

**1 Linear Regression**  
Simple  $y_n \approx f(\mathbf{x}_n) := w_0 + w_1 x_{n1}$   
Multiple  
 $f(\mathbf{x}_n) := w_0 + \sum_{j=1}^D w_j x_{nj} = \tilde{\mathbf{x}}_n^T \mathbf{w}$   
If  $D > N$  the task is under-determined (more dimensions than data)  $\rightarrow$  regularization.

**2 Cost functions**  
 $MSE = \frac{1}{N} \sum_{n=1}^N [y_n - f(\mathbf{x}_n)]^2$  Not good with outliers,  
 $MAE = \frac{1}{N} \sum_{n=1}^N |y_n - f(\mathbf{x}_n)|$   
Error  $e_n = y_n - f(\mathbf{x}_n)$

**2.1 Convexity**  
A line joining two points never intersects with the function anywhere else.  
 $f(\lambda \mathbf{u} + (1 - \lambda)\mathbf{v}) \leq \lambda f(\mathbf{u}) + (1 - \lambda)f(\mathbf{v})$  with  $\lambda \in [0; 1]$ . A strictly convex function has a unique global minimum  $\mathbf{w}^*$ . Sums of convex functions are convex.

A function must always lie above its linearisation:  
 $\mathcal{L}(u) \geq \mathcal{L}(w) + \nabla \mathcal{L}(w)^T (u - w) \forall u, w$ .  
A set is convex iff line segment between any two points of  $\mathcal{C}$  lies in  $\mathcal{C}$  :  
 $\theta u + (1 - \theta)v \in \mathcal{C}$

**3 Optimisation**  
Find  $\mathbf{w}^* \in \mathcal{R}^D$  which  $\min \mathcal{L}(\mathbf{w})$ .  
Gradient  $\nabla \mathcal{L} := \begin{bmatrix} \frac{\partial \mathcal{L}(\mathbf{w})}{\partial w_1} & \dots & \frac{\partial \mathcal{L}(\mathbf{w})}{\partial w_D} \end{bmatrix}$

**3.1 Gradient descent**  
 $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \gamma \nabla \mathcal{L}(\mathbf{w}^{(t)})$ . Very sensitive to ill-conditioning.  
GD - Linear Reg  
 $\mathcal{L}(\mathbf{w}) = \frac{1}{2N} (\mathbf{y} - X\mathbf{w})^T (\mathbf{y} - X\mathbf{w}) \rightarrow$   
 $\nabla \mathcal{L}(\mathbf{w}) = -\frac{1}{N} X^T (\mathbf{y} - X\mathbf{w})$ . Cost:  
 $O_{err} = 2ND + N$  and  $O_w = 2ND + D$ .

**3.2 SGD**  
 $\mathcal{L} = \frac{1}{N} \sum \mathcal{L}_n(\mathbf{w})$  with update  $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \gamma \nabla \mathcal{L}_n(\mathbf{w}^{(t)})$ .

**3.3 Mini-batch SGD**  
 $\mathbf{g} = \frac{1}{|B|} \sum_{n \in B} \nabla \mathcal{L}_n(\mathbf{w}^{(t)})$  with update  
 $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \gamma \mathbf{g}$ .

**3.4 Subgradient at w**  
 $\mathbf{g} \in \mathbb{R}^D$  such that  $\mathcal{L}(u) \geq \mathcal{L}(w) + \mathbf{g}^T (u - w)$ . Example subgradient for MAE:  $h(e) = |e| \rightarrow g(e) = \text{sgn}(e)$  if  $e \neq 0, \lambda$  otherwise. We get the gradient:  
 $\nabla \mathcal{L}_{MAE} = -\frac{1}{N} \sum_n \text{sgn}(x_n) \nabla f(x_n)$ .

**3.5 Projected SGD**  
 $\mathbf{w}^{(t+1)} = \mathcal{P}_{\mathcal{C}}[\mathbf{w}^{(t)} - \gamma \nabla \mathcal{L}(\mathbf{w}^{(t)})]$   
**3.6 Newton's method**  
Second order (more expensive  $O(ND^2 + D^3)$  but faster convergence).

