

0 Basic Math Fundamentals

0.1 Linear Algebra

Transpose:

$$(AB)^T = B^T A^T \quad (A^T)^T = A \quad (A + B)^T = A^T + B^T$$

Trace:

$$A \in \mathbb{R}^{n \times n}: \quad \text{Tr}A = \sum_{i=1}^n A_{ii}, \quad \text{Tr}A = \text{Tr}A^T$$

$$a = \text{Tr}(a) \text{ for } a \in \mathbb{R}, \quad \text{Tr}ABC = \text{Tr}BCA = \text{Tr}CAB$$

$$\frac{\partial}{\partial A} \text{Tr}(AB) = B^T$$

Determinant:

$$\det(AB) = \det(A) \det(B) \quad \frac{\partial}{\partial A} (\log(\det X)) = (X^{-1})^T$$

$$\log(\det A) = -\log(\det A^{-1}) \quad \det(kA) = k^N \det(A)$$

Norms:

$$\|x\|_p = \left(\sum_{i=1}^n |x_i|^p \right)^{\frac{1}{p}} \quad \text{with } p > 1$$

Inverse:

$$(ABC)^{-1} = C^{-1}B^{-1}A^{-1} \quad (A^{-1})^T = (A^T)^{-1}$$

$$A^{-1} \text{ exists} \Leftrightarrow \text{rank}(A) \text{ is full} \Leftrightarrow \det(A) \neq 0 \Leftrightarrow \forall \lambda_i: \lambda_i \neq 0$$

Matrix-Sum Notation:

$$\begin{aligned} x^T A x &= \sum_{i=1}^n x_i (Ax)_i = \sum_{i=1}^n x_i \left(\sum_{j=1}^n A_{ij} x_j \right) \\ &= \sum_{i=1}^n \sum_{j=1}^n A_{ij} x_i x_j \end{aligned}$$

Positive and Negative Definiteness:

$$\text{PD: } x^T A x > 0 \quad \text{PSD: } x^T A x \geq 0$$

$$\text{ND: } x^T A x < 0 \quad \text{NSD: } x^T A x \leq 0$$

Hessian:

$$\nabla_x^2 f(x) \in \mathbb{R}^{n \times n} = \begin{bmatrix} \frac{\partial^2 f(x)}{\partial x_1^2} & \frac{\partial^2 f(x)}{\partial x_1 \partial x_2} & \frac{\partial^2 f(x)}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f(x)}{\partial x_2 \partial x_1} & \frac{\partial^2 f(x)}{\partial x_2^2} & \frac{\partial^2 f(x)}{\partial x_2 \partial x_n} \\ \frac{\partial^2 f(x)}{\partial x_n \partial x_1} & \frac{\partial^2 f(x)}{\partial x_n \partial x_2} & \frac{\partial^2 f(x)}{\partial x_n^2} \end{bmatrix}$$

Hessian of non-symmetric matrix: $f(x) = x^T A x$

$$H_f = \frac{1}{2}(A + A^T)$$

Lagrangian:

$$\mathcal{L}(x, y, \lambda) = f(x, y) - \lambda g(x, y)$$

$$\nabla_{x,y,\lambda} \mathcal{L}(x, y, \lambda) = 0$$

Jacobian and Gradient:

$$J_a = \frac{\partial \mathbf{a}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial a_1}{\partial x_1} & \frac{\partial a_1}{\partial x_2} & \cdots & \frac{\partial a_1}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial a_m}{\partial x_1} & \frac{\partial a_m}{\partial x_2} & \cdots & \frac{\partial a_m}{\partial x_n} \end{bmatrix} \in \mathbb{R}^{m \times n}$$

$$\nabla_{\mathbf{x}} c = \frac{\partial c}{\partial \mathbf{x}} = \frac{\partial c}{\partial \mathbf{a}} \frac{\partial \mathbf{a}}{\partial \mathbf{x}} = \left(\frac{\partial \mathbf{a}}{\partial \mathbf{x}} \right)^T \nabla_{\mathbf{a}} c \quad \text{with} \quad \left[\frac{\partial \mathbf{a}}{\partial \mathbf{x}} \right]_{ij} = \frac{\partial a_i}{\partial x_j}$$

$$\nabla_{\mathbf{a}} c = \left(\frac{\partial c}{\partial \mathbf{a}} \right)^T = \begin{bmatrix} \frac{\partial c}{\partial a_1} & \cdots & \frac{\partial c}{\partial a_m} \end{bmatrix}^T \in \mathbb{R}^m$$

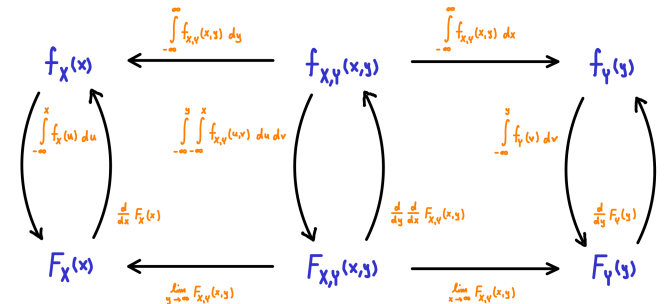
0.2 Probability Theory

Bayes' Theorem: $P(A|B) = \frac{P(A,B)}{P(B)} = \frac{P(B|A) \cdot P(A)}{P(B)}$

Independence: $P(A \cap B) = P(A)P(B)$

Marginalization:

$$p(a) = \int \int p(a, b, c) db dc$$



Mean and Variance:

Mean: $\mathbb{E}[f] = \sum_x p(x) f(x) = \int p(x) f(x) dx$

\mathbb{E} is a lin. operator. For scalar x : $\mathbb{E}[x] = \mathbb{E}[\text{tr}(x)] = \text{tr}(\mathbb{E}[x])$

Law of iterated expectations: $\mathbb{E}_{z \sim p(z)} \left[\mathbb{E}_{x \sim p(x|z)} [x|z] \right]$

Variance: $\begin{aligned} \text{Var}[f] &= \mathbb{E}[(f(x) - \mathbb{E}[f(x)])^2] \\ &= \mathbb{E}[f(x)^2] - \mathbb{E}[f(x)]^2 \end{aligned}$

$$\text{Var}[A] + \text{Var}[B] = \text{Var}[A + B]$$

Covariance: $\text{Cov}[X, Y] = \mathbb{E}[XY] - \mathbb{E}[X] \mathbb{E}[Y]$

Chain Rule:

$$P(A, B|C) = P(A|B, C) \cdot P(B|C) = P(B|A, C) \cdot P(A|C)$$

$$P(A, B|C) = \frac{P(A, B, C)}{P(C)} = P(A|B, C) \cdot P(B|C)$$

1 K-Nearest Neighbor Classification

Indicator Variable:

$$\mathbb{I}(e) = \begin{cases} 1 & \text{if } e \text{ is true} \\ 0 & \text{if } e \text{ is false} \end{cases}$$

Most probable class label \hat{y} :

$$\hat{y} = \arg \max_c p(y = c | x, k)$$

Classification:

$$p(y = c | x, k) = \frac{1}{k} \sum_{i \in \mathcal{N}_k(x)} \mathbb{I}(y_i = c)$$

Weighted Classification:

$$p(y = c | x, k) = \frac{1}{Z} \sum_{i \in \mathcal{N}_k(x)} \frac{1}{d(x, x_i)} \mathbb{I}(y_i = c)$$

With $Z = \sum_{i \in \mathcal{N}_k(x)} \frac{1}{d(x, x_i)}$

Regression:

$$\hat{y} = \frac{1}{Z} \sum_{i \in \mathcal{N}_k(x)} \frac{1}{d(x, x_i)} y_i$$

Classification Performance:

Accuracy: $acc = \frac{TP+TN}{TP+TN+FP+FN}$

Precision: $prec = \frac{TP}{TP+FP}$

Sensitivity/Recall: $rec = \frac{TP}{TP+FN}$

Specificity: $tnr = \frac{TN}{FP+TN}$

False Negative Rate: $fnr = \frac{FN}{TP+FN}$

False Positive Rate: $fpr = \frac{FP}{FP+TN}$

F1 Score: $f1 = \frac{2 \cdot prec \cdot rec}{prec+rec}$

Distance Measures:

L_1 norm: $\diamond \quad \sum_i |u_i - v_i|$

L_2 norm: $\bigcirc \quad \sqrt{\sum_i (u_i - v_i)^2}$

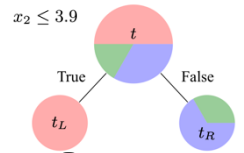
L_∞ norm: $\square \quad \max_i |u_i - v_i|$

Mahalanobis: $\sqrt{(u-v)^T \Sigma^{-1} (u-v)}$

Data Standardization (zero mean & unit variance):

Z-Score: $x_{i,std} = \frac{x_i - \mu_i}{\sigma_i}$

2 Decision Trees



Inference on decision trees:

1. Test attributes of x to find region \mathcal{R}
2. New unseen sample x is given the most common label in corresponding region \mathcal{R}

$$\hat{y} = \arg \max_c p(y = c | \mathcal{R}) = \frac{n_{c,\mathcal{R}}}{\sum_{c_i \in \mathcal{C}} n_{c_i,\mathcal{R}}}$$

Building an optimal decision tree:

Grow the tree top-down and choose the best split node-by-node using a greedy heuristic on the training data. \rightarrow NP-complete

Stopping criteria (pre-pruning):

- Distribution in branch is pure: $i(t) = 0$
- Maximum depth reached
- Number of samples in each branch below t_n
- Benefit of splitting too low: $\Delta i(s, t) < t_\Delta$
- Accuracy on validation set

Improvement heuristic / Information gain:

$$\Delta i(s, t) = i(t) - p_L \cdot i(t_L) - p_R \cdot i(t_R)$$

Where p_L and p_R are the percentages of original data

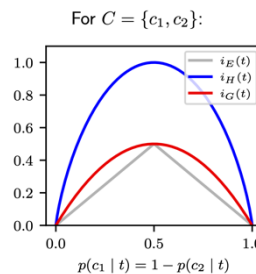
Impurity measures:

Misclassification rate: $i_E(t) = 1 - \max_c \pi_c$

Entropy: $i_H(t) = - \sum_{c_i \in \mathcal{C}} \pi_{c_i} \log_2 \pi_{c_i}$

Gini index: $i_G(t) = 1 - \sum_{c_i \in \mathcal{C}} \pi_{c_i}^2$

with $\pi_c = p(y = c | t)$



Reduced error pruning:

- Delete all descendant nodes of t but not t itself: $T \setminus T_t$
- Use val. set to get an error estimate: $err_{D_V}(T)$
 - For each node t calculate $err_{D_V}(T \setminus T_t)$
 - Prune tree at node with highest err-reduction
 - Repeat until $\forall t: err_{D_V}(T) < err_{D_V}(T \setminus T_t)$

Ensembles (aggregate predictions from multiple models):

Bagging: Combine predictions of many classifiers trained on sampled sub-data-set \rightarrow Random forests

Boosting: Incrementally train weak classifiers that correct previous mistakes.

Stacking: Train a meta-classifier with the base classifier's predictions as features.

3 Probabilistic Inference

General:

Independent and identically distributed (i.i.d.):

$$p(\mathcal{D} | \theta) = \prod_{i=1}^N p(x_i | \theta)$$

Logarithm preserves maxima:

$$\arg \max_{\theta} p(\theta | \mathcal{D}) = \arg \max_{\theta} \log p(\theta | \mathcal{D})$$

Hoeffding's Inequality: $N \geq \frac{\ln(\frac{2}{\delta})}{2e^2}$

Maximum Likelihood Estimation (MLE):

$$\theta_{MLE} = \arg \max_{\theta} p(\mathcal{D} | \theta)$$

1. Find matching distribution $p(\mathcal{D} | \theta)$
2. Apply i.i.d. formula
3. Take logarithm of distribution ($\prod \rightarrow \sum$)
4. Derivate w.r.t. θ
5. Solve for θ

Coin Flip Example: $\theta_{MLE} = \frac{|T|}{|T| + |H|}$

Maximum A Posteriori Estimation (MAP):

Performs better than MLE if less data is available.

$$\theta_{MAP} = \arg \max_{\theta} p(\theta | \mathcal{D})$$

Bayes Formula:

$$p(\theta | \mathcal{D}) = \frac{\underset{\text{Posterior Distribution}}{p(\mathcal{D} | \theta)} \cdot \underset{\text{Likelihood}}{p(\theta)}}{\underset{\text{Prior}}{p(\mathcal{D})}} \propto p(\mathcal{D} | \theta) \cdot p(\theta)$$

1. Find matching distribution
2. Find matching prior
3. Apply i.i.d. formula
4. Take logarithm of distribution
5. Derivate w.r.t. θ
6. Solve for θ

Coin Flip Example: $\theta_{MAP} = \frac{|T| + a - 1}{|T| + |H| + a + b - 2}$

MLE corresponds to having a uniform prior (which holds no information on θ ; "improper").

Posterior Distribution Estimate:

Find full distribution $p(\theta | \mathcal{D})$:

1. Calculate $p(\theta | \mathcal{D}) \propto p(\mathcal{D} | \theta) \cdot p(\theta)$
 - a. Apply i.i.d. formula (no need for log)
 - b. Drop all factors independent of θ
2. Find normalization constant
 - a. Compute integral $\int p(\theta | \mathcal{D}) d\theta$
 - b. Match pattern of unnormalized posterior to known PDF.

Coin Example: $p(\theta | \mathcal{D}) = \text{Beta}(\theta | a + |T|, b + |H|)$

The mean of a Gaussian posterior coincides with its mode/MAP.

Posterior Predictive Distribution (Fully Bayesian):

$$p(x_{new} | \mathcal{D}, \dots) = \int_{-\infty}^{\infty} p(x_{new}, \theta | \mathcal{D}, \dots) d\theta = \int_{-\infty}^{\infty} \underset{\text{Likelihood}}{p(x_{new} | \theta)} \underset{\text{Posterior Distribution}}{p(\theta | \mathcal{D}, \dots)} d\theta$$

1. Calculate Posterior Distribution Estimate
2. Find matching distribution for likelihood
3. Substitute these expressions

Coin Flip Example:

$$p(x_{new} | \mathcal{D}, a, b) = \text{Ber}\left(x_{new} \mid \frac{|T| + a}{|T| + |H| + a + b}\right)$$

Convolution formula:

$$p(x) = \int_{-\infty}^{\infty} p(x - z) p(z) dz$$

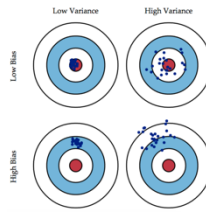
For two Gaussians:

$$p(x_{new} | \mathcal{D}, \dots) = \int_{-\infty}^{\infty} \mathcal{N}(x - z | 0, b_1^{-1}) \mathcal{N}(z | m, b_2^{-1}) dz = \mathcal{N}(x | m, b_1^{-1} + b_2^{-1})$$

with $\mathcal{N}(x - z | 0, b_1^{-1}) = \mathcal{N}(x | z, b_1^{-1})$

4 Linear Regression

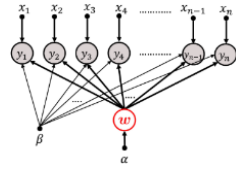
$$\hat{y} = f_w(\mathbf{x}) = w_0 + \sum_{j=1}^M w_j \phi_j(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x})$$



Probabilistic Interpretation:

Ridge regression is equivalent to Maximum a posteriori estimation with gaussian prior $p(\mathbf{w})$.

