# 1 Regression

## 1.1 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 $w^*$. 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 $\nabla\mathcal{L} := \left[\frac{\partial\mathcal{L}(\mathbf{w})}{\partial w_1} \quad \ldots \quad \frac{\partial\mathcal{L}(\mathbf{w})}{\partial w_D}\right]$

## 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: $h(e) = |e| \rightarrow g(e) = sgn(e)$ if $e \neq 0, \lambda$ otherwise . We get the gradient:
$\nabla\mathcal{L}_{MAE} = -\frac{1}{N}\sum_n sgn(x_n)\nabla f(x_n)$.

## 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

$X^T(\mathbf{y} - X\mathbf{w}) = 0 \Rightarrow$
$\mathbf{w}^* = (X^TX)^{-1}X^T\mathbf{y}$ and $\hat{\mathbf{y}}_\mathbf{m} = \mathbf{x_m}^T\mathbf{w}^*$
Graham matrix invertible iff $rank(X) = D$ (use SVD $X = USV^T \in \mathbb{R}^{N\times D}$ if this is not the case to get pseudo-inverse $\mathbf{w}^* = V\tilde{S}U^T$ with $\tilde{S}$ 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
$\mathcal{L}_{LL} = -\frac{1}{2\sigma^2}\sum(y_n - x_n^Tw)^2 + cst.$

# 6 Regularization

## 6.1 Ridge Regression

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

## 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

$\hat{y}(\mathbf{x}) = argmax_{y\in\mathcal{Y}}\, p(y|\mathbf{x})$

## 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
$p(y|X, w) = \prod p(y_n|x_n) = \prod_{n:y_n=0}p(y_n = 0|x_n)\ldots\prod_{n:y_n=K}p(y_n = K|x_n) = \prod_k^K\prod_n^N[p(y_n = k|x_n, w)]^{\tilde{y}_{nk}}$
where $tildey_{nk} = 1$ if $y_n = k$.
For binary classification
$p(y|X, w) = \prod p(y_n|x_n) = \prod_{n:y_n=0}p(y_n = 0|x_n)\prod_{n:y_n=1}p(y_n = 1|x_n) = \prod_n^N\sigma(x_n^Tw)^{y_n}[1 - \sigma(x_n^Tw)]^{1-y_n}$
Loss
$\mathcal{L}(w) = \sum_{n=1}^N ln(1 + exp(x_n^Tw)) - y_nx_n^Tw$
which is convex in $w$.
Gradient
$\nabla\mathcal{L}(w) = \sum_{n=1}^N x_n(\sigma(x_n^Tw) - y_n) = X^T[\sigma(Xw) - y]$ (no closed form solution).
Hessian
$H(w) = X^TSX$ with $S_{nn} = \sigma(x_n^Tw)[1 - \sigma(x_n^Tw)]$

## 8.3 Exponential family

General form
$p(y|\eta) = h(y)exp[\eta^T\psi(y) - A(\eta)]$ where
Cumulant
$A(\eta) = ln[\int_y h(y)exp[\eta^T\psi(y)]dy]$
$\nabla A(\eta) = \mathbb{E}[\psi(y)] = g^{-1}(\eta)$
$\nabla^2 A(\eta) = \mathbb{E}[\psi\psi^T] - \mathbb{E}[\psi]\mathbb{E}[\psi^T]$
Link function
$\eta = g(\mu) \Leftrightarrow \mu = g^{-1}(\eta)$
$\eta_{gaussian} = (\mu/\sigma^2, -1/2\sigma^2)$ ; $\eta_{poisson} = ln(\mu)$ ; $\eta_{bernoulli} = ln(\mu/1 - \mu)$ ;
$\eta_{general} = g^{-1}(\frac{1}{N}\sum_{n=1}^N\psi(y_n))$
$\nabla\mathcal{L}(w)X^T[g^{-1}(Xw) - \psi(y)] = 0$

## 8.4 Nearest Neighbor Models

Performs best in low dimensions.