$w^{(t+1)} = w^{(t)} - \gamma^{(t)} (H^{(t)})^{-1} \nabla \mathcal{L}(w^{(t)})$   
**3.7 Optimality conditions**  
Necessary :  $\nabla \mathcal{L}(\mathbf{w}^*) = 0$  Sufficient :  
Hessian PSD  $\mathbf{H}(\mathbf{w}^*) := \frac{\partial^2 \mathcal{L}(\mathbf{w}^*)}{\partial w \partial w^T}$

**4 Least Squares**  
**4.1 Normal Equation**  
 $X^T (\mathbf{y} - X\mathbf{w}) = 0 \Rightarrow$   
 $\mathbf{w}^* = (XX^T)^{-1} X^T \mathbf{y}$  and  $\hat{\mathbf{y}}_m = \mathbf{x}_m^T \mathbf{w}^*$   
Graham matrix invertible iff  $\text{rank}(X) = D$  (use SVD  $X = USV^T$  if this is not the case to get pseudo-inverse  $\mathbf{w}^* = V\hat{S}U^T$  with  $\hat{S}$  pseudo-inverse of  $S$ ).

**5 Likelihood**  
Probabilistic model  $y_n = \mathbf{x}_n^T \mathbf{w} + \epsilon_n$ .  
Probability of observing the data given a set of parameters and inputs :  $p(\mathbf{y}|X, \mathbf{w}) = \prod p(y_n | \mathbf{x}_n, \mathbf{w}) = \prod \mathcal{N}(y_n | \mathbf{x}_n^T \mathbf{w}, \sigma^2)$   
Best model maximises log-likelihood  
 $\mathcal{L}_{LL} = -\frac{1}{2\sigma^2} \sum (y_n - \mathbf{x}_n^T \mathbf{w})^2 + cst$ .

**6 Regularization**  
**6.1 Ridge Regression**  
 $\mathcal{L}(\mathbf{w}) = \frac{1}{2} (\mathbf{y} - X\mathbf{w})^T (\mathbf{y} - X\mathbf{w}) + \frac{\lambda}{2} \|\mathbf{w}\|_2^2 \rightarrow$   
 $\mathbf{w}_{\text{ridge}}^* = (XX^T + \lambda I_D)^{-1} X^T \mathbf{y} = X^T (XX^T + \lambda I_N)^{-1} \mathbf{y}$   
Can be considered a MAP estimator :  
 $\mathbf{w}_{\text{ridge}}^* = \text{argmin}_w - \log(p(w|X, y))$

**6.2 Lasso**  
Sparse solution.  $\mathcal{L}(w) = \frac{1}{2N} (y - Xw)^T (y - Xw) + \lambda \|w\|_1$

**7 Model Selection**  
**7.1 Bias-Variance decomposition**  
Small dimensions : large bias, small variance. Large dimensions : small bias, large variance. Error for the val set compared to the emp distr of the data goes down like  $\frac{1}{\sqrt{|\text{validation points}|}}$  and goes up like  $\sqrt{\ln(|\text{hyper parameters}|)}$

