COMP 652 - Machine Learning
by Carlos G. Oliver, page 1 of 2

1 Math

Bayes rule: $P(A|B)=\frac{P(A,B)}{P(B)}=\frac{P(B|A)P(A)}{P(B)}$ $P(A,B|C)=\frac{P(A,B,C)}{P(C)}$

Cond indep: $P(A,B|C)=P(A|C)P(B|C)$

Chain rule: $P(A_n,\dots,A_1) = P(A_n|A_{n-1},\dots,A_1)\cdot P(A_{n-1},\dots,A_1)$

Joint: $\mathrm{P}\left(\bigcap_{k=1}^n A_k\right) = \prod_{k=1}^n \mathrm{P}\left(A_k \,\middle|\, \bigcap_{j=1}^{k-1} A_j\right)$

Posterior: unobserved θ, observed x: $p(\theta|x)=\frac{p(x|\theta)p(\theta)}{p(x)}$

Prior: prior belief in dist. of θ: $p(\theta)$

Marginal: $\Pr(X=x)=\sum_y \Pr(X=x,Y=y)=\sum_y \Pr(X=x\mid Y=y)\Pr(Y=y)$

Likelihood: prob of observed given parameters: $p(x|\theta)$

Covariance: $Cov(X,Y)=E\{(X-E(X))(Y-E(Y))\}$

MAP: $h_{MAP}=\arg\max_{h\in H} P(h|D) = \arg\max_{h\in H} P(D|h)P(h)$

Lagrange: $L(\mathbf{x},\lambda)=f(\mathbf{x})+\lambda g(\mathbf{x})$ Set constraint $g(\mathbf{x})$ to zero and add multiplier.

Multinomial: $(x_1+x_2+\cdots+x_m)^n = \sum_{k_1+k_2+\cdots+k_m=n}\binom{n}{k_1,k_2,\ldots,k_m}\prod_{t=1}^m x_t^{k_t}$ $(a+b+c)^3=a^3+b^3+c^3+3a^2b+3a^2c+3b^2a+3b^2c+3c^2a+3c^2b+6abc.$

Entropy: $H(X)=-\sum_{i=1}^n p(x_i)\log p(x_i)$

Cond. entropy: $H(Y|X) = \sum_{x\in X} p(x)H(Y|X=x) = -\sum_{x\in X} p(x)\sum_{y\in Y} p(y|x)\log p(y|x) = \sum_{x\in X,y\in Y} p(x,y)\log\frac{p(x)}{p(x,y)}$ $H(Y|X) = H(X|Y)-H(X)+H(Y)$ $H(X_1,\ldots X_n) = \sum_i^n H(X_i|X_1,\ldots X_{n-1})$ **KL Divergence:** $D_{KL}=\sum_x P(x)\log\frac{P(x)}{Q(x)}$

Mutual info: $I(X;Y) = \sum_{y\in Y}\sum_{x\in X} p(x,y)\log\frac{p(x,y)}{p(x)p(y)}$ $I(X;Y)=H(X)-H(X|Y)=H(X,Y)-H(X|Y)-H(Y|X)$

$$\mathbf{x}\mathbf{y}^{\mathrm{T}} = \begin{bmatrix}x_1\\ \vdots \\ x_m\end{bmatrix}[x_1\ldots x_n] = \begin{bmatrix}x_1y_1 & \cdots & x_1y_n\\ \vdots & \ddots & \vdots\\ x_my_1 & \cdots & x_my_n\end{bmatrix}$$

Matrix-vector product: $\mathbf{A}\mathbf{x}=\begin{bmatrix}a_1^T\mathbf{x}\\ \vdots\\ a_m^T\mathbf{x}\end{bmatrix}$

Matrix mult': $(\mathbf{AB})_{ij}=\sum_{k=1}^m A_{ik}B_{kj}$

$$\mathbf{AB}=\begin{bmatrix}a_1^Tb_1 & a_1^Tb_2 & \dots & a_1^Tb_p\\ \vdots & \vdots & \ddots & \vdots\\ a_m^Tb_1 & a_m^Tb_2 & \dots & a_m^Tb_n\end{bmatrix}$$

Multip'n dimensions: $(n\times p)(p\times m)=n\times p$

$\sin(x)'=\cos x$

$\cos(x)'=-\sin x$

$\sigma(x)'=\sigma(x)(1-\sigma(x))$ $(AB)^T=B^TA$

$\log_b(x^y)=y\log_b(x)$

Unif: $P_{\theta_1,\theta_2}(x)=\frac{1}{\theta_2-\theta_1}$

$AA^{-1}=A^{-1}A=I$ for square matrices.

