1 Bayesian Decision Theory

Bernoulli : $P(X) = p_0^X(1-p_0)^{1-X}$, p_0 is the param.

Estimation of p_0 form $\mathcal{X} = \{x^{(l)}\}_{l=1}^N$:
$\hat{p_0} = \frac{\#\text{heads}}{\#\text{tosses}} = \frac{\sum_{l=1}^N x^{(l)}}{N}$

Predict outcome = head if $P_0 > 1/2$, tail otherwise.

Baye's Rule :
Posterior$P(C|\mathbf{x}) = \frac{\text{likelihood}\times\text{prior}}{\text{evidence}} = \frac{p(\mathbf{x}|C)P(C)}{p(\mathbf{x})}$

Baye for $K > 2$ classes:
$P(C_i|\mathbf{x}) = \frac{p(\mathbf{x}|C_i)P(C_i)}{\sum_{k=1}^K p(\mathbf{x}|C_k)P(C_k)}$

Optimal decision: Choose C_i if $P(C_i|\mathbf{x}) = \max_k(C_k|\mathbf{x})$

Losses and Risks :
$R(\alpha_i|\mathbf{x}) = \sum_{k=1}^K \lambda_{ik}P(C_k|\mathbf{x})$,
α_i the action assigned to class C_i, λ_{ik} loss for α_i if C_k

Optimal:α_i if $R(\alpha_i|\mathbf{x}) = \min_k R(\alpha_k|\mathbf{x})$

0-1 loss : $R(\alpha_i|\mathbf{x}) = \sum_{k=1}^K \lambda_{ik}P(C_k|\mathbf{x}) = 1 - P(C_i|\mathbf{x})$,
$\lambda_{ik} = 1$ if $i \neq k$, 0 otherwise

Reject Option : Loss:

$$\lambda_{ik} = \begin{cases} 1 & \text{if } i = k \\ \lambda & \text{if } i = K+1 \\ 0 & \text{otherwise} \end{cases}$$

λ is the loss for choosing reject.

Expected risk:

$$R(\alpha_i|\mathbf{x}) = \begin{cases} \sum_{k=1}^K \lambda P(C_k|\mathbf{x}) = \lambda & \text{if } i = K+1 \\ \sum_{k\neq 1} P(C_k|\mathbf{x}) = 1 - P(C_i|\mathbf{x}) & \text{if } i \in 1,\dots,K \end{cases}$$

Optimal Decision: choose C_i if
$R(\alpha_i|x) = \min_{1\le k\le K} R(\alpha_i|\mathbf{x}) < R(\alpha_{K+1}|\mathbf{x})$,
Reject otherwise.

Discriminant Functions : choose C_i if $g_i(\mathbf{x}) = \max_k g_k(x)$

$g_i(\mathbf{x}) = -R(\alpha_i|x) = P(C_i|\mathbf{x}) = p(\mathbf{x}|C_i)P(C_i)$

Two class may define single discriminant functions: $g(\mathbf{x}) = g_1(\mathbf{x}) - g_2(\mathbf{x})$, choose C_i if it greater than zero.

Decision Regions $\mathcal{R}_i = \{\mathbf{x}|g_i(\mathbf{x}) = \max_k g_k(\mathbf{x})\}$

Bayesian Network joint:
$P(X_1,\dots,X_d) = \prod_{i=1}^d P(X_i|parents(X_i))$

2 Parameter Estimation

Setting Assume data follow a distribution model.$\mathcal{X} = \{\mathbf{x}^{(l)}\}$, $\mathbf{x}^l \sim p(\mathbf{x})$, Assume some parametric form for $p(\mathbf{x}|\theta)$, θ is estimated using $\mathcal{X}$

Param Approach to classification In Baye's rule for classification, $p(\mathbf{x}|C_i)$(likelihood) and $P(C_i)$ (prior) need to be estimated from the sample $\mathcal{X}$

MLE seeks to find θ that makes sampling from $p(\mathbf{x}|\theta)$ as likely as possible

likelihood:
$L(\theta|\mathcal{X}) \equiv p(\mathcal{X}|\theta) = \prod_{l=1}^N p(\mathbf{x}^{(l)}|\theta)$

log likelihood: $\mathcal{L} \equiv \log L(\theta|X) = \sum_{l=1}^N \log p(\mathbf{x}^{(l)}|\theta)$

Max Likelihood estimate: $\hat{\theta} = arg\max_\theta \mathcal{L}(\theta|\mathcal{X})$

Bernoulli $x \in \{0,1\}, P(x=1)$ for $p(C_1)$,
$P(x|p_0) = p_0^x(1-p_0)^{1-x}$

Log Likelihood:
$\mathcal{L}(p_0|\mathcal{X}) = \sum_{l=1}^N [x^{(l)}\log p_0 + (1-x^{(l)})\log(1-p_0)]$

ML estimation: $\hat{p_0} = \frac{1}{N}\sum_{l=1}^N x^{(l)}$

Multinomial Ran Var. $\mathbf{x}$ with $K \geq 2$ possible value

Indicator var: $x_i = 1$ if outcome is state i, 0 if not

$P(\mathbf{x}|\theta) = P(X_1,\dots,x_K|p_1,\dots,p_K) = \prod_{i=1}^K p_i^{x_i}$, $\sum_{i=1}^K p_i = 1$

Log likelihood: $\mathcal{L}(p_0|\mathcal{X}) = \sum_l^N \sum_{i=1}^K x_i^{(l)}\log p_i$

ML estimate: $\hat{p_i} = \frac{1}{N}\sum_{l=1}^N x_i^{(l)}$

Normal pdf: $p(x|\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp[-\frac{(x-\mu)^2}{2\sigma^2}]$

$\mathcal{L}(\mu,\sigma|\mathcal{X}) = -\frac{N}{2}\log(2\pi) - N\log\sigma - \frac{1}{2\sigma^2}\sum_{l=1}^N(x^{(l)}-\mu)^2$

ML Estimates: $\hat{\mu} = \frac{1}{N}\sum_{l=1}^N x^{(l)}$
$\hat{\sigma^2} = \frac{1}{N}\sum_{l=1}^N (x^{(l)} - \hat{\mu})^2$

Multivariable Normal $\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})$, $\boldsymbol{\mu}$ mean vec, $\boldsymbol{\Sigma}$ covariance matrix

$p(\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}|^{1/2}}\exp[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})]$

$\mathcal{L}(\boldsymbol{\mu},\boldsymbol{\Sigma}|\mathcal{X}) = \frac{Nd}{2}\log(2\pi) - \frac{N}{2}\log|\boldsymbol{\Sigma}| - \frac{1}{2}\sum_{l=1}^N(\mathbf{x}^{(l)} - \boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}^{(l)} - \boldsymbol{\mu})$

ML estimates: $\hat{\boldsymbol{\mu}} = \frac{1}{N}\sum_{l=1}^N \mathbf{x}^{(l)}$,
$\hat{\boldsymbol{\Sigma}} = \frac{1}{N}\sum_{l=1}^N(\mathbf{x}^{(l)} - \hat{\boldsymbol{\mu}})(\mathbf{x}^{(l)} - \hat{\boldsymbol{\mu}})^T$

Bias and Variance Bias: $b_\theta(d) = E[d] - \theta$
Variance: $E[d - E[d]^2]$(d is estimator of param θ)

Mean Squared error:
$r(d,\theta) = E[(d-\theta)^2] = \text{bias}^2 + \text{variance}^2$

Bayesian Estimation Treat θ as a Ran Var. Prior $p(\theta)$

Posterior:
$p(\theta|\mathcal{X}) = \frac{p(\mathcal{X}|\theta)p(\theta)}{p(\mathcal{X})} = \frac{p(\mathcal{X}|\theta)p(\theta)}{\int p(\mathcal{X}|\theta')p(\theta')d\theta'}$

Estimation of density at x:
$p(x|\mathcal{X}) = \int p(x|\theta)p(\theta|\mathcal{X})d\theta$

Regression $y = g(x|\theta)$: $y = \int g(x|\theta)p(\theta|\mathcal{X})d\theta$

Computational Considerations

Max a posteriori (MAP): $\theta_{MAP} = arg\max_\theta p(\theta|\mathcal{X})$,
$p(x|\mathcal{X}) \approx p(x|\theta_{MAP})$, $y \approx y_{MAP} = g(x|\theta_{MAP})$

ML estimation: $\theta_{ML} = arg\max_\theta p(\theta|\mathcal{X})$

Bayes' estimation – expectation w.r.t. posterior density:

$$\theta_{Bayes} = E[\theta|\mathcal{X}] = \int \theta p(\theta|\mathcal{X})d\theta$$

Example: Bayesian estimation with known μ, σ and σ_0
$x^{(l)} \sim \mathcal{N}(\theta,\sigma_0^2)$, $\theta \sim \mathcal{N}(\mu,\sigma^2)$
MLE: $\theta_{ML} = \frac{1}{N}\sum_{l=1}^N \mathbf{x}^{(l)} = m$
$\theta_{Map} = \theta_{Bayes} = E(\theta|\mathcal{X}) = \frac{N/\sigma_0^2}{N/\sigma_0^2+1/\sigma^2}m + \frac{1/\sigma^2}{N/\sigma_0^2+1/\sigma^2}\mu$

Classification with Discriminant Functions Gaussian density for each class: $p(x|C_i) = \frac{1}{\sqrt{2\pi\sigma_i^2}}\exp\left[-\frac{(x-\mu_i)^2}{2\sigma_i^2}\right]$