**8 Classification**  
**8.1 Optimal**  
 $\hat{y}(\mathbf{x}) = \text{argmax}_{v \in \mathcal{Y}} p(y|\mathbf{x})$

**8.2 Logistic regression**  
 $\sigma(z) = \frac{e^z}{1+e^z}$  to limit the predicted values  $y \in [0; 1]$  ( $p(1|\mathbf{x}) = \sigma(\mathbf{x}^T \mathbf{w})$  and  $p(0|\mathbf{x}) = 1 - \sigma(\mathbf{x}^T \mathbf{w})$ ). We decide with respect to 0.5  
Likelihood  
 $p(y|X, w) = \prod p(y_n | x_n) = \prod_{n: y_n=0} p(y_n = 0 | x_n) \dots \prod_{n: y_n=K} p(y_n = K | x_n) = \prod_k^K \prod_n [p(y_n = k | x_n, w)]^{\tilde{y}_{nk}}$  where  $\tilde{y}_{nk} = 1$  if  $y_n = k$ .  
For binary classification  
 $p(y|X, w) = \prod p(y_n | x_n) = \prod_{n: y_n=0} p(y_n = 0 | x_n) \prod_{n: y_n=1} p(y_n = 1 | x_n) = \prod_n \sigma(x_n^T w)^{y_n} [1 - \sigma(x_n^T w)]^{1-y_n}$   
Loss  
 $\mathcal{L}(w) = \sum_{n=1}^N \ln(1 + \exp(x_n^T w)) - y_n x_n^T w$  which is convex in  $w$ .  
Gradient  
 $\nabla \mathcal{L}(w) = \sum_{n=1}^N x_n (\sigma(x_n^T w) - y_n) = X^T [\sigma(Xw) - y]$  (no closed form solution).  
Hessian  
 $H(w) = X^T S X$  with  $S_{nn} = \sigma(x_n^T w) [1 - \sigma(x_n^T w)]$

**8.3 Exponential family**  
General form  
 $p(y|\eta) = h(y) \exp[\eta^T \psi(y) - A(\eta)]$  where  
Cumulant  
 $A(\eta) = \ln[\int_y h(y) \exp[\eta^T \psi(y)] dy]$   
 $\nabla A(\eta) = \mathbb{E}[\psi(y)]$   
 $\nabla^2 A(\eta) = \mathbb{E}[\psi \psi^T] - \mathbb{E}[\psi] \mathbb{E}[\psi^T]$   
Link function  
 $\eta = g^{-1}(\mu) \Leftrightarrow \mu = g(\eta)$

- $\eta_{\text{gaussian}} = (\mu/\sigma^2, -1/2\sigma^2)$
- $\eta_{\text{poisson}} = \ln(\mu)$
- $\eta_{\text{bernoulli}} = \ln(\mu/1 - \mu)$
- $\eta_{\text{general}} = g^{-1}(\frac{1}{N} \sum_{n=1}^N \psi(y_n))$

**8.4 Nearest Neighbor Models**  
Performs best in low dimensions.

**8.4.1 k-NN**  
 $f_{St,k}(x) = \frac{1}{k} \sum_{n: x_n \in \text{ngb}_{St,k}(x)} y_n$  Pick odd  $k$  so there is a clear winner. Large  $k \rightarrow$  large bias small variance (inv.)

**8.4.2 Error bound**  
 $\mathbb{E}[\mathcal{L}_{St}] \leq 2\mathcal{L}_{f^*} + 4c\sqrt{d}N^{-1/d+1}$

**8.5 Support Vector Machines (SVM)**  
Logistic regression with hinge loss :  $\min_w \sum_{n=1}^N [1 - y_n x_n^T w]_+ + \frac{\lambda}{2} \|w\|^2$  where  $y \in [-1; 1]$  is the label and  $\text{hinge}(x) = \max\{0, x\}$ . Convex but not differentiable so need subgradient.  
We can also use duality :  $\mathcal{L}(w) = \max_{\alpha} G(w, \alpha)$ . For SVM  $\min_w \max_{\alpha \in [0,1]^N} \sum \alpha_n (1 - y_n x_n^T w) + \frac{\lambda}{2} \|w\|^2$  differentiable and convex.  
Can switch *max* and *min* when convex in  $w$  and concave in  $\alpha$ . This can make the formulation simpler:  
 $w(\alpha) = \frac{1}{\lambda} \sum \alpha_n y_n x_n = \frac{1}{\lambda} X^T \text{diag}(y) \alpha$  which yields the optimisation problem:  $\max_{\alpha \in [0,1]^N} \alpha^T \mathbf{1} - \frac{1}{2\lambda} \alpha^T Y X X^T Y \alpha$  The solution is sparse ( $\alpha_n$  is the slope of the lines that are lower bounds to the hinge loss).