Chain rule: $F'(x)=f'(g(x))g'(x)$

Product rule: $(f\cdot g)'=f'\cdot g+f\cdot g'$

Norm: $\|\mathbf{x}\|_d=\sum_i x_i^d$

Bias vs Variance: $\mathrm{E}\Big[\big(y-\hat{f}(x)\big)^2\Big] = \mathrm{E}\Big[\hat{f}(x)-f(x)\Big]^2+\mathrm{E}[\hat{f}(x)^2]-\mathrm{E}[\hat{f}(x)]^2+\epsilon^2$

2 Regression

2.1 Regularization

Higher λ more bias. With more data, variance decreases can afford weaker regularization (less bias).

2.2 L2 regularization

Weights do not reach zero. Faster.

$$J_w=\frac{1}{2}(\Phi\mathbf{w}-\mathbf{y})^T(\Phi\mathbf{w}-\mathbf{y})+\frac{\lambda}{2}\mathbf{w}^T\mathbf{w}$$

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

2.3 L1 regularization

Some weights set to zero. More expensive.

2.4 Gradient Descent

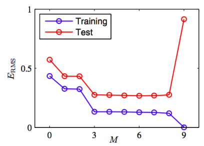
$\mathbf{w}\leftarrow\mathbf{w}-\alpha\nabla\log L(\mathbf{w})$ $\mathbf{w}\leftarrow\mathbf{w}-\alpha\nabla\log J(\mathbf{w})$

2.5 Logistic Regression

$\sigma(t)=\frac{e^t}{e^t+1}=\frac{1}{1+e^{-t}}$ $L(\mathbf{w}) = -\frac{1}{N}\sum_{n=1}^N\Big[y_n\log\hat{y}_n+(1-y_n)\log(1-\hat{y}_n)\Big]$

2.6 Bayesian regularization

Can sample hypotheses from prior distribution on parameters. When get data, posterior changes where we draw hypotheses from. No need for regularization. Use sequential bayesian updating to get new prior on parameters given data. $new\ prior \propto current prior\times new likelihood$ i.e. $\pi_{n+1}\propto\pi_n\times p(x_{n+1}|\theta_n,x_n)$



3 Kernels

$k(\mathbf{x},\mathbf{z})=\phi(\mathbf{x})\cdot\phi(\mathbf{z})$ **Mercer theorem:** $K(\mathbf{x},\mathbf{z})$ is kernel iff Gram matrix $\mathbf{K}$ symmetric and positive semidefinite: $\mathbf{K_{ij}}=\mathbf{K_{ji}}$ and $\mathbf{z}^T\mathbf{K}\mathbf{z}\geq 0$ $\mathbf{K}=\Phi\Phi^T$ where $\mathbf{K_{nm}}=\phi(\mathbf{x_n})^T\phi(\mathbf{x_m})=k(\mathbf{x_n},\mathbf{x_m})$ Gram matrix size of input. Try to find a feature map whose dot product yields the kernel.

4 SVM

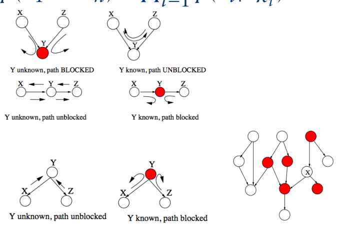
Minimize absolute error. More robust to outliers. Max margin is convex optimization. $h_{\mathbf{w}}(\mathbf{x})=\mathrm{sign}(\sum_{i=1}^m\alpha_iy_i(\mathbf{x_i}\mathbf{x})+\mathbf{w_0})$ $\alpha_i>0$ only for support vectors. Soft error SVM: $0<\zeta\leq 1$ if inside margin. $\zeta>1$ if misclassified. Total errors: $C\sum_i\zeta$. Large C higher variance.

