K-medoids algorithm:

IN: $\{\underline{x}_i\}_{i=1}^N$, distance measure $D(\underline{x}_i, \underline{\mu})$

Randomly initialise $\underline{\mu}$

- Iterate until $\underline{c}$ is no longer changing:-

1. For each c_i, set $c_i = \underset{k}{\operatorname{argmin}}\ D(\underline{x}_i, \underline{\mu}_k)$

2. For each μ_k, set $\underline{\mu}_k = \underset{\underline{\mu}}{\operatorname{argmin}} \sum_{i:c_i=k} D(\underline{x}_i, \underline{\mu})$ (*)

- Extension using general distance metric
- K-means $D(\underline{x}_i, \underline{\mu}) = \|\underline{x}_i - \underline{\mu}\|_2^2$ (Euclidean); L_1: $D(\underline{x}_i, \underline{\mu}) = \|\underline{x}_i - \underline{\mu}\|_1$ (robust to outliers)

LIS-Key equations

Probabilistic data modelling: - Bayes classifier, logistic regression, least-squares, ridge-reg (ML/MAP interp.), Bayesian LR

Data modelling: - Perceptron, decision trees, SVM, K-means (non-probabilistic)

ML estimation (situation I): - 4 ingredients 1) model param. 2) data $\{\underline{x}_i\}_{i=1}^N$ 3) distr family $p(\underline{x}|\underline{\theta})$ 4) iid $\underline{x}_i \sim p(\underline{x}|\underline{\theta})$

(ML) - $\underline{\theta}_{ML} = \underset{\underline{\theta}}{\operatorname{argmax}}\ p(\underline{x}_1, \dots, \underline{x}_N|\underline{\theta}) \overset{(a)}{=} \underset{\underline{\theta}}{\operatorname{argmax}} \prod_{i=1}^N p(\underline{x}_i|\underline{\theta}) \overset{(b)}{=} \underset{\underline{\theta}}{\operatorname{argmax}} \sum_{i=1}^N \ln p(\underline{x}_i|\underline{\theta})$

a) iid b) $f(y) > f(x) \Rightarrow \ln f(y) > \ln f(x)$; $\underset{y}{\operatorname{argmax}} \ln g(y) = \underset{y}{\operatorname{argmax}}\ g(y)$

LSLR + Bayes class (Analytic sol.): $\underline{x}_i \overset{iid}{\sim} N(\underline{x}|\underline{\mu}, \underline{\Sigma})$ $\underline{\theta} = (\underline{\mu}, \underline{\Sigma})$ $\nabla_{\underline{\theta}} \ln\left(\prod_{i=1}^N p(\underline{x}_i|\underline{\theta})\right) = 0$ yields:-

$$\hat{\underline{\mu}}_{ML} = \frac{1}{N}\sum_{i=1}^N \underline{x}_i \qquad \hat{\underline{\Sigma}}_{ML} = \frac{1}{N}\sum_{i=1}^N (\underline{x}_i - \hat{\underline{\mu}}_{ML})(\underline{x}_i - \hat{\underline{\mu}}_{ML})^T \text{ (LSLR)} \qquad \text{(BC)}$$

$$\hat{\underline{\mu}}_k = \frac{1}{n_k}\sum_{i=1}^N \mathbb{I}(y_i = k)\underline{x}_i \qquad \hat{\underline{\Sigma}}_k = \frac{1}{n_k}\sum_{i=1}^N \mathbb{I}(y_i = k)(\underline{x}_i - \hat{\underline{\mu}}_k)(\underline{x}_i - \hat{\underline{\mu}}_k)^T \qquad \hat{\pi}_k = \frac{1}{N}\sum_{i=1}^N \mathbb{I}(y_i = k)$$

ML estimation (situation II): - split parameters into two groups $\underline{\theta}_1, \underline{\theta}_2$, maximise likelihood

(ML) $\hat{\underline{\theta}}_{1,ML}, \hat{\underline{\theta}}_{2,ML} = \underset{\underline{\theta}_1, \underline{\theta}_2}{\operatorname{argmax}} \sum_{i=1}^N p(\underline{x}_i|\underline{\theta}_1, \underline{\theta}_2)$

- can solve one given the other, cannot solve simultaneously

Co-ordinate ascent(?): (General, probabilistic) For $t = 1, 2, \dots$

1. Optimise $\underline{\theta}_1^{(t)} = \underset{\underline{\theta}_1}{\operatorname{argmax}} \sum_{i=1}^N \ln p(\underline{x}_i|\underline{\theta}_1, \underline{\theta}_2^{(t-1)})$

2. Optimise $\underline{\theta}_2^{(t)} = \underset{\underline{\theta}_2}{\operatorname{argmax}} \sum_{i=1}^N \ln p(\underline{x}_i|\underline{\theta}_1^{(t)}, \underline{\theta}_2)$

ML situation (situation III): (III): Find $\hat{\underline{\theta}}_{1,ML} = \underset{\underline{\theta}_1}{\operatorname{argmax}} \sum_{i=1}^N \ln p(\underline{x}_i|\underline{\theta}_1)$ via:

(E-M) $\hat{\underline{\theta}}_{1,ML} = \underset{\underline{\theta}_1}{\operatorname{argmax}} \sum_{i=1}^N \ln p(\underline{x}_i, \underline{\theta}_2|\underline{\theta}_1)$ (*)

(add a second variable $\underline{\theta}_2$) LH side condition. → prior on $\underline{\theta}_2$

· $\underline{\theta}_2$ is a 'latent variable'; as marginal distri. $p(\underline{x}_i|\underline{\theta})$ is tricky to optimise, we integrate over the joint distribution of $\underline{x}_i$ and latent $\underline{\theta}_2$
· E-M algorithm finds $\hat{\underline{\theta}}_1, \underline{m}$ via this method.

General Expectation-Maximisation:

setup: Two parameter sets $\underline{\theta}_1, \underline{\theta}_2$: $p(\underline{x}|\underline{\theta}_1) = \int p(\underline{x}, \underline{\theta}_2|\underline{\theta}_1)\, d\underline{\theta}_2$ (marginal distri.)

objective criteria: define objective criteria
1) optimise marginal $p(\underline{x}_i|\underline{\theta}_1)$ on $\underline{\theta}_1$
2) $p(\underline{x}, \underline{\theta}_2|\underline{\theta}_1)$ used for comp. convenience
3) we know (and) posterior $p(\underline{\theta}_2|\underline{x}, \underline{\theta}_1)$ given data $\underline{x}, \underline{\theta}_1$

EM-objective function:

$$\ln p(\underline{x}|\underline{\theta}_1) = \underbrace{\int q(\underline{\theta}_2) \ln \frac{p(\underline{x}, \underline{\theta}_2|\underline{\theta}_1)}{q(\underline{\theta}_2)}\, d\underline{\theta}_2}_{1.\ \mathcal{L}(\underline{x}, \underline{\theta}_1)} + \underbrace{\int q(\underline{\theta}_2) \ln \frac{q(\underline{\theta}_2)}{p(\underline{\theta}_2|\underline{x}, \underline{\theta}_1)}\, d\underline{\theta}_2}_{2.\ KL(q\|p)}$$

1. $\mathcal{L}(\underline{x}, \underline{\theta}_1)$ can be calculated (evaluated as integral), leaving only function of $\underline{\theta}_1$ for a particular setting $q(\underline{\theta}_2)$
2. $KL(q\|p)$: $KL \geq 0$, $KL = 0$ when $q = p$

EM-objective: (identity?) Product rule of prob.; conditional prob, $\ln p(\underline{x}|\underline{\theta}_1) \neq f(\underline{\theta}_2)$ (factor out)

EM algorithm output: · Heuristicly optimise $\ln p(\underline{x}|\underline{\theta}_1)$ wrt $\underline{\theta}_1 \Rightarrow$
· Generate a sequence of values $\underline{\theta}_1^{(1)}, \ldots, \underline{\theta}^{(T)}$: 1. sequence $\{\underline{\theta}_1^{(t)}\}_{t=1}^{T}$: $\ln p(\underline{x}|\underline{\theta}_1^{(t)}) \geq \ln p(\underline{x}|\underline{\theta}_1^{(t-1)})$
2. We want $\underline{\theta}_1^{(t)}$ converge to a local maximum in $p(\underline{x}|\underline{\theta}_1)$
· EM generates #1, satisfies #2

EM-Algorithm:

Given: $\underline{\theta}_1^{(t)}$, find $\underline{\theta}_1^{(t+1)}$:-

E-step: set $q_t(\underline{\theta}_2) = p(\underline{\theta}_2|\underline{x}, \underline{\theta}_1^{(t)})$, evaluate

$$\mathcal{L}_t(\underline{x}, \underline{\theta}_1) = \int q_t(\underline{\theta}_2) \ln \frac{p(\underline{x}, \underline{\theta}_2|\underline{\theta}_1)}{q_t(\underline{\theta}_2)}\, d\underline{\theta}_2 = \int q_t(\underline{\theta}_2) \ln p(\underline{x}, \underline{\theta}_2|\underline{\theta}_1)\, d\underline{\theta}_2 - \underbrace{\int q_t(\underline{\theta}_2) \ln q_t(\underline{\theta}_2)\, d\underline{\theta}_2}_{\text{ignore}}$$

