Lecture 9 - scribbles

Tuesday, October 3, 2017
14:27

today: kernel trick
unsupervised learning: K-means
GMM & EM

Kernel trick

main idea: trick to do non-linear methods using linear techniques

motivation/background: recall for least-square regression, gradient: $\sum_i x_i \overbrace{(y_i - w^T x_i)}^{\text{scalar}} = X^T \tilde{\alpha}_{LS}$

" logistic regression, " $\sum_i x_i (y_i - \sigma(w^T x_i)) = X^T \tilde{\alpha}_{LR}$

$\Rightarrow$ during gradient descent, $w_t = \sum_{i=1}^{n} \underbrace{\alpha_i^t}_{\text{scalar}} x_i$

also, for ridge regression solution: $w^* = (X^T X + \lambda I_d)^{-1} X^T y$

$= X^T (X X^T + \lambda I_n)^{-1} y$

$= X^T \tilde{\alpha}_{LS}^*$

Lemma:
$\forall X$ $n \times d$
$(X^T X + \lambda I_d)^{-1} X^T = X^T (X X^T + \lambda I_n)^{-1}$
(could prove from "matrix inversion lemma")

$(X X^T)_{ij} = \langle x_i, x_j \rangle = k(x_i, x_j)$
"Gram matrix" / "kernel matrix"
$\rightarrow$ "kernelized least squares"

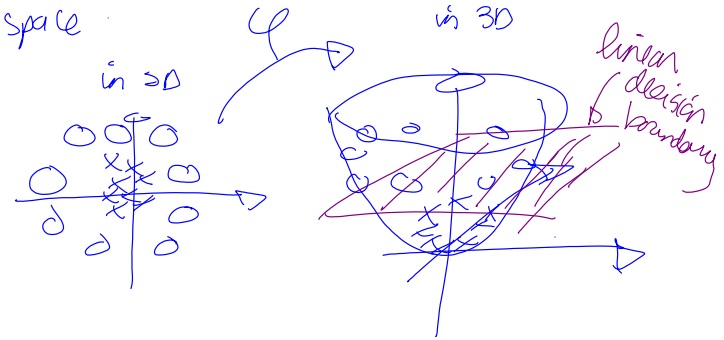
* for prediction, $w^T x = \sum_{i=1}^{n} \alpha_i (x_i^T x)$
$\langle x_i, x \rangle$ is all we need

suppose have mapping $X \xrightarrow{\varphi} \varphi(X)$
"feature space embedding"

$\tilde{w}^T \varphi(x) = \sum_{i=1}^{n} \alpha_i \langle \varphi(x_i), \varphi(x) \rangle$

$i=1$ — $K(x_i, x)$ "kernel"

"kernel trick": you can implicitly work in high dimensional space only using $K(\cdot,\cdot)$ evaluation



for example: from 2D to 3D

$$\varphi(x_1, x_2) \triangleq (x_1^2, \sqrt{2}x_1x_2, x_2^2)$$

$$\langle \varphi(z), \varphi(z') \rangle = \left\langle \begin{pmatrix} x_1^2 \\ \sqrt{2}x_1x_2 \\ x_2^2 \end{pmatrix}, \begin{pmatrix} x_1'^2 \\ \sqrt{2}x_1'x_2' \\ x_2'^2 \end{pmatrix} \right\rangle = (x_1x_1')^2 + 2(x_1x_1')(x_2x_2') + (x_2x_2')^2$$

$$z = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \quad z' = \begin{pmatrix} x_1' \\ x_2' \end{pmatrix}$$

$$= (x_1x_1' + x_2x_2')^2$$

$$= (\langle z, z' \rangle)^2 = K(z, z')$$

d could even be ∞-dim for some kernels

e.g. Gaussian kernel / RBF kernel

$$K(x, x') = \exp\left(\frac{-\|x - x'\|^2}{2\sigma^2}\right)$$ } bandwidth parameter

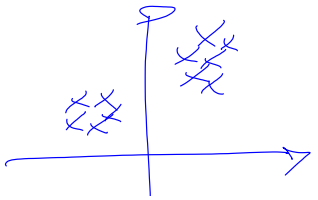
i.e. $\exists\ \varphi: \mathcal{X} \to \mathcal{H}$ ($\mathcal{H}$: RKHS Hilbert space)

s.t. $\langle \varphi(x), \varphi(x') \rangle_{\mathcal{H}} = K(x, x') =$ ☆

* note: neural networks can be seen as "learning" $\varphi(x)$ ie. learning the kernel ...

Unsupervised learning

here X without labels Y...