5 EM/Active Learning/Missing Data

When missing data: many local maxima (normally likelihood has unique max), no closed form solutions. So we do gradient ascent or EM. $\log L(\theta)=\sum_{\text{complete data}}\log P(\mathbf{x_i},y_i|\theta)+\sum_{\text{incomplete data}}\log\sum_y(\mathbf{x_i}|\theta)$. **E step:** compute expected assignment (hard or soft, in soft we compute the weight for each distribution accounting for each point.) of points to distributions (estimate $p(y_i=k|\theta)$). **M step:** recompute parameters to maximize likelihood of current assignments $p(\theta|y_i)$. Good for low dimensionality data. **Active learning sampling strategies:** 1. generate examples for oracle. 2. query if instance in region of uncertainty (costly to maintain region) 3. uncertainty sampling. 4. query by committee (set of hypotheses vote. take examples for which KL divergence between distributions predicted by each hypothesis is high). 5. Expected error reduction/ max info gain (consider impact of labelling $\mathbf{x}$ with all labels, measure impact on other examples. 6. Density based (queries far from major concentration of data less useful)

6 Bayes Nets

$p(x_1,\ldots,x_n)=\prod_{i-1}^n p(x_i|x_{\pi_i})$



Markov blanket: parents, children, spouses.

Moral graph: graph U: edge $(X,Y)\in U$ if X in Y's markov blanket.

Belief propagation: This is exact inference $m_{ji} = \sum_{x_j}\left(\psi^E(x_j)\psi(x_i,x_j)\prod_{k\in\text{nghbr}(x_j)}m_{kj}(x_j)\right)$ $p(y|\hat{x}_E)\propto\psi^E(y)\prod_{k\in\text{nghbr}(Y)}m_{ky}(y)$

Gibbs Sampling (approximate inference): 1. set evidence nodes $E=e$, all others random. 2. Sample x_i' from $P(X_i|x_1,..,x_n)$ i.e. markov blanket. 3. Obtain new $x_1'..x_n'$. Converges to true steady state distribution using makov chain properties.

Learning: likelihood of whole graph decomposes to likelihood over each node's parameter $L(\theta|D)=\prod_{i\in\text{nodes}}^n L(\theta_i|D)$. Use EM.

7 Markov Chains

Markov property: $P(s_{t+1}|s_t) = P(s_{t+1}|s_0,..s_t)$ $P(s_{t+1=s'})=\sum_s P(s_0=s)P(s_1=s'|s_0=s)$ in matrix form: $\mathbf{p_t} = vekT^T\mathbf{p_{t-1}} = (\mathbf{T}^T)^t\mathbf{p_0}$

8 Hidden Markov Models

Parameters: states, observations: $\mathcal{S},\mathcal{O}$, $\mathbf{b_0}\in|\mathcal{S}|$, transition probs $\mathbf{T}\in|\mathcal{S}|\times|\mathcal{S}|$, emission probs $\mathbf{Q}\in|\mathcal{S}|\times|\mathcal{O}|$ $P(o_1,..,o_T,s_1,..,s_T) = P(s_1)P(o_1|s_1)\prod_{t=2}^T P(s_t|s_{t-1})P(o_t|s_t)$

Forward alg: Compute $P(o_{1:t},S_t=s,)$ $\alpha_t(S_t)=P(o_1,..,o_t,S_t=s)=\sum_{s_{t-1}}P(o_t|s_t)P(s_t|s_{t-1})\alpha_{t-1}(s_{t-1}),\alpha_t(s_1)=P(o_1,s_1)=P(s_1)p(o_1|s_1)$

Backward alg: obtain $\beta_t(s_t) = p(o_{t+1:n}|s_t)$. $\beta_t(s_t) = \sum_{s_{t+1}}\beta_{t+1}(s_{t+1})P(o_{t+1}|s_{t+1})P(s_{t+1}|s_t)$