Discriminant functions: $g_i(x) = \log[p(x|C_i)p(C_i)] = -\frac{1}{2}\log 2\pi - \log\sigma_i - \frac{(x-\mu_i)^2}{2\sigma_i^2} + \log P(C_i)$

Sample $\mathcal{X} = \left\{(x^{(l)},\mathbf{y}^{(l)})\right\}_{l=1}^N (y_i^{(l)} = 1 \text{ if } x^{(l)} \in C_i)$

ML Estimates: $\hat{P}(C_i) = \frac{1}{N}\sum_{l=1}^N y_i^{(l)}$
$m_i = \frac{\sum_{l=1}^N x^{(l)}y_i^{(l)}}{\sum_{l=1}^N y_i^{(l)}}$ $\quad s_i^2 = \frac{\sum_{l=1}^N (x^{(l)}-m_i)^2 y_i^{(l)}}{\sum_{l=1}^N y_i^{(l)}}$

Discriminant Functions:

$$g_i(x) = -\log s_i - \frac{(x-m_i)^2}{2s_i^2} + \log\hat{P}(C_i)$$

Additive Parametric Model Functional relationship in additive form: $r = f(x) + \epsilon$

Parametric modeling: $f(x) \approx g(x|\theta)$, $\epsilon \sim \mathcal{N}(0,\sigma^2)$

Conditional probability of output given input:

$$p(r|x) \sim \mathcal{N}(g(x|\theta),\sigma^2)$$

Log likelihood given: $\mathcal{X} = \left\{(x^{(l)},r^{(l)})\right\}$:

$$\mathcal{L}(\theta|\mathcal{X}) = \log\prod_{l=1}^N p(x^{(l)},r^{(l)}) = \frac{1}{2\sigma^2}\sum_{l=1}^N\left[r^{(l)} - g(x^{(l)}|\theta)\right]^2 + \text{const}$$

Equivalent to minimizing error function:

$$E(\theta|\mathcal{X}) = \frac{1}{2}\sum_{l=1}^N\left[r^{(l)} - g(x^{(l)}|\theta)\right]^2$$

Called least squares estimates

Polynomial Regression

$$g(x^{(l)}|w_0,w_1,\dots,w_k) = w_k(x^{(l)})^k + \dots + w_2(x^{(l)})^2 + w_1x^{(l)} + w_0$$

Least square estimate: $\hat{\mathbf{w}} = (\mathbf{D}^T\mathbf{D})^{-1}\mathbf{D}^T\mathbf{r}$, where:

$$D = \begin{bmatrix} 1 & x^{(1)} & (x^{(1)})^2 & \dots & (x^{(l)})^k \\ 1 & x^{(2)} & (x^{(2)})^2 & \dots & (x^{(l)})^k \\ \dots \\ 1 & x^{(N)} & (x^{(N)})^N & \dots & (x^{(l)})^k \end{bmatrix}$$

$$\mathbf{r} = \left(r^{(1)}, r^{(2)}, \dots, r^{(N)}\right)^T$$

Bias and Variance Expected squared error of sample $\mathcal{X}$ $E\left[(r-g(x))^2|x\right] = (E[r|x] - g(x))^2 + E\left[(r - E[r|x])^2|x\right] = \text{squared err} + \text{noise}$
Average over $\mathcal{X}$: $E_\mathcal{X} = \left[(E[r|x] - g(x))^2|x\right] = (E[r|x] - E_\mathcal{X}[g(x)])^2 + E_\mathcal{X}\left[(g(x) - E_\mathcal{X}[g(x)])^2\right] = \text{bias} + \text{variance}$

3 Multivariate Method

Multivariate Data N i.i.d. instances:

$$\mathbf{X} = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & \dots x_d^{(1)} \\ x_1^{(2)} & x_2^{(2)} & \dots x_d^{(2)} \\ \dots \\ x_1^{(N)} & x_2^{(N)} & \dots x_d^{(N)} \end{bmatrix}$$

Parameters Mean Vector: $E[x] = \mu = (\mu_1,\dots,\mu_d)^T$

Covariance of x_i and x_j:

$$\sigma_{ij} = E\left[(x_i - \mu_i)(x_j - \mu_j)\right] = E[x_ix_j] - \mu_i\mu_j$$

Variance of x_i: $\sigma_i^2 = E\left[(x_i - \mu_i)^2\right]$

Covariance matrix:

$$\boldsymbol{\Sigma} = Cov(\mathbf{x}) = E\left[(\mathbf{x}-\boldsymbol{\mu})(\mathbf{x}-\boldsymbol{\mu})^T\right] = \begin{bmatrix} \sigma_1^2 & \sigma_{12} & \dots & \sigma_{1d} \\ \sigma_{21} & \sigma_2^2 & \dots & \sigma_{2d} \\ \dots \\ \sigma_{d1} & \sigma_{d2} & \dots & \sigma_d^2 \end{bmatrix}$$

Correlation: $\rho_{ij} = \frac{\sigma_{ij}}{\sigma_i\sigma_j}$

x_i and x_j are independent $\Rightarrow \sigma_{ij} = \rho_{ij} = 0$

Parameter Estimation Sample Mean $\mathbf{m} = \frac{1}{N}\sum_{l=1}^N \mathbf{x}^{(l)}$

Sam. Cov.: $\mathbf{S} = [s_{ij}]_{i,j=1}^d = \frac{1}{N}\sum_{l=1}^N(\mathbf{x}^{(l)} - \mathbf{m})(\mathbf{x}^{(l)} - \mathbf{m})^T$

Multivariate Normal Distribution $\mathbf{x} \sim \mathcal{N}_d(\boldsymbol{\mu},\boldsymbol{\Sigma})$

$p(x) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}|^{1/2}}\exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right]$

Mahalanobis distance: $(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})$(d-dimensional hyperellipsoid.)

Bivariate Normal Distribution Covariance matrix:

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix}$$

$p(x_1,x_2) = \frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}}\exp\left[-\frac{1}{2(1-\rho^2)}(z_1^2 - 2\rho z_1z_2 + z_2^2)\right]$,
where $z_i = \frac{x_i - \mu_i}{\sigma_i}$

Parametric Classification Class-conditional densities:$p(\mathbf{x}|C_i) \sim \mathcal{N}_d(\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)$:

$$p(\mathbf{x}|C_i) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}_i|^{1/2}}\exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T\boldsymbol{\Sigma}_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)\right]$$

Discriminant functions: $g_i(\mathbf{x}) = \log p(\mathbf{x}|C_i) + \log P(C_i)$
$= -\frac{1}{2}\log 2\pi - \frac{1}{2}\log|\boldsymbol{\Sigma}_i| - \frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T\boldsymbol{\Sigma}_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i) + \log P(C_i)$

Estimation of Parameters $\hat{P}(C_i) = \frac{1}{N}\sum_l r_i^{(l)}$
$\mathbf{m}_i = \frac{\sum_l r_i^{(l)}\mathbf{x}^{(l)}}{\sum_l r_i^{(l)}}$
$\mathbf{S}_i = \frac{\sum_l r_i^{(l)} r_i^{(l)}(\mathbf{x}^{(l)} - \mathbf{m}_i)(\mathbf{x}^{(l)} - \mathbf{m}_i)^T}{\sum_l r_i^{(l)}}$

Quadratic Discriminant Functions

$$g_i(\mathbf{x}) = \mathbf{x}^T\mathbf{W}_i\mathbf{x} + \mathbf{w}_i^T\mathbf{x} + w_0$$

where $\mathbf{W}_i = -\frac{1}{2}\mathbf{S}_i^{-1}$
$\mathbf{w}_i = \mathbf{S}_i^{-1}\mathbf{m}_i$
$w_{i0} = -\frac{1}{2}\mathbf{m}_i^T\mathbf{S}_i^{-1}\mathbf{m}_i - \frac{1}{2}\log|\mathbf{S}_i| + \log\hat{P}(C_i)$

4 Dimensionality reduction

Forward Search Start with no features, add them one by one, at each step adding the one that decreases most.

Backward Search Start with all features and so a similar process

Principle Component Analysis Projection of $\mathbf{x}$ on the direction of $\mathbf{w}$: $z = \mathbf{w}^T\mathbf{x}$

Finding the first principle component $\mathbf{w_1}$ such that $Var(z_1)$ is maximized:

$$Var(z_1) = Var(\mathbf{w}^T\mathbf{x}) = E[(\mathbf{w}^T\mathbf{x} - \mathbf{w}^T\boldsymbol{\mu})^2] = \mathbf{w}^TE[(\mathbf{x}-\boldsymbol{\mu})(\mathbf{x}-\boldsymbol{\mu})^T]\mathbf{w} = \mathbf{w}^T\boldsymbol{\Sigma}\mathbf{w}$$

$Cov(\mathbf{x}) = E[(\mathbf{x}-\boldsymbol{\mu})(\mathbf{x}-\boldsymbol{\mu})^T] = \boldsymbol{\Sigma}$

Lagrangian: $\mathbf{w}_1^T\boldsymbol{\Sigma}\mathbf{w}_1 - \alpha(\mathbf{w}_1^T\mathbf{w}_1 - 1)$

Taking derivative:$\boldsymbol{\Sigma}\mathbf{w}_1 = \alpha\mathbf{w}_1$ (eigenvalue equation) $\mathbf{w}_1^T\boldsymbol{\Sigma}\mathbf{w}_1 = \alpha\mathbf{w}_1^T\mathbf{w}_1 = \alpha(\mathbf{w}_1$ is unit)

We choose the eigenvector with the largest eigenvalue for the variance to be maximum.