M-step: set $\underline{\theta}_1^{(t+1)} = \underset{\underline{\theta}_1}{\operatorname{argmax}}\ \mathcal{L}_t(\underline{x}, \underline{\theta}_1)$

EM results in monotonic increases in observed data log-likelihood $\ln p(\underline{x}|\underline{\theta}_1)$

- In other presentations, E-step is framed as taking expected value of the complete-data log likelihood with respect to the conditional distribution of θ_2 (or $\underline{z}$) given $\underline{x}$ and current estimates of parameters $\underline{\theta}_1^{(t)}$

i.e. $\mathbb{E}_{\underline{\theta}_2 | \underline{x}, \underline{\theta}_1^{(t)}} \left[\log L(\underline{\theta}_1; \underline{x}, \underline{\theta}_2) \right]$ (?) came back (*) harmonise with other sources

EM-missing data:-

- $\underline{x}_i \in \mathbb{R}^d$ - a vector with missing data
- $\underline{x}_i$ { 1. $\underline{x}_i^o$ (observed sub-vector of $\underline{x}_i$) 2. $\underline{x}_i^m$ (unobserved sub-vector of $\underline{x}_i$) } - missing dim. can be different for different $\underline{x}_i$
- Assuming $\underline{x}_i \overset{iid}{\sim} N(\underline{\mu}, \underline{\Sigma})$, we require $\hat{\underline{\mu}}_{ML}, \hat{\underline{\Sigma}}_{ML} = \underset{\underline{\mu}, \underline{\Sigma}}{\operatorname{argmax}} \sum_{i=1}^{N} \ln p(\underline{x}_i^o | \underline{\mu}, \underline{\Sigma})$
- **Intractable:** as $\underline{x}_i^o$ (each) has different dimensionality; each dim corresponds to different subsets of dimensions in $\underline{\mu}$ and $\underline{\Sigma}$

Tractable optimisation: If we knew $\underline{x}_i^m$ (and $\underline{x}_i$), following is tractable:- (?) (*)

$\hat{\underline{\mu}}_{ML}, \hat{\underline{\Sigma}}_{ML} = \underset{\underline{\mu}, \underline{\Sigma}}{\operatorname{argmax}} \sum_{i=1}^{N} \ln p(\underline{x}_i^o, \underline{x}_i^m | \underline{\mu}, \underline{\Sigma})$ easy to optimise

ML estimation + imputation of missing values: - A method for optimising $\sum_{i=1}^{N} \ln p(\underline{x}_i^o | \underline{\mu}, \underline{\Sigma})$ and imputing $\{\underline{x}_1^m, \ldots, \underline{x}_N^m\}$

Setup: $p(\underline{x}_i^o | \underline{\mu}, \underline{\Sigma}) = \int p(\underline{x}_i^o, \underline{x}_i^m | \underline{\mu}, \underline{\Sigma}) \, d\underline{x}_i^m = N(\underline{\mu}_i^o, \underline{\Sigma}_i^o)$ from $\underline{x}_i^o$ { $\underline{\mu}_i^o$ - subvector of $\underline{\mu}$; $\underline{\Sigma}_i^o$ - submatrix of $\underline{\Sigma}$ }

Single Gaussian EM: (missing data)

$$\sum_{i=1}^{N} \ln p(\underline{x}_i^o | \underline{\mu}, \underline{\Sigma}) = \sum_{i=1}^{N} \int q(\underline{x}_i^m) \ln \frac{p(\underline{x}_i^o, \underline{x}_i^m | \underline{\mu}, \underline{\Sigma})}{q(\underline{x}_i^m)} d\underline{x}_i^m + \sum_{i=1}^{N} \int q(\underline{x}_i^m) \ln \frac{q(\underline{x}_i^m)}{p(\underline{x}_i^m | \underline{x}_i^o, \underline{\mu}, \underline{\Sigma})} d\underline{x}_i^m$$

E-step (I): set $q(\underline{x}_i^m) = p(\underline{x}_i^m | \underline{x}_i^o, \underline{\mu}, \underline{\Sigma})$ under current $\underline{\mu}, \underline{\Sigma}$

- $\underline{x}_i = \begin{bmatrix} \underline{x}_i^o \\ \underline{x}_i^m \end{bmatrix} \sim N\left(\begin{bmatrix} \underline{\mu}_i^o \\ \underline{\mu}_i^m \end{bmatrix}, \begin{bmatrix} \underline{\Sigma}_i^{oo} & \underline{\Sigma}_i^{om} \\ \underline{\Sigma}_i^{mo} & \underline{\Sigma}_i^{mm} \end{bmatrix} \right) \rightarrow p(\underline{x}_i^m | \underline{x}_i^o, \underline{\mu}, \underline{\Sigma}) = N(\hat{\underline{\mu}}_i, \hat{\underline{\Sigma}}_i)$

via cond. Gaussian results: $\begin{cases} \hat{\underline{\mu}}_i = \underline{\mu}_i^m + \underline{\Sigma}_i^{mo} (\underline{\Sigma}_i^{oo})^{-1} (\underline{x}_i^o - \underline{\mu}_i^o) \\ \hat{\underline{\Sigma}}_i = \underline{\Sigma}_i^{mm} - \underline{\Sigma}_i^{mo} (\underline{\Sigma}_i^{oo})^{-1} \underline{\Sigma}_i^{om} \end{cases}$

E-step (II): - $\mathbb{E}_{q(\underline{x}_i^m)} \left[\ln p(\underline{x}_i^o, \underline{x}_i^m | \underline{\mu}, \underline{\Sigma}) \right]$

- for each i, evaluate:-

$$\mathbb{E}_q \left[(\underline{x}_i - \underline{\mu})^T \underline{\Sigma}^{-1} (\underline{x}_i - \underline{\mu}) \right] = \mathbb{E}_q \left[\operatorname{trace} \left\{ \underline{\Sigma}^{-1} (\underline{x}_i - \underline{\mu})(\underline{x}_i - \underline{\mu})^T \right\} \right]$$
$$= \operatorname{trace} \left\{ \underline{\Sigma}^{-1} \mathbb{E}_q \left[(\underline{x}_i - \underline{\mu})(\underline{x}_i - \underline{\mu})^T \right] \right\}$$

- Calculated using $q(\underline{x}_i^m) = p(\underline{x}_i^m | \underline{x}_i^o, \underline{\mu}, \underline{\Sigma})$ - only $\underline{x}_i^m$ portion of $\underline{x}_i$ integr.
- $q(\underline{x}_i^m) = N(\hat{\underline{\mu}}_i, \hat{\underline{\Sigma}}_i)$

M-step: Define $\hat{x}_i$ - vector where we replace missing values in x_i with $\hat{\mu}_i$

$\hat{V}_i$ - matrix of 0s plus sub-matrix $\hat{\Sigma}_i$ in missing dimensions

- Maximise $\sum_{i=1}^{N} \mathbb{E}_q[\ln p(x_i^o, x_i^m | \mu, \Sigma)]$
- $\hat{\mu}_{up}, \hat{\Sigma}_{up} = \underset{\mu,\Sigma}{\operatorname{argmax}} \sum_{i=1}^{N} \mathbb{E}_q[\ln p(x_i^o, x_i^m | \mu, \Sigma)]$
- $\hat{\mu}_{up} = \frac{1}{N}\sum_{i=1}^{N} \hat{x}_i \quad \hat{\Sigma}_{up} = \frac{1}{N}\sum_{i=1}^{N} \{(\hat{x}_i - \hat{\mu}_{up})(\hat{x}_i - \hat{\mu}_{up})^T + \hat{V}_i\}$

Return to E-step: calculate new $p(x_i^m | x_i^o, \mu_{up}, \Sigma_{up})$

High-level: - Hoover mechanics again (*), (**), not fully clear.

