varepsilon} \\ \varepsilon \ge \dfrac{\ln|H| + \ln\frac{1}{\delta}}{N} \Rightarrow O(\frac{1}{N}) \end{cases}$$

(more generally, decrease as $O(\frac{1}{\sqrt{N}})$.

|H| large: bias low, variance large

|H| small: bias high, variance low

$\ln|H|$: VC-dimension

<list $x_1 \cdots x_m \sim y$ table, compute number of poss. of y>

boolean formula with m binary features:

$$\ln|H| = 2^m \ln 2.$$

decision tree of depth k: $\ln|H| \le 2^k \log m$.

decision tree with up to k leaves: $\ln|H| \le k \ln m + 2k \ln k$

(2) Chernoff bound:

$$P\left(\theta - \frac{1}{N}\sum_{j=1}^{N} x^j > \varepsilon\right) \le e^{-2N\varepsilon^2}.$$

$\theta = error_{true}(h)$. $\quad \hat{\theta} = error_{train}(h) \Leftarrow \frac{1}{N}\sum_{j=1}^{N} \mathbb{1}\, h(x^j) \ne y^j$.

$$= \int_x p(x)\, \mathbb{1}(h(x) \ne t(x))\, dx$$

(3) general:

$$P(error_{true}(h_i) - error_{train}(h_i) > \varepsilon) \le e^{-2N\varepsilon^2}.$$

$$\forall h: P(error_{true}(h) - error_{train}(h) > \varepsilon) \le |H| e^{-2N\varepsilon^2}.$$

$$\Rightarrow N \ge \frac{\ln|H| + \ln\frac{1}{\delta}}{2\varepsilon^2} \qquad \Rightarrow \varepsilon \ge \sqrt{\frac{\ln|H| + \ln\frac{1}{\delta}}{2N}} \Rightarrow O(\frac{1}{\sqrt{N}}).$$

~~Neural Networks~~ MDPs.

X = state
V: value, R: reward.
π: policy = a: action

reinforcement learning: training by feedback.

$$V_\pi(x_0) = E_\pi\left[\sum_{t=0}^{\infty} \gamma^t R(x_t)\right] = E_\pi\left[R(x_0) + \gamma \sum_{t=1}^{\infty} \gamma^{t-1} R(x_t)\right]$$

$$= R(x_0) + \gamma \sum_{x_1} P(x_1 | x_0, a = \pi(x_0))\, V_\pi(x_1).$$

stop when $\|V_{t+1} - V_t\|_\infty < \varepsilon$. for computing the value of a policy.

compute the optimal value V: Bellman equation

$$V^*(x) = \max_a \left\{R(x,a) + \gamma \sum_{x'} P(x' | x,a)\, V^*(x')\right\}$$

value iteration on Bellman equation to solve for $V^*(x) \Rightarrow$ then we get $\pi^*(x)$
(dynamic programming).

$$\pi^*(x) = \arg\max_a \left\{R(x,a) + \gamma \sum_{x'} P(x' | x,a)\, V^*(x')\right\}$$

- start with some guess V^0 (for all states).
- iteratively: $V^{t+1}(x) \leftarrow \max_a \left\{R(x,a) + \gamma \sum_{x'} P(x' | x,a)\, V^t(x')\right\}$.
- stop when $\|V^{t+1} - V^t\| < \varepsilon$.

given or guessed.

reinforcement learning: $R(x,a)$ rewards & $P(x'|x,a)$ trans. Prob. are unknown

goal: learn policy that (approx.) maximizes $E_\pi\left[\sum_{t=0}^{\infty} \gamma^t R(x_t)\right]$

interact with the world at each time step t.

- exploitation: stick with the known world.
- exploration: at the risk of getting no big reward.

Rmax algorithm: initialize. $R(x_0, a) = R_{max}$, $P(x_0 | x_0, a) = 1$. at each T, either exploit or explore

connectivity models:

hierarchical clustering builds models based on distance connectivity

centroid models:

K-means represents each cluster by a single mean vector

distribution models:

clusters are modeled using statistical distributions such as Gaussians E-M

---

no "correct" clustering.