Second PC: $\mathbf{w}_2^T\boldsymbol{\Sigma}\mathbf{w}_2 - \alpha(\mathbf{w}_2^T\mathbf{w}_2 - 1) - \beta(\mathbf{w}_2^T\mathbf{w}_1 - 0)$

Derivative: $2\boldsymbol{\Sigma}\mathbf{w}_2 - 2\alpha\mathbf{w}_2 - \beta\mathbf{w}_1 = 0$ (times $\mathbf{w}_1^T$ on both side, $\mathbf{w}_1^T\mathbf{w}_2 = 0$

Then $\Sigma\mathbf{w}_2 = \alpha\mathbf{w}_2$, $\mathbf{w}_2$ is the second largest eigenvalue.

Proportion of variance (PoV) Explained:$\frac{\lambda_1,\dots,\lambda_k}{\lambda_1,\dots,\lambda_d}$

Factor Analysis Assume latent factors z_j Sample $\mathcal{X} = \{\mathbf{x}^{(l)}\}$, $E(\mathbf{x}) = \boldsymbol{\mu}$, $Cov(\mathbf{x}) = \boldsymbol{\Sigma}$
Factors z_j, $E[z_j] = 0$, $Var(z_j) = 1$
Noise: ϵ_i: $E[\epsilon_i] = 0$, $Var(\epsilon_i) = \Psi_i$, $Cov(\epsilon_i,\epsilon_j) = 0$
$x_i - \mu_i = \sum_{j=1}^k v_{ij}z_j + \epsilon_i$, assume $\boldsymbol{\mu} = 0$, v_{ij} are called factor loading
$Var(x_i) = \sum_{j=1}^k v_{ij}^2Var(z_j) + Var(\epsilon_i) = \sum_{j=1}^k v_{ij}^2 + \Psi_i$
Covariance Matrix: $\boldsymbol{\Sigma} = Cov(\mathbf{Vz} + \boldsymbol{\epsilon}) = \mathbf{VV}^T + \boldsymbol{\Psi}$
Factor loading: $Cov(\mathbf{x},\mathbf{z}) = \mathbf{V}$
Dim reduc: Given $\mathbf{S}$ as the estimator of $\boldsymbol{\Sigma}$, we want to find $\mathbf{V}$ and $\boldsymbol{\Psi}$ s.t. $\mathbf{S} = \mathbf{VV}^T + \boldsymbol{\Psi}$, $\boldsymbol{\Psi} = diag(\Psi_i)$

Multidimensional Scaling lower dimension preserve pairwise distances.
Sample $\mathcal{X} = \left\{\mathbf{x}^{(l)} \in \mathbb{R}^d\right\}_{l=1}^N$
Squared Euclidean distance between point r and s:

$$d_{rs}^2 = \sum_{j=1}^d (x_j^{(r)} - x_j^{(s)})^2 = b_{rr} + b_{ss} - 2b_{rs}$$

$b_{rs} = \sum_{j=1}^d x_j^{(r)}x_j^{(s)}$, or matrix form $\mathbf{B} = \mathbf{XX}^T$
Constraint: $\sum_l^N x_j^{(l)} = 0, \forall j$, define: $T = \sum_{l=1}^N b_{ll}$
Then $\sum_r d_{rs}^2 = T + Nb_{ss}$, $\sum_r\sum_s = d_{rs}^2 = 2NT$
defining:
$d_{ss}^2 = \frac{1}{N}\sum_r d_{rs}^2$, $d_{rs} = \frac{1}{N}\sum_s d_{rs}^2$, $d_{**}^2 = \frac{1}{N^2}\sum_r\sum_s d_{rs}^2$
So $b_{rs} = \frac{1}{2}(d_{r*}^2 + d_{ss}^2 - d_{**}^2 - d_{rs}^2)$ $\mathbf{B} = \mathbf{XX}^T$ is p.s.d. :

$$\mathbf{B} = \mathbf{CDC}^T = (\mathbf{CD}^{1/2})(\mathbf{CD}^{1/2})^T$$

Ignore small eigenvalues, let $\mathbf{c}_j$ be the k eigenvectors chosen with eigenvalues λ_j, the new dimensions:$z_j^{(l)} = \sqrt{\lambda_j}c_j^{(l)}$

LDA Sample mean after projection:

$$m_1 = \mathbf{w}^T\mathbf{m}_1, m_2 = \mathbf{w}^T\mathbf{m}_2$$

Between class scatter: $(m_1 - m_2) = \mathbf{w}^T\mathbf{S}_B\mathbf{w}$, $\mathbf{S}_B = (\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^T$
Within Class Scatter: $s_1^2 = \mathbf{w}^T\mathbf{S}_1\mathbf{w}$, $\mathbf{S}_1 = \sum_l(\mathbf{x}^{(l)} - \mathbf{m}_1)(\mathbf{x}^{(l)} - \mathbf{m}_1)^Ty^{(l)}$, similarly $\mathbf{S}_2 = \sum_l(\mathbf{x}^{(l)} - \mathbf{m}_2)(\mathbf{x}^{(l)} - \mathbf{m}_2)^T(1 - y^{(l)})$, so

$$s_1^2 + s_2^2 = \mathbf{w}^T\mathbf{S}_w\mathbf{w}$$

, $\mathbf{S}_w = \mathbf{S}_1 + \mathbf{S}_2$

Fisher's LD $J(\mathbf{w}) = \frac{(m_1 - m_2)^2}{s_1^2 + s_2^2} = \frac{\mathbf{w}^T\mathbf{S}_B\mathbf{w}}{\mathbf{w}^T\mathbf{S}_W\mathbf{w}}$, take derivative of J w.r.t. $\mathbf{w}$ setting it to 0: $\mathbf{S}_B\mathbf{w} = \lambda\mathbf{S}_W\mathbf{w}$,
or $\mathbf{S}_W^{-1}\mathbf{S}_B\mathbf{w} = \lambda\mathbf{w}$ (Eigen equation)

$K > 2$ Within-class scatter $\mathbf{S}_i = \sum_l y_i^{(l)}(\mathbf{x}^{(l)} - \mathbf{m}_i)(\mathbf{x}^{(l)} - \mathbf{m}_i)^T$, $y_i^{(l)} = 1$ if $\mathbf{x}^{(l)} \in C_i$
Total class scatter: $\mathbf{S}_W = \sum_{i=1}^K \mathbf{S}_i$
Between Class Scatter: $\mathbf{S}_B = \sum_{i=1}^K N_i(\mathbf{m}_i - \mathbf{m})(\mathbf{m}_i - \mathbf{m})^T$
Optimal is $\mathbf{W}$ that max: $J(\mathbf{W}) = \frac{Tr(\mathbf{W}^T\mathbf{S}_B\mathbf{W})}{Tr(\mathbf{W}^T\mathbf{S}_W\mathbf{W})}$ Corresponds to eigenvectors of $\mathbf{S}_W^{-1}\mathbf{S}_B$

5 Clustering

Mixture Densities $p(\mathbf{x}) = \sum_{i=1}^K p(\mathbf{x}|\mathcal{G}_i)P(\mathcal{G}_i)$ $\mathcal{G}_i$ mixture components, $p(\mathbf{x}|\mathcal{G}_i)$ component densities, $P(\mathcal{G}_i)$ mixing proportions

Gaussian mixture: $p(\mathbf{x}|\mathcal{G}_i) = \mathcal{N}(\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)$,

parameters $\boldsymbol{\Phi} = \{p(\mathcal{G}_i),\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i\}_{i=1}^k$

k-Means Initialize $m_i, i = 1,\dots,k$ (random)
Repeat
For all $\mathbf{x} \in \mathcal{X}$

$$b_i^{(l)} = \begin{cases} 1 & \text{if } i = arg\min_j \|\mathbf{x}^{(l)} - \mathbf{m}_j\| \\ 0 & \text{otherwise} \end{cases}$$

For all $\mathbf{m}_i, i = 1,\dots,k$

$$\mathbf{m}_i = \frac{\sum_l b_i^{(l)}\mathbf{x}^{(l)}}{\sum_l b_i^{(l)}}$$

Until $\mathbf{m}_i$ converges

EM Algorithm

Log likelihood:

$$\mathcal{L}(\boldsymbol{\Phi}|\mathcal{X}) = \log\prod_l p(\mathbf{x}^{(l)}|\boldsymbol{\Phi}) = \sum_l \log\sum_l^k p(\mathbf{x}^{(l)}|\mathcal{G}_i)P(\mathcal{G}_i)$$

where $\boldsymbol{\Phi}$ include $P(\mathcal{G}_i)$ and sufficient statistic $\boldsymbol{\Theta}_i$ of $p(\mathbf{x}^{(l)}|\mathcal{G}_i)$

Complete-data likelihood: $\mathcal{L}_C(\boldsymbol{\Phi}|\mathcal{X},\mathcal{Z})$, $\mathcal{Z}$ hidden variables (not observable)