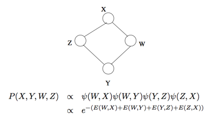
F-B alg: 1. compute $\alpha_t(s)$. 2. compute $\beta_t(s)$ 3. for an s and t: $P(S_t|o_1,\ldots,o_T)=\frac{P(o_1,..o_T,S_t=s)P(o_{t+1},\ldots,o_T|S_t=s)}{P(o_1,\ldots,o_T)}=\frac{\alpha_t(s)\beta_t(s)}{\sum_{s'}\alpha_T(s')}$ Complexity: $O(|S|T)$

Baum-Welch: EM for missing parameters. Given obs. and initial parameters $\lambda=(\beta_0(s),p_{ss'},q_{so})$. 1. (E-step) Compute $P(S_t|o_1,\ldots o_T)\forall s,t$, $P(S_t=s,S_{t+1}=s'|o_1,\ldots,o_T)\forall s,s',t$ (using F-B, $O(|S|T+|S|^2T)$). 2. (M-step) $b_0(s)=P(S_1=s|o_1,..,o_T)$, $p_{ss'}=\frac{\text{expected \# of s to s'}}{\text{expected s occurences}} = \frac{\sum_{t<T}P(S_t=s,S_{t+1}=s'|o_1,\ldots,o_T)}{\sum_{t<T}P(S_t=s|o_1,\ldots,o_T)}$, $q_{so}=\frac{\text{expected \# of o from s}}{\text{expected s occurences}}=\frac{\sum_{t:o_t=o}P(S_t=s|o_1,\ldots,o_T)}{\sum_t P(S_t=s|o_1,\ldots,o_T)}$, $O(|S|^2T+|S||O||T|)$

9 Undirected Graphical Models

$X\perp\!\!\!\perp Z|Y$ if every path from X to Z goes through Y. Capture correlations, not causality. Can't always go from bayes to undirected and back. If two nodes not connected by arc, they are conditionally independent given rest of graph. Express joint as product of maximal clique potentials: $p(X_1=x_1,\ldots,X_n=x_n)=\frac{1}{Z}\prod_{\text{cliques}C}\psi_C(\mathbf{x_C})$ where $\mathbf{x_C}$ is the values if nodes in C, and $Z=\sum_x\prod_C\psi_C(\mathbf{x_C})$. $\psi_C(\mathbf{x_C})=e^{-H_C(\mathbf{x_C})}$ We define H to be anything. $p(\mathbf{x})=Z^{-1}\prod_C e^{-H_C(\mathbf{x_C})}=Z^{-1}e^{-\sum_C H_C(\mathbf{x_C})}=Z^{-1}e^{-H(\mathbf{x})}$ where H_C is the energy of the clique. For a 2D spin glass: $H(\mathbf{x})=\sum_{i,j}\beta_{ij}x_ix_j+\sum_i\alpha_ix_i$ can do belief propagation like in bayes net with the messages. Order of updates is important. Potentials energy of agreement or disagreement in clique.

Parameter Learning: Because of normalization learning can't be broken down, can use gradient based. Max likelihood: $\log L(\psi|D) = \sum_{i=1}^N\log p(x_1^i,\ldots,x_n^i) = Z^{-1}\psi_C(\mathbf{x_C}) = (\sum_C\sum_{x_C}N(x_C)\log\psi_C(x_C))-N\log Z$ for each clique, $N(x_C)$ are sufficient statistics. Take derivative and get $P_{ML}=\frac{N(x_C)}{N}$. At max $\frac{\hat{p}}{\psi_C(x_C)}=\frac{p}{\psi_C(x_C)}$ so we compute marginal under current guess $p^0(x_C)$ and recompute to get closer to equality above. $\psi_C^{t+1}(x_C)=\psi_C^t(x_C)\frac{\hat{p}(x_C)}{p^t(x_C)}$ Will converge in the limit. Need initial guess ψ^0.