K-NN centroid:

find the k cluster centers and assign the objects to the nearest cluster center, such that the squared distances from the cluster are minimized.

x_i $\vec{X}$ k.

$$\underset{S}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{j=1}^{K \cdot N_i} \left\| x_{ij} - C_i \right\|^2$$

NP-hard.

approximate heuristics $\Rightarrow$ local optimum.

problem: partition n observations into k clusters

$$\underset{S}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{\vec{x}_i \in S_i} \left\| \vec{x}_i - \vec{C}_i \right\|^2$$

$$\vec{C}_i = \frac{1}{|S_i|} \sum_{\vec{x}_i \in S_i} \vec{x}_i$$

each $\vec{x}_i$ is assigned to exactly one S_i.

expectation assignment step: (Euclidean distance)

$$S_i^{(t)} = \{ \vec{x}_i : \left\| \vec{x}_i - \vec{C}_i^{(t)} \right\|^2 \le \left\| \vec{x}_i - \vec{C}_\ell^{(t)} \right\| \quad \forall \ell : 1 \le \ell \le k. \}$$

maximization update step:

$$\vec{C}_i^{(t+1)} = \frac{1}{|S_i^{(t)}|} \sum_{\vec{x}_i \in S_i^{(t)}} \vec{x}_i$$

guarantee.

convergence: the assignments no longer change.

hard part:
- initialization
- the choice of k.

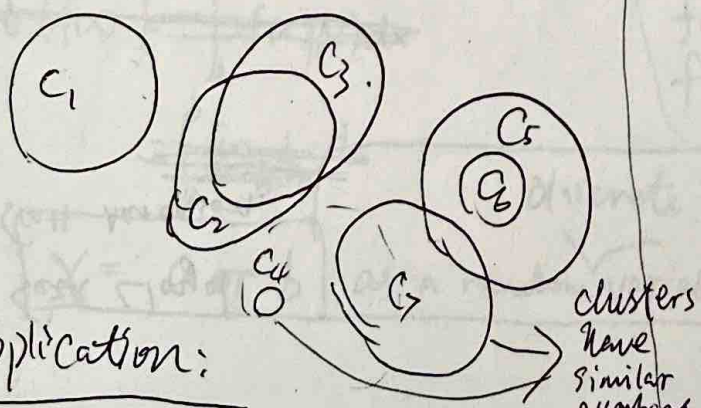
working scenarios (limitations):

△ clusters are expected to be of similar size.

△ spherical clusters separable in a way such that MEAE converges to the cluster center

△ can't find non-convex clusters

failed example:



application:

Vector quantization ⇒
Color quantization ⇒ color palette.

feature learning ⇒
dist($\vec{X}$, $\vec{C}$) as feature
⇒ named entity recognition

distribution models:

clusters defined as objects belonging most likely to the same distribution

overfitting
⇓
Choosing the appropriate model → difficult

capture correlation and dependence between attributes

Naive Bayes classifier

target $f: X \to Y$, $X = X(a_1 \cdots a_n)$.

$$y_{map} = \underset{y_j \in Y}{\operatorname{argmax}} P(y_j \mid a_1 \cdots a_n)$$

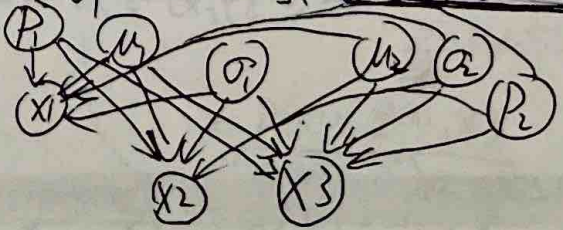
$$= \underset{y_j \in Y}{\operatorname{argmax}} \frac{P(a_1 \cdots a_n \mid y_j) P(y_j)}{P(a_1 \cdots a_n)}$$

$$= \underset{y_j \in Y}{\operatorname{argmax}} P(a_1 \cdots a_n \mid y_j) P(y_j).$$

$$= \underset{y_j \in Y}{\operatorname{argmax}} P(y_j) \prod_i P(a_i \mid y_j)$$

(Naive Bayes assumption: a_i conditionally independent on one another)

Mixture of Gaussians: (Bayes nets)



Jensen's inequality

f: convex: $f''(x) \geq 0$ for all $x \in \mathbb{R}$

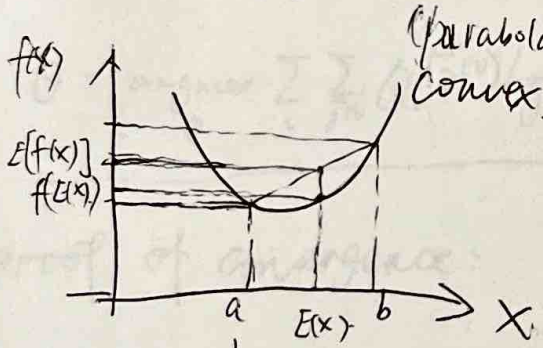
$f(\vec{x})$: hessian $H \geq 0$ semi-definite

$$\boxed{E[f(x)] \geq f(E(x))}$$

if f is strictly convex

then $E[f(x)] = f(E(x))$ iff $P(X = E(X)) = 1$.

(x = constant).