(strategy) 1. maximise log-likelihood only considering observed point x_i^o over unknown μ, Σ

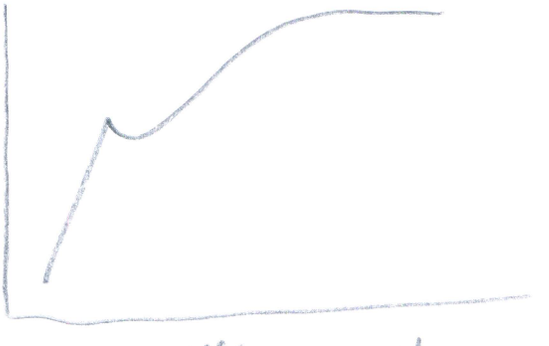
2. use q-distribution (conditional posterior on missing x_i^m $p(x_i^m | x, \theta)$) given data and parameter estimates to make probabilistic state about conjectured values for x_i^m

Implementation: - Each EM update → monotonic increase in log likelihood of incomplete dataset ~~$p(x_i|\theta$~~ $\log p(x_i^o | \theta)$

Convergence: - marginal improvements in $\ln p(x_i^o | \theta)$

- Initialise μ and Σ (set x_i^m (missing) = 0 and use initial $\hat{\mu}_{ML}, \hat{\Sigma}_{ML}$ or random initialisation)

- EM objective after each iteration update to μ, Σ



(or) 1) - Output $\hat{\mu}_{ML}, \hat{\Sigma}_{ML}$ that maximises $\sum_{i=1}^{N} \ln p(x_i^o | \mu, \Sigma)$

(or) 2) Output conditional posterior $q(x_i^m) = p(x_i^m | x_i^o, \hat{\mu}_{ML}, \hat{\Sigma}_{ML})$ on missing dimensions which gives mean and uncertainty of missing data x_i^m i.e. probability distri on x_i^m → impute $\{x_i^m, \ldots, x_N^m\}$

L16 - Key equations

K-means: Hard clustering model, each observation $\underline{x}_i$ assigned to only one cluster $c_i = k$ for some $k \in \{1, \dots, K\}$, no accounting for boundary cases probabilist.

Soft clustering: Data can be clustered semi-probabilistically (K-means is non-prob.)
(probabilistic semantics, but no formal distributions defined)

Weighted K-means (algorithm)

IN: data $\{(\underline{x}_i)\}_{i=1}^N$ $\underline{x}_i \in \mathbb{R}^d$

GOAL: minimise $\mathcal{L} = \sum_{i=1}^{N} \sum_{k=1}^{K} \phi_i(k) \frac{\|\underline{x}_i - \underline{\mu}_k\|_2^2}{\beta}$ over all ϕ_i $i = 1, \dots, N$, $\underline{\mu}_k$ $k = 1, \dots, K$

conditions: $\phi_i(k) > 0$, $\sum_{k=1}^{K} \phi_i(k) = 1$, $\beta > 0$

Iterate:

1. update ϕ: For each i, update word allocation weights

$$\phi_i(k) = \frac{\exp\left\{-\frac{1}{\beta}\|\underline{x}_i - \underline{\mu}_k\|_2^2\right\}}{\sum_j \exp\left\{-\frac{1}{\beta}\|\underline{x}_i - \underline{\mu}_k\|_2^2\right\}} \quad \text{for } k = 1, \dots, K$$

2. update μ: For each k, update $\underline{\mu}_k$ with weighted average:

$$\underline{\mu}_k = \frac{\sum_{i=1}^{N} \underline{x}_i \phi_i(k)}{\sum_{i=1}^{N} \phi_i(k)}$$

- substitute $\mathbb{I}\{c_i = k\}$ (K-means) $\leftarrow \frac{\phi_i(k)}{\beta}$ into $\mathcal{L}$ and $\underline{\mu}_k$ update with $\phi_i(k) = \frac{\exp\left\{-\frac{1}{\beta}\|\underline{x}_i - \underline{\mu}_k\|_2^2\right\}}{\sum_j \exp\left\{-\frac{1}{\beta}\|\underline{x}_i - \underline{\mu}_j\|_2^2\right\}}$ (normalised RBF kernel)
- $\phi_i(k) = \{0, 1\} \rightarrow$ hard clustering
- $\phi_i(k)$ large - $\underline{x}_i$ close to $\underline{\mu}_k$; $\phi_i(k)$ - $\underline{x}_i$ far from $\underline{\mu}_k$

Mixture models: Above: weight vector ϕ_i is akin to probability $\underline{x}_i$ assigned to cluster

mixture models: A probabilistic model where ϕ_i is actually a probability distribution according to model

Mixture model (high-level): 1. Prior distribution on cluste assignment indicator c_i
2. Likelihood —"— on observation $\underline{x}_i$, given assign. c_i

Analog with SL (Bayes class.): · cluste prior $\overset{SL}{\rightarrow}$ class prior$_{SL}$
· cluste-cond. likelihood $\rightarrow$ class-cond likelihood } but cluste c_i is latent/auxiliary

Mixture models (Features): 1. Generative model (defines p.d. on data $\underline{x}$)
2. A weighted (discrete p.d.) combination of simple distributions (same distri. family)

(see diagram - complex distrn)

generic mixture model (generative process) :- Data: $\{\underline{x}_i\}_{i=1}^N$ $\underline{x}_i \in \mathcal{X} / \mathbb{R}^d$

· model param: $\underline{\pi} \in \mathbb{R}^K$ $\underline{\pi} = \begin{bmatrix} \pi_1 \\ \pi_K \end{bmatrix}$; $\underline{\theta}_1, \dots, \underline{\theta}_K$

generative process: For obs $i = 1, \dots, N$:

1. Generate cluste assign. $c_i \overset{iid}{\sim} Dis(\underline{\pi})$ $p(c_i = k | \underline{\pi}) = \pi_k$
2. Generate obs. $\underline{x}_i \sim p(\underline{x} | \underline{\theta}_{c_i})$

· Each $\underline{x}_i$ randomly assigned to cluste c_i via distri $\underline{\pi}$

$\underline{\pi}$ - mixing weights · c_i indexes cluste assignment for $\underline{x}_i \rightarrow$ i) selects index of

· 2 $\underline{x}_i$s shae param → clustered together $\underline{\theta}$ to gen. $\underline{x}_i$ $(\underline{\theta}_1, \dots, \underline{\theta}_K)$

EM-generic mixture algorithm:

IN: $\{\underline{x}_i\}_{i=1}^N$ $\underline{x} \in \mathcal{X}$

GOAL: Maximise $\mathcal{L} = \ln p(\underline{x}_i | \underline{\pi}, \underline{\theta})$ where $p(\underline{x} | \underline{\theta}_k)$ is problem spec.

Iterate until incremental improvement to $\mathcal{L}$ is small :-

1. E-step: For $i = 1, \dots, n$ set :- (posterior prob.)

$$\phi_i(k) = \frac{\pi_k p(\underline{x}_i | \underline{\theta}_k)}{\sum_j p(\underline{x}_i | \underline{\theta}_j) \pi_j} \quad \text{for } k = 1, \dots, K$$

2. M-step: For $k = 1, \dots, K$ define $n_k = \sum_{i=1}^N \phi_i(k)$ and set (*)(~)

$$\pi_k = \frac{n_k}{n} \qquad \underline{\theta}_k = \underset{\underline{\theta}}{\operatorname{argmax}} \sum_{i=1}^N \phi_i(k) \ln p(\underline{x}_i | \underline{\theta})$$

- Similar to generalisation of Bayes clas for any $p(\underline{x} | \underline{\theta}_k)$
- comments: mixture model → specify cluste prior and cluste cond. likelihood;

(*) (~) (?)

Gaussian Mixture Models (GMMs): Parameters: $\underline{\pi} \in \mathbb{R}^K$ $\underline{\pi} = [\pi_1, \dots, \pi_K]$

(μ_k, Σ_k) - mean and covariance of k^{th} Gaussian

$\mu_k \in \mathbb{R}^d$, $\Sigma_k \in \mathbb{R}^{d \times d}$; $k = 1, \dots, K$

generate data: For i^{th} obs $\underline{x}_i$:-

1. Assign i^{th} obs → to cluster $c_i \sim$ discrete$(\underline{\pi})$
2. Generate $\underline{x}_i \sim N(\mu_{c_i}, \Sigma_{c_i})$

$\underline{\mu} = [\mu_1, \dots, \mu_K]$ $\underline{\underline{\Sigma}} = \{\Sigma_1, \dots, \Sigma_K\}$ goal: Estimate $\underline{\pi}, \underline{\mu}, \underline{\underline{\Sigma}}$

(matrix) $\in \mathbb{R}^{d \times K}$ $\in \mathbb{R}^{d \times d \times K}$ (numpy)

Generic EM objective:

$$\ln p(\underline{x}|\underline{\theta}_1) = \int q(\underline{\theta}_2) \ln \frac{p(\underline{x}, \underline{\theta}_2|\underline{\theta}_1)}{q(\underline{\theta}_2)} + \int q(\underline{\theta}_2) \ln \frac{q(\underline{\theta}_2)}{p(\underline{\theta}_2|\underline{x}, \underline{\theta}_1)} d\underline{\theta}_2$$