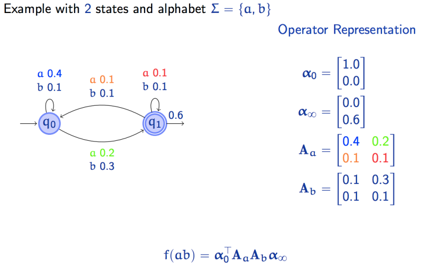
10 PCA

Try to minimize reconstruction error. **Kernel PCA:** 1. pick kernel 2. Construct normalized kernel matrix $\tilde{\mathbf{K}}\in m\times m$ size of data 3. Get eigenvalues λ_j and eigenvectors $\mathbf{a_j}$ 4. Represent points as $y_j=\sum_{i=1}^m a_{ji}K(\mathbf{x},\mathbf{x_i}),j=1,..,m$. Each y_j is the coordinate of $\phi(\mathbf{x})$ in one of feature space axes $\mathbf{v_j}$. $\mathbf{v_j}=\sum_{j=1}^m a_ji\phi(\mathbf{x})$

11 Weighted Automata

Spectral methods faster, not subject to local minima. Don't always have unique parametrization or probabilistic interpretation.

Example with 2 states and alphabet $\Sigma=\{a,b\}$



$$f_A(\mathbf{x}) = f_A(x1, \ldots x_N) = \alpha_0{}^T \mathbf{A}_{\mathbf{x}} \alpha_\infty$$

Definition: p prefix, s suffix $\Rightarrow$ $\mathbf{H}_f(p,s) = f(p \cdot s)$

Example $f(x) = |x|_a$
(number of a's in x)

$$\mathbf{H}_f = \begin{array}{c|ccccc} & \lambda & a & b & aa & \cdots \\ \hline \lambda & 0 & 1 & 0 & 2 & \cdots \\ a & 1 & 2 & 1 & 3 & \\ b & 0 & 1 & 0 & 2 & \\ aa & 2 & 3 & 2 & 4 & \\ \vdots & \vdots & & & & \ddots \end{array}$$

$$\mathbf{H}_f(\lambda, aa) = \mathbf{H}_f(a, a) = \mathbf{H}_f(aa, \lambda) = 2$$

If $\text{rank}(H_f) = n$ then there exists WFA A with n states s.t. $f = f_A$.

Estimate hankel matrix from data. Perform SVD of H, solve for parameters with pseudo-inverses.

$$\begin{array}{cc|cccccc} & \Sigma^\star & \lambda & a & b & aa & ab & \cdots \\ & & & \mathcal{S} & \mathcal{S} & & & \\ \hline & \lambda & 1 & 0.3 & 0.7 & 0.05 & 0.25 & \cdots \\ \mathcal{P} & a & 0.3 & 0.05 & 0.25 & 0.02 & 0.03 & \cdots \\ \mathcal{P} & b & 0.7 & 0.6 & 0.1 & 0.03 & 0.2 & \cdots \\ & aa & 0.05 & 0.02 & 0.03 & 0.017 & 0.003 & \cdots \\ & ab & 0.25 & 0.23 & 0.02 & 0.11 & 0.12 & \cdots \\ & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{array}$$

$\mathbf{h}_{\lambda,\mathcal{S}}$ $\mathbf{h}_{\mathcal{P},\lambda}$ $\mathbf{H}$ $\mathbf{H}_a$

- $\mathbf{H} \in \mathbb{R}^{\mathcal{P}\times\mathcal{S}}$ for finding $\mathbf{P}$ and $\mathbf{S}$
- $\mathbf{H}_\sigma \in \mathbb{R}^{\mathcal{P}\times\mathcal{S}}$ for finding $\mathbf{A}_\sigma$
- $\mathbf{h}_{\lambda,\mathcal{S}} \in \mathbb{R}^{1\times\mathcal{S}}$ for finding $\boldsymbol{\alpha}_0$
- $\mathbf{h}_{\mathcal{P},\lambda} \in \mathbb{R}^{\mathcal{P}\times 1}$ for finding $\boldsymbol{\alpha}_\infty$

12 Method of Moments

Yields consistent estimators (approach true distribution in limit of infinite data) in contrast to EM. Is not subject to local optima. Sample and computational complexity are polynomial.