If $p(\mathbf{w}|\alpha) = \mathcal{N}(\mathbf{w}|0, \alpha^{-1}I)$ with $\lambda = \frac{\alpha}{\beta}$:



$$\mathbf{w}_{MAP} = \arg \max_{\mathbf{w}} p(\mathbf{w}|\mathbf{X}, \mathbf{y}, \alpha, \beta)$$

$$= \arg \min_{\mathbf{w}} (-\ln p(\mathbf{y}|\mathbf{X}, \mathbf{w}, \beta) - \ln p(\mathbf{w}|\alpha))$$

$$E_{MAP}(\mathbf{w}) \propto E_{Ridge}(\mathbf{w}) + const.$$

4.3 Weighted Least Squares Regression

Loss Function:

$$E(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^N t_i (\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_i) - y_i)^2$$

Optimal Weight Vector / Normal Equation:

$$\begin{aligned} \mathbf{w}^* &= \arg \min_{\mathbf{w}} \frac{1}{2} (\boldsymbol{\Phi} \mathbf{w} - \mathbf{y})^T \mathbf{T} (\boldsymbol{\Phi} \mathbf{w} - \mathbf{y}) \\ &= (\boldsymbol{\Phi}^T \mathbf{T} \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^T \mathbf{T} \mathbf{y} \end{aligned}$$

Probabilistic Interpretation:

Weighted least squares is equivalent to probabilistic least squares where we choose $\beta = t_i$, therefore making the regression targets no longer identically distributed but still independent.

$$y_i \sim \mathcal{N}(\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_i), t_i^{-1})$$

4.4 Fully Bayesian Posterior Distribution

$$p(\mathbf{w}|\mathcal{D}) = p(\mathbf{w}|\mathbf{X}, \mathbf{y}, \beta, \cdot) = \frac{p(\mathbf{y}|\mathbf{X}, \mathbf{w}, \beta) \cdot p(\mathbf{w}|\cdot)}{p(\mathbf{y}|\mathbf{X}, \beta, \cdot)}$$

If $p(\mathbf{w}|\alpha) = \mathcal{N}(\mathbf{w}|0, \alpha^{-1}I)$ with $\lambda = \frac{\alpha}{\beta}$:

$$p(\mathbf{w}|\mathcal{D}) = \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

with $\boldsymbol{\mu} = \mathbf{w}_{MAP} = \beta \boldsymbol{\Sigma} \boldsymbol{\Phi}^T \mathbf{y}$ and $\boldsymbol{\Sigma}^{-1} = \alpha I + \beta \boldsymbol{\Phi}^T \boldsymbol{\Phi}$

4.5 Predicting New Data

Maximum likelihood: \mathbf{w}_{ML} and β_{ML}

$$p(\hat{y}_{new} | \mathbf{x}_{new}, \mathbf{w}_{ML}, \beta_{ML}) = \mathcal{N}(\hat{y}_{new} | \mathbf{w}_{ML}^T \boldsymbol{\phi}(\mathbf{x}_{new}), \beta_{ML}^{-1})$$

Maximum a posteriori: \mathbf{w}_{MAP}

$$p(\hat{y}_{new} | \mathbf{x}_{new}, \mathbf{w}_{MAP}, \beta) = \mathcal{N}(\hat{y}_{new} | \mathbf{w}_{MAP}^T \boldsymbol{\phi}(\mathbf{x}_{new}), \beta^{-1})$$

Posterior Predictive Distribution:

$$p(\hat{y}_{new} | \mathbf{x}_{new}, \mathcal{D})$$

$$= \mathcal{N}(\hat{y}_{new} | \boldsymbol{\mu}^T \boldsymbol{\phi}(\mathbf{x}_{new}), \beta^{-1} + \boldsymbol{\phi}(\mathbf{x}_{new})^T \boldsymbol{\Sigma} \boldsymbol{\phi}(\mathbf{x}_{new}))$$

$$\mathbf{X} \in \mathbb{R}^{N \times (D+1)} = \begin{bmatrix} -(\mathbf{X}^{(1)})^T & - \\ -(\mathbf{X}^{(2)})^T & - \\ \vdots & \vdots \\ -(\mathbf{X}^{(n)})^T & - \end{bmatrix} \quad \mathbf{w} \in \mathbb{R}^{D+1} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_{D+1} \end{bmatrix} \quad \mathbf{y} \in \mathbb{R}^N = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

4.1 Ordinary Least Squares Regression

Loss Function:

$$E_{LS}(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^N (f_w(\mathbf{x}_i) - y_i)^2 = \frac{1}{2} \sum_{i=1}^N (\mathbf{w}^T \mathbf{x}_i - y_i)^2$$

Optimal Weight Vector/ Normal Equation:

$$\mathbf{w}^* = \arg \min_{\mathbf{w}} E_{LS}(\mathbf{w}) = \arg \min_{\mathbf{w}} \frac{1}{2} \sum_{i=1}^N (\mathbf{w}^T \mathbf{x}_i - y_i)^2$$

$$= \arg \min_{\mathbf{w}} \frac{1}{2} (\mathbf{X} \mathbf{w} - \mathbf{y})^T (\mathbf{X} \mathbf{w} - \mathbf{y})$$

$$= (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y} = \mathbf{X}^+ \mathbf{y}$$

$$\begin{aligned} \nabla_{\mathbf{w}} E_{LS}(\mathbf{w}) &= \nabla_{\mathbf{w}} \frac{1}{2} (\mathbf{X} \mathbf{w} - \mathbf{y})^T (\mathbf{X} \mathbf{w} - \mathbf{y}) \\ &= \nabla_{\mathbf{w}} \frac{1}{2} (\mathbf{w}^T \mathbf{X}^T \mathbf{X} \mathbf{w} - 2 \mathbf{w}^T \mathbf{X}^T \mathbf{y} + \mathbf{y}^T \mathbf{y}) \\ &= \mathbf{X}^T \mathbf{X} \mathbf{w} - \mathbf{X}^T \mathbf{y} \end{aligned}$$

With non-lin. basis functions

$$\begin{aligned} \mathbf{w}^* &= \arg \min_{\mathbf{w}} \frac{1}{2} \sum_{i=1}^N (\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_i) - y_i)^2 \\ &= (\boldsymbol{\Phi}^T \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^T \mathbf{y} = \boldsymbol{\Phi}^+ \mathbf{y} \quad \text{with } \phi_0 = 1 \text{ (absorb bias)} \end{aligned}$$

$$\text{Design M.: } \boldsymbol{\Phi} = \begin{bmatrix} \phi_0(\mathbf{x}_1) & \cdots & \phi_M(\mathbf{x}_1) \\ \vdots & \ddots & \vdots \\ \phi_0(\mathbf{x}_N) & \cdots & \phi_M(\mathbf{x}_N) \end{bmatrix} \in \mathbb{R}^{N \times (M+1)}$$

Probabilistic Interpretation:

Least squares regression is equivalent to maximum likelihood estimation of \mathbf{w}, β . Also equivalent to sampling from i.i.d. samples with gaussian noise β :

$$\text{Data likelihood: } p(\mathbf{y}|\boldsymbol{\Phi}, \mathbf{w}, \beta) = \prod_{i=1}^N p(y_i | \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_i), \beta)$$

Gaussian mean Precision

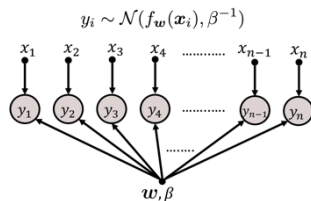
$$\mathbf{w}_{ML} = \arg \max_{\mathbf{w}} p(\mathbf{y}|\boldsymbol{\Phi}, \mathbf{w}, \beta)$$

$$= \arg \min_{\mathbf{w}} -\ln p(\mathbf{y}|\boldsymbol{\Phi}, \mathbf{w}, \beta) = \arg \min_{\mathbf{w}} E_{LS}(\mathbf{w})$$

Find precision of Gaussian:

$$\beta_{ML} = \arg \min_{\beta} E_{LS}(\mathbf{w}_{ML}, \beta)$$

$$\frac{1}{\beta_{ML}} = \frac{1}{N} \sum_{i=1}^N (\mathbf{w}_{ML}^T \boldsymbol{\phi}(\mathbf{x}_i) - y_i)^2$$



4.2 Ridge Regression

Loss Function:

$$E_{Ridge}(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^N (\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_i) - y_i)^2 + \frac{\lambda}{2} \|\mathbf{w}\|_2^2$$

Optimal Weight Vector / Normal Equation:

$$\begin{aligned} \mathbf{w}^* &= \arg \min_{\mathbf{w}} \frac{1}{2} (\boldsymbol{\Phi} \mathbf{w} - \mathbf{y})^T (\boldsymbol{\Phi} \mathbf{w} - \mathbf{y}) + \frac{\lambda}{2} \mathbf{w}^T \mathbf{w} \\ &= (\boldsymbol{\Phi}^T \boldsymbol{\Phi} + \lambda I)^{-1} \boldsymbol{\Phi}^T \mathbf{y} \end{aligned}$$

If $N < M$, $\boldsymbol{\Phi}^T \boldsymbol{\Phi}$ is not invertible $\rightarrow +\lambda I$ fixes this.