**8.6 Kernel Ridge Regression**  
From duality  $w^* := X^T \alpha^*$  where  $\alpha^* := (K + \lambda I_N)^{-1} y$  and  $K = XX^T = \phi^T(x) \phi(x) = \kappa(x, x')$  (needs to be PSD and symmetric).

**9 Unsupervised Learning**  
**9.1 K-means clustering**  
 $\min \mathcal{L}(z, \mu) = \sum_n \sum_k^K z_{nk} \|x_n - \mu_k\|_2^2$  with  $z_{nk} \in \{0, 1\}$  (unique assignments:  $\sum_k z_{nk} = 1$ ).  
Algorithm (Coordinate Descent)

- $\forall n$  compute  $z_n = \begin{cases} 1 & \text{if } k = \text{argmin}_j \|x_n - \mu_j\|^2 \\ 0 & \text{otherwise} \end{cases}$
- $\forall k$  compute  $\mu_k = \frac{\sum_n z_{nk} x_n}{\sum_n z_{nk}}$

Issues

- Heavy computation
- Spherical clusters
- Hard clusters

Probabilistic model  
 $p(X|\mu, z) = \prod_n \mathcal{N}(x_n | \mu_k, I) = \prod_n \prod_k^K \mathcal{N}(x_n | \mu_k, I)^{z_{nk}}$

**9.2 Gaussian Mixture Models**  
 $p(X|\mu, z) = \prod_n p(x_n | z_n, \mu_k, \Sigma_k) p(z_n | \pi) = \prod_n \prod_k^K [\mathcal{N}(x_n | \mu_k, \Sigma_k)]^{z_{nk}} \prod_k^K [\pi_k]^{z_{nk}}$  where  $p_i k = p(z_n = k)$   
Marginal likelihood:  $z_n$  are latent variables so they can be factored

out from the likelihood  $p(x_n | \theta) = \sum \pi_k \mathcal{N}(x_n | \mu_k, \Sigma_k)$ . (number of parameters reduced from  $O(N)$  to  $O(D^2 K)$ ).

**9.3 EM**  
**9.3.1 GMM**  
Intialize  $\mu^{(1)}, \Sigma^{(1)}, \pi^{(1)}$ .

- E-step: Compute the assignments.  $q_{kn}^{(t)} := \frac{\pi_k^{(t)} \mathcal{N}(x_n | \mu_k^{(t)}, \Sigma_k^{(t)})}{\sum_k^K \pi_k^{(t)} \mathcal{N}(x_n | \mu_k^{(t)}, \Sigma_k^{(t)})}$

2. Compute Marginal Likelihood

3. M-step: Update

$$\mu^{(t+1)} = \frac{\sum_n q_{kn}^{(t)} x_n}{\sum_n q_{kn}^{(t)}}$$

$$\Sigma^{(t+1)} = \frac{\sum_n q_{kn}^{(t)} (x_n - \mu^{(t+1)})(x_n - \mu^{(t+1)})^T}{\sum_n q_{kn}^{(t)}}$$

$$\pi^{(t+1)} = \frac{1}{N} \sum_n q_{kn}^{(t)}$$

**9.3.2 General**  
 $\theta^{(t+1)} := \text{argmax}_{\theta} \sum_n^N \mathbb{E}_{p(z_n | x_n, \theta^{(t)})} [\log p]$