Expectation of complete data likelihood given $\mathcal{X}$ and the current (iteration t) parameter val $\boldsymbol{\Phi}^t$(E-step)
$\mathcal{Q}(\boldsymbol{\Phi}|\boldsymbol{\Phi}^t) = E\left[\mathcal{L}_C(\boldsymbol{\Phi}|\mathcal{X},\mathcal{Z})|\mathcal{X},\boldsymbol{\Phi}^t\right] = \sum_\mathcal{Z} p(\mathcal{Z}|\mathcal{X},\boldsymbol{\Phi}^t)\log p(\mathcal{Z},\mathcal{X}|\boldsymbol{\Phi})$

m-step:$\boldsymbol{\Phi}^{t+1} = arg\max_{\boldsymbol{\Phi}}\mathcal{Q}(\boldsymbol{\Phi},\boldsymbol{\Phi}^t)$

EM in Gaussian Mixtures Indicator $\mathbf{z}^{(l)}$, $z_i^{(l)} = 1$ if $\mathbf{x}^{(l)}$ belongs to cluster $\mathcal{G}_i$

Gaussian Component densities: $p_i(\mathbf{x}^{(l)}) = p(\mathbf{x}|\mathcal{G}_i) = \mathcal{N}(\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)$

Prior $P(\mathcal{G}_i) = \pi_i$, so: $p(\mathbf{z}^{(l)}) = \prod_{i=1}^k \pi_i^{z_i^{(l)}}$

probability given hidden: $p(\mathbf{x}|\mathbf{z}^{(l)}) = \prod_{i=1}^k p_i(\mathbf{x}^{(l)})^{z_i^{(l)}}$

Joint density: $p(\mathbf{x}^{(l)},\mathbf{z}^{(l)}) = P(\mathbf{z}^{(l)})p(\mathbf{x}^{(l)}|\mathbf{z}^{(l)})$

Complete log:

$$\mathcal{L}_C(\boldsymbol{\Phi}|\mathcal{X},\mathcal{Z}) = \prod_l p(\mathbf{x}^{(l)},\mathbf{z}^{(l)}|\boldsymbol{\Phi}) = \sum_l\sum_i z_i^{(l)}\left[\log\pi_i + \log p_i(\mathbf{x}^{(l)}|\boldsymbol{\Phi})\right]$$

$$\mathcal{Q}(\boldsymbol{\Phi}|\boldsymbol{\Phi}^t) = \sum_l\sum_i E\left[z_i^{(l)}|\mathcal{X},\boldsymbol{\Phi}^t\right]\left[\log\pi_i + \log p_i(\mathbf{x}^{(l)}|\boldsymbol{\Phi})\right]$$

where $E\left[z_i^{(l)}|\mathcal{X},\boldsymbol{\Phi}^t\right] = E\left[z_i^{(l)}|\mathbf{x}^{(l)},\boldsymbol{\Phi}^t\right] = \frac{p(\mathbf{x}|\mathcal{G}_i,\boldsymbol{\Phi}^t)P(\mathcal{G}_i)}{\sum_j p(\mathbf{x}^{(l)}|\mathcal{G}_j,\boldsymbol{\Phi}^t)P(\mathcal{G}_j)} = P(\mathcal{G}_i|\mathbf{x}^{(l)},\boldsymbol{\Phi}^t) = h_i^{(l)}$

For Gaussian components, $\hat{p}_i(\mathbf{x}^{(l)}|\boldsymbol{\Theta}_i) = \mathcal{N}(\mathbf{m}_i,\mathbf{S}_i)$, so

$$h_i^{(l)} = \frac{\pi_i|\mathbf{S}_i|^{-1/2}\exp\left[-\frac{1}{2}(\mathbf{x}^{(l)} - \mathbf{m}_i)^T\mathbf{S}_i^{-1}(\mathbf{x}^{(l)} - \mathbf{m}_i)\right]}{\sum_j\pi_j|\mathbf{S}_j|^{-1/2}\exp\left[-\frac{1}{2}(\mathbf{x}^{(l)} - \mathbf{m}_j)^T\mathbf{S}_j^{-1}(\mathbf{x}^{(l)} - \mathbf{m}_j)\right]}$$

Max Step: $\boldsymbol{\Phi}^{t+1} =$
$arg\max_{\boldsymbol{\Phi}}\left[\sum_l\sum_i h_i^{(l)}\log\pi_i + \sum_l\sum_i h_i^{(l)}\log p_i(\mathbf{x}^{(l)}|\boldsymbol{\Phi})\right]$

Using constraint $\sum_i\pi_i = 1$:

$$\Delta_{\pi_i}\left[\sum_l\sum_i h_i^{(l)}\log\pi_i - \lambda(\sum_i\pi_i - 1)\right] = 0$$

we get $\pi_i = \frac{\sum_l h_i^{(l)}}{N}$

Then solve for $\Delta_{\boldsymbol{\Theta}_i}\sum_l\sum_i h_i^{(l)}\log p_i(\mathbf{x}^{(l)}|\boldsymbol{\Phi}) = 0$

For Gaussian component, $\hat{p}_i(\mathbf{x}^{(l)}|\boldsymbol{\Theta}_i) = \mathcal{N}(\mathbf{m}_i,\mathbf{S}_i)$, so $\mathbf{m}_i^{t+1}$ and S_i^{t+1} is estimator with value in t-th iteration.

6 Non Parametric Method

Nonparametric Density Estimation Sample: $\mathcal{X} = \{x^{(l)}\}_{l=1}^N$, Probability density: $p(X)$, cumulative distribution: $F(X)$

Estimator of $F(X)$: $\hat{F}(x) = \frac{\#\{x^{(l)}leqx)\}}{N}$

Estimator of $p(X)$: $\hat{p}(x) = \frac{1}{h}[\frac{\#\{x^{(l)}+h\le x\} - \#\{x^{(l)}\le x\}}{N}]$, where h is the interval and instances $x^{(l)}$ that fall in this interval are assumed to be close enough

Histogram Estimator bin: $[x_0 + mh, x_0 + (m+1)h]$, x_0 origin, h bin width

$\hat{p}(x) = \frac{\#\{x^{(l)}\text{in the same bin as x}\}}{Nh}$

Naive Estimator: $\hat{p}(x) = \frac{\#\{x - h/2 < x^{(l)} \le x + h/2\}}{Nh}$

Alternative form: $\hat{p}(x) = \frac{1}{Nh}\sum_{t=1}^N w(\frac{x - x^{(l)}}{h})$ with weight function

$$w = \begin{cases} 1 & if|u| < 1/2 \\ 0 & otherwise \end{cases}$$

Kernel Estimator Kernel $K(u) = \frac{1}{2\pi}\exp(-\frac{u^2}{2})$,
Estimator:$\hat{p} = \frac{1}{Nh}\sum_{l=1}^N K(\frac{x - x^{(l)}}{h})$

KNN $\hat{p}(x) = \frac{k}{2Nd_k(x)}$, $d_k(x)$ is the distance from x to the kth nearest instannce.

KNN with kernel:$\hat{p}(\mathbf{x}) = \frac{1}{Nh^d}\sum_{l=1}^N K(\frac{\mathbf{x} - \mathbf{x}^{(l)}}{h})$, with $\int_{\mathbb{R}^d} K(\mathbf{x})d\mathbf{x} = 1$,

Multivariate ellipsoidal Gaussin kernel:
$K(\mathbf{u}) = \frac{1}{(2\pi)^{d/2}|\mathbf{S}|^{1/2}}\exp\left(-\frac{1}{2}\mathbf{u}^T\mathbf{S}^T\mathbf{u}\right)$

Nonparammetric Classification Kernel estimator of class-conditional densities: $\hat{p}(\mathbf{x}|C_i) = \frac{1}{N_ih^d}\sum_{l=1}^N K(\frac{\mathbf{x} - \mathbf{x}^{(l)}}{h})y_i^{(l)}$, $y_i^{(l)} = 1$ if $\mathbf{x}^{(l)}$ is in C_i, and $N = \sum_l y_i^{(l)}$, $\hat{P}(C_i) = \frac{N_i}{N}$

Discriminant:
$g_i(\mathbf{x}) = \hat{p}(\mathbf{x}|C_i)\hat{P}(C_i) = \frac{1}{Nh^d}\sum_{l=1}^N K(\frac{\mathbf{x} - \mathbf{x}^{(l)}}{h})y_i^{(l)}$

KNN classifier $\hat{p}(\mathbf{x}|C_i) = \frac{k_i}{N_iV^k(\mathbf{x})}$, $\hat{P}(C_i|\mathbf{x}) = \frac{k_i}{k}$, Choose C_i if $i = arg\max_j\hat{P}(C_j|\mathbf{x}) = arg\max_j k_j$

Non param regression $y^{(l)} = g(\mathbf{x}^{(l)}) + \epsilon$,
$\hat{g}(x) = \frac{\sum_{l=1}^N K(\frac{x - x^{(l)}}{h})y^{(l)}}{\sum_{l=1}^N K(\frac{x - x^{(l)}}{h})}$

Regularized cost function balance bias and variance:
$\sum_l\left[y^{(l)} - \hat{g}(x^{(l)})\right]^2 + \lambda\int_a^b[\hat{g}''(x)]^2dx$

7 Decision Trees

Classification Trees for node m, N_m training instances, N_m^i instances belong to class C_i, estimate for the probability of class C_i $\hat{P}(C_i|\mathbf{x},m) = \frac{N_m^i}{N_m} = p_m^j$, pure if $p_m^j = 1$

Entropy $\mathcal{T}_m = -\sum_{i=1}^K p_m^j\log_2 p_m^j$, assume $0\log 0 = 0$, largest is $\log_2 K$ when all $p_m^i = 1/K$