5 Linear Classification

5.1 Perceptron

General Idea:

SGD with minibatch size 1: $\min_{w,b} \sum_i L(y_i, \mathbf{w}^T \mathbf{x}_i + b)$

Hinge Loss:

$$L(u, v) = \max(0, \varepsilon - uv) = \begin{cases} \varepsilon - uv, & \text{if } uv < \varepsilon \\ 0 & \text{else} \end{cases}$$

Decision Rule:

$$\hat{y} = f(\mathbf{w}^T \mathbf{x} + w_0) \quad \text{where} \quad f(t) = \begin{cases} 1 & \text{if } t > 0 \\ 0 & \text{otherwise} \end{cases}$$

Learning Rule (Init. $\mathbf{w}, w_0 \leftarrow \mathbf{0}$):

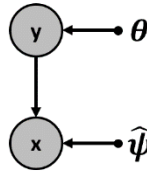
$$\mathbf{w} \leftarrow \begin{cases} \mathbf{w} + \mathbf{x}_i & \text{if } y_i = 1 \\ \mathbf{w} - \mathbf{x}_i & \text{if } y_i = 0 \end{cases} \quad \& \quad w_0 \leftarrow \begin{cases} w_0 + 1 & \text{if } y_i = 1 \\ w_0 - 1 & \text{if } y_i = 0 \end{cases}$$

Multiple Classes:

- One-Versus-Rest: Compare to all
- One-Versus-One: Compare pairwise
- Multiclass Discriminant: $\hat{y} = \arg \max_{c \in \mathcal{C}} \mathbf{w}_c^T \mathbf{x} + x_{0c}$

Limitations of hard-decision based classifiers:

- No measure of uncertainty
- Cannot handle noisy data
- Poor generalization
- Difficult to optimize



5.2 Probabilistic Generative Model

Models the joint probability

$$p(y = c | \mathbf{x}, \psi, \theta) \propto \overbrace{p(\mathbf{x} | y = c, \psi) \cdot p(y = c | \theta)}$$

Class Conditional/ Likelihood: (Distr. of points in class c)

Multiv. normal (*shared Σ*): $p(\mathbf{x} | y = c) = \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_c, \Sigma)$

Class Prior: (Prior probability of a point belonging to a class c)

Categorical Distribution: $p(y) = \prod_{c=1}^C \theta_c^{\mathbb{I}(y=c)}$

Data Likelihood: (joint distribution)

$$p(D | \{\pi_c, \theta_c\}_{c=1}^C) = \prod_{n=1}^N \prod_{c=1}^C \left(\pi_c p(\mathbf{x}^{(n)} | \boldsymbol{\mu}_c, \Sigma) \right)^{y_c^{(n)}}$$

(Apply log and take derivative for specific c to compute the MLE)

$$\text{MLE of } \theta: \quad \theta_c^{MLE} = \frac{1}{N} \sum_{i=1}^N \mathbb{I}(y_i = c) = \frac{N_c}{N} = \pi_c$$

$$\text{MLE of } \boldsymbol{\mu}_c: \quad \boldsymbol{\mu}_c = \frac{1}{N_c} \sum_{\substack{n=1 \\ y^{(n)}=c}}^N \mathbf{x}^{(n)}$$

MLE of the shared Covariance Matrix Σ :

$$\Sigma = \sum_{c=1}^C \frac{N_c}{N} \mathbf{S}_c \quad \text{with} \quad \mathbf{S}_c = \frac{1}{N_c} \sum_{\substack{n=1 \\ y^{(n)}=c}}^N (\mathbf{x}^{(n)} - \boldsymbol{\mu}_c)(\mathbf{x}^{(n)} - \boldsymbol{\mu}_c)^T$$

New Data: $y_{new} = \arg \max_c (p = c | n_{new}, \psi, \theta)$

5.3 Linear Discriminant Analysis (LDA)

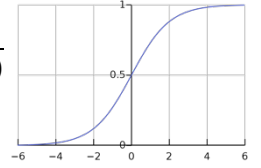
For Gaussian class conditionals with shared covariance matrices.

LDA for two classes: Linear Decision Boundary

Rewrite the posterior as:

$$p(y = 1 | \mathbf{x}) = \sigma(a) = \frac{1}{1 + \exp(-a)}$$

$$\text{where } a = \log \frac{p(\mathbf{x} | y = 1)p(y = 1)}{p(\mathbf{x} | y = 0)p(y = 0)}$$



$$\text{Solution:} \quad p(y = 1 | \mathbf{x}) = \sigma(\mathbf{w}^T \mathbf{x} + w_0)$$

is a Sigmoid ($y = 1$ if $a > 0$) with these parameters:

$$\mathbf{w} = \Sigma^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)$$

$$w_0 = -\frac{1}{2} \boldsymbol{\mu}_1^T \Sigma^{-1} \boldsymbol{\mu}_1 + \frac{1}{2} \boldsymbol{\mu}_0^T \Sigma^{-1} \boldsymbol{\mu}_0 + \log \frac{p(y = 1)}{p(y = 0)}$$

LDA for more classes: Linear Decision Boundaries

The posterior over the class label is:

$$p(y = c | \mathbf{x}) = \frac{p(\mathbf{x} | y = c)p(y = c)}{\sum_{c'=1}^C p(\mathbf{x} | y = c')p(y = c')}$$

The solution is a SoftMax:

$$p(y = c | \mathbf{x}) = \frac{\exp(\mathbf{w}_c^T \mathbf{x} + w_{c0})}{\sum_{c'=1}^C \exp(\mathbf{w}_{c'}^T \mathbf{x} + w_{c'0})}$$

$$\mathbf{w}_c = \Sigma^{-1} \boldsymbol{\mu}_c \quad w_{c0} = -\frac{1}{2} \boldsymbol{\mu}_c^T \Sigma^{-1} \boldsymbol{\mu}_c + \log p(y = c)$$

Decision Boundary: $\mathbf{w}_1^T \mathbf{x} + w_{10} = \mathbf{w}_2^T \mathbf{x} + w_{20}$

5.4 Naive Bayes Classification

Class prior: $p(y = c) = \pi_c$

Class conditionals: $p(\mathbf{x} | y = c) = \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_c, \Sigma_c)$

We use *per-class means* and *diagonal per-class covariance matrices*. Naive Bayes can handle mixed data types (*features are cond. independent*).

Assume: $p(x_1, x_2, \dots, x_d | y = c) = \prod_{i=1}^d p(x_i | y = c)$

NBC for two classes: Quadratic Decision Boundary

Start at: $p(y = 1 | \mathbf{x}) = \sigma(a)$

Solution: $p(y = 1 | \mathbf{x}) = \sigma(\mathbf{x}^T \mathbf{W}_2 \mathbf{x} + \mathbf{w}_1^T \mathbf{x} + w_0)$

$$\mathbf{W}_2 = \frac{1}{2} [\Sigma_0^{-1} - \Sigma_1^{-1}] \quad \mathbf{w}_1 = \Sigma_1^{-1} \boldsymbol{\mu}_1 - \Sigma_0^{-1} \boldsymbol{\mu}_0$$

$$w_0 = -\frac{1}{2} \boldsymbol{\mu}_1^T \Sigma_1^{-1} \boldsymbol{\mu}_1 + \frac{1}{2} \boldsymbol{\mu}_0^T \Sigma_0^{-1} \boldsymbol{\mu}_0 + \log \frac{\pi_1}{\pi_0} + \frac{1}{2} \log \frac{|\Sigma_0|}{|\Sigma_1|}$$

At the decision bound.: $p(y = 1 | \mathbf{x}) = p(y = 0 | \mathbf{x})$

5.5 Ordinary 2-class Logistic Regression (Probabilistic Generative Model)

Directly model the posterior distribution $p(y|x)$ by treating w and w_0 as free parameters. Best for pure classification task because no assumptions are needed.