Consider the Gaussian mixture model (GMM) (can be obtained from FLD):

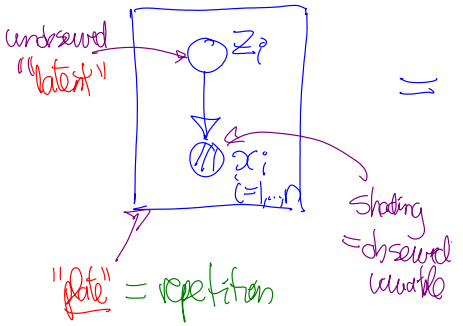
$$Y \sim \text{Mult}(\pi), \quad \pi \in \Delta_K$$

$$X \mid Y = j \sim N(\mu_j, \Sigma)$$

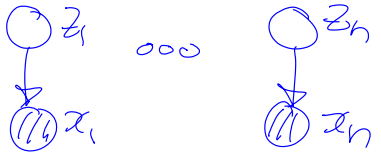
$$p(x) = \sum_y p(x, y) = \sum_{j=1}^{K} \pi_j N(x \mid \mu_j, \Sigma)$$

GMM model, more generally could use different covariance per class Σ_j

graphical model for this "latent variable model" (use z instead of Y)

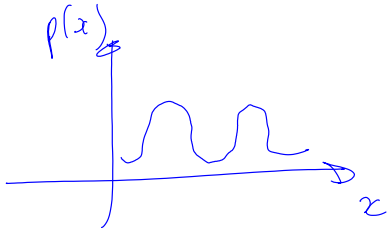


$=$

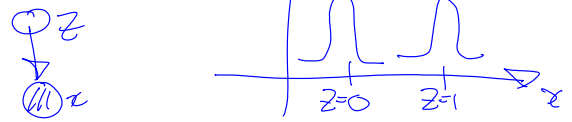


two views on $p(x)$

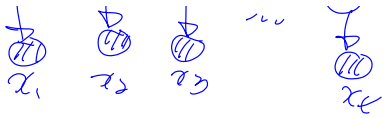
mixture distribution



latent variable model



(later in class, we will add time structure: HMM)

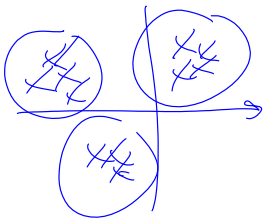


K-means → to do clustering i.e. group data

(can be seen as a limit of GMM)

we want to get cluster assignment for every data point x_i

intuition:



represent $z_{ij} = 1$ to mean x_i in cluster j

$j = 1, \dots, k$ # of clusters (specified in advance for K-means)

applications: • vector quantization

• in computer vision: use K-means to get "bag of visual words" representation of image patches

K-mean algorithm → can be seen as block-coordinate minimization of objective function

$$J(z, \mu) \triangleq \sum_{i=1}^{n} \left(\sum_{j=1}^{k} z_{ij} \, \|x_i - \mu_j\|^2 \right)$$

"distersion measure"

z: cluster assignment $z_1, \dots, z_n \in$ corners of Δ_k

μ: $\mu_1, \dots \mu_k \in \mathbb{R}^d$ cluster means

$\|x_i - \mu_{z_i}\|^2$

alg: 1) initialize $\mu^{(1)}$

2) iterate to convergence:

"E" step: $z^{(t+1)} = \underset{z \in \text{valid clustering}}{\operatorname{argmin}} J(z, \mu^{(t)})$

$\Rightarrow z_{ij}^{(t+1)} = 1$ for $j^* = \underset{j}{\operatorname{argmin}} \|x_i - \mu_j^{(t)}\|$

$\Rightarrow z_{ij}^{(t+1)} = 1$ for $j^* = \arg\min_j \|x_i - \mu_j^{(t)}\|$

"M" step: $\mu^{(t+1)} = \arg\min_{\mu \in \mathbb{R}^{dxk}} J(z^{(t+1)}, \mu)$

$\Rightarrow \mu_j^{(t+1)} = \left(\frac{1}{\sum_i z_{ij}}\right) \sum_i z_{ij} x_i$ (empirical mean with cluster)

demo:

http://home.deib.polimi.it/matteucc/Clustering/tutorial_html/AppletKM.html

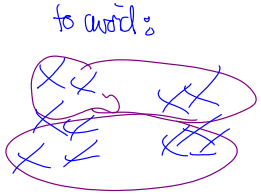
properties:

1) converge in finite # of iteration to local min

2) NP hard in general to find best z

K-means++: is a clever (random) initialization scheme which guarantees objective is within $\log K$ of global optimum (w.h.p.)

→ idea: spread out as much as possible the initial means



choice of k?