Other Impurity Measures Properties: $\phi(1/2,1/2) \geq \phi(p,1-p)$, $\phi(0,1) = \phi(1,0) = 0$, $\phi(p,1-p)$ increase in p on $[0,1/2]$ and decrease o$[1/2,1]$

Best Split Node m, N_{mj} take branch j, if $f_m(\mathbf{x}) = j$, the estimate for the probability of class C_i is: $\hat{P}(C_i|\mathbf{x},m,j) = p_{mj}^i = \frac{N_{mj}^i}{N_{mj}}$,total impurity after split:
$\mathcal{T}_m' = -\sum_{i=1}^n\frac{N_{mj}}{N_m}\sum_{i=1}^K p_{mj}^i\log p_{mj}^i$

Regression Trees $b_m(\mathbf{x}) = 1$ if in $\mathcal{X}_m$,
estimated value at node m: $g_m = \frac{\sum_l b_m(\mathbf{x}^{(l)})y^{(l)}}{\sum_l b_m(\mathbf{x}^{(l)})}$,
mean squaresquare error after split: $E_m = \frac{1}{N_m}\sum_l(y^{(l)} - g_m)^2b_m(\mathbf{x}^{(l)})$, tree expansion: $b_{mj}(\mathbf{x}) = 1$if $\mathbf{x} \in \mathcal{X}_{mj}$,
estimate val in branch j: $g_{mj} = \frac{\sum_l b_{mj}(\mathbf{x}^{(l)})y^{(l)}}{\sum_l b_{mj}(\mathbf{x}^{(l)})}$,
error after split $E_m' = \frac{1}{N_m}\sum_j\sum_l(y^{(l)} - g_{mj}^2b_{mj}(\mathbf{x}^{(l)})$
Best split: split that resuts in smallest error, or worst possible error.

Pruning Prepruning stop split when the number of instances reaching a node is belong a certain percentage. Post pruning: Replace subtree by a leaf node, if the leaf node does not perform worse, the subtree is pruned and replaced by leaf.

8 Linear Discrimination

Linear Discriminant functions $g_i(\mathbf{x}|\mathbf{w}_i,w_{i0}) = \mathbf{w}_i^T\mathbf{x} + w_{i0}$, generalized use base function $g_i(\mathbf{x}) = \sum_{j=1}^k w_j\phi_{ij}(\mathbf{x})$

Two class $g(\mathbf{x}) = g_1(\mathbf{x}) - g_2(\mathbf{x})$, C_1 if $g(\mathbf{x}) > 0$

Geometry Interpretation Express any point as $\mathbf{x} = \mathbf{x}_p + r\frac{\mathbf{w}}{\|\mathbf{w}\|}$, x_p projection of $\mathbf{x}$ onto hyperplane. r distance from $\mathbf{x}$ to hyper plane.plane, we have $r = \frac{g(\mathbf{x})}{\|\mathbf{w}\|}$

Multi classes K discriminant$g_i(\mathbf{x}|\mathbf{w}_i, w_{i0}) = \mathbf{w}_i^T + w_{i0}$, linear separable $g_i(\mathbf{x}|\mathbf{w}_i, w_{i0}) > 0$ if $\mathbf{x} \in C_i$, choose C_i if $g_i(\mathbf{x}) = \max_{j=1}^K g_i(\mathbf{x})$

Pairwise Separation Discriminant function for class i and j:

$$g_{ij}(\mathbf{x}|\mathbf{w}_{ij}, w_{ij0}) = \mathbf{w}_{ij}^T\mathbf{x} + w_{ij0} = \begin{cases} > 0 & \text{if } \mathbf{x} \in C_i \\ \le 0 & \text{if } \mathbf{x} \in C_j \\ \text{don't care} & \text{if } \mathbf{x} \in C_k, k \ne i, k \ne j \end{cases}$$

Logistic Discrimination

Two Classes Assume that the log likelihood ratio is linear:

$$\log \frac{p(\mathbf{x}|C_1)}{p(\mathbf{x}|C_2)} = \mathbf{w}^T\mathbf{x} + w_0^o$$

Using Baye's rule we have;

$$\begin{aligned} logit(P(C_1|x)) &= \log \frac{p(C_1|\mathbf{x})}{1 - p(C_1|\mathbf{x})} \\ &= \log \frac{(\mathbf{x}|C_1)}{p(\mathbf{x}|C_2)} + \log \frac{p(C_1)}{p(C_2)} \\ &= \mathbf{w}^T\mathbf{x} + w_0 \end{aligned}$$

where $w_0 = w_0^o + \log \frac{P(C_1)}{P(C_2)}$ Rearranging terms:

$$\begin{aligned} y &= sigmoid(\mathbf{w}^T\mathbf{x} + w_0) \\ &= \hat{P}(C_1|\mathbf{x}) = \frac{1}{1 + \exp\left[-(\mathbf{w}^T\mathbf{x} + w_0)\right]} \end{aligned}$$

As our estimator of $P(C_1|\mathbf{x})$

Gradient Decent In the discriminant-based approach, the parameters are those of the discriminants, and they are *optimized to minimize the classification error*

Error w denotes the set of parameters and $E(\mathbf{w}|\mathcal{X})$ is the error parameters $\mathbf{w}$ on the given training set $\mathcal{X}$, we look for:

$$\mathbf{w}* = arg \min_{\mathbf{w}} E(\mathbf{w}|\mathcal{X})$$

No analytical solution

Gradient Vector When $E(\mathbf{w})$ is a differentiable function of a vector of variables, we have the gradient vector composed of the partial derivatives:

$$\nabla_{\mathbf{w}} E = \left[\frac{\partial E}{\partial w_1}, \frac{\partial E}{\partial w_2}, \ldots, \frac{\partial E}{\partial w_d}\right]^T$$

Gradient Descent starts from a *random* $\mathbf{w}$, at each step, update w in a *opposite direction* of the gradient:

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i}, \forall i$$

$$w_i = w_i + \Delta w_i$$

η is *step size*, or *learning factor*

When we get to minimum, the derivative is 0 and the procedure terminates.

This indicates that the procedure finds the nearest minimum that can be *local minimum*. There is no guarantee of finds the nearest minimum that can be a local minimum

Learning parameters Given a sample of two classes, $\mathcal{X} = \mathbf{x}^{(l)}, \mathbf{r}^{(l)}$, where $\mathbf{r}^{(l)} = 1$ if $\mathbf{x} \in C_1$

We assume $\mathbf{r}^{(l)}$, given $\mathbf{x}^{(l)}$ is Bernoulli with probability $y^{(l)} = p(C_1|\mathbf{x}^{(l)})$:

$$\mathbf{r}^{(l)}|\mathbf{x}^{(l)} \sim Bernoulli(y^{(l)})$$

Note that in this discriminant-based approach, we model directly $\mathbf{r}|\mathbf{x}$ The sample likelihood is:

$$L(\mathbf{w}, w_0|\mathcal{X}) = \prod_t (y^{(l)})^{(r^{(l)})}(1 - y^{(l)})^{1 - r^{(l)}}$$

We can always turn it in an error function to minimize: $E = -\log L$ So we have *cross-entropy*:

$$E(\mathbf{w}, w_0|\mathcal{X}) = -\sum_t \mathbf{r}^{(l)} \log y^{(l)} + (1 - r^{(l)}) \log(1 - y^{(l)})$$

We use gradient descent to minimize cross-entropy If $y = sigmoid(a) = \frac{1}{1+exp(-a)}$, its derivative is given as:

$$\frac{dy}{da} = y(1 - y)$$

and we get the following update equations:

$$\begin{aligned} \Delta w_j &= -\eta \frac{\partial E}{\partial w_j} = \eta \sum_t \left(\frac{r^{(l)}}{y^{(l)}} - \frac{1 - r^{(l)}}{1 - y^{(l)}} x_j^{(l)}\right. \\ &= \eta \sum (r^{(l)} - y^{(l)}) x_j^{(l)}, \ = 1, \ldots, d \end{aligned}$$

$$\Delta w_0 = -\eta \frac{\partial E}{\partial w_0} = \eta \sum (r^{(l)} - y^{(l)})$$

Multiple Classes Take one of the classes C_k, as reference class and assume that:
$\log \frac{p(\mathbf{x}|C_i)}{p(\mathbf{x}|C_K)} = \mathbf{w}_i^T\mathbf{x} + w_{i0}^o$, $i = 1, \ldots, K - 1$
Then we have:

$$\frac{P(C_i|\mathbf{x})}{P(C_K|\mathbf{x})} = \exp[\mathbf{w}_i^T\mathbf{x} + w_{i0}]$$

With $w_{i0} = w_{i0}^o + \log \frac{P(C_i)}{P(C_K)}$

Generalization of sigmoid Summing over i we can deduce:
$P(C_K|\mathbf{x}) = \frac{1}{1 + \sum_{i=1}^{K-1} \exp(\mathbf{w}_i^T\mathbf{x} + w_{i0})}$
$P(C_i|\mathbf{x}) = \frac{\exp(\mathbf{w}_i^T + w_{i0})}{1 + \sum_{j=1}^{K-1} \exp(\mathbf{w}_j^T\mathbf{x} + w_{j0})}$, $k = 1, \ldots, K - 1$

Softmax Treat all classes uniformly
$y_i = \hat{P}(C_i|\mathbf{x}) = \frac{\exp(\mathbf{w}_i^T + w_{i0})}{\sum_{j=1}^K \exp(\mathbf{w}_j^T + w_{j0})}$, $i = 1, \ldots, K$