Posterior:

$$p(y|w, X) = \prod_{i=1}^N \sigma(w^T x_i)^{y_i} (1 - \sigma(w^T x_i))^{1-y_i}$$

Loss Function: Binary Cross Entropy

Negative log-likelihood of Binary Logistic Regression

$$\begin{aligned} E(w) &= -\log p(y|w, X) \\ &= -\sum_{i=1}^N (y_i \log \sigma(w^T x_i) \\ &\quad + (1 - y_i) \log (1 - \sigma(w^T x_i))) \end{aligned}$$

Gradient of Loss Function:

$$\nabla E(w) = \nabla(-\ln p(y|w, X)) = \sum_{i=1}^N x_i (\sigma(w^T x_i) - y_i)$$

5.6 2-class Logistic Ridge Regression

Posterior:

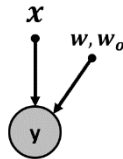
$$p(y|w, X) = \prod_{i=1}^N \sigma(w^T x_i)^{y_i} (1 - \sigma(w^T x_i))^{1-y_i}$$

Loss Function: Binary Cross Entropy

$$E(w) = -\log p(y|w, X) + \lambda \|w\|_2^2$$

Gradient of Loss Function:

$$\nabla E_{Ridge}(w) = \sum_{i=1}^N x_i (\sigma(w^T x_i) - y_i) + \lambda w$$



5.7 Ordinary Multi-Class Logistic Regression

Likelihood:

$$\begin{aligned} p(\mathcal{D}|\{\pi_c, \theta_c\}_{c=1}^C) &= \prod_{n=1}^N p(x^{(n)}|y_n, \theta) p(y_n|\pi) \\ &= \prod_{n=1}^N \prod_{c=1}^C (p(x^{(n)}|\theta_c) \pi_c)^{y_c^{(n)}} \end{aligned}$$

Loss Function: Cross Entropy

Negative log-likelihood of Multiclass Logistic Regression

$$E(w) = -\sum_{i=1}^N \sum_{c=1}^C y_{ic} \log \frac{\exp(w_c^T x)}{\sum_{c'} \exp(w_{c'}^T x)}$$

One-Hot Encoding: $y_{ic} = f(x) = \begin{cases} 1, & \text{if } x_i \in c \\ 0, & \text{else} \end{cases}$

5.8 Optimizing Logistic Regression

Gradient Descent (no closed-form solution):

$$w_{t+1} \leftarrow w_t - \tau \nabla E(w_t)$$

5.9 Sigmoid and SoftMax

SoftMax:

$$\sigma(x)_i = \frac{\exp x_i}{\sum_{c=1}^C \exp x_c} = \sigma(Wx + w_0)$$

where $W \in \mathbb{R}^{C \times D}$, $x \in \mathbb{R}^D$, $w_0 \in \mathbb{R}^C$

$$\frac{\partial}{\partial w_i} \sigma(j) = \sigma(j) (\delta_{ij} - \sigma(i)) x$$

Sigmoid:

$$\sigma(a) = \frac{1}{1 + \exp(-a)}$$

$$\frac{\partial \sigma(a)}{\partial a} = \sigma(a)(1 - \sigma(a))$$

$$\sigma(-a) = 1 - \sigma(a)$$

$$\sigma^{-1} = \ln \frac{a}{1-a}$$

$$\tanh a = 2\sigma(2a) - 1$$

$$a = \ln \frac{\sigma}{1-\sigma}$$

5.10 Generative vs. Discriminative Models

- Discriminative models achieve better performance in pure classification tasks.
- Generative models are fragile when assumptions are violated (e.g. non-Gaussian class likelihoods).
- Generative models handle missing data and outliers better and can generate new data.

6 Optimization

6.1 Convexity

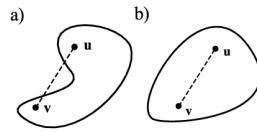
First Order Convexity Conditions:

Function f is convex for $x_1, x_2 \in X$ if for $\lambda \in [0, 1]$:

- $f(\lambda x_1 + (1 - \lambda)x_2) \leq \lambda f(x_1) + (1 - \lambda)f(x_2)$
- $f(x_2) - f(x_1) \geq \frac{f((1-\lambda)x_1 + \lambda x_2) - f(x_1)}{\lambda}$
- $f(x_2) - f(x_1) \geq (x_2 - x_1)^T \nabla f(x_1)$
- $H_f = \nabla_x^2 f(x)$ is PSD
- $\frac{\partial^2 f(x)}{\partial x^2} > 0$

Set X is convex if for $\lambda \in [0, 1]$:

$$\forall x, y \in X: \quad \lambda x + (1 - \lambda)y \in X$$



- If A and B are convex sets, then $A \cap B$ is convex.

Convexity Preserving Operations:

Let $f_1, f_2: \mathbb{R}^d \rightarrow \mathbb{R}$ be convex and $g: \mathbb{R}^d \rightarrow \mathbb{R}$ be concave, then:

- $h(x) = f_1(x) + f_2(x)$ is convex
- $h(x) = \max\{f_1(x), f_2(x)\}$ is convex
- $h(x) = c \cdot f_1(x)$ is convex if $c \geq 0$
- $h(x) = c \cdot g(x)$ is convex if $c \leq 0$
- $h(x) = f_1(Ax + b)$ is convex (A matrix, b vector)
- $h(x) = m(f_1(x))$ is convex if $m: \mathbb{R} \rightarrow \mathbb{R}$ is convex and nondecreasing
- $h(x) = \text{const}$, $h(x) = w^T x$ and $h(x) = e^x$ are convex
- $\pm w^T x$ is convex and concave - $\log(x)$ is concave
- $\min(\cdot) = \max(-\cdot)$

6.2 Gradient Descent

General Idea:

1. Given a starting point $\theta \in \text{Dom}(f)$
2. $\Delta\theta := -\nabla f(\theta)$
3. Do line search to find t^*
4. Update $\theta := \theta + t^* \Delta\theta$
5. Repeat 2-4 until stopping criterion

Line Search:

$$t^* = \arg \min_{t \geq 0} f(\theta + t \cdot \Delta\theta)$$

Backtracking Line Search:

With $\alpha \in (0, 0.5)$ and $\beta \in (0, 1)$, start at $t = 1$ and repeat $t = \beta \cdot t$ until

$$f(\theta + t\Delta\theta) < f(\theta) + t\alpha \nabla f(\theta)^T \Delta\theta$$

Learning Rate to Avoid Line Search:

$$\theta_{t+1} \leftarrow \theta_t - \tau \cdot \nabla f(\theta_t)$$

- τ too small: Slow convergence or ending up in a saddle point or local minima
- τ too big: Oscillations without convergence

Learning Rate Schedule:

$$\tau_{t+1} \leftarrow \alpha \cdot \tau_t \text{ for } 0 < \alpha < 1$$

Momentum:

Incorporates history of gradients. Accelerates search in the direction where many previous gradients point to.

$$m_t \leftarrow \tau \cdot \nabla f(\theta_t) + \gamma \cdot m_{t-1}$$

$$\theta_{t+1} \leftarrow \theta_t - m_t$$

Stochastic Gradient Descent:

Use a minibatch of entire data to compute a noisy gradient estimate and use it to update the parameters.

- Randomly pick a small subset \mathcal{S} from the entire data \mathcal{D} , i.e., the so-called mini batch
- Compute the gradient based on the mini batch
- Update $\theta_{t+1} \leftarrow \theta_t - \tau \cdot \frac{n}{|\mathcal{S}|} \cdot \nabla f(\theta_t)$
- Pick a new mini-batch and repeat

6.3 Newton Method

Uses first-order and second-order derivative. f must be convex and Hessian $\nabla^2 f(\theta) \succcurlyeq 0$ (PSD). Used only for low dimensional problems.

Taylor-Expansion of f at point θ_t :

For small δ

$$f(\theta_t + \delta) = f(\theta_t) + \delta^T \nabla f(\theta_t) + \frac{1}{2} \delta^T \nabla^2 f(\theta_t) \delta + O(\delta^3)$$

Minimize Approximation:

$$\theta_{t+1} \leftarrow \theta_t - [\nabla^2 f(\theta_t)]^{-1} \nabla f(\theta_t) = \theta_t - \frac{f(\theta_t)}{f'(\theta_t)}$$

$$w_{t+1} \leftarrow w_t - H_E^{-1} \nabla E(w_t)$$

$$H = \nabla^2 E(w_t) = \Phi^T R \Phi \text{ where } R \in \mathbb{R}^{n \times n} = R_{nn} \text{ with } R_{nn} = \sigma(w^T \phi(x_n))(1 - \sigma(w^T \phi(x_n)))$$

7 Deep Learning 1

7.1 Loss Functions

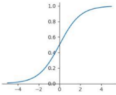
Output type	Output distribution	Output layer	Cost function
Binary	Bernoulli	Sigmoid	Binary cross-entropy
Discrete	Multinomial	Softmax	Cross-entropy
Continuous	Gaussian	Linear	Gaussian cross-entropy (Mean squared error)
Continuous	Arbitrary	GAN, VAE, ...	Various

The loss function must allow gradient-based training.