**10 Matrix Factorizations**  
**10.1 Prediction**  
Find  $\mathbf{X} \approx \mathbf{WZ}^T$  where  $\mathbf{W} \in \mathbb{R}^{D \times K}$  and  $\mathbf{Z} \in \mathbb{R}^{N \times K}$  with  $K \ll D, N$ . Large  $K \rightarrow$  overfitting. If  $K \geq \max\{D, N\}$  trivial solution ( $\mathbf{W} = \mathbf{1}_D$  or  $\mathbf{Z} = \mathbf{1}_N$ ).  
Quality of reconstruction (not jointly convex nor identifiable):  
 $\mathcal{L}(\mathbf{W}, \mathbf{Z}) := \frac{1}{2} \sum_{(d,n) \in \Omega} [x_{dn} - (\mathbf{WZ}^T)_{dn}]^2 =$

$$\sum_{(d,n) \in \Omega} f_{dn}(w, z)$$

$$\text{Regularizer: } \Omega(W, Z) = \frac{\lambda_w}{2} \|\mathbf{W}\|_{Frob}^2 + \frac{\lambda_z}{2} \|\mathbf{Z}\|_{Frob}^2$$

Optimisation with SGD (compute  $\nabla_w$  for a fixed user  $d'$  and  $\nabla_z$  for a fixed item  $n'$ ). ALS (assume no missing ratings):  $\mathbf{Z}_*^T = (\mathbf{W}^T \mathbf{W} + \lambda_z I_K)^{-1} \mathbf{W}^T \mathbf{X}$   
 $\mathbf{W}_*^T = (\mathbf{Z}^T \mathbf{Z} + \lambda_w I_K)^{-1} \mathbf{Z}^T \mathbf{X}$

**10.2 Text Representation**  
Factorize the co-occurence matrix to get each row forming a representation of a word ( $\mathbf{W}$ ) or a context word ( $\mathbf{Z}$ ) respectively.

## 10.2.1 GloVe

$$f_{dn} := \min\{1, (n_{dn}/n_{max})^\alpha\}, \alpha \in [0; 1]$$

## 10.2.2 Skipgram/CBOW

Binary classification to separate real word pairs from fake ones.

## 10.3 FastText

Supervised sentence-level BoW.

## 11 Dimensionality reduction

### 11.1 SVD

$\mathbf{X} = \mathbf{USV}^T$ , with  $\mathbf{X} : D \times N$ ,  $\mathbf{U} : D \times D$  orthonormal,  $\mathbf{V} : N \times N$  orthonormal,  $\mathbf{S} : D \times N$  diagonal PSD, values in descending order ( $s_1 \geq s_2 \geq \dots \geq s_D \geq 0$ ).

Reconstruction  $\|\mathbf{X} - \hat{\mathbf{X}}\|_F^2 \geq \|\mathbf{X} - \mathbf{U}_K \mathbf{U}_K^T \mathbf{X}\|_F^2 = \sum_{i \geq K+1} s_i^2$   $\forall$  rank- $K$  matrix  $\hat{\mathbf{X}}$  (i.e. we should compress the data by projecting it onto these left singular vectors.)

Truncated SVD:  $\mathbf{U}_K \mathbf{U}_K^T \mathbf{X} = \mathbf{US}_K \mathbf{V}^T$

Application to MF:  $\mathbf{U} = \mathbf{W}$  and  $\mathbf{SV}^T = \mathbf{Z}^T$ . Reconstruction limited by the rank- $K$  of  $\mathbf{W}, \mathbf{Z}$ .

### 11.2 PCA

Decorrelate the data. Empirical mean before:  $\mathbf{NK} = \mathbf{XX}^T = \mathbf{US}_D^2 \mathbf{U}^T$ . After  $\tilde{\mathbf{X}} = \mathbf{U}^T \mathbf{X}$ :  $\mathbf{N}\tilde{\mathbf{K}} = \tilde{\mathbf{X}}\tilde{\mathbf{X}}^T = \mathbf{S}_D^2$  (the components are uncorrelated).

Pitfalls: not invariant under scalings.

## 12 Neural Networks

The output at the node  $j$  in layer  $l$  is  $x_j^{(l)} = \phi(\sum_i w_{i,j}^{(l)} x_i^{(l-1)} + b_j^{(l)})$

### 12.1 Representation power

Error bound  $\leq \frac{(2Cr)^2}{n}$  where  $C$  is the smoothness bound,  $n$  the number of nodes. We can approximate any sufficiently smooth 2-dimensional function on a bounded domain (on average) with  $\sigma$  activation, "pointwise" with ReLU).