Learning $\frac{\partial y_i}{\partial a_j} = y_i(\delta_{ij} - y_i)$, $\Delta \mathbf{w}_j = \eta \sum_l (r_j^{(l)} - y_j^{(l)})\mathbf{x}^{(l)}$, $\Delta w_{j0} = \eta \sum_l (r_j^{(l)} - y_j^{(l)})$

Regression for two-class Classification $r^{(l)} = y^{(l)} + \epsilon$, $r^{(l)} \in \{0, 1\}$, $\epsilon \sim \mathcal{N}(0, \sigma^2)$, $y^{(l)} = sigmoid(\mathbf{w}^T\mathbf{x}^{(l)} + w_0)$,

Likelihood:$L(\mathbf{w}, w_0|\mathcal{X}) = \prod_l \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{(r^{(l)} - y^{(l)})^2}{2\sigma^2}\right]$,

Error: $E(\{\mathbf{w}_i, w_{i0}\}_i |\mathcal{X}) = \frac{1}{2}\sum_l (r^{(l)} - y^{(l)})^2$,

Learning $\Delta\mathbf{w} = \eta \sum_l (r^{(l)} - y^{(l)}) y^{(l)}(1 - y^{(l)})\mathbf{x}^{(l)}$,
$\Delta w_0 = \eta \sum_l (r^{(l)} - y^{(l)}) y^{(l)}(1 - y^{(l)})$

$K > 2$ **classes** $\mathbf{r}^{(l)} = \mathbf{y}^{(l)} + \boldsymbol{\epsilon}$,
$\epsilon \sim \mathcal{N}_K(0, \sigma^2\mathbf{I}_K)$, $y_i^{(l)} = \frac{1}{1 + \exp\left[-(\mathbf{w}_i^T\mathbf{x}^{(l)} + w_{i0})\right]}$,
Likelihood:
$L(\{\mathbf{w}_i, w_{i0}\}_i|\mathcal{X}) = \prod_l \frac{1}{(2\pi)^{K/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\left[-\frac{\|\mathbf{r}^{(l)} - \mathbf{y}^{(l)}\|^2}{2\sigma^2}\right]$
Error Func: $E(\{\mathbf{w}_i, w_{i0}\}|\mathcal{X}) = \frac{1}{2}\sum_l \|\mathbf{r}^{(l)} - \mathbf{y}^{(l)}\|^2$
Learning $\Delta\mathbf{w}_i = \eta \sum_l (r_i^{(l)} - y_i^{(l)}) y_i^{(l)}(1 - y_i^{(l)})\mathbf{x}^{(l)}$,
$\Delta w_0 = \eta \sum_l (r_i^{(l)} - y_i^{(l)}) y_i^{(l)}(1 - y_i^{(l)})$

9 Multilayer Perceptrons

Perceptron The output y is a **weighted sum** of the input $\mathbf{x} = (x_0, x_1, \ldots, x_d)^T$:

$$y = \sum_{j=1}^d w_j x_j + w_0 = \mathbf{w}^T\mathbf{x}$$

where x_0 is a special *bias unit* with $x_0 = 1$ and $\mathbf{w} = (w_0, w_1, \ldots, w_d)^T$ are called **connection weight or synaptic weights**

to implement a linear discriminant function, we need threshold function:

$$s(a) = \begin{cases} 1 & \text{if } a > 0 \\ 0 & \text{otherwise} \end{cases}$$

to define the following decision rule:

$$\text{Choose} \begin{cases} C_1 & \text{if } s(\mathbf{w}^T\mathbf{x}) = 1 \\ C - 2 & \text{otherwise} \end{cases}$$

Use **sigmoid** instead of threshold to gain differentiability:

$$y = sigmoid(\mathbf{w}^T\mathbf{x}), \text{ where } sigmoid(a) = \frac{1}{1 + \exp(-a)}$$

The output may be interpreted as the *posterior probability* that the input $\mathbf{x}$ belongs to C_1

$K > 2$ **Outputs** K perceptrons, each with a weight vector $\mathbf{w}_i$,

$$y_i = \sum_{j=1}^d w_{ij} x_j + w_{i0} = \mathbf{w}_i^T\mathbf{x}, \text{ or } \mathbf{y} = \mathbf{W}\mathbf{x}$$

where w_{ij} is the weight from input x_j to output y_i and each row of the $K \times (d+1)$ matrix $\mathbf{W}$ is the weight vector of one perceptron.

Choose C_i if $y_i = \max_k y_k$

Posterior probability, use softmax to define y_i as:

$$y_i = \frac{\exp(\mathbf{w}_i^T)}{\sum_{k=1}^K \exp(\mathbf{w}_k^T\mathbf{x})}$$

Stochastic Gradient Descent Gradient descent for online learning, for regression, the error on a single instance

$$E^{(l)}(\mathbf{w}|\mathbf{x}^{(l)}, r^{(l)}) = \frac{1}{2}(r^{(l)} - y^{(l)})^2 = \frac{1}{2}\left[r^{(l)} - (\mathbf{w}^T\mathbf{x}^{(l)})\right]^2$$

, gives the online update rule:
$\Delta w_j^{(l)} = \eta(r^{(l)} - y^{(l)}) x_j^{(l)}$,$\eta$ is step size.

For Binary classification:
Likelihood: $L = (y^{(l)})^{r^{(l)}}(1 - y^l)^{1 - r^{(l)}}$
Cross Entropy: $E^{(l)}(\mathbf{w}|\mathbf{x}^{(l)}, r^{(l)}) =$
$-\log L = -r^{(l)} \log y^{(l)} - (1 - r^{(l)}) \log(1 - y^{(l)})$
Online update rule:
$\Delta w_j^{(l)} = \eta(r^{(l)} - y^{(l)}) x_j^{(l)}$

$K > 2$ $y_i^{(l)} = \frac{\exp(\mathbf{w}_i^T\mathbf{x}^{(l)})}{\sum_k \exp(\mathbf{w}_k^T\mathbf{x}^{(l)})}$
Likelihood: $L = \prod_i (y_i^{(l)})^{r_i^{(l)}}$
Cross Entropy: $E^{(l)}(\{\mathbf{w}_i\}|\mathbf{x}^{(l)}, \mathbf{r}^{(l)}) = -\sum_i r_i^{(l)} \log y_i^{(l)}$
Online update rule: $\Delta w_{ij}^{(l)} = \eta(r_i^{(l)} - y^{(l)}) x_j^{(l)}$

Multilayer Perceptron MLP has a hidden layer between the input and output. Input to hidden: $z_h = sigmoid(\mathbf{w}_h^T\mathbf{x}) = \frac{1}{1 + \exp\left[-(\sum_{j=1}^d w_{hj} x_j + w_{h0})\right]}$
Hidden to output: $y_i = \mathbf{v}_i^T\mathbf{z} = \sum_{h=1}^H v_{ih} z_h + v_{i0}$
Backward, hidden-to-output weight: treating hidden unit as input
input-to-hidden, chain rule: $\frac{\partial E}{\partial w_{hj}} = \frac{\partial E}{\partial y_i}\frac{\partial y_i}{\partial z_h}\frac{\partial z_h}{\partial w_{hj}}$

MLP for Nonlinear Regression(Multi output) outputs:
$y_i^{(l)} = \sum_{h=1}^H v_{ih} z_h^{(l)} + v_{i0}$,
$z_h^{(l)} = sigmoid(\mathbf{w}_h^T\mathbf{x}^{(l)})$
Error function: $E(\mathbf{W}, \mathbf{v}|\mathcal{X}) = \frac{1}{2}\sum_l \sum_i (r_i^{(l)} - y_i^{(l)})^2$
Update rule for second layer: $\Delta v_{ih} = \eta \sum_l (r_i^{(l)} - y_i^{(l)}) z_h^{(l)}$
Update rule for first layer: $\Delta w_{hj} = -\eta \frac{\partial E}{\partial w_{hj}} =$
$\eta \sum_l \left[\sum_i (r_i^{(l)} - y_i^{(l)}) v_{ih}\right] z_h^{(l)}(1 - z_h^{(l)}) x_j^{(l)}$

MLP for NonLinear Multi-class Discrimination
Outputs: $y_i^{(l)} = \frac{\exp(o_i^{(l)})}{\sum_k \exp(o_k^{(l)})}$, $o_i^{(l)} = \sum_h^H v_{ih} z_h^{(l)} + v_{i0}$
Error Function: $E(\mathbf{W}, \mathbf{V}|\mathcal{X}) = -\sum_l \sum_i r_i^{(l)} \log y_i^{(l)}$, update rules are the same us regression.

10 Support Vector Machine

Optimal Separating Hyperplane results from *statistical learning* theory showing that the separating hyperplane with the *largest margin* generalizes best.