EM objective for GMM: (joint)

$$\sum_{i=1}^{N} \ln p(\underline{x}_i|\underline{\pi}, \underline{\mu}, \underline{\underline{\Sigma}}) = \sum_{i=1}^{N}\sum_{k=1}^{K} q(c_i=k) \ln \frac{p(\underline{x}_i, c_i=k|\underline{\pi}, \underline{\mu}, \underline{\underline{\Sigma}})}{q(c_i=k)} + \sum_{i=1}^{N}\sum_{k=1}^{K} q(c_i=k) \ln \frac{q(c_i=k)}{p(c_i|\underline{x}_i, \underline{\pi}, \underline{\mu}, \underline{\underline{\Sigma}})}$$

Maximum likelihood EM for GMM:

IN: $\{\underline{x}_i\}_{i=1}^N$ $\underline{x}_i \in \mathbb{R}^d$

GOAL: Maximise $\mathcal{L} = \sum_{i=1}^{N} \ln p(\underline{x}_i|\underline{\pi}, \underline{\mu}, \underline{\underline{\Sigma}})$

- Iterate until incremental improvement to $\mathcal{L}$ is small:-

1. E-step: For $i=1,\dots,N$, set:-

$$\phi_i(k) = \frac{\pi_k N(\underline{x}_i|\underline{\mu}_k, \underline{\underline{\Sigma}}_k)}{\sum_j \pi_j N(\underline{x}_i|\underline{\mu}_j, \underline{\underline{\Sigma}}_j)} \qquad k=1,\dots,K$$

2. M-step: For $k=1,\dots,K$,

a) Define $n_k = \sum_{i=1}^{N} \phi_i(k)$

b) Update values:- $\hat{\pi}_k = \frac{n_k}{n}$ $\hat{\underline{\mu}}_k = \frac{1}{n_k}\sum_{i=1}^{N} \phi_i(k)\underline{x}_i$ $\hat{\underline{\underline{\Sigma}}}_k = \frac{1}{n_k}\sum_{i=1}^{N} \phi_i(k)(\underline{x}_i - \hat{\underline{\mu}}_k)(\underline{x}_i - \hat{\underline{\mu}}_k)^T$

EM-GMM: (one it.)

PRELIM: Set N q-distributions over auxiliary variables $c_1, \dots, c_N$ via Bayes

$q(c_i=k) \leftarrow p(c_i=k|\underline{x}_i, \underline{\pi}, \underline{\mu}, \underline{\underline{\Sigma}})$ as N condit. post $p(c_i=k|\underline{x}_i, \underline{\pi}, \underline{\mu}, \underline{\underline{\Sigma}})$ indexed by $\underline{x}_i$

$p(c_i=k|\underline{x}_i, \underline{\pi}, \underline{\mu}, \underline{\underline{\Sigma}}) \propto p(c_i=k|\underline{\pi}) p(\underline{x}_i|c_i=k, \underline{\mu}, \underline{\underline{\Sigma}})$

$$q(c_i=k) = \frac{\pi_k N(\underline{x}_i|\underline{\mu}_k, \underline{\underline{\Sigma}}_k)}{\sum_j \pi_j N(\underline{x}_i|\underline{\mu}_j, \underline{\underline{\Sigma}}_j)} \rightarrow \phi_i(k)$$

(M) solve for posterior of c_i given $\underline{\pi}, \underline{\mu}, \underline{\underline{\Sigma}}$

E-step: Take expectations using updated qs

$$Q = \sum_{i=1}^{N}\sum_{k=1}^{K} \phi_i(k) \ln p(\underline{x}_i, c_i=k|\underline{\pi}, \underline{\mu}_k, \underline{\underline{\Sigma}}_k) + \text{constant wrt } \underline{\pi}, \underline{\mu}, \underline{\underline{\Sigma}}$$

M-step: Maximise Q wrt $\underline{\pi}$ and each $\underline{\mu}_k, \underline{\underline{\Sigma}}_k$

Reasons for using EM:

original objective:- $\mathcal{L} = \sum_{i=1}^{N} \ln\left(\sum_{k=1}^{K} p(\underline{x}_i, c_i = k \mid \underline{\pi}, \underline{\mu}_k, \underline{\Sigma}_k)\right) = \sum_{i=1}^{N} \ln \sum_{k=1}^{K} \pi_k N(\underline{x}_i \mid \underline{\mu}_k, \underline{\Sigma}_k)$

log-sum

· log-sum ⇒ optimising $\underline{\pi}$, $\underline{\mu}_k$, $\underline{\Sigma}_k$ hard.

EM-M-step: $Q = \sum_{n=1}^{N} \sum_{k=1}^{K} q_i(k) \{\ln \pi_k + \ln N(\underline{x}_i \mid \underline{\mu}_k, \underline{\Sigma}_k)\}$ + constant wrt $\underline{\pi}, \underline{\mu}, \underline{\Sigma}$

$\ln p(\underline{x}_i, c_i = k \mid \underline{\pi}, \underline{\mu}_k, \underline{\Sigma}_k)$

cluster no. selection: · ML for GMM → overfitting · use Dirichlet priors on $\underline{\pi}$ (mixing weight vector) which encourages many Gaussians to disappear

L17 - Key equations:

object recommendation: / content filtering
\ collaborative filtering

content filtering: · use known information about products, uses to make recommend. (e.g. movie info, price info, demograp, questionnaire info) (a priori)

collaborative filtering: use previous use's input/behav. to make recommend. (use feedback, domain fee)

· collaborative filtering: → item-based (evaluate use pref on 'neighbouring' items by same user)
(neighbourhood based) → use-based (like-minded users who complement ratings, similarity score between needed them; let them vote on recommends)

· collaborative-filtering: · ratings → characterise items and uses from factors inferred from ratings
(location/latent factor) · factors measure how much user likes movies that score high on that factor → interp or uninterp.

(EII): location-based/latent factor models embed objects and uses into points in $\mathbb{R}^d$
(OII): Recommendations based on proximity of uses and objects in latent embedding space inferred from data

· Embeddings estimated via matrix factorisation, used to produce estimated ratings

Matrix factorisation: · form ratings matrix $\underline{M}$
(technique, prob, goal) · many M_{ij} = NaN (sparse matrix)

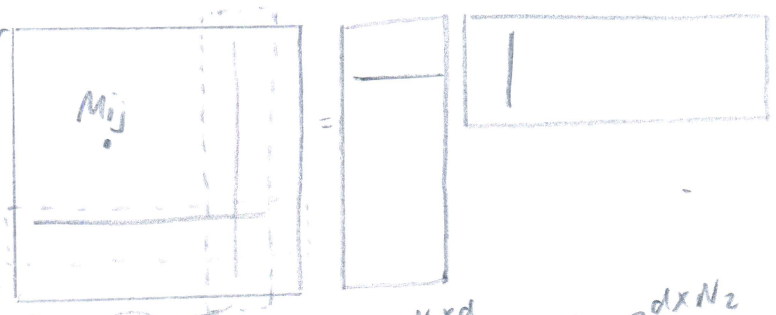
GOAL: Probabilistically fill in missing values

SOLUTION: Hope to have prediction of missing ratings for recommend highly rated obj j amongst pred.

(S7):
· MF - Estimate a low-rank factorisation using observed data, ignoring unobserved.

Low-rank matrix factorisation

1. Account for sparsity in $\underline{M}$
2. Low rank $\Rightarrow d \ll \min\{N_1, N_2\}$
3. Estimate
   - N_1 user-location vectors $\underline{u}_i \in \mathbb{R}^d$ (N_1 users)
   - N_2 object-location vectors $\underline{v}_j \in \mathbb{R}^d$
4. d - no. of factors (latent things considered)



N_2 objs

$\underline{M} \in \mathbb{R}^{N_1 \times N_2}$ $\quad \underline{U} \in \mathbb{R}^{N_1 \times d}$ $\quad \underline{V} \in \mathbb{R}^{d \times N_2}$

Low rank factorisation of $\underline{M}$:

(SVD) MF model maps users, items to a joint latent factor space of dimensionality d, determined here by low rank (rank($\underline{M}$))

- user-item interactions (i.e. ratings) modelled as inner products in that space
- user i, object j associated with $\underline{u}_i \in \mathbb{R}^d$, $\underline{v}_j \in \mathbb{R}^d$ respectively
- elements of $\underline{u}_i$ and $\underline{v}_j$ measure extent of interest user has in items high in those factors / extent to which item possesses those factors

→ $M_{ij} \approx \underline{u}_i^T \underline{v}_j$ $\left(\hat{M}_{ij} = \underline{u}_i^T \underline{v}_j\right)$ (SVD)

- challenge → mapping $i, j \to \underline{u}_i, \underline{v}_j \in \mathbb{R}^d$ (factor-vectors)
- Conventional SVD fails due to sparsity; missing data imputation - comp. expensive

Low-rank factors - intuition:

- sparsity, subset of N_1 users rated Moneyball, another subset 24 (think columns of $\underline{M}$)
- some ratings will overlap
- similar movies (e.g. spy thriller / along any factor) ⇒ rating across columns of $\underline{M}$ highly correlated
- low-rank factorisation enforce this correlation by requiring users and objects to be in a smaller latent embedding space $\mathbb{R}^d$ → restrictions on $\underline{U}$ and $\underline{V}$
- low-rank ⇒ N_1 dim colums of $\underline{M}$ do not 'fill up' $\mathbb{R}^{N_1}$, span region of $\mathbb{R}^{N_1}$ that is smaller (i.e. $\mathbb{R}^d$)
- low-rank restriction / high correlation ⇒ 'borrow' information across movies to learn locations
- same argmts for user-oriented. (rows of $\underline{M}$) (users with similar tastes across movies)

Probabilistic Matrix factorisation: model for estimating low-rank factorisation (setup/gen. model)

Observation/index sets

- define observation set Ω containing observed (i,j)

$\Omega = \{(i,j) : M_{ij} \text{ is measured } (\neq NaN)\}$ $(i,j) \in \Omega$ if measured

- define index sets:
  - Index set of objects rated by user i Ω_{u_i}

$\Omega_{u_i} = \{j : M_{ij} \text{ is measured } (\neq NaN)\}$ $j \in \Omega_{u_i}$ if measured

  - Index set of users who rated object j Ω_{v_j}

$\Omega_{v_j} = \{i : M_{ij} \text{ is measured } (\neq NaN)\}$

PMF (generative model)

- For N_1 users, N_2 objects

$$\underline{u}_i \sim N(0, \lambda^{-1}\underline{\underline{I}}) \quad i = 1, \dots, N_1$$
$$\underline{v}_j \sim N(0, \lambda^{-1}\underline{\underline{I}}) \quad j = 1, \dots, N_2$$

} priors with zero-mean isotropic covariance

- Data distribution: $M_{ij} \sim N(\underline{u}_i^T \underline{v}_j, \sigma^2)$ for each $(i,j) \in \Omega$

Criteria for E-M (discussion): $\underline{M}_o$ - observed $\underline{M}_m$ missing (conclude, EM not necessary)

$$p(\underline{M}_o | \underline{U}, \underline{V}) = \int p(\underline{M}_o, \underline{M}_m | \underline{U}, \underline{V}) \, d\underline{M}_m \rightarrow \text{EM tool for max } p(\underline{M}_o | \underline{U}, \underline{V}) \text{ over } \underline{U}, \underline{V}$$

- Criteria: 1. $p(\underline{M}_o | \underline{U}, \underline{V})$ - diffi to max (marginal likel.) 2. $p(\underline{M}_o, \underline{M}_m | \underline{U}, \underline{V})$ - easy to use (joint likel.)

3. $p(\underline{M}_m | \underline{M}_o, \underline{U}, \underline{V})$ - known

- If $p(\underline{M}_o | \underline{U}, \underline{V})$ no issues for inference → NO (i.e. closed form updates for ML/MAP estimates)
- Try max $p(\underline{M}_o, \underline{U}, \underline{V})$ over $\underline{U}, \underline{V}$ (joint like): $p(\underline{M}_o, \underline{U}, \underline{V}) = \prod_{(i,j)\in\Omega} p(M_{ij} | \underline{u}_i, \underline{v}_j) \times \left[\prod_{i=1}^{N_1} p(\underline{u}_i)\right] \times \left[\prod_{j=1}^{N_2} p(\underline{v}_j)\right]$

PMF (log-joint like. and MAP objective)

$$\underline{U}_{MAP}, \underline{V}_{MAP} = \underset{\underline{U},\underline{V}}{\operatorname{argmax}} \ln p(\underline{M}_o, \underline{U}, \underline{V})$$
$$= \underset{\underline{U},\underline{V}}{\operatorname{argmax}} \sum_{(i,j)\in\Omega} \ln p(M_{ij} | \underline{u}_i, \underline{v}_j) + \sum_{i=1}^{N_1} \ln p(\underline{u}_i) + \sum_{j=1}^{N_2} \ln p(\underline{v}_j)$$

MAP objective:

$$\mathcal{L} = -\sum_{(i,j)\in\Omega} \frac{1}{2\sigma^2} \|M_{ij} - \underline{u}_i^T \underline{v}_j\|_2^2 - \sum_{i=1}^{N_1} \frac{\lambda}{2} \|\underline{u}_i\|_2^2 - \sum_{j=1}^{N_2} \frac{\lambda}{2} \|\underline{v}_j\|_2^2$$

- $\nabla_{\underline{u}_i} \mathcal{L}$ and $\nabla_{\underline{v}_j} \mathcal{L}$ yields solutions for $\underline{u}_i$ and $\underline{v}_j$, but they are dependent on each other → coordinate ascent (K-means / GMM)

PMF (MAP inference co-ordinate ascent): Algorithm:

IN: Incomplete ratings matrix $\underline{M}$, indexed by Ω, rank d

OUT: N_1 user locations, N_2 object locations, $\underline{u}_i, \underline{v}_j \in \mathbb{R}^d$

INIT: Each $\underline{v}_j$ e.g. $\underline{v}_j \sim N(0, \lambda^{-1}\underline{\underline{I}})$ $\forall j = 1, \dots, N_2$

For each iteration do:

For $i = 1, \dots, N_1$ update user location:

$$\underline{u}_i = \left(\lambda\sigma^2 \underline{\underline{I}} + \sum_{j\in\Omega_{u_i}} \underline{v}_j \underline{v}_j^T\right)^{-1} \left(\sum_{j\in\Omega_{u_i}} M_{ij} \underline{v}_j\right)$$

For $j = 1, \dots, N_2$

$$\underline{v}_j = \left(\lambda\sigma^2 \underline{\underline{I}} + \sum_{i\in\Omega_{v_j}} \underline{u}_i \underline{u}_i^T\right)^{-1} \left(\sum_{i\in\Omega_{v_j}} M_{ij} \underline{u}_i\right)$$

PREDICT: that $\hat{M}_{ij} = \hat{\underline{u}}_{i,MAP}^T \hat{\underline{v}}_{j,MAP}$ (i.e. with estimated $\underline{v}_j, \underline{u}_i$ after algo conv.)

PMF: · consider updating for object j factor vector $\underline{v}_j$

Analogies with SL (RR): · "minimise sum of squared error" · $\sum_{(i,j)\in\Omega} \frac{1}{\sigma^2}(M_{ij} - \underline{u}_i^T\underline{v}_j)^2$ with penalty $\lambda\|\underline{v}_j\|_2^2$

- use location ($\underline{u}_i$) is covariate vector for rating M_{ij}
- M_{ij} analogous to SL output we want to predict

· Ridge regression (update) for $\underline{v}_j$: $\underline{v}_j = \left(\lambda\sigma^2\underline{I} + \sum_{(i\in\Omega_{\underline{v}_j})} \underline{u}_i\underline{u}_i^T\right)^{-1}\left(\sum_{i\in\Omega_{\underline{v}_j}} M_{ij}\underline{u}_i\right)$

· Analogy - $N_1 + N_2$ coupled ridge regression problems

· No covariates → we estimate them from the distribution family defining our model

· $\hat{\underline{w}}_{RR} = (\lambda\underline{I} + \underline{X}^T\underline{X})^{-1}\underline{X}^T\underline{y} \iff \underline{v}_j = \left(\lambda\sigma^2\underline{I} + \sum_{(i\in\Omega_{\underline{v}_j})} \underline{u}_i\underline{u}_i^T\right)^{-1}\left(\sum_{i\in\Omega_{\underline{v}_j}} M_{ij}\underline{u}_i\right)$ · By symmetry, holds for $\underline{u}_i$

PMF: · Remove Gaussian priors on $\underline{u}_i$ and $\underline{v}_j$

(Analogies with SL LS) · $\underline{v}_j = \left(\sum_{i\in\Omega_{\underline{v}_j}} \underline{u}_i\underline{u}_i^T\right)^{-1}\left(\sum_{i\in\Omega_{\underline{v}_j}} M_{ij}\underline{u}_i\right) \iff \hat{\underline{w}}_{LS} = (\underline{X}^T\underline{X})^{-1}\underline{X}^T\underline{y}$

· sequence of updates for columns and rows → maximises condit. likeli $p(\underline{M}_o|\underline{U},\underline{V})$

· ML only max conditional likelihood

· MAP accounts for priors, estimate from mode of posterior → maximise joint likelihood $p(\underline{M}_o,\underline{U},\underline{V})$

· Additional restriction: invertibility of $\underline{u}_i\underline{u}_i^T$ and $\underline{v}_j\underline{v}_j^T$ ⇒ all users rated > d objects, all objects rated > d times

LB - Key equations

Topic Models: used in info retrieval - discover themes, annotate documents, organise etc.

Probabilistic topic model: 1. Topics - Estimate probability distributions on words (topic-specific word distributions) on a fixed vocabulary. Each p.d assigns most probability mass to subsets of co-occurring words on a 'theme'

2. Topic proportions - Estimates a probability distribution on topics for each (document-specific topic distrib.) document. Documents contain variable proportions of no. of topics

3) Topic assignment - Assigns every word in a document to a topic.