7.2 Activation Functions

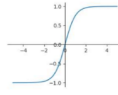
Sigmoid

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$



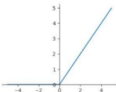
tanh

$$\tanh(x)$$



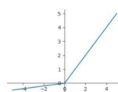
ReLU

$$\max(0, x)$$



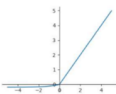
Leaky ReLU

$$\max(0.1x, x)$$



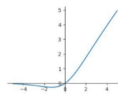
ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



Swish

$$x \cdot \sigma(x)$$



7.3 Parameter Learning with Backprop

Advantages:

- Reuse computations of common ancestors
- Only pass through the computation graph twice
- Modular structure

Backpropagation of Affine Layer ($a = Wx + b$):

$$\frac{\partial E}{\partial W} = \begin{bmatrix} \frac{\partial E}{\partial W_{11}} & \dots & \frac{\partial E}{\partial W_{1H}} \\ \vdots & \ddots & \vdots \\ \frac{\partial E}{\partial W_{D1}} & \dots & \frac{\partial E}{\partial W_{DH}} \end{bmatrix} = \dots = x \frac{\partial E}{\partial a}$$

$$\frac{\partial E}{\partial x} = \frac{\partial E}{\partial a} W$$

$$\frac{\partial E}{\partial b} = \frac{\partial E}{\partial a}$$

Derivative of function f : input \rightarrow output w.r.t. input:

		Input is a ...		
		scalar	vector	matrix
Output is a...	scalar	scalar	vector	matrix
	vector	vector	matrix	3-way tensor
	matrix	matrix	3-way tensor	4-way tensor

8 Deep Learning 2

8.1 Convolution Neural Network (CNN)

$$(x * k)(t) = \int_{-\infty}^{\infty} x(\tau)k(t - \tau)d\tau \equiv \sum_{\tau=-\infty}^{\infty} x(\tau)k(t - \tau)$$

CNNs for images are based on discrete 2D convolution:

$$(x * k)(i, j) = \sum_l \sum_m x(l, m)k(i - l, j - m)$$

Cross Correlation:

$$\hat{x}(i, j) = \sum_{l=1}^L \sum_{m=1}^M x(i + l, j + m)k(l, m)$$

Padding: Either reduce the output or pad the input (e.g., with zeros or constants). (VALID, SAME, FULL)

Stride: Distance between positions the kernel applied.

$$S > 1 \text{ is downsampling } D_{L+1} = \left\lfloor \frac{D_L + 2P - K}{S} \right\rfloor + 1$$

Pooling: Calculate summary statistics in sliding window (e.g., max-, mean-, L_p -norm-pooling)

8.2 Training Deep Neural Networks

Weight Symmetry: Two hidden units have the same bias and weights \rightarrow Same gradients \rightarrow Do not learn different features \rightarrow Break symmetry by initializing with small random values

Weight Scale: Activation function may saturate for hidden units with large fan-in \rightarrow Vanishing or exploding gradients \rightarrow Init. weights with good mean & variance.

Xavier Glorot Initialization: Preserve mean & variance of incoming i.i.d. signal \rightarrow Init. weight matrices with zero mean and variance $Var(W) = \frac{2}{fan-in + fan-out}$

$$W \sim Uniform\left(-\sqrt{\frac{6}{fan-in + fan-out}}, \sqrt{\frac{6}{fan-in + fan-out}}\right)$$

Regularization: Use L_2 norm penalty. Can be combined with dataset augmentation, injecting noise, dropout

BatchNorm: Stabilize the distribution of each layer's activations with minibatch statistics:

$$\hat{x} = \frac{x - \mathbb{E}_B[x]}{\sqrt{Var_B[x] + \epsilon}} \text{ and } y = \gamma \hat{x} + \beta, \quad \gamma, \beta \text{ learned w/ BP.}$$

Skip Connections: Improve information flow in NN

$$y = f(x, W) T(x, W_T) + x(1 - T(x, W_T))$$

Tips:

- Use only differentiable operations
- Always try to overfit the model on a small batch of the training set to make sure that the model is right
- Start with small models and gradually add complexity while monitoring how the performance improves
- Be aware of the properties of activation functions, e.g., no sigmoid output when doing regression
- Monitor the training procedure and use early stopping

9 SVM and Kernels

9.1 Constraint Optimization Problem

Feasibility:

A point $\theta \in \mathbb{R}^d$ is feasible iff it satisfies the constraints of the optimization problem, $f_i(\theta) \leq 0 \forall i$.

Lagrangian:

Minimize: $f_0(\theta)$ (Min. is $p^* = f_0(\theta^*)$)

Subject to: $f_i(\theta) \leq 0, i = 1, \dots, M$

$$L(\theta, \alpha) = f_0(\theta) + \sum_{i=1}^M \alpha_i f_i(\theta) \text{ with } \alpha_i \geq 0$$

Lagrange Dual Function:

For every α , the corresponding unconstrained and concave $g(\alpha)$ is a lower bound on the optimal value of the constrained problem: $\forall \alpha \quad f_0(\theta^*) \geq g(\alpha)$.

Maximize: $g(\alpha)$

Subject to: $\alpha_i \geq 0, i = 1, \dots, m$

$$g(\alpha) = \min_{\theta \in \mathbb{R}^d} L(\theta, \alpha) = \min_{\theta \in \mathbb{R}^d} \left(f_0(\theta) + \sum_{i=1}^M \alpha_i f_i(\theta) \right)$$

The maximum d^* of the Lagrange dual problem is the best possible lower bound on p^* .

Weak Duality: $d^* \leq p^*$ Duality Gap: $p^* - d^* \geq 0$

Strong Duality: $d^* = p^*$, holds for SVM due to the complementary slackness condition $\alpha_i^* f_i(\theta^*) = 0$.

9.2 Hard Margin SVM

Objective: (Max. the margin)

Minimize: $f_0(w, b) = \frac{1}{2} w^T w = \frac{1}{2} \|w\|^2$

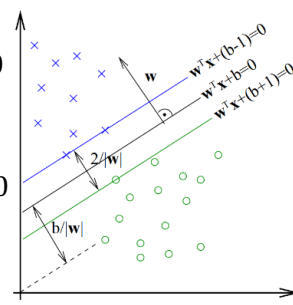
Subject to: $f_i(w, b) = y_i(w^T x_i + b) - 1 \geq 0$
(w points in dir. of +1 class)

1. Lagrangian:

$$L(w, b, \alpha) = \frac{1}{2} w^T w - \sum_{i=1}^N \alpha_i [y_i(w^T x_i + b) - 1]$$

2. Minimize w.r.t. w and b to construct the Dual:

$$\nabla_w L(w, b, \alpha) = w - \sum_{i=1}^N \alpha_i y_i x_i = 0$$

$$\frac{\partial L}{\partial b} = \sum_{i=1}^N \alpha_i y_i = 0$$


3. Dual Problem:

$$\max_{\alpha} g(\alpha) = \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N y_i y_j \alpha_i \alpha_j x_i^T x_j$$

subject to:

$$\sum_{i=1}^N \alpha_i y_i = 0$$

$$\alpha_i \geq 0, \text{ for } i = 1, \dots, N$$

Solving dual problems of SVM is a quadratic problem:

$$g(\alpha) = \frac{1}{2} \alpha^T Q \alpha + \alpha^T \mathbf{1}_N \text{ with } Q = -yy^T \odot XX^T \preceq 0$$

4. Substitute solution for α^* in w and find w^*, b^* :

$$w^* = \sum_{i=1}^N \alpha_i^* y_i x_i \quad b^* = y_i - w^{*T} x_i$$

Complementary slackness condition:

$$\alpha_i^* f_i(\theta^*) = 0 \Rightarrow \alpha_i^* [y_i(w^{*T} x_i + b^*) - 1] = 0$$

(A training sample only contributes to w if $\alpha_i \neq 0$, such that $y_i(w^T x_i + b) = 1$. Then x_i lies on the margin and is called a *support vector*.)