Hard-margin case points are **linearly separable** with proper scaling of $\mathbf{w}$ and w_0, the points closet to the hyper plane satisfy $|\mathbf{w}^T\mathbf{x} + w_0| = 1 \Rightarrow$ **canonical separating hyperplane.**

The one max the margin: **canonical optimal separating hyperplane**

$\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ be two closest point on each side:

$$\mathbf{w}^T\mathbf{x}^{(1)} + w_0 = +1$$
$$\mathbf{w}^T\mathbf{x}^{(2)} + w_0 = -1$$

So $\mathbf{w}^T(x^{(1)} - \mathbf{w}^{(2)}) = 2$, the margin is given by:

$$\gamma = \frac{1}{2}\frac{\mathbf{w}^T(\mathbf{x}^{(1)} - \mathbf{x}^{(2)})}{\|\mathbf{w}\|} = \frac{1}{\|\mathbf{w}\|}$$

Maximizing the margin is equivalent to minimizing $\|\mathbf{w}\|$

Inequality constraint

$$\mathbf{w}^T\mathbf{x}^{(l)} + w_0 \begin{cases} \ge +1 & \text{if } y^{(l)} = +1 \\ \le -1 & \text{if } y^{(l)} = -1 \end{cases}$$

Equivalent to:

$$y^{(l)}(\mathbf{w}^T\mathbf{x}^{(l)} + w_0) \ge 1$$

Primal Optimization

$$\begin{aligned} &\text{Minimize} && \frac{1}{2}\|\mathbf{w}\|^2 \\ &\text{Subject to} && y^{(l)}(\mathbf{w}^T\mathbf{x}^{(l)} + w_0) \ge 1, \forall l \end{aligned}$$

Lagrangian

$$\begin{aligned} L_p(\mathbf{w}, w_), \alpha_l) &= \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{l=1}^N \alpha_l \left[y^{(l)}(\mathbf{w}^T\mathbf{x}^{(l)} + w_0) - 1\right] \\ &= \frac{1}{2}\mathbf{w}^T\mathbf{w} - \mathbf{w}^T \sum_l \alpha_l y^{(l)}\mathbf{x}^{(l)} - w_0 \sum_l \alpha_l y^{(l)} + \sum_l \alpha_l \end{aligned}$$

Eliminating Primal Var

$$\frac{\partial L_p}{\partial \mathbf{w}} = 0 \Rightarrow \mathbf{w} = \sum_l \alpha_l y^{(l)}\mathbf{x}^{(l)}$$

$$\frac{\partial L_p)}{\partial w_0} = 0 \Rightarrow \sum_l \alpha_l y^{(l)} = 0$$

Dual Optimization Problem Setting gradient of L_p w.r.t. $\mathbf{w}$ and w_0 to 0, then plugging, we get duel optimization problem:

$$\begin{aligned} &\text{Maximize} && \sum_l \alpha_l - \frac{1}{2}\sum_l \sum_{l'} \alpha_l \alpha_{l'} y^{(l)} y(l')(\mathbf{x}^{(l)})^T\mathbf{x}^{(l')} \\ &\text{subject to} && \sum_l \alpha_l y^{(l)} = 0 \text{and } \alpha_l \ge 0, \forall l \end{aligned}$$

Support Vector Most of the Dual Variables vanish with $\alpha_l = 0$. They are points lying beyond the margin, Support vectors: $\mathbf{x}^{(l)}$ with $\alpha_l > 0$.
Computation of primal variables: $\mathbf{w} = \sum_{l=1}^N \alpha_l y^{(l)}\mathbf{x}^{(l)} = \sum_{\mathbf{x}^{(l)} \in SV} \alpha_l y^{(l)}\mathbf{x}^{(l)}$
Support vector on margin:$y^{(l)}(\mathbf{w}^T\mathbf{x}^{(l)} + w_0) = 1$ or $w_0 = y^{(l)} - \mathbf{w}^T\mathbf{x}^{(l)}$, Then:

$$w_0 = \frac{1}{|SV|} \sum_{x^{(l)} \in SV} (y^{(l)} - \mathbf{w}^T\mathbf{x}^{(l)})$$

Discriminant Function $g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0$ (plug in $\mathbf{w}$ and w_0 above), choose C_1 if $g(\mathbf{x}) > 0$

$K > 2$ An SVM $g_i(\mathbf{x})$ is learned for each two-class problem. Choose C_j if $j = arg \max_k g_k(\mathbf{x})$

Slack Variables Relaxed constraint:
$y^{(l)}(\mathbf{w}^T\mathbf{x}^{(l)} + w_0) \ge 1 - \zeta_l$,
minimize: $C\sum_l \zeta_l + \frac{1}{2}\|\mathbf{w}\|^2$
Lagrangian(Primal):
$L_p = \frac{1}{2}\|\mathbf{w}\|^2 +$
$C\sum_l \zeta_l - \sum_{l=1}^N \alpha_l \left[y^{(l)}(\mathbf{w}^T\mathbf{x}^{(l)} + w_0) - 1 + \zeta_l\right] - \sum \mu_l \zeta_l$,
C is the regularization parameter, μ_l is the new Lagrange multipiler to guarantee that $\zeta \ge 0$
Dual: Max $\sum_l \alpha_l - \frac{1}{2}\sum_l \sum_{l'} \alpha_l \alpha_{l'} y^{(l)} y^{(l')}(\mathbf{x}^{(l)})^T\mathbf{x}^{(l')}$
subject to: $\sum_l \alpha_l y^{(l)} = 0$ and $0 \le \alpha_l \le C, \forall l$

Kernel Functions Dual Problem:
Max: $\sum_l \alpha_l - \frac{1}{2}\sum_l \sum_{l'} \alpha_l \alpha_{l'} y^{(l)} y^{(l')} \phi(\mathbf{x}^{(l)})^T \phi(\mathbf{x}^{(l')})$
subject to: $\sum_l y^{(l)} = 0$ and $0 \le \alpha_l \le C, \forall l$
Kernel: $K(\mathbf{x}^{(l)}, \mathbf{x}^{(l')} = \phi(\mathbf{x}^{(l)})^T \phi(\mathbf{x}^{(l')})$

$\epsilon - InsensitiveLoss$

$$\varepsilon_\epsilon(y^{(l)}, f(\mathbf{x}^{(l)}) = \begin{cases} 0 & \text{if } |y^{(l)} - f(\mathbf{x}^{(l)})| \le \epsilon \\ |y^{(l)} - f(\mathbf{x}^{(l)})| - \epsilon & otherwise \end{cases}$$

Primal: Max $\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_l (\zeta_l^+ + \zeta_l^-)$
Subject to: $y^{(l)} - (\mathbf{w}^T\mathbf{x}^{(l)} + w_0) \le \epsilon + \zeta_l^+, \forall l$
$(\mathbf{w}^T\mathbf{x}^{(l)} + w_0) - y^{(l)} \le \epsilon + \zeta_l^-, \forall l$, $\zeta_l^+, \zeta_l^- \ge 0, \forall l$
Two slack variables: ζ_l^+ such that $y^{(l)} - (\mathbf{w}^T\mathbf{x}^{(l)} + w_0) > \epsilon$, ζ_l^- such that $(\mathbf{w}^T\mathbf{x}^{(l)} + w_0) - y^{(l)} > \epsilon$

11 Performance Evaluation and Comparison

K-Fold Cross Validation The data set $\mathcal{X}$ is randomly partitioned into K equal-sized subsets $\mathcal{X}_i$, **Stratification** the class distribution different subsets are kept roughly the same.
Use $\mathcal{X}_1, \ldots, \mathcal{X}_K$ as validation sets, and the remaining as training respectively.
If N is small, K should be large to allow large enough training sets.
Leave one out: one instance if left out as validation and $N - 1$ for training.

5×2 **Cross Validation** For each fold i, Split into two equal-sized parts, $\mathcal{X}_i^{(1)}$, $\mathcal{X}_i^{(2)}$, 10 training/validation set pairs: $\mathcal{T}_1 = \mathcal{X}_1^{(1)}, \mathcal{V}_1 = \mathcal{X}_1^{(2)}, \mathcal{T}_2 = \mathcal{X}_1^{(2)}, \mathcal{V}_2 = \mathcal{X}_1^{(1)}, \mathcal{T}_3 = \mathcal{X}_2^{(1)}, \mathcal{V}_3 = \mathcal{X}_2^{(2)}, \mathcal{T}_4 = \mathcal{X}_2^{(2)}, \mathcal{V}_4 = \mathcal{X}_2^{(1)}, \ldots$

Bootstrapping Generates new samples, each of size N, by drawing randomly with replacement. Multiple bootstrap sammples are used to max the change that the system is trained on all instances.

Error Measure TP: True positive, FP: False postivie, FN:False Negative, TN: True Negative, Error rate: $\frac{|FN| + |FP|}{N}$

Receiver Operating characteristics curve Hit rate: $\frac{|TP|}{|TP| + |FN|}$ False alarm rate: $\frac{|FP|}{|FP| + |TN|}$, area under the curve is often used.

Point vs Interval estimator Point specifies a value for θ, intervale specifies an interval within which θ lines with a certain degree of confidence.

Confidence Interval Define z_α,
Two sided: $P(\mathcal{Z} > z_\alpha) = \alpha$, then for level of confidence $1 - \alpha$,

$$P(-z_{\alpha/2} < \sqrt{N}\frac{m - \mu}{\sigma} < z_{\alpha/2}) = 1 - \alpha$$

One-sided $P(m - z_{alpha}\frac{\sigma}{\sqrt{N}} < \mu) = 1 - \alpha$

t Distribution σ^2 replaced by sample var $S^2 = \frac{\sum_l (x^{(l)} - m)^2}{N - 1}$, $\sqrt{N}(m - \mu)/S$ follows a t distribution with $N - 1$ degree of freedom: $\sqrt{N}\frac{m - \mu}{S} \sim t_{N-1}$ For $\alpha \in (0, 1/2)$,

$$P(m - t_{\alpha/2, N-1}\frac{S}{\sqrt{N}} < \mu < m + t_{\alpha/2, N-1}\frac{S}{\sqrt{N}}) = 1 - \alpha$$

Hypothesis Testing Test some hypothesis concerning the parameters.