· 1), 2), 3) must be estimated

Latent Dirichlet Allocation - most well-known topic model

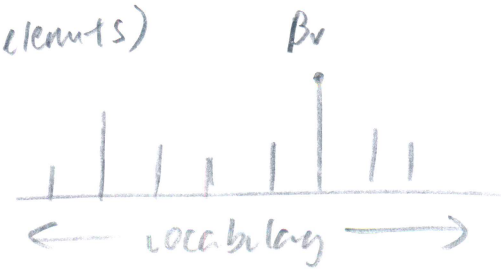
· use of bag-of-words representation for a document (sequential dependencies between words binned)

LDA (ingredients): 1. collection of topics 2. collection of topic proportions for each document

Topics: · Represent, by a probability-distribution on a global, corpus-wide (vocabulary) same set of vocabulary words (V words indexed by $v = 1, \dots, V$)

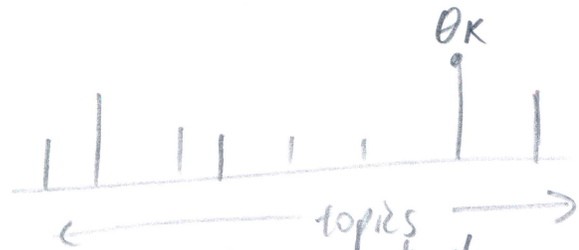
· k^{th} topic, vector out of K topics - $\underline{\beta}_k = [\beta_1, \dots, \beta_V]_k$ (V elements)

· Each element β_v is probability of v^{th} word occurrence in topic k



document/topic proportion:

- Represent a document as a document-specific distribution on topics/signature topic proportion.
- For d^{th} document out of D documents:
  d^{th} topic proportion vector $\underline{\theta}_d$
- $\underline{\theta}_d = [\theta_1, \ldots, \theta_K]_d$ (K-elements)
- Element θ_K is probability of K^{th} topic occurrence in document d.



LDA (Generative process):

1. Generate each topic, a distri on words:- (topic)
   $\beta_K \sim$ Dirichlet(γ) $K = 1, \ldots, K$
2. For each document $d = 1, \ldots, D$, generate distri on topics:- (topic proport.)
   $\underline{\theta}_d \sim$ Dirichlet(α) $d = 1, \ldots, D$
3. For n^{th} word in d^{th} document, (topic assignment) (*) (a)
   a) Allocate word to a topic $c_{dn} \sim$ Discrete($\underline{\theta}_d$)
   b) Generate word from selected topic $x_{dn} \sim$ Discrete($\beta_{c_{dn}}$)

- 1./2. Priors ⇒ topic and topic proportions drawn iid from a Dirichlet distri placed on K topics and D documents
- 3. For d^{th} document, n^{th} word, decide which topic word comes from i.e. a topic indicator from a discrete distri. (multinomial) using distribution on topics for d^{th} document ($\underline{\theta}_d$) [i.e. $\underline{\theta}_d$ acts as a parameter of discrete distri.)

$c_{dn} = \{1, \ldots, K\}$

topic chosen ⇒ at data level, n^{th} word in d^{th} document is generated from a discrete distri using topic selected by topic indicator c_{dn}.

(*): Do not KNOW c_{dn} (topic indicators), $\underline{\theta}_d$ (topic proportions), β_K (topics)

Dirichlet distribution: A continuous distribution over discrete probability vectors

- β_K - probability vector
- $\underline{\gamma}$ - positive parameter vector
  $\underline{\gamma} \in \mathbb{R}^V$ $\beta_K \in \mathbb{R}^V$

$$p(\beta_K | \underline{\gamma}) = \frac{\Gamma\left(\sum_{v=1}^{V} \gamma_v\right)}{\prod_{v=1}^{V} \Gamma(\gamma_v)} \prod_{v=1}^{V} \beta_{K,v}^{\gamma_v - 1}$$

$V \geq 2$ (no. of words)
γ_v - concentration param.

shared indexes 1 dimensionality

concentration param (γ):

→ As K-topics, there will be K draws from Dirichlet distri
- γ controls the extent to which probability mass is concentrated across each element $\beta_{K,v}$ for a particular β_K vector.

- $\gamma \to \infty$: each element $\beta_{k,v}$ of topic vector $\underline{\beta}_k$ becomes uniform for all K topics
- $\gamma \to 0$: probability mass concentrated on subset of dimensions V ($\underline{\beta}_k$ element sparsity)
- $\gamma < 1$: A priori belief that topics place probability mass on small subset of total words in vocab (V).

- Prior distribution on topic proportions parametrised by $\alpha(k) < 1 \Rightarrow$ a priori, document only uses subset of topics available

Inference algorithm: variational inference

LDA outputs: 1) A set of K topic specific word distributions $\{\hat{\underline{\beta}}_1, \dots, \hat{\underline{\beta}}_K\} = \hat{\underline{\beta}}$
2) A set of D document-specific topic distributions $\{\hat{\underline{\theta}}_1, \dots, \hat{\underline{\theta}}_D\}$

LDA and MF: Q) For a document, what is $p(x_{dn} = i \mid \underline{\beta}, \underline{\theta}_d)$ (probability of n^{th} word in d^{th} doc equal to in vocab list, given set of topics and topic proportion for d^{th} doc?)

A) - Marginal distribution on x_{dn} with c_{dn} integrated out:-

$$p(x_{dn} = i \mid \underline{\beta}, \underline{\theta}_d) = \sum_{k=1}^{K} p(x_{dn} = i, c_{dn} = k \mid \underline{\beta}, \underline{\theta}_d) = \sum_{k=1}^{K} p(x_{dn} = i \mid \underline{\beta}, c_{dn} = k)\, p(c_{dn} = k \mid \underline{\theta}_d)$$

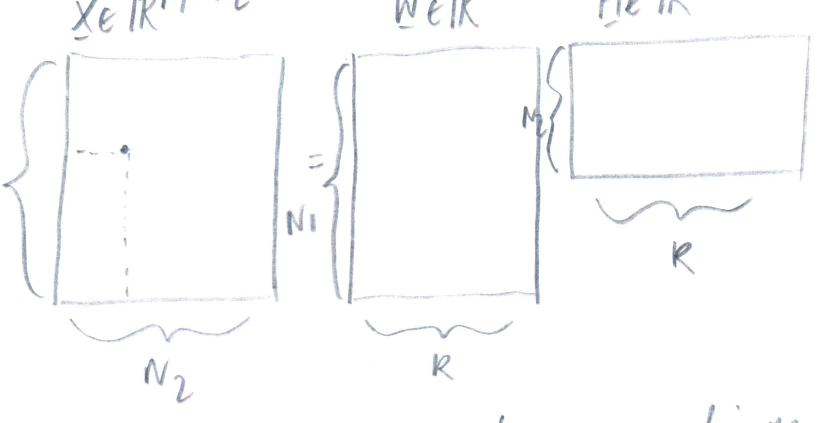
(β_{ki} under the first factor; θ_{dk} under the second)

- Let $B = [\underline{\beta}_1, \dots, \underline{\beta}_K]$, $B \in \mathbb{R}^{V \times K}$ $\Theta = [\underline{\theta}_1, \dots, \underline{\theta}_D]$ $\Theta \in \mathbb{R}^{K \times D}$
- $p(x_{dn} = i \mid \beta, \theta) = (B\Theta)_{id}$ = product of 2 matrices with non-negative entries
- LDA is an example of NMF

Non-negative matrix factorisation:

- Data $\underline{X}$ - some missing $(\underline{X})_{ij}$
- $X_{ij} \geq 0 \ \forall i, j$
- Factors $\to W_{ik}, H_{kj} \geq 0 \ \forall k, j, i$

(ish): $X_{ij} \approx \sum_{k=1}^{K} W_{ik} H_{kj}$



NMF application domains:

1) Text data - X_{ij} word term freq (count word i in doc j)
$\underline{X}_{:,j}$ - ~~word~~ histogram word term freq in doc j
$\underline{X}_{i,:}$ - histo of word i occurrences

2) Images - vectorise N×M image into $X_{:,j}$ (colum)

3) Discrete grouped data - Quantise continuous set of K means (compress info → prototype) (cluster medoids)
- X_{ij} - count on times group j uses cluster i or cluster i in group j e.g. indexing ft, j index songs

2 NMF objectives: NMF #1: $\|\underline{X} - \underline{W}\underline{H}\|_2^2 = \sum_i \sum_j (X_{ij} - (\underline{W}\underline{H})_{ij})^2$ (squared error)

NMF #2: $D(X \| WH) = -\sum_i \sum_j \left[X_{ij} \ln (\underline{W}\underline{H})_{ij} - (\underline{W}\underline{H})_{ij} \right]$ (KL-diverg.)

s.t. $W_{ik}, H_{kj} \geq 0 \ \forall i, k, j$

NMF Algorithm · works to minimise objective $\min_n F(h)$ by generating seq $F(h^1) \geq F(h^2)$...
and ensuring it converges to a local minimum.