### 12.2 Learning

Problem is not convex but SGD is stable. Backpropagation: Let  $\mathcal{L}_n = (y_n - f^{(L+1)} \circ \dots \circ f^{(1)}(\mathbf{x}_n^{(0)}))^2$ .

#### Forward pass

$\mathbf{x}^{(0)} = \mathbf{x}_n$ . For  $l = 1, \dots, L+1$   $\mathbf{z}^{(l)} = (\mathbf{W}^{(l)})^T \mathbf{x}^{(l-1)} + \mathbf{b}^{(l)}$ ,  $\mathbf{x}^{(l)} = \phi(\mathbf{z}^{(l)})$

### Backward pass

$\delta^{(L+1)} = -2(y_n - \mathbf{x}^{(L+1)})\phi'(\mathbf{z}^{(L+1)})$  and  $\forall l : \delta^{(l)} = (\mathbf{W}^{(l+1)})^T \delta^{(l+1)} \odot \phi'(\mathbf{z}^{(l)})$

### Final pass

$$\frac{\partial \mathcal{L}_n}{\partial w_{i,j}^{(l)}} = \delta_j^{(l)} \mathbf{x}_i^{(l-1)}, \frac{\partial \mathcal{L}_n}{\partial b_j^{(l)}} = \delta_j^{(l)}$$

### 12.3 Activations

- sigmoid  $\phi(x) = 1 - \sigma(x)$

$$\tanh \frac{e^x + e^{-x}}{e^x + e^{-x}} = 2\phi(2x) - 1$$

- ReLU, Leaky ReLU ( $\max\{ax, x\}$ )

### 12.4 Convolutional Neural Nets

Convolution with filter  $f$ :  $x^{(1)}[n, m] = \sum_{k,l} f[k, l] x^{(0)}[n-k, m-l]$ . Filter is local so no need for fully connected layers. We can use same filter at every position: *weight sharing*. Learning: run backprop by computing different weights, then sum the gradients of shared weights.

### 12.5 Overfitting

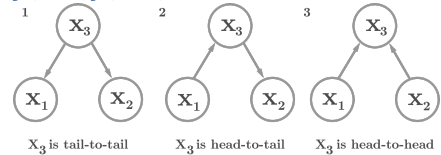
Adding regularisation is equivalent to weight decay (by  $(1-\eta\lambda)$ ). Can also use dataset augmentation, dropout.

## 13 Graphical Models

### 13.1 Bayes Nets

$p(X_1, \dots, X_D) = p(X_1)p(X_2|X_1)\dots p(X_D|X_1, \dots, X_{D-1})$ . One node is a random variable, directed edge from  $X_j$  to  $X_i$  if  $X_j$  appears in the conditioning  $p(X_i|\dots, X_j, \dots)$ . The graph must be *acyclic*.

Conditional independence:  $p(X, Y) = p(X)p(Y)$  or given  $Z$   $p(X, Y|Z) = p(X|Z)p(Y|Z)$ .



- $p(X_1, X_2, X_3) = p(X_3)p(X_1|X_3)p(X_2|X_3)$  :  $X_1$  and  $X_2$  are independent given  $X_3$
- $p(X_1, X_2, X_3) = p(X_1)p(X_3|X_1)p(X_2|X_3)$  :  $X_1$  and  $X_2$  are independent given  $X_3$
- $p(X_1, X_2, X_3) = p(X_1)p(X_2)p(X_3|X_1, X_2)$  :  $X_1$

and  $X_2$  are not independent given  $X_3$

$X \rightarrow Y$  path blocked by  $Z$  if it contains a variable such that either

- variable is in  $Z$  and it is head-to-tail or tail-to-tail
- node is head-to-head and neither this node nor any of its descendants are in  $Z$ .

$X$  and  $Y$  are D-separated by  $Z$  iff every path  $X \rightarrow Y$  is blocked by  $Z$ .