5. Classification:

$$h(x) = \text{sign} \left(\left(\sum_{i=1}^N \alpha_i^* y_i x_i^T \right) x + b^* \right)$$

The solution is sparse, since most α_i are 0.

9.1 Soft Margin SVM

Can deal with noisy data through the slack variable ξ_i (Dist. how far the margin is violated in units of $\|w\|$).

Objective: (Max. the margin & min. violations)

Minimize: $f_0(w, b) = \frac{1}{2} w^T w + C \sum_{i=1}^N \xi_i$

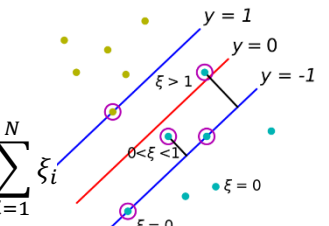
C says how hard a violation is punished. In hard margin SVM: $C \rightarrow \infty$.

Subject to: $f_i(w, b) = y_i(w^T x_i + b) - 1 + \xi_i \geq 0$
 $\xi_i \geq 0$

1. Lagrangian:

$$L(w, b, \xi, \alpha, \mu) = \frac{1}{2} w^T w + C \sum_{i=1}^N \xi_i$$

$$- \sum_{i=1}^N \alpha_i [y_i(w^T x_i + b) - 1 + \xi_i]$$

$$- \sum_{i=1}^N \mu_i \xi_i$$


2. Minimize w.r.t. \mathbf{w} , \mathbf{b} and ξ to construct the Dual:

$$\nabla_{\mathbf{w}} L(\mathbf{w}, b, \xi, \boldsymbol{\alpha}, \boldsymbol{\mu}) = \mathbf{w} - \sum_{i=1}^N \alpha_i y_i \mathbf{x}_i = 0$$

$$\frac{\partial L}{\partial b} = \sum_{i=1}^N \alpha_i y_i = 0$$

$$\frac{\partial L}{\partial \xi_i} = C - \alpha_i - \mu_i = 0 \quad \text{for } i = 1, \dots, N$$

3. Dual Problem:

$$\max_{\boldsymbol{\alpha}} g(\boldsymbol{\alpha}) = \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N y_i y_j \alpha_i \alpha_j \mathbf{x}_i^T \mathbf{x}_j$$

subject to:

$$\sum_{i=1}^N \alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq C, \text{ for } i = 1, \dots, N$$

9.2 Hinge Loss Formulation

The Hinge Loss penalizes the points that lie within the margin. We can optimize the hinge loss function directly, using standard gradient descent.

Objective:

$$\text{Minimize: } f_0(\mathbf{w}, b) = \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{i=1}^N \xi_i$$

$$\text{Subject to: } f_i(\mathbf{w}, b) = y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 + \xi_i \geq 0$$

$$\xi_i \geq 0$$

Hinge Loss Formulation:

$$\min_{\mathbf{w}, b} \frac{1}{2C} \mathbf{w}^T \mathbf{w} + \sum_{i=1}^N \max\{0, 1 - y_i(\mathbf{w}^T \mathbf{x}_i + b)\}$$

Can be solved with SGD.

SVMs are L2-regularized perceptrons with a Hinge-loss function.

9.3 Kernels

Encode similarity between arbitrary numerical or non-numerical data. High outcome \rightarrow High similarity.

$$k(\mathbf{x}_i, \mathbf{x}_j) := \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j) \text{ with } K: \mathbb{R}^D \times \mathbb{R}^D \rightarrow \mathbb{R}$$

Where $\phi(\mathbf{x})$ is a feature map (non-lin. transf.).

Mercer's Theorem:

A kernel is valid if it gives rise to a symmetric, PSD kernel matrix \mathbf{K} (Gram matrix) for any input data \mathbf{X} .

$$\mathbf{K} = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & \cdots & k(\mathbf{x}_1, \mathbf{x}_N) \\ \vdots & \ddots & \vdots \\ k(\mathbf{x}_N, \mathbf{x}_1) & \cdots & k(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix}$$

\mathbf{K} is PSD when $\det \mathbf{K} \geq 0$.

Or a Kernel $k: \mathcal{M} \times \mathcal{M} \rightarrow \mathbb{R}$ is valid, if there exists a feature map $\phi: \mathcal{M} \rightarrow \mathcal{H}$, such that $k(\mathbf{x}, \mathbf{y}) = \langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle_{\mathcal{H}}$ where $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ is the inner product on \mathcal{H} .

Kernel Preserving Operations:

Let $k_1: \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ and $k_2: \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ be kernels, with $\mathcal{X} \subseteq \mathbb{R}^N$. Then the following functions k are kernels as well:

- $k(\mathbf{x}_1, \mathbf{x}_2) = k_1(\mathbf{x}_1, \mathbf{x}_2) + k_2(\mathbf{x}_1, \mathbf{x}_2)$
- $k(\mathbf{x}_1, \mathbf{x}_2) = c \cdot k_1(\mathbf{x}_1, \mathbf{x}_2)$, with $c > 0$
- $k(\mathbf{x}_1, \mathbf{x}_2) = k_1(\mathbf{x}_1, \mathbf{x}_2) \cdot k_2(\mathbf{x}_1, \mathbf{x}_2)$
- $k(\mathbf{x}_1, \mathbf{x}_2) = k_3(\phi(\mathbf{x}_1), \phi(\mathbf{x}_2))$, with the kernel k_3 on $\mathcal{X}' \subseteq \mathbb{R}^M$ and $\phi: \mathcal{X} \rightarrow \mathcal{X}'$
- $k(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^T \mathbf{A} \mathbf{x}_2$, with $\mathbf{A} \in \mathbb{R}^N \times \mathbb{R}^N$ symmetric and positive semidefinite

Non-valid kernels loose convexity!

Kernel Types:

$$\text{Polynomial: } k(\mathbf{a}, \mathbf{b}) = (\mathbf{a}^T \mathbf{b})^p \text{ or } (\mathbf{a}^T \mathbf{b} + 1)^p$$

$$\text{Gaussian: } k(\mathbf{a}, \mathbf{b}) = \exp\left(-\frac{\|\mathbf{a} - \mathbf{b}\|^2}{2\sigma^2}\right)$$

$$\text{Sigmoid: } k(\mathbf{a}, \mathbf{b}) = \tanh(\kappa \mathbf{a}^T \mathbf{b} - \delta), \quad \kappa, \delta > 0$$

The sigmoid kernel is not PSD but works in practice.

$$\text{Linear: } k(\mathbf{a}, \mathbf{b}) = \mathbf{a}^T \mathbf{b}$$

Classifying a New Point with a Kernelized SVM:

Behaves like kNN-classifier with distance measure $k(\cdot)$

$$h(\mathbf{x}) = \text{sign} \left(\sum_{\{j | \mathbf{x}_j \in \mathcal{S}\}} \alpha_j y_j k(\mathbf{x}_j, \mathbf{x}) + b \right)$$

where \mathcal{S} is the set of support vectors with $\xi_i = 0$.

Multiple Classes:

One-vs-Rest: Train C SVM models for C classes. Winner is the class, where the distance from the hyperplane is maximal.

One-vs-One: Train all possible pairings and evaluate all. The winner is the class with the weighted majority vote.

10 Dimensionality Reduction

10.1 Principal Component Analysis

Find a coordinate system in which the features are lin. uncorrelated. The goal is to transform the data, s.t. the covariance between the new dimensions is 0.