### 8.4.1 k-NN

$f_{S_{t,k}}(x) = \frac{1}{k}\sum_{n:x_n\in ngbh_{S_{t,k}(x)}} y_n$ Pick odd $k$ so there is a clear winner. Large $k \rightarrow$ large bias small variance (inv.)

### 8.4.2 Error bound

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

## 8.5 Support Vector Machines (SVM)

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

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

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

Issues

1. Heavy computation
2. Spherical clusters
3. Hard clusters

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

## 9.2 Gaussian Mixture Models

$p(X|\mu, z) = \prod_n^N p(x_n|z_n, \mu_k, \Sigma_k)p(z_n|\pi) = \prod_n^N\prod_k^K[\mathcal{N}(x_n|\mu_k, \Sigma_k)]^{z_{nk}}\prod_k^K[\pi_k]^{z_{nk}}$
where $pi_k = p(z_n = k)$
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 $O(D^2K)$.

## 9.3 EM

### 9.3.1 GMM

Intialize $\mu^{(1)}, \Sigma^{(1)}, \pi^{(1)}$.

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

$\theta^{(t+1)} := argmax_\theta\sum_n^N\mathbb{E}_{p(z_n|x_n,\theta^{(t)})}[log\ p$

# 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 $K << D, N$. Large $K \rightarrow$ overfitting. If $K \geq max\{D, N\}$ trivial solution ($W = \mathbf{1}_D$ or $Z = \mathbf{1}_N$).
Quality of reconstruction (not jointly convex nor identifiable):
$\mathcal{L}(\mathbf{W}, \mathbf{Z}) := \frac{1}{2}\sum_{(d,n)\in\Omega}[x_{dn} - (\mathbf{WZ}^T)_{dn}]^2 = \sum_{(d,n)\in\Omega} f_{dn}(w, z)$
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): $\mathbf{Z}_*^T = (\mathbf{W}^T\mathbf{W} + \lambda_zI_K)^{-1}\mathbf{W}^T\mathbf{X}$ $\mathbf{W}_*^T = (\mathbf{Z}^T\mathbf{Z} + \lambda_wI_K)^{-1}\mathbf{ZX}^T$

## 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 ($s_1 \ge s_2 \ge \dots \ge s_D \ge 0$).

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

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

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

### 11.2 PCA

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

Pitfalls: not invariant under scalings.

## 12 Neural Networks

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

### 12.1 Representation power

Error bound $\le \frac{(2Cr)^2}{n}$ where $C$ is the smoothness bound, $n$ the number of nodes. We can approximate any sufficiently smooth 2-dimensional function on a bounded domain (ön 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 $\forall l: \delta^{(l)} = (\mathbf{W}^{(l+1)}\delta^{(l+1)})\odot\phi'(\mathbf{z}^{(l)})$

**Final pass**

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

### 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 ($max\{\alpha x, x\}$)

### 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

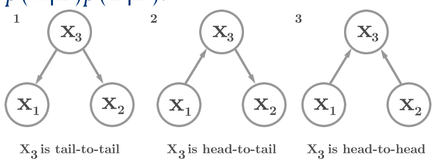
### 13.1 Bayes Nets

$p(X_1,\dots,X_D) = p(X_1)p(X_2|X_1)\dots p(X_D|X_{\dots})$

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 $Z$ $p(X,Y|Z) = p(X|Z)p(Y|Z)$.