~~$E[f(x)] = \int_a^b f(x) p(x) dx$~~

$= \int_a^b f(x) \frac{1}{b-a} dx$

X = a or b as a discrete random variable

---

training data: $\vec{X} \in \mathbb{R}^{(n)}$. $i = 1, \ldots, m$

hidden/latent variables: $\vec{z}$ (~~discrete~~ cluster assignment for each i).

known variables: $\vec{\theta}$. (~~cluster assignment~~ distribution param.)

$$l(\vec{\theta}) = \sum_{i=1}^{m} \log P(\vec{x}^{(i)}; \vec{\theta})$$

$$= \sum_{i=1}^{m} \log \sum_{\vec{z}} P(\vec{x}^{(i)}, \vec{z}; \vec{\theta})$$

goal: finding MLE of $\vec{\theta}$ (~~gmm~~ cluster assignment)

$\vec{\theta}$ = discrete

$$\vec{\theta} = \arg\max_{\theta} l(\vec{\theta}).$$

$$l(\vec{\theta}) = \sum_{i=1}^{m} \log P(x^{(i)}; \vec{\theta}).$$

$$= \sum_{i=1}^{m} \log \sum_{\vec{z}^{(i)}} P(x^{(i)}, \vec{z}^{(i)}; \vec{\theta})$$

$$= \sum_{i=1}^{m} \log \sum_{\vec{z}^{(i)}} Q_i(\vec{z}^{(i)}) \frac{P(x^{(i)}, \vec{z}^{(i)}; \vec{\theta})}{Q_i(\vec{z}^{(i)})}$$

where $Q_i(\vec{z}^{(i)})$ is some distribution over the $\vec{z}^{(i)}$'s.

($\sum_{\vec{z}^{(i)}} Q_i(\vec{z}^{(i)}) = 1$).

$$= \sum_{i=1}^{m} f\left[E(\vec{z}^{(i)})\right]$$

f is strictly concave $\geq$

$$\sum_{i=1}^{m} E[f(\vec{z}^{(i)})]$$

$f(x) = \log(x)$

$f''(x) = -\frac{1}{x^2} < 0$

$$= \sum_{i=1}^{m} \sum_{\vec{z}^{(i)}} Q_i(\vec{z}^{(i)}) \log \frac{P(x^{(i)}, \vec{z}^{(i)}; \vec{\theta})}{Q_i(\vec{z}^{(i)})}$$

(lower bound on $l(\vec{\theta})$

for = to hold. we have.

$$\frac{P(x^{(i)}, \vec{z}^{(i)}; \vec{\theta})}{Q_i(\vec{z}^{(i)})} = \text{constant}.$$

$\therefore Q_i(\vec{z}^{(i)}) \propto P(x^{(i)}, \vec{z}^{(i)}; \vec{\theta})$

$\because \sum_{\vec{z}^{(i)}} Q_i(\vec{z}^{(i)}) = 1$

$\therefore Q_i(\vec{z}^{(i)}) = \frac{P(x^{(i)}, \vec{z}^{(i)}; \vec{\theta})}{\sum_{\vec{z}} P(x^{(i)}, \vec{z}; \vec{\theta})}$

$= \frac{P(x^{(i)}, \vec{z}^{(i)}; \vec{\theta})}{P(x^{(i)}; \vec{\theta})}$

$= P(\vec{z}^{(i)} | x^{(i)}; \vec{\theta})$

rewrite:

$$l(\vec{\theta}) \geq \sum_{i=1}^{m} \sum_{\vec{z}^{(i)}} Q_i(\vec{z}^{(i)}) \log \frac{P(\vec{x}^{(i)}, \vec{z}^{(i)}; \vec{\theta})}{Q_i(\vec{z}^{(i)})}$$

lower bound: $Q_i(\vec{z}^{(i)}) = P(\vec{z}^{(i)} | \vec{x}^{(i)}; \vec{\theta})$

$\therefore l(\vec{\theta})$ lower bounded by

$$\sum_{i=1}^{m} \sum_{\vec{z}^{(i)}} P(\vec{z}^{(i)} | \vec{x}^{(i)}; \vec{\theta}) \log P(\vec{x}^{(i)}; \vec{\theta})$$

---

maximization:

$$\theta = \arg\max_{\theta} \sum_{i} \sum_{\vec{z}^{(i)}} Q_i(\vec{z}^{(i)}) \log \frac{P(\vec{x}^{(i)}, \vec{z}^{(i)}; \vec{\theta})}{Q_i(\vec{z}^{(i)})}$$

---

proof of convergence:

$$l(\theta^{(t+1)}) \geq \sum_{i} \sum_{\vec{z}^{(i)}} Q_i^{(t)}(\vec{z}^{(i)}) \log \frac{P(\vec{x}^{(i)}, \vec{z}^{(i)}; \vec{\theta}^{(t+1)})}{Q_i^{(t)}(\vec{z}^{(i)})}$$