$$f(x; \theta_1, \cdots, \theta_k)$$
$$\downarrow$$
$$\mathcal{S} = \{x_1, \cdots, x_n\}$$
$$\downarrow$$
$$\begin{cases} \mathbb{E}[x] &= g_1(\theta_1, \cdots, \theta_k) \quad \simeq \frac{1}{n}\sum_{i=1}^n x_i \\ \mathbb{E}[x^2] &= g_2(\theta_1, \cdots, \theta_k) \quad \simeq \frac{1}{n}\sum_{i=1}^n x_i^2 \\ &\vdots \\ \mathbb{E}[x^k] &= g_k(\theta_1, \cdots, \theta_k) \quad \simeq \frac{1}{n}\sum_{i=1}^n x_i^k \end{cases}$$
$$\downarrow$$
$$\widehat{\theta}_1, \cdots \widehat{\theta}_k$$

- What if the random variable $\mathbf{x}$ takes its values in $\mathbb{R}^d$?
- Let's look at the multivariate normal. If $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, the first and second moments are

$$\mathbb{E}[\mathbf{x}] = \boldsymbol{\mu} \quad \text{and} \quad \mathbb{E}[\mathbf{x}\mathbf{x}^\top] = \boldsymbol{\Sigma} + \boldsymbol{\mu}\boldsymbol{\mu}^\top$$

- What if we need higher order moments? The second order moment is $\mathbb{E}[\mathbf{x}\mathbf{x}^\top]$, but what is e.g. the third order moment?

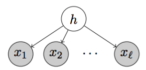
$$\mathbb{E}[\mathbf{x} \otimes \mathbf{x} \otimes \mathbf{x}]$$

Latent Variable Model:
$$f(\mathbf{x}) = \sum_{i=1}^k w_i f_i(\mathbf{x}; \boldsymbol{\mu}_i)$$
$$\downarrow$$
$$\mathcal{S} = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\} \subset \mathbb{R}^d$$
$$\Big\downarrow \text{Structure in the Low Order Moments}$$
$$\begin{cases} \mathbb{E}[\mathbf{x} \otimes \mathbf{x}] &= g_1(\sum_{i=1}^k w_i \boldsymbol{\mu}_i \otimes \boldsymbol{\mu}_i) \\ \mathbb{E}[\mathbf{x} \otimes \mathbf{x} \otimes \mathbf{x}] &= g_2(\sum_{i=1}^k w_i \boldsymbol{\mu}_i \otimes \boldsymbol{\mu}_i \otimes \boldsymbol{\mu}_i) \end{cases}$$
$$\Big\downarrow \text{Tensor Power Method}$$
$$\widehat{w}_i, \widehat{\boldsymbol{\mu}}_i$$

Single Topic Model

- Documents modeled as bags of words:
  - Vocabulary of d words
  - k different topics
  - ℓ words per document
- Documents are drawn as follows:
  (1) Draw a topic h randomly with probability $\mathbb{P}[h = j] = w_j$ for $j \in [k]$
  (2) Draw ℓ word independently according to the distribution $\boldsymbol{\mu}_h \in \Delta^{d-1}$

$\Rightarrow$ Words are independent given the topic:



- Using one-hot encoding for the words $\mathbf{x}_1, \cdots, \mathbf{x}_\ell \in \mathbb{R}^d$ in a document we also have

$$\mathbb{E}[\mathbf{x}_1 \mid h = j] = \boldsymbol{\mu}_j$$
$$\mathbb{E}[\mathbf{x}_1 \otimes \mathbf{x}_2 \mid h = j] = \boldsymbol{\mu}_j \otimes \boldsymbol{\mu}_j$$
$$\mathbb{E}[\mathbf{x}_1 \otimes \mathbf{x}_2 \otimes \mathbf{x}_3 \mid h = j] = \boldsymbol{\mu}_j \otimes \boldsymbol{\mu}_j \otimes \boldsymbol{\mu}_j$$

From which we can deduce

$$\mathbb{E}[\mathbf{x}_1 \otimes \mathbf{x}_2] = \sum_{j=1}^k w_j \boldsymbol{\mu}_j \otimes \boldsymbol{\mu}_j$$
$$\mathbb{E}[\mathbf{x}_1 \otimes \mathbf{x}_2 \otimes \mathbf{x}_3] = \sum_{j=1}^k w_j \boldsymbol{\mu}_j \otimes \boldsymbol{\mu}_j \otimes \boldsymbol{\mu}_j$$

- Under which conditions can we recover the weights w_j and vectors $\boldsymbol{\mu}_j$ for $j \in [k]$ from $\mathbf{M}_2 = \sum_j w_j \boldsymbol{\mu}_j \otimes \boldsymbol{\mu}_j$?
  (i) If the $\boldsymbol{\mu}_j$ are orthonormal and the w_j are distinct, they are the unit eigenvectors of $\mathbf{M}_2$ and the weights are its eigenvalues.
  $\rightarrow$ We would still need to recover the signs of the $\boldsymbol{\mu}_j$...
  (ii) Otherwise, this is not possible!