1. $p(X_1,X_2,X_3) = p(X_3)p(X_1|X_3)p(X_2|X_3)$ : $X_1$ and $X_2$ are independent given $X_3$
2. $p(X_1,X_2,X_3) = p(X_1)p(X_3|X_1)p(X_2|X_3)$ : $X_1$ and $X_2$ are independent given $X_3$
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 $X_3$

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

1. variable is in $Z$ and it is head-to-tail or tail-to-tail
2. 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 $X \to Y$ 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 $h = f(g(w)) \to \partial h(w) = \partial f(g(w))\nabla g(w)$

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

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

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

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

Marginal Likelihood $p(\mathbb{X}|\alpha) = \int_\theta p(\mathbb{X}|\theta)p(\theta|\alpha)\, d\theta$
$p(X = x) = \sum_y p(X = x, Y = y) = \sum_y p(X = x \mid Y = y)p(Y = y)$
Posterior probability $\propto$ Likelihood $\times$ Prior
Maximising over a Gaussian is equivalent to minimising MSE: $\beta^*_{MAP} = argmax_\beta p(y|X,\beta)p(\beta) \Leftrightarrow \beta^* = argmin_\beta \mathcal{L}(\beta)$
Identifiable model $\theta_1 = \theta_2 \to P_{\theta_1} = P_{\theta_2}$

### 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).
Jensen's inequality: $log(\sum a) \ge \sum q log(\frac{a}{q})$

## 15 Mock Exam Notes

### 15.1 Normal equation

Unique if convex.
$\frac{1}{\sigma_k^2}X(X^Tw_k - y_k) + w_k = 0 \Leftrightarrow$
$w_k^* = (\frac{1}{\sigma_k^2}XX^T + I_D)^{-1}\frac{1}{\sigma_k^2}Xy_k$

### 15.2 MAP solution

$\mathcal{L}(w) = \sum_k\sum_n \frac{1}{2\sigma_k^2}(y_{nk} - x_n^Tw_k)^2 + \frac{1}{2}\sum_k\|w_k\|_2^2 \to$ Likelihood $p(y|X,w) = \prod_n\prod_k \mathcal{N}(y_{nk}|w_k^Tx_n,\sigma_k^2)$ and prior $p(w) = \prod_k \mathcal{N}(w_k|0,I_D)$

### 15.3 Convexity

$ln[\sum_k^K e^{t_k}]$ 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 $p(r_n = k|\pi)$. Derive the marginal $p(y_n|x_n,\beta,\pi)$. $p(y_n|x_n, r_n = k,\beta) = \sum_k^K p(y_n, r_n = k|x_n,\beta,\pi) = \sum_k^K p(y_n|r_n = k, x_n,\beta,\pi)\cdot\pi_k = \sum_k^K \mathcal{N}(y_n|\beta_k^T\tilde{x}_n,\sigma^2)\cdot\pi_k$

### 15.5 MF

$\hat{r}_{um} = \langle\mathbf{v}_u,\mathbf{w}_m\rangle + b_u + b_m$ $\mathcal{L} = \frac{1}{2}\sum_{u\ m}(\hat{r}_{um}-r_{um}) + \frac{\lambda}{2}\left[\sum_u(b_u^2+\|\mathbf{v}_u\|^2) + \sum_m(b_m^2+\|\mathbf{w}_m\|^2)\right]$. The optimal value for $b_u$ for a particular user $u'$: $\sum_{u'\ m}(\hat{r}_{u'm} - r_{u'm}) + \lambda b_{u'} = 0$.
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.
- $max\{0,x\} = max_{\alpha\in[0,1]}\alpha x$
- $min\{0,x\} = min_{\alpha\in[0,1]}\alpha x$
- $g(x) = min_y f(x,y) \Rightarrow g(x) \le f(x,y)$
- $max_x g(x) \le max_x f(x,y)$
- $max_x min_y f(x,y) \le min_y max_x f(x,y)$
- $\nabla_W(\mathbf{x}^T\mathbf{Wx}) = \mathbf{xx}^T$
- $\nabla_x(\mathbf{x}^T\mathbf{Wx}) = (\mathbf{W}+\mathbf{W}^T)\mathbf{x}$
- K-means: optimal cluster (resp. centers) init $\to$ 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 $O(K^2L)$
- 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

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

### 16.3 Non-convex functions

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

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

- $\sum \mathcal{N}$

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