· Minimise via an auxiliary function → multiplicative update for W, H

NMF #1 (multiplicative update for $\|X - WH\|_2^2$) Algorithm:

Minimise $\min_{W,H} \sum_{ij} (X_{ij} - (WH)_{ij})^2$ s.t. $W_{ik} \geq 0$ $H_{kj} \geq 0$

· Randomly init H, W with non-negative values
· Iterate first for all values in H, then all in W:-

$$H_{kj} \leftarrow H_{kj} \frac{(W^T X)_{kj}}{(W^T WH)_{kj}}$$

$$W_{ik} \leftarrow W_{ik} \frac{(XH^T)_{ik}}{(WHH^T)_{ik}}$$

· until change in $\|X - WH\|_2^2$ is small

NMF #2 (multiplicative update for $D(X\|WH)$) Algorithm

Minimise $\min_{W,H} \sum_{ij} \left[X_{ij} \ln \frac{1}{(WH)_{ij}} + (WH)_{ij} \right]$ s.t. $W_{ik} \geq 0$ $H_{kj} \geq 0$

Randomly initialise W and H with non-negative values
· Iterate first for all values in H, then all in W:-

$$H_{kj} \leftarrow H_{kj} \left(\frac{\sum_i W_{ik} X_{ij} / (WH)_{ij}}{\sum_i W_{ik}} \right)$$

$$W_{ik} \leftarrow W_{ik} \left(\frac{\sum_j H_{kj} X_{ij} / (WH)_{ij}}{\sum_j H_{kj}} \right)$$

until change in $D(X\|WH)$ is small

NMF #1 & #2 through Prob. lens · NMF #1 - squared error penalty $\Rightarrow X_{ij} \overset{iid}{\sim} N(\sum_k W_{ik} H_{kj}, \sigma^2)$ ($X_{ij} \geq 0 \Rightarrow$ not true') s.t. non-negating

· NMF #2 - divergence penalty $\Rightarrow X_{ij} \overset{iid}{\sim} Po((WH)_{ij})$ $Po(x|\lambda) = \frac{\lambda^x}{x!} e^{-\lambda}$

#2 - NMF → Estimating expected value for data matrix $\mathbb{E}[X]$
under generative assumption that likelihood is poisson, param by $(i,j)^{th}$ element of product of NMF factors $(WH)_{ij}$

LDA-NMF: 1. Form tem freq matrix X (X_{ij} - # times word i appears in doc j)
2. Run NMF for estimates $\underline{W}, \underline{H}$
3. For $k=1,\dots,K$:-
i) Set $a_k = \sum_i W_{ik}$
ii) Divide W_{ik} by a_k for all i (k^{th} column of $\underline{W}$)
iii) Multiply H_{kj} by a_k for all j (k^{th} row of $\underline{H}$)
} 'post processing'

(NTF): Each column vector $\underline{w}_k$ of $\underline{W}$ (after post-processing) → p.d. on words for topic k (i.e. topic)
· Each column vector $\underline{h}_j$ of H (— " —) → p.d. on topics for document j

Application (Face modelling): · Put face images along columns of data matrix $\underline{X}$
$\underline{X} \approx \underline{W}\underline{H}$; $X_{ij} \approx \sum_{k=1}^{K} W_{ik} H_{kj}$
- K columns of $\underline{W}$ - basis images
- Columns of H - encodings (1-1 corres with a face $X_{:,j}$)
- encoding-coefficient where face is a linear comb. of basis images

K-means (Face modelling): - Face (column of $\underline{X}$) represented by prototypical basis image (column of $\underline{W}$) nod.
- Columns of H (binary) (in cluster assign.)

SVD: (Face modelling): · Face (column of $\underline{X}$) represented by an eigenface basis image (colum of $\underline{W}$) and a non-sparse ratings $\underline{m}$. of encodings $\underline{H}$
· Face is a linear combo of eigenfaces
· Difficult to interpret

NMF: Face (column of $\underline{X}$) represented by interpretable parts based localised features as single basis image (col. of $\underline{W}$) col sparse encodgs mat. H
NMF ⇒ non-negativity of $\underline{W}$ and $\underline{H}$
· Face is linear combo of parts/localised features

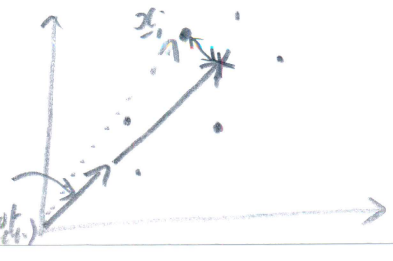
· Topic modelling - Each col. of $\underline{X}$ (doc j) as linear combination of topics

L19 - Key equations

Dimensionality reduction: · High dimensional data $\underline{x}_i \in \mathbb{R}^d$, where 'info' does not use full d dimensions

Principal components Analysis: A way of automatically mapping data $\underline{x}_i$ into a low-dimensional co-ordinate system
1. captures most info in data → few dimensions 2. Extensions allow us to handle missing data and unwrap data

PCA (2D→1D intuition) (minimum error form): · Approximate $\underline{x}_i \in \mathbb{R}^2$ via unit length vector $\underline{q}$ $(*)(q)$ - projecting vector onto a scalar
· unit length $\underline{q}$: $\underline{q}^T\underline{q} = 1$
· X (length) - $(\underline{q}^T\underline{x}_i)$ after projecting $\underline{x}$ onto line defined by $\underline{q}$
· $X - (\underline{q}^T\underline{x}_i)\underline{q}$ - takes vector $\underline{q}$ and stretches it to X (scalar multiple)



PCA (2D → 1D): selecting q : minimise squared approximation error

(min. error form) $q = \underset{q}{\operatorname{argmin}} \sum_{i=1}^{N} \|x_i - qq^T x_i\|_2^2$ s.t. $q^T q = 1$ · $qq^T x_i = (q^T x_i) q$

- approximation of x_i by stretching q to X.

$= \underset{q}{\operatorname{argmin}} \sum_{i=1}^{N} x_i^T x_i - q^T \left(\sum_{i=1}^{N} x_i x_i^T \right) q$

PCA (2D → 1D) (eigenvector, eigenvalue, maximise data cov.)

- defining $X = [x_1 \dots x_N]$ so $X \in \mathbb{R}^{d \times N}$ (so in a different way to previous stacking)

$\Leftrightarrow \underset{q}{\operatorname{argmax}}\ q^T (XX^T) q$ s.t. $q^T q = 1$ · eigendecomposition problem q - 1st eigenvec. of XX^T

λ - $q(XX^T)q$ - 1st eigenvalue

(*) · Note this is analagous to maximising projected variance $q(XX^T)q$

PCA (General): consider K eigenvectors:

$q = \underset{q}{\operatorname{argmin}} \sum_{i=1}^{N} \|x_i - \sum_{k=1}^{K} (x_i^T q_k) q_k\|_2^2$ s.t. $q_k^T q_{k'} = \begin{cases} 1 & k = k' \\ 0 & k \neq k' \end{cases}$ (XX^T)

(*) - links to eigenvalues as optimisation images229

$= \underset{q}{\operatorname{argmin}} \sum_{i=1}^{N} x_i^T x_i - \sum_{k=1}^{K} q_k^T \left(\sum_{i=1}^{N} x_i x_i^T \right) q_k$

K vectors $Q = [q_1 \dots q_K]$ define a K dimensional sub-space with which to represent data

- $x_{proj} = \begin{bmatrix} q_1^T x \\ \vdots \\ q_K^T x \end{bmatrix}$ (scalars that eventually stretch q_k by magnitude/ projection of x onto line defined by q_k (see eqn 1.C)

(07) reconstruction/ approx: $x \approx \sum_{k=1}^{K} (q_k^T x) q_k = Q x_{proj}$

(*) call built-in functions to find K largest eigenvectors and eigenvalues

PCA (Eigenvalue/eigenvectors / SVD)

- Reformulate problem as:- Find (λ, q): $(XX^T) q = \lambda q$
- Since $(XX)^T$ is P.S.D. $\in S_+^n$ $r \leq \min\{d, N\}$ pairs $\Rightarrow \exists \lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_r > 0$ (as covariance matrix)

$q_k^T q_k = 1$ $q_k^T q_{k'} = 0$

Justifying $XX^T \in S_+^n$ via SVD:

X has SVD → USV^T with $U \in \mathbb{R}^{d \times d}$ $S \in \mathbb{R}^{d \times n}$ $V^T \in \mathbb{R}^{n \times n}$ (Full-SVD) $s_{ii} \geq 0$ (sqrt)