1. Define a statistics obey a certain distribution
2. Random sample is consistent with the hypothesis under consideration, accept(not reject)
3. Otherwise reject

Given normal $\mathcal{N}(\mu, \sigma^2)$, σ known, μ unknown. **Null Hypothesis**:$H_0 : \mu = \mu_0$, $H_1 : \mu \ne \mu_0$ Accept H_0 if the sample mean is not too far from μ_0. Accept H_0 with level of significance α if μ_0 lies in the $100(1 - \alpha)$,

$$\sqrt{N}\frac{m - \mu_0}{\sigma} \in (-z_{\alpha/2}, z_{\alpha/2})$$

Error Type I, rejected when it is correct, α defines how much type I can tolerate. Type II accepted when incorrect.

One Sided $H_0 : \mu \le \mu_0$, $H_1 : \mu > \mu_0$, accept $\frac{\sqrt{N}(m - \mu_0)}{\sigma} \in (-\infty, z_\alpha)$

t-test If σ^2 is not known, $H_0 : \mu = \mu_0$, $H_1 : \mu \ne \mu_0$, accept at α if

$$\sqrt{N}\frac{m - \mu_0)}{S} \in (-t_{\alpha, N-1}, t_{\alpha, N-1})$$

Binomial Test Single test/validation:Classifier is trained on a training set $\mathcal{T}$ and tested on validation set $\mathcal{V}$. p be the probability that the classifier makes a misclassification error..Define $x^{(l)}$ as a B ernoulli variable to denote the correctness, $x^{(l)} = 1$ with probability p and $x^{(l)} = 0$ with probability $1 - p$, point estimation: $\hat{p} = \frac{\sum_l x^{(l)}}{|\mathcal{V}|} = \frac{\sum_l x^{(l)}}{N}$

Hypothesis test: $H_0 : p \le p_0$ vs. $H_1 : p > p_0$, $X = \sum_{l=1}^N x^{(l)}$ denote the number of errors on $\mathcal{V}$.,

$$P(X = j) = \binom{N}{j} p^j (1 - p)^{N-j}$$

Under the null hypothesis $p \le p_0$, so the probability that there are e errors or less is:

$$P(X \le e) = \sum_{j=1}^e \binom{N}{j} p_0^j (1 - p_0)^{N-j}$$

Binomial test: accept H_0 if $P(X \le e) < 1 - \alpha$

Paired t Test Multiple training/validation set pairs: run the algorithm K times on K T/V set pairs, we get K error probabilities, $p_i, i = 1, \ldots, K$ on K validation sets.

Paired t test to determine whether to accept the null hypothesis H_0 that the classifier has error probability p_0 or less at significane level α.

$$x_i^{(l)} = \begin{cases} 1 & \text{if classifier trained on } \mathcal{V}_i \text{ makes an error on instance} \\ 0 & \text{otherwise} \end{cases}$$

Test statistic: $\sqrt{K}\frac{(m - p_0)}{S} \sim t_{K-1}$, where $p_i = \frac{\sum_{l=1}^N x_i^{(l)}}{N}$, $m = \frac{\sum_{i=1}^K p_i}{K}$, $S^2 = \frac{\sum_{i=1}^K (p_i - m)^2}{K - 1}$

K-Fold Cross-Validated Paired *t* Test Given two classification algorithms and a data set, we want to compare and test whether the two algorithms constuct classifiers that have the same expected error rate on a new instance.

K-fold cross validation is used to obtain K training/validation set pairs $\{(\mathcal{T}_i, \mathcal{V}_i)\}_{i=1}^K$, run two algorithms, error probabilities p_i^1 and p_i^2, define $p_i = p_i^1 - p_i^2$, p_i with mean μ

$H_0 = \mu = 1$, $H_1 = \mu \ne 0$. Test statistic: $\sqrt{K}\frac{(m - 0)}{S} \sim t_{K-1}$, where $m = \frac{\sum_{i=1}^K p_i}{K}$, $S^2 = \frac{\sum_{i=1}^K (p_i - m)^2}{K - 1}$, accept H_0 at significance level α if $\sqrt{K}m/S \in -t_(\alpha/2, K - 1, t_{\alpha/2, K-1})$

12 Ensemble Learning

Combining Base Learners Multiexpert combination method: base learners work in parallel, give decision and combined to give a final.

Multistage combinations: Base learners work serially, sorted increasing complexity, complex is used when simple is not confident.

Voting Takes a convex combination of the base learners:

$$y = f(d_1, \ldots, d_L|\boldsymbol{\Phi}) = \sum_{j=1}^L w_j d_j(\mathbf{x})$$

, with $w_j \ge 0$ and $\sum_{j=1}^L w_j = 1$, $\boldsymbol{\Phi} = (w_1, \ldots, w_L)^T$ are the parameters and y is the final prediction.

Voting for Classification for class C_i $y_i = \sum_{j=1}^L w_j d_{ij}(\mathbf{x})$, where d_{ij} is the vote of learner j for C_i

Simple voting $w_j = \frac{1}{L}$

Bayesian model combination: $P(C_i|x) = \sum_{models \mathcal{M}_j} P(C_i|x, \mathcal{M}_j) P(\mathcal{M}_|)$, weight w_j can be seen as approximation of the prior $P(\mathcal{M}_j)$, Analysis, as L increase, bias does not change but the variance decreases.

Bagging bootstrap aggregation, a voting method whereby the base learners are made different by training on slightly different training sets.

AdaBoost Modifies the probabilities of drawing instances for classifier training as a function of the error of the previous base learner.

Training:
For all $\{x^t, r^t\}_{t=1}^N \in \mathcal{X}$, initialize $p_1^t = 1/N$
For all base-learners $j = 1, \ldots, L$
Randomly draw $\mathcal{X}_j$ from $\mathcal{X}$ with probabilities p_j^t
Train d_j using $\mathcal{X}_j$
For each (x^t, r^t), calculate $y_j^t \leftarrow d_j(x^t)$
Calculate error rate: $\epsilon_j \leftarrow \sum_t p_j^t \cdot 1(y_j^t \ne r^t)$
If $\epsilon_j > 1/2$, then $L \leftarrow j - 1$; stop
$\beta_j \leftarrow \epsilon_j/(1 - \epsilon_j)$
For each (x^t, r^t), decrease probabilities if correct:
If $y_j^t = r^t$ $p_{j+1}^t \leftarrow \beta_j p_j^t$ Else $p_{j+1}^t \leftarrow p_j^t$
Normalize probabilities:
$Z_j \leftarrow \sum_t p_{j+1}^t$; $p_{j+1}^t \leftarrow p_{j+1}^t/Z_j$
Testing:
Given x, calculate $d_j(x), j = 1, \ldots, L$
Calculate class outputs, $i = 1, \ldots, K$:
$y_i = \sum_{j=1}^L \left(\log \frac{1}{\beta_j}\right) d_{ji}(x)$

13 Matrices Properties

Basic Matrix $(\mathbf{AB})^T = \mathbf{B}^T\mathbf{A}^T$, $(\mathbf{AB})^{-1} = \mathbf{B}^{-1}\mathbf{A}^{-1}$, $(\mathbf{A}^T)^{-1} = (\mathbf{A}^{-1})^T$, $\mathbf{P}^{-1} + \mathbf{B}^T\mathbf{R}^{-1}\mathbf{B})^{-1}\mathbf{B}^T\mathbf{R}^{-1} = \mathbf{PB}^T(\mathbf{BPB}^T + \mathbf{R})^{-1}$,

Traces and Determinants $Tr(\mathbf{AB}) = Tr(\mathbf{BA})$, $Tr(\mathbf{ABC}) = Tr(\mathbf{CBA}) = Tr(\mathbf{BCA})$, $|\mathbf{A}^{-1}| = \frac{1}{|\mathbf{A}|}$, $\mathbf{a}^T\mathbf{A}\mathbf{a} = Tr(\mathbf{A}\mathbf{a}\mathbf{a}^T)$

Matrix Derivatives $\frac{\partial}{\partial \mathbf{x}}(\mathbf{x}^T\mathbf{a}) = \frac{\partial}{\partial \mathbf{x}}(\mathbf{a}^T\mathbf{x}) = \mathbf{a}$, $\frac{\partial}{\partial \mathbf{x}}(\mathbf{AB}) = \frac{\partial \mathbf{A}}{\partial \mathbf{x}}\mathbf{B} + \mathbf{A}\frac{\partial \mathbf{B}}{\partial \mathbf{x}}$, $\frac{\partial}{\partial x}(\mathbf{A}^{-1}) = -\mathbf{A}^{-1}\frac{\partial \mathbf{A}}{\partial x}\mathbf{A}^{-1}$, $\frac{\partial}{\partial x} \ln|\mathbf{A}| = Tr(\mathbf{A}^{-1}\frac{\partial \mathbf{A}}{\partial x})$, $\frac{\partial}{\partial A_{ij}} Tr(\mathbf{AB}) = B_{ji}$, $\frac{\partial}{\partial \mathbf{A}} Tr(\mathbf{AB}) = \mathbf{B}^T$, $\frac{\partial}{\partial \mathbf{A}} Tr(\mathbf{A}) = \mathbf{I}$, $\frac{\partial}{\partial \mathbf{A}} Tr(\mathbf{ABA}^T) = \mathbf{A}(\mathbf{B} + \mathbf{B}^T)$, $\frac{\partial}{\partial \mathbf{A}} \ln|\mathbf{A}| = (\mathbf{A}^{-1})^T$