$X$  is conditionally independent of  $Y$  conditioned on the  $Z$  if  $X$  and  $Y$  are D-separated by  $Z$ . Independence is symmetric.

The Markov blanket of a node  $X_i$  is the set of parents, children, and co-parents of the node  $X_i$  (other parents of its children).

### 14 Quick maff

Chain rule  $h = f(g(w)) \rightarrow \partial h(w) = \partial f(g(w)) \nabla g(w)$

Gaussian  $\mathcal{N}(y|\mu, \sigma^2) \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{(y-\mu)^2}{\sigma^2})$

Multivariate Gaussian  $\mathcal{N}(y|\mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^D \det(\Sigma)}} \exp(-\frac{1}{2}(y-\mu)^T \Sigma^{-1}(y-\mu))$

Bayes rule  $p(x|y) = \frac{p(y|x)p(x)}{p(y)}$

Logit  $\sigma(x) = \frac{\partial \ln[1+e^x]}{\partial x}$

Naming Joint distribution  $p(x, y) = p(x|y)p(y) = p(y|x)p(x)$  where

- $p(x|y) \rightarrow$  likelihood
- $p(y) \rightarrow$  prior
- $p(y|x) \rightarrow$  posterior
- $p(x) \rightarrow$  marginal likelihood

Marginal Likelihood

$$p(\mathbf{X}|\alpha) = \int_{\theta} p(\mathbf{X}|\theta) p(\theta|\alpha) d\theta$$

Posterior probability  $\propto$  Likelihood  $\times$  Prior

Maximising over a Gaussian is equivalent to minimising MSE:

$$\beta_{MAP}^* = \operatorname{argmax}_{\beta} p(y|X, \beta) p(\beta) \Leftrightarrow \beta^* = \operatorname{argmin}_{\beta} \mathcal{L}(\beta)$$

Identifiable model  $\theta_1 = \theta_2 \rightarrow P_{\theta_1} = P_{\theta_2}$

## 14.1 Algebra

$$(PQ + I_N)^{-1}P = P(QP + I_M)^{-1}$$

$$\sum_n (y_n - \beta^T \mathbf{x}_n)^2 = (\mathbf{y} - \mathbf{X}\beta)^T (\mathbf{y} - \mathbf{X}\beta)$$

$$\sum_j \beta^2 = \beta^T \beta$$

Unitary / orthogonal:  $\mathbf{UU}^T = \mathbf{U}^T \mathbf{U} = \mathbf{I}$  and  $\mathbf{U}^T = \mathbf{U}^{-1}$ . Rotation matrix (preserves length of vector).

## 15 Mock Exam Notes

### 15.1 Normal equation

Unique if convex.

$$\frac{1}{\sigma_k^2} X(X^T w_k - y_k) + w_k = 0 \Leftrightarrow$$

$$w_k^* = (\frac{1}{\sigma_k^2} X X^T + I_D)^{-1} \frac{1}{\sigma_k^2} X y_k$$

### 15.2 MAP solution

$$\mathcal{L}(w) = \sum_k \sum_n \frac{1}{2\sigma_k^2} (y_{nk} - x_n^T w_k)^2 +$$

$$\frac{1}{2} \sum_k \|w_k\|^2 \rightarrow \text{Likelihood } p(y|X, w) =$$

$$\prod_n \prod_k \mathcal{N}(y_{nk} | w_k^T x_n, \sigma_k^2) \text{ and prior } p(w) = \prod_k \mathcal{N}(w_k | 0, I_D)$$

### 15.3 Convexity

$\ln[\sum_k e^{t_k}]$  is convex. Linear sum of parameters is convex.