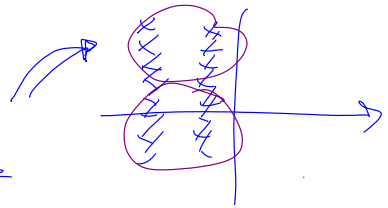
hyperparameter

- one heuristic is: $J(\mu, z, K) = \sum_{i=1}^{n} \sum_{j=1}^{K} z_{ij} \|x_i - \mu_j\|^2 + \lambda K$

→ we'll see later in class "non-parametric models" where "K" is basically infinite and can get $p(K|\text{data})$ e.g. Dirichlet process mixture model

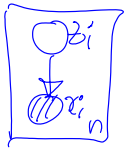
(see also model selection lecture later)

3) K-mean is very sensitive on distance measure: if assumes spherical cluster



↳ GMM addresses that

EM – maximum likelihood in latent variable models

setup 

X is observed
z is latent (unobserved)

log-likelihood $\log p(x_{1:n};\theta) = \log \prod_{i=1}^{n} p(x_i;\theta)$

$= \sum_{i=1}^{n} \log p(x_i;\theta)$

$= \sum_{i=1}^{n} \log \left[\sum_{z_i} p(x_i, z_i;\theta) \right]$

problem??

→ gives multi-modal difficult optimization problem

options for ML in latent variable model:

1) do gradient ascent on non-concave objective
2) EM algorithm ↔ block-coordinate ascent on auxiliary function that lower bounds $\log p(x_{1:n};\theta)$

nice interpretation in terms of filling missing data

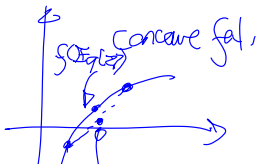
i.e. E step → fills z with "soft-values"
M step → maximize w.r.t. θ (with respect) for fully observed model

trick overview:

$$\log \sum_z p(x,z) = \log \sum_z q(z) \frac{p(x,z)}{q(z)}$$

Jensen's inequality

$$E_q[f(z)] \leq f(E_q[z])$$



where $q(z)$ is some distribution on z | when f is concave | $f(\mathbb{E}_q[g(z)])$

$$\log\left(\mathbb{E}_q\left[\frac{p(x,z)}{q(z)}\right]\right) \geq \mathbb{E}_q\left[\log \frac{p(x,z)}{q(z)}\right] = \sum_z q(z)\log p(x,z;\theta) - \sum_z q(z)\log q(z)$$

Jensen's inequality trick?

$$\triangleq \mathcal{L}(q,\theta) \triangleq \mathbb{E}_{q(z)}[\log p(x,z;\theta)] + H(q)$$

$H(q)$: entropy of q

we have $\log p(x;\theta) \geq \mathcal{L}(q,\theta) \quad \forall q,\theta$

we get equality when "f" is linear ie. $\frac{p(x,z)}{q(z)} = \text{constant with respect to } z = p(x)$

ie. $q(z) \propto p(x,z)$

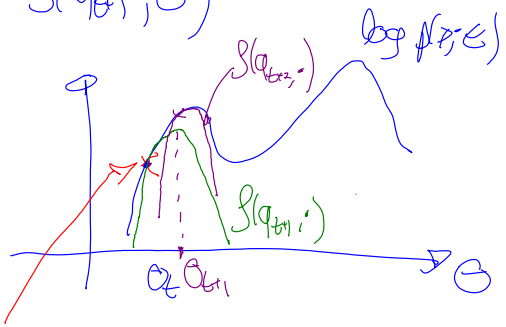
$$\Rightarrow q^*(z) = \frac{p(x,z)}{\sum_{z'} p(x,z')} = \frac{p(x,z)}{p(z)} = p(z|x)$$

this means that $\underset{q \in \text{distributions}}{\text{argmax}}\ \mathcal{L}(q,\theta_t) = p(z|x;\theta_t)$

EM algorithm: E step: $q_{t+1} = \underset{q}{\text{argmax}}\ \mathcal{L}(q,\theta_t) \Rightarrow q_{t+1}(z) \triangleq p(z|x;\theta_{t+1})$

M step: $\theta_{t+1} = \underset{\theta}{\text{argmax}}\ \mathcal{L}(q_{t+1},\theta)$

block coordinate ascent on $\mathcal{L}(q,\theta) \leq \log p(x;\theta)$



we have that $\mathcal{L}(q_{t+1},\theta_t) = \mathcal{L}(q_t,\theta_t)$

we have $\log p(x;\theta_t) = \mathcal{L}(q_{t+1}, \theta_t)$

$\hookrightarrow q_{t+1}(z) = p(z|x;\theta_t)$

properties:

a) $\log p(x;\theta_{t+1}) \geq \log p(x;\theta_t)$ $\Rightarrow$ EM is increasing the likelihood

b) θ_t in EM converges to a stationary point of $\log p(x;\theta)$

i.e. $\nabla_\theta \log p(x;\theta^*) = 0$

Like k-means, initialization is crucial

→ usually need random restarts

for GMM, could use k-mean to initialize the μ_k's (means of Gaussian)