- Under which conditions can we recover the weights w_j and vectors $\boldsymbol{\mu}_j$ for $j \in [k]$ from $\boldsymbol{\mathcal{M}}_3 = \sum_j w_j \boldsymbol{\mu}_j \otimes \boldsymbol{\mu}_j \otimes \boldsymbol{\mu}_j$?
  $\rightarrow$ We can recover $\pm w_j^{1/3} \boldsymbol{\mu}_j$ if the $\boldsymbol{\mu}_j$ are linearly independent using Jennrich's algorithm (this is sufficient for e.g. single topics model)
  $\rightarrow$ For any vector $\mathbf{v} \in \mathbb{R}^d$ we have

$$\boldsymbol{\mathcal{M}}_3 \bullet_1 \mathbf{v} = \sum_{j=1}^k w_j (\mathbf{v}^\top \boldsymbol{\mu}_j) \boldsymbol{\mu}_j \otimes \boldsymbol{\mu}_j = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^\top.$$

thus if the $\boldsymbol{\mu}_j$ are orthonormal we can recover the $\boldsymbol{\mu}_j$ as eigenvectors and the w_j by solving the linear equation $\lambda_j = w_j(\mathbf{v}^\top \boldsymbol{\mu}_j)$.
(No more ambiguity for the signs of the $\boldsymbol{\mu}_j$ since the w_j are positive.)
idea: Use $\mathbf{M}_2$ to whiten the tensor $\boldsymbol{\mathcal{M}}_3$, then recover the parameters using eigen-decomposition or tensor power method.

Tensor Power Method / (Simultaneous) Diagonalization

We want to solve the following system of equations in $w_i, \boldsymbol{\mu}_i$:

$$\begin{cases} \mathbf{M}_2 &= \sum_{i=1}^k w_i \boldsymbol{\mu}_i \otimes \boldsymbol{\mu}_i \\ \boldsymbol{\mathcal{M}}_3 &= \sum_{i=1}^k w_i \boldsymbol{\mu}_i \otimes \boldsymbol{\mu}_i \otimes \boldsymbol{\mu}_i \end{cases}$$

Overview:

1. Use $\mathbf{M}_2$ to transform the tensor $\boldsymbol{\mathcal{M}}_3$ into an orthogonally decomposable tensor: i.e. find $\mathbf{W} \in \mathbb{R}^{k\times d}$ such that

$$\boldsymbol{\mathcal{T}} = \boldsymbol{\mathcal{M}}_3 \times_1 \mathbf{W} \times_2 \mathbf{W} \times_3 \mathbf{W} = \sum_{i=1}^k \tilde{w}_i \tilde{\boldsymbol{\mu}}_i \otimes \tilde{\boldsymbol{\mu}}_i \otimes \tilde{\boldsymbol{\mu}}_i$$

where the $\tilde{\boldsymbol{\mu}}_i \in \mathbb{R}^k$ are orthonormal.

2. Use (simultaneous) diagonalization or the tensor power method to recover the weights $\tilde{w}_i$ and vectors $\tilde{\boldsymbol{\mu}}_i$.
3. Recover the original weights w_i and vectors $\boldsymbol{\mu}_i$ by 'reverting' the transformation from step 1.