$\forall Q_i, \theta$ (because of Jensen's inequality)

$$\geq \sum_{i} \sum_{\vec{z}^{(i)}} Q_i^{(t)}(\vec{z}^{(i)}) \log \frac{P(\vec{x}^{(i)}, \vec{z}^{(i)}; \vec{\theta}^{(t)})}{Q_i^{(t)}(\vec{z}^{(i)})}$$

(because of the maximization step)

$$= l(\theta^{(t)})$$

$\therefore l(\theta)$ converges monotonically under EM

$z = 0, 1$ in MoG example.

in Mixture of Gaussians

~~in M-step, we solve for~~

~~ϕ, μ, Σ, i, e~~

~~$\vec{\theta} = (\phi, \mu, \Sigma)$~~

k-clusters

in E-step, we calculate

$$Q_i(z^{(i)} = j) = P(z^{(i)} = j | \vec{x}^{(i)}; \vec{\theta})$$

$$= \frac{P(z^{(i)} = j; \vec{x}^{(i)}, \vec{\theta})}{P(\vec{x}^{(i)}, \vec{\theta})}$$

$$= \frac{P(\vec{x}^{(i)}, \vec{\theta} | z^{(i)} = j) P(z^{(i)} = j)}{\sum_{j=1}^{k} P(\vec{x}^{(i)}, \vec{\theta} | z^{(i)} = j) P(z^{(i)} = j)}$$

$$\triangleq w_j^{(i)}$$

---

in M-step, we solve for

$\vec{\theta} = (\phi, \vec{\mu}, \Sigma)$

rewrite: lower bound.

$$l(\vec{\theta}) =$$

$$\sum_{i=1}^{m} \sum_{z^{(i)}} Q_i(z^{(i)}) \log \frac{P(\vec{x}^{(i)}, z^{(i)}; \vec{\theta})}{Q_i(z^{(i)})}$$

$$= \sum_{i=1}^{m} \sum_{j=1}^{k} Q_i(z^{(i)} = j) \log \frac{P(\vec{x}^{(i)} | z^{(i)}; \vec{\theta}) P(z^{(i)}; \vec{\theta})}{Q_i(z^{(i)} = j)}$$

$$= \sum_{i=1}^{m} \sum_{j=1}^{k} w_j^{(i)} \log \frac{\frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma_j|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(\vec{x}^{(i)} - \vec{\mu}_j)^T \Sigma_j^{-1} (\vec{x}^{(i)} - \vec{\mu}_j)\right] \phi_j}{w_j^{(i)}}$$

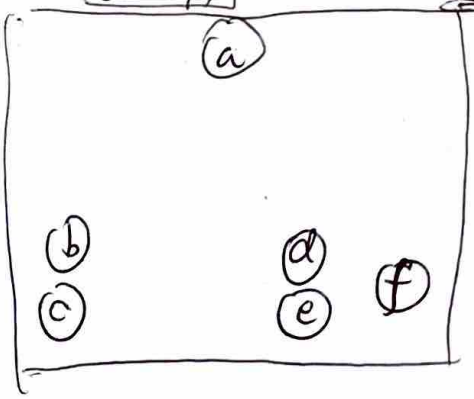
$$\begin{cases} \mu_j = \\ \phi_j = \\ \Sigma_j = \end{cases}$$

agglomerative: bottom-up. tree divisive: top-down.

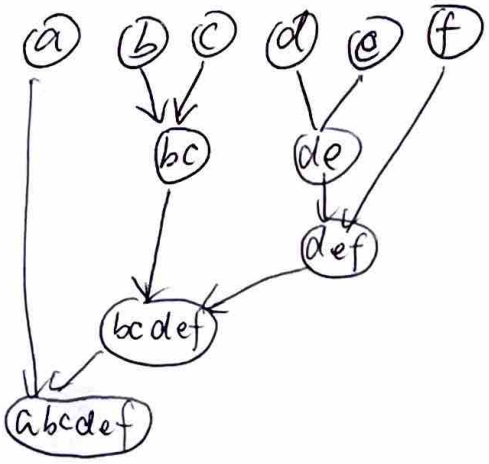
$O(n^3) - O(n^2)$

merge by distance of two clusters:

- $\|a-b\|_2 = \sqrt{\sum_i (a_i - b_i)^2}$
- $\|a-b\|_1 = \sum_i |a_i - b_i|$
- $\|a-b\|_\infty = \max_i |a_i - b_i|$
- non-numeric: hamming distance
- K-L divergence (e.g. decision tree)
- entropy



tree.



$O(2^n)$.

- min-cut.
- K-means ..
- etc.

evaluating clustering results