### 15.4 Deriving marginal distribution

$$p(y_n | x_n, r_n = k, \beta) = \mathcal{N}(y_n | \beta_k^T \tilde{x}_n, 1)$$

Assume  $r_n$  follows a multinomial  $p(r_n = k | \pi) = \pi_k$ . Derive the marginal

$$p(y_n | x_n, \beta, \pi) = \sum_k p(y_n, r_n = k | x_n, \beta, \pi) =$$

$$\sum_k^K p(y_n | r_n = k, x_n, \beta, \pi) \cdot \pi_k =$$

$$\sum_k^K \mathcal{N}(y_n | \beta_k^T \tilde{x}_n, \sigma^2) \cdot \pi_k$$

### 15.5 MF

$$\hat{r}_{um} = \langle \mathbf{v}_u, \mathbf{w}_m \rangle + b_u + b_m \quad \mathcal{L} =$$

$$\frac{1}{2} \sum_u m (\hat{r}_{um} - r_{um})^2 + \frac{\lambda}{2} \left[ \sum_u (b_u^2 + \|\mathbf{v}_u\|^2) + \sum_m (b_m^2 + \|\mathbf{w}_m\|^2) \right].$$

The optimal value for  $b_u$  for a particular user  $u'$  :  $\sum_u m (\hat{r}_{um} - r_{um}) + \lambda b_{u'} = 0$ .

Problem jointly convex? Compute

$$H(\hat{r}(v, w)) = \begin{bmatrix} 2w^2 & 4vw - 2r \\ 4vw - 2r & 2v^2 \end{bmatrix}$$

which is not PSD in general.

## 16 Multiple Choice Notes

### 16.1 True statements

- Regularisation term sometimes renders the min. problem into a strictly concave/convex problem.

- k-NN can be applied even if the data cannot be linearly separated.

$$\max\{0, x\} = \max_{\alpha \in [0, 1]} \alpha x$$

$$\min\{0, x\} = \min_{\alpha \in [0, 1]} \alpha x$$

$$g(x) = \min_y f(x, y) \Rightarrow g(x) \leq f(x, y)$$

$$\max_x g(x) \leq \max_x f(x, y)$$

$$\max_x \min_y f(x, y) \leq \min_y \max_x f(x, y)$$

$$\nabla_W (\mathbf{x}^T \mathbf{W} \mathbf{x}) = \mathbf{x} \mathbf{x}^T$$

$$\nabla_x (\mathbf{x}^T \mathbf{W} \mathbf{x}) = (\mathbf{W} + \mathbf{W}^T) \mathbf{x}$$

K-means: optimal cluster (resp. centers) init  $\rightarrow$  one step optimal representation points (resp. clusters).

Logistic loss is typically preferred over  $L_2$  loss in classification tasks.

For optimizing a MF of a  $D \times N$  matrix, for large  $D, N$  : per iteration, ALS has an increased computational cost over SGD and per iteration, SGD cost is independent of  $D, N$ .

The complexity of backprop for a nn with  $L$  layers and  $K$  nodes/layer is  $O(K^2 L)$

CNN where the data is laid out in a one-dimensional fashion and the filter/kernel has  $M$  non-zero terms. Ignoring the bias terms, there are  $M$  parameters.

## 16.2 Convex functions

- $f(x) = x^\alpha, x \in \mathbb{R}^+, \forall \alpha \geq 1$  or  $\leq 0$
- $f(x) = -x^3, x \in [-1, 0]$
- $f(x) = e^{ax}, \forall x, a \in \mathbb{R}$
- $f(x) = \ln(1/x), x \in \mathbb{R}^+$
- $f(x) = g(h(x)), x \in \mathbb{R}, g, h$  convex and increasing over  $\mathbb{R}$
- $f(x) = ax + b, x \in \mathbb{R}, \forall a, b \in \mathbb{R}$
- $f(x) = |x|^p, x \in \mathbb{R}, p \geq 1$
- $f(x) = x \log(x), x \in \mathbb{R}^+$

## 16.3 Non-convex functions

$$f(x) = x^3, x \in [-1, 1]$$

$$f(x) = e^{-x^2}, x \in \mathbb{R}$$

$$\sum \mathcal{N}$$

$$\sin(x) \forall x \in \mathbb{R}$$