MLE for Bayes nets

Generalizing, for any Bayes net with variables $X_1, \ldots X_n$, we have:

$$\begin{aligned} L(\theta|D) &= \prod_{j=1}^m p(x_1(j), \ldots x_n(j)|\theta) \text{ (from i.i.d)} \\ &= \prod_{j=1}^m \prod_{i=1}^n p(x_i(j)|x_{\pi_i}(j)), \theta) \text{ (factorization)} \\ &= \prod_{i=1}^n \prod_{j=1}^m p(x_i(j)|x_{\pi_i}(j)) \\ &= \prod_{i=1}^n L(\theta_i|D) \end{aligned}$$

13 HMM notes

- Where will the chain be on the first time step, $t = 1$?

$$P(s_{t+1} = s') = \sum_s P(s_0 = s) P(s_1 = s' | s_0 = s)$$

by using the graphical model for the first time step: $s_0 \rightarrow s_1$.
- We can put this in matrix form as follows:

$$\mathbf{p}_1' = \mathbf{p}_0' \mathbf{T} \longrightarrow \mathbf{p}_1 = \mathbf{T}' \mathbf{p}_0$$

where $\mathbf{T}'$ denotes the transpose of $\mathbf{T}$
- Similarly, at $t = 2$, we have:

$$\mathbf{p}_2 = \mathbf{T}'\mathbf{p}_1 = (\mathbf{T}')^2 \mathbf{p}_0$$

- By induction, the probability distribution over possible states at time step t can be computed as:

$$\mathbf{p}_t = \mathbf{T}'\mathbf{p}_{t-1} = (\mathbf{T}')^t \mathbf{p}_0$$

- If $\lim_{t\to\infty} \mathbf{p}_t$ exists, it is called the *stationary or steady-state distribution* of the chain.
- If the limit exists, $\pi = \lim_{t\to\infty} \mathbf{p}_t$, then we have:

$$\pi = \mathbf{T}'\pi, \sum_{s\in S} \pi_s = 1$$

Uncertainty sampling strategies

- Classification:
  1. Ask about the instance for which the most likely class is very uncertain
     E.g., in a probabilistic classifier, the best input $\mathbf{x}$ is given by:
     $$\mathbf{x}^* = \arg\max_{\mathbf{x}} (1 - \max_{y_i} P(y_i|\mathbf{x}))$$
  2. Ask about the instance where the class label has the highest entropy
     $$\mathbf{x}^* = \arg\max_{\mathbf{x}} \left(- \sum_{y_i} P(y_i|\mathbf{x}) \log P(y_i|\mathbf{x}) \right)$$
  3. Ask about the instance for which the top two classes have close probability
  4. Ask based on the margin (in margin-based classifiers)
- Regression: ask about the instance with highest variance.

1. Guess initial parameters p_k, μ_k, Σ_k for each class k
2. Repeat until convergence:
   (a) *E-step:* For each instance i and class k, compute the probabilities of class membership:
   $$w_{ik} = P(y_i = k|\mathbf{x}_i) = \frac{P(\mathbf{x}_i|y_i = k)P(y_i = k)}{P(\mathbf{x}_i)}$$
   I.e., instances are "partially assigned" to each class, according to w_{ik}
   (b) *M-step:* Update the parameters of the model to maximize the likelihood of the data:
   $$p_k = \frac{1}{m}\sum_{i=1}^m w_{ik} \qquad \mu_k = \frac{\sum_{i=1}^m w_{ik}\mathbf{x}_i}{\sum_{i=1}^m w_{ik}}$$
   $$\Sigma_k = \frac{\sum_{i=1}^m w_{ik} (\mathbf{x}_i - \mu_k)(\mathbf{x}_i - \mu_k)^T}{\sum_{i=1}^m w_{ik}}$$

14 Miscellaneous

- ML overfits as it prefers more parameters. Need regularization.
- Mean square error is ML estimator of error given gaussian noise assumption.
- If hypothesis is linear, gradient descent converges to unique global optimum.
- Gibbs sampling with no evidence: stationary distribution is the joint over all variables.
- $\frac{P(X=x'|y)}{P(X=s|y)} = \frac{P(X=x',y)}{P(X=s,y)} = \frac{\prod_C \psi_C(X=x',y)}{\prod_C \psi_C(X=x,y)}$
- Bayes net has at most 2^k parameters where k is the max number of parents in a node.
- HMM has unique steady state distribution if it ergodic (all states can be reached from any other state and there are no periodic cycles). Equilibrium reached regardless of initial distribution.
- Gradient descent not necessarily gives legal parameters.