$X \in \mathbb{R}^{d \times n}$ (orthog.) $\Rightarrow u_{:,i}^T u_{s,j} = 0$ $\|u_{:,i}\|^2 = 1$ $\forall i, j, i \neq j \Rightarrow U^T U = UU^T = I$

$(XX^T) = USV^T VS^T U^T = US^2 U^T$ with $Q = U$ $\lambda_i = (s^2)_{ii} \geq 0$ $\Leftrightarrow \boxed{(XX^T)U = US^2}$

PCA (pre-processing):- PCA sensitive to scaling

- mean centering - subtract mean from every dimension of x_i

- Ensures every principal component passes through origin

PCA - digits example: · 16×16 handwritten 3s $x_i \in \mathbb{R}^{256}$; stack $X \in \mathbb{R}^{256 \times N}$ (images)

· $XX^T \in \mathbb{R}^{256 \times 256}$ and demean

(reconstr.) $\sum_{k=1}^{K} (q_k \underline{x}_{i,proj}) \underline{q}_k$ ... of eigenvectors / principal components used to determine degree of compression

$= \text{mean} + \underline{Q}\underline{x}_{proj}$ with $(K-(M-1))$ dimensions of $\underline{Q}$ and $\underline{x}_{proj}$ switched off

Probabilistic PCA:

- PCA, SVD → probabilistic model on PCA, estimate param via EM (with gen model
- Allows 1. Missing data 2. Additional param e.g noise 3. Modularity + point est.)
  4. Distributions → characterise uncertainty in precision

Probabilistic PCA (setup): · Analogous to matrix factorisation, with gen process and distr exp. defined.

(or) Approximate $\underline{X} \approx \underline{W}\underline{Z}$ $X \in \mathbb{R}^{d \times n}$ = { $\underline{W} \in \mathbb{R}^{d \times K}$ - factor loadings (similar to eigenvects with no orthonorm, unit legth constr.); $\underline{Z} \in \mathbb{R}^{K \times n}$ - ith column of $\underline{Z}$, $\underline{z}_{:,i} = \underline{z}_i \in \mathbb{R}^K$, a low-dim represent. of $\underline{x}_i$ }

(☆) better pca ellende (Bish), papers

PCA: $\underline{x}_i \sim N(\underline{W}\underline{z}_i, \sigma^2 \underline{I})$

(gen. process) $\underline{z}_i \sim N(0, \underline{I})$

Do not know $\underline{W}$, $\underline{z}_i$ - NO orthogonality and unit legth constraints

PCA:

(Intractability of marginal likelihood)

$\underline{W}_{ML} = \underset{\underline{W}}{\text{argmax}} \ln p(\underline{x}_1, \ldots, \underline{x}_N | \underline{W}) = \underset{\underline{W}}{\text{argmax}} \sum_{i=1}^{N} \ln p(\underline{x}_i | \underline{W})$ (not consistent with Bishop, lit.) (☆)

· Intractable as $p(\underline{x}_i | \underline{W}) = N(\underline{x}_i | 0, \sigma^2 \underline{I} + \underline{W}\underline{W}^T)$ (log-sum eventually ??) and cannot find gradient wrt $\underline{W}$ (??)

CA · maginal log-likelihood:-

nsetup:)

$\ln p(\underline{x}_i | \underline{W}) = \sum_{i=1}^{N} \ln \int p(\underline{x}_i, \underline{z}_i | \underline{W}) d\underline{z}_i = \sum_{i=1}^{N} \int q(\underline{z}_i) \ln \frac{p(\underline{x}_i, \underline{z}_i | \underline{W})}{q(\underline{z}_i)} d\underline{z}_i + \sum_{i=1}^{N} \int q(\underline{z}_i) \ln \frac{q(\underline{z}_i)}{p(\underline{z}_i | \underline{x}_i, \underline{W})} d\underline{z}_i$

t $q(\underline{z}_i) = p(\underline{z}_i | \underline{x}_i, \underline{W})$ $\forall i = 1, \ldots, N$ } requires calc. of $p(\underline{z}_i | \underline{x}_i, \underline{W})$ and tractable max. of $\mathcal{L}$

aximise $\mathcal{L}$ wrt $\underline{W}$

for PPCA (Algorithm):

$\{\underline{x}_i\}_{i=1}^N$, $\underline{x}_i \in \mathbb{R}^d$, model $\underline{x}_i \sim N(\underline{W}\underline{z}_i, \sigma^2 \underline{I})$ $\underline{z}_i \sim N(0, \underline{I})$ $\underline{z}_i \in \mathbb{R}^K$

tep: For all $i = 1, \ldots, N$ set $q(\underline{z}_i) = p(\underline{z}_i | \underline{x}_i, \underline{W}) = N(\underline{x}_i | \underline{\mu}_i, \underline{\Sigma}_i)$ where

$\underline{\Sigma}_i = \left((\underline{I} + \underline{W}\underline{W}^T)/\sigma^2 \right)^{-1}$ $\underline{\mu}_i = \underline{\Sigma}_i \underline{W}^T \underline{x}_i / \sigma^2$

tep: update $\underline{W}$ ($D \times K$) by maximising $\mathcal{L}$ from E-step ($K \times K$):

$\underline{W} = \left[\sum_{i=1}^{N} \underline{x}_i \underline{\mu}_i^T \right] \left[\sigma^2 \underline{I} + \sum_{i=1}^{N} (\underline{\mu}_i \underline{\mu}_i^T + \underline{\Sigma}_i) \right]^{-1}$

Iterate until increase in $\sum_{i=1}^{N} \ln p(\underline{x}_i | \underline{W})$ (marg likel) is small

Point estimate ~~(and K, σ)~~ $\hat{\underline{W}}$ (and $\hat{K}, \hat{\sigma}^2$), conditional posterior $p(\underline{z}_i | \underline{x}_i, \underline{W})$ on each $\underline{z}_i$

- Conditional posterior $p(z_i|x_i, W)$ captures belief about what low-dim embedding should be; but also a distribution with uncertainty

PPCA Applications: #1 - image processing, 8×8 overlapping patch in image, vectorise → $x_i \in \mathbb{R}^{64}$

- Then stack into columns of X: $X \in \mathbb{R}^{256 \times 262,144}$
- Factor model: Approx. $x_i \approx W\mu_i$ where μ_i is posterior mean of z_i ($z_{:,i}$ - ith column of Z)
- Reconstruct image by replacing x_i with $W\mu_i$ (and averaging) ⊖ (*)
- Noise from denoised patches → captured in noise variance param σ^2 #2 denoising

$x_i \sim N(Wz_i, \sigma^2 I)$; σ^2 estimated

ass' denoised mean ~ noise variance soaks up noise

- #3 - Missing data

Kernel PCA

- dot products, generalise with non-linear kernel
- map to high-dim feature space, then execute dimensionality reduction
- PCA → find eigenvalues, eigenvectors of $XX^T = \sum_{i=1}^{N} x_i x_i^T$ → due to definition of $X = \begin{bmatrix} | & & | \\ x_1 & \dots & x_N \\ | & & | \end{bmatrix}$
- Define $\phi(x)$ as feature mapping $\phi: \mathbb{R}^d \to \mathbb{R}^D$
- Solve the eigendecomposition: $-\left[\sum_{i=1}^{N} \phi(x_i)\phi(x_i)^T\right] q_k = \lambda q_k \quad \forall k = 1, \dots, K$ without using $\phi(\cdot)$ and $q_k \in \mathbb{R}^D$

(see derivations) $\Rightarrow q_k = \sum_{i=1}^{N} a_{ki}\phi(x_i)$ where $a_{ki} = \frac{\phi(x_i)^T q_k}{\lambda_k}$ $a_k = \begin{bmatrix} a_{k1} \\ \vdots \\ a_{kN} \end{bmatrix}$

KPCA: Perform PCA on kernel matrix K rather than XX^T -

$Ka_k = \lambda_k a_k$ where K is kernel matrix constructed on data with $K \in \mathbb{R}^{N \times N}$, S_+^N (i.e. rewrite earlier eigendecomposition in terms of a_k, K)

Kernel PCA (Algorithm):

IN: $\{x_i\}_{i=1}^N$ $x_i \in \mathbb{R}^d$, $K(x_i, x_j)$ - kernel fn

- construct K, kernel matrix on data eg $K_{ij} = a\exp\left\{-\frac{1}{b}\|x_i - x_j\|_2^2\right\}$
- Solve: eigendecomposition

$Ka_k = \lambda a_k$ for first $r \ll n$ eigenvalue/vector pairs $(\lambda_1, a_1), \dots (\lambda_r, a_r)$

OUT: new co-ordinate system for x_i by implicitly mapping $\phi(x_i)$ and projecting $(q_k^T \phi(x_i))$

$x_i \xrightarrow{\text{proj}} \begin{bmatrix} \lambda_1 a_{1i} \\ \vdots \\ \lambda_r a_{ri} \end{bmatrix}$ with a_{ki} - i^{th} dimension of k^{th} eigenvector a_k