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:
\[ \sum_i (y_i - wx_i) = X^T \hat{\theta}_{ls} \]
for logistic regression,
\[ \sum_i (y_i - \sigma(wx_i)) = X^T \hat{\theta}_{ls} \]

\[ \Rightarrow \text{ during gradient descent, } W_t = \sum_{i=1}^{n} \alpha_i \hat{x}_i \]

Also for ridge regression, solution:
\[ \hat{\theta} = (X^T X + \lambda I_d)^{-1} X^T y \]
\[ = X^T (X^T X + \lambda I_n)^{-1} y \]
\[ = \hat{\theta}_{ls} \]
\[ (X^T X)_{ij} = \langle x_i, x_j \rangle = k(x_i, x_j) \]
"Gram matrix" / "Kernel matrix"

\[ \Rightarrow \"\text{Kernalized least squares}\"

Suppose have mapping
\[ X \rightarrow \Phi(x) \]
"feature space embedding"

\[ \hat{\theta} = \sum_{i=1}^{n} \alpha_i \langle \Phi(x_i), \Phi(x) \rangle \]
"Kernel trick": you can implicitly work in high dimensional space only using $K(\cdot, \cdot)$ evaluation.

For example: from 2D to 3D

$$\psi(x_1, x_2) = (x_1^2, \sqrt{2}x_1x_2, x_2^2)$$

$$<\psi(z_1), \psi(z_2)> = \begin{bmatrix} x_1^2 \\ \sqrt{2}x_1x_2 \\ x_2^2 \end{bmatrix} \begin{bmatrix} x_1^2 \\ \sqrt{2}x_1x_2 \\ x_2^2 \end{bmatrix}$$

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

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

$$= <z_1, z_2>^2 = K(z_1, z_2)$$

A kernel could even be co-dim for some kernels e.g. Gaussian Kernel / RBF Kernel

$$K(x, z) = \exp(-\frac{||x-z||^2}{2\sigma^2})$$

i.e. $\psi: X \rightarrow \mathcal{H}$

$s.t. <\psi(x), \psi(x')>_{\mathcal{H}} = K(x, x') = \psi$ 

*Note: Neural networks can be seen as "learning" $\psi(x)$ i.e. learning the kernel $\psi$. 

Unsupervised Learning

Here $X$ without labels $Y$...
Consider the Gaussian mixture model (GMM):

\[ Y \sim \text{Mult}(\pi), \quad \pi \in \Delta_K \]
\[ X | Y = y \sim N(\mu_y, \Sigma) \]
\[ p(x) = \sum_{y \in \Delta_K} \frac{\pi(y)}{\pi(y)} N(x | \mu_y, \Sigma) \]

GMM model, more generally could use different covariance per-class \( \Sigma \)

Graphical model for this "latent variable model" (use \( Z \) instead of \( Y \))

(\( Z \) = repeated)

Two views on \( p(x) \)

Mixture distribution

Latent variable model

(\( \pi \) = probability of class)

(\( \Sigma \) = covariance for class)

(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 \( j \)
\( j = 1, \ldots, K \) # 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
\[
J(z, \mu) = \sum_{i=1}^{N} \min_{j \in \{1, \ldots, K\}} \| x_i - \mu_j \|_2^2
\]
cluster assignment \( z_1, \ldots, z_N \) in \( \{0, 1\}^K \) \( \mu_1, \ldots, \mu_K \) cluster means
alg: 1) initialize \( \mu_0 \)
2) iterate to convergence:
"E" step: \( z^{(t+1)} = \arg \min_{z \in \{0, 1\}^K} J(z, \mu^{(t)}) \)
\( z_{ij}^{(t+1)} = 1 \) for \( j = \arg \min_{j \in \{1, \ldots, K\}} \| x_i - \mu_j^{(t)} \|_2^2 \)
demo: http://home.deib.polimi.it/matteucc/Clustering/tutorial_html/AppletKM.html

1) converge in finite # of iteration to local min
2) NP hard in general to find best \( \mu \)

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

\[ J(\mu, z, K) = \sum_{i=1}^{n} \sum_{j=1}^{K} z_{ij} ||x_i - \mu_j||^2 + \lambda K \]

we'll see later in class "non-parametric models"

where "\( K \)" is basically infinite

e.g. Dirichlet process

and can get \( p(K \mid data) \)

(see also model selection lecture later)

3) K-mean is very sensitive on distance measure & it assumes spherical cluster

To avoid:

Choice of \( K \):

1) heuristic is:
   \[ J(\mu, z, K) = \sum_{i=1}^{n} \sum_{j=1}^{K} z_{ij} ||x_i - \mu_j||^2 + \lambda K \]
EM - maximum likelihood in latent variable model

\[ \log \text{likelihood} = \log p(x, z) = \log \left( \sum_z q(z) p(x, z) \right) \]

- Gave multi-modal & difficult optimization problem

Options for ML in latent variable model:

1) Do gradient ascent on non-concave objective

2) EM algorithm: coordinate ascent on auxiliary function that lower bounds \( \log p(x, z) \)

Nice interpretation in terms of filling missing data

i.e. \( E \) step \( \rightarrow \) fill \( z \) with \( q(z) \) values

M step \( \rightarrow \) maximize \( \psi(z \| B) \) for fully observed model with respect

\[ \log \sum_z p(z, x) = \log \sum_z q(z) p(z, x) \]

Jensen's inequality

\[ \mathbb{E}[\mathcal{S}(x)] \leq \mathcal{S}(\mathbb{E}[x]) \]

\( \rightarrow \) concave set,
where \( q(z) \) is some distribution on \( z \) when \( f \) is constant

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

we have \( \log p(x; \theta) \geq \mathbb{E}_q \left[ \log \frac{p(x,z; \theta)}{q(z)} \right] + H(q) \)

we get equality when \( f \) is linear i.e. \( f(z) = \text{constant} = p(z) \)

dd. \( q(z) \) or \( p(x,z) \)

\[
\Rightarrow q^{*}(z) = p(x,z) = p(x,z) = p(z|x) \\
\geq p(z|x) = p(z)
\]

this means that \( \arg \max_{q \text{distribution}} \mathbb{E}_q q(z) = p(z|x; \theta) \)

EM algorithm: E step: \( q_{t+1} = \arg \max_{q} \mathcal{L}(q; x, \theta_t) \Rightarrow q_{t+1}(z) = p(z|x, \theta_t) \)

M step: \( \theta_{t+1} = \arg \max_{\theta} \mathcal{L}(q_{t+1}; x, \theta) \)

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

we have \( \lim_{t \to \infty} \mathcal{L}(q_t; x, \theta) = \log p(x; \theta) \)
We have \[ \log p(x; \Theta) = \ell(\Theta_{\text{ML}}) \]
\[ \ell(\Theta_{\text{ML}}) = \ell(x) \]

Properties:

1. \[ \log p(x; \Theta) > \log p(x; \tilde{\Theta}) \implies \text{EM is increasing the likelihood} \]
2. \( \Theta_{\text{ML}} \) in EM converges to a stationary point of \( \log p(x; \Theta) \)

i.e. \[ \nabla \log p(x; \Theta_{\text{ML}}) = 0 \]

Like k-means, initialization is crucial.

- Usually need random restarts.

For GMM, could use k-means ++ to initialize the means of Gaussian.

Teaching Page 8