1. Center the Data (zero mean):

$$\tilde{x}_i = x_i - \bar{x} \quad \text{with } \bar{x} = \frac{1}{N} \mathbf{X}^T \mathbf{1}_N = \frac{1}{N} \sum_{n=1}^N x_n$$

2. Compute the Covariance Matrix $\Sigma_{\tilde{X}}$:

$$\text{Var}(X_j) = \frac{1}{N} \sum_{i=1}^N (x_{ij} - \bar{x}_j)^2 = \frac{1}{N} \mathbf{X}_j^T \mathbf{X}_j - \bar{x}_j^2$$

$$\begin{aligned} \text{Cov}(X_{j_1}, X_{j_2}) &= \frac{1}{N} \sum_{i=1}^N (x_{ij_1} - \bar{x}_{j_1})(x_{ij_2} - \bar{x}_{j_2}) \\ &= \frac{1}{N} \mathbf{X}_{j_1}^T \mathbf{X}_{j_2} - \bar{x}_{j_1} \bar{x}_{j_2} \end{aligned}$$

$$\Sigma_{\tilde{X}} = \begin{bmatrix} \text{var}(X_1) & \text{cov}(X_1, X_2) & \cdots & \text{cov}(X_1, X_d) \\ \text{cov}(X_2, X_1) & \text{var}(X_2) & \cdots & \text{cov}(X_2, X_d) \\ \vdots & \cdots & \ddots & \vdots \\ \text{cov}(X_d, X_1) & \text{cov}(X_d, X_2) & \cdots & \text{var}(X_d) \end{bmatrix}$$

$$= \frac{1}{N} \tilde{\mathbf{X}}^T \tilde{\mathbf{X}} - \underbrace{\bar{\mathbf{x}} \bar{\mathbf{x}}^T}_{=0} = \frac{1}{N} \tilde{\mathbf{X}}^T \tilde{\mathbf{X}}$$

3. Transform the data s.t. the covariances between the new dimensions are 0:

$$\text{Eigendecomposition:} \quad \Sigma_{\tilde{X}} = \Gamma \cdot \Lambda \cdot \Gamma^T$$

Γ : Orthonormal matrix with column-eigenvectors γ_i

Λ : Diagonal matrix of eigenvalues λ_i .

$$\text{After transf.:} \quad \Sigma_{\tilde{X}'} = \Gamma \cdot \Sigma_{\tilde{X}} \cdot \Gamma^T = \Gamma^T \cdot \Gamma \Lambda \Gamma^T \cdot \Gamma = \Lambda$$

Dimensions in the new coordinate system (given by the eigenvectors/ *principal components*) have variance λ_i .

4. Dimensionality Reduction of Original Data:

$$\mathbf{Y} = \tilde{\mathbf{X}} \cdot \Gamma \quad \quad \bar{\mathbf{y}} = \Gamma \cdot \bar{\mathbf{x}}$$

$$\text{Keep } k \text{ largest var. dims:} \quad \mathbf{Y}_{\text{reduced}} = \tilde{\mathbf{X}} \cdot \Gamma_{\text{truncated}}$$

$$\text{with 90\% of the energy:} \quad k = \sum_{i=1}^k \lambda_i \geq 0.9 \cdot \sum_{i=1}^d \lambda_i$$

Power Iteration:

Iteratively compute the eigenvector with the largest absolute value. Initialize with arbitrary normalized vector v and iterate until convergence:

$$v \leftarrow \frac{\Sigma_{\tilde{X}} \cdot v}{\|\Sigma_{\tilde{X}} \cdot v\|}$$

$$\text{Deflated Matrix:} \quad \hat{\Sigma}_{\tilde{X}} = \Sigma_{\tilde{X}} - \lambda_1 \cdot \gamma_1 \gamma_1^T$$

Maximum Variance Formulation (Alt. view of PCA):

Project the data to a lower dimensional space \mathbb{R}^k with $k \ll d$ while maximizing the variance of the projected data.

Minimum Error Formulation (Alt. view of PCA):

Find an orthogonal set of k linear basis functions $w_j \in \mathbb{R}^d$ and corresponding low-dimensional projections $z_j \in \mathbb{R}^k$ such that the average reconstruction error is minimized with $\hat{x}_i = W z_i + \mu$. In other words, find one low dimensional projection which allows us to reconstruct the original data with the most information being preserved.

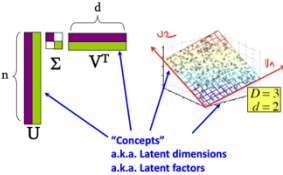
10.2 Singular Value Decomposition

The goal of SVD is to find the best low rank approximation by minimizing the reconstruction error.

$$\min_{\substack{B \\ \text{rank}(B)=k}} \|\mathbf{X} - \mathbf{B}\|_F^2 = \sum_{i=1}^N \sum_{j=1}^D (x_{ij} - b_{ij})^2$$

Here, the reconstruction error is the Frobenius norm. This is a constraint optimization problem.

Matrix Decomposition of $\mathbf{X} \in \mathbb{R}^{n \times d}$:

$$\mathbf{X} = \mathbf{U} \cdot \Sigma \cdot \mathbf{V}^T = \sum_{i=1}^{r=\text{Rank}(\mathbf{X})} \sigma_i \mathbf{u}_i \circ \mathbf{v}_i^T$$


$\mathbf{U} \in \mathbb{R}^{n \times r}$: column orthonormal
User/Entry-to-Concept Similarity
 $\mathbf{U} \leftarrow$ Eigenvectors of $\mathbf{X}\mathbf{X}^T$

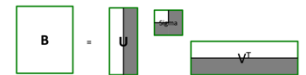
$\Sigma \in \mathbb{R}^{r \times r}$: diag. w/ decr.-order singular values σ_i
Strength of each concept
 $\Sigma \leftarrow \sqrt{\text{Eigenvalues of } \mathbf{X}^T \mathbf{X} \text{ or } \mathbf{X}\mathbf{X}^T}$

$\mathbf{V} \in \mathbb{R}^{d \times r}$: column orthonormal
Movie/Feature-to-Concept Similarity
 $\mathbf{V} \leftarrow$ Eigenvectors of $\mathbf{X}^T \mathbf{X}$

If \mathbf{X} has full rank, $r = d$.

Projected Data (Coordinates in new reference frame):

$$\mathbf{X}_{\text{proj}} = \mathbf{X} \cdot \mathbf{V} = \mathbf{U} \cdot \Sigma$$



For truncated SVD, $\mathbf{X}\mathbf{V}$ is preferable since we only compute the top k singular values. Columns of \mathbf{V} are the axes of the new reference frame.

SVD vs. PCA:

Transform the data such that dimensions of new space are uncorrelated and discard new dimensions with smallest variance is equivalent to finding the optimal low-rank approximation regarding the Frobenius norm.

$$\mathbf{X}^T \mathbf{X} = \mathbf{V} \Sigma^2 \mathbf{V}^T \rightarrow \Gamma = \mathbf{V} \quad \Sigma^2 = \Lambda$$

$$\text{Frobenius norm: } \|\mathbf{X}\|_F = \|\mathbf{X}^T\|_F, \quad \|\mathbf{X}\|_F^2 = \text{tr}(\mathbf{X}^T \mathbf{X})$$

11 Matrix Factorization

11.1 Latent Factor Model

RMSE for Evaluating Recommender Systems:

$$RMSE = \frac{1}{|S|} \sqrt{\sum_{(u,i) \in S} (r_{ui} - \hat{r}_{ui})^2} \propto \text{Squared Error}$$

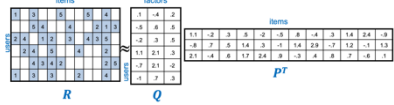
SVD on Rating Data:

Standard SVD cannot be used because missing elements are treated as zeros and lack of sparsity.

$$R \approx Q \cdot P^T \text{ with } Q = U\Sigma \in \mathbb{R}^{n \times k} \text{ and } P = V \in \mathbb{R}^{d \times k}$$

Latent Factor Minimization Problem:

Sum over existing entries $S = \{(u, i) | r_{ui} \neq \text{missing}\}$ & do not require P, Q to be orthogonal and unit length.

$$\min_{P, Q} \sum_{(u,i) \in S} (r_{ui} - q_u \cdot p_i^T)^2$$


11.2 Alternating Optimization

Alternatingly fix one variable and solve for the other.

1. Initialization $P^{(0)}, Q^{(0)}, t = 0$:

- Use SVD & replace missing entries by 0.
- With mean/ random/ Gauss-distr. values.

2. Alternatingly optimize $P^{(t+1)}$ and $Q^{(t+1)}$:

$$\begin{aligned} P^{(t+1)} &= \argmin_P f(P, Q^{(t)}) \\ &= \argmin_P \sum_{(u,i) \in S} (r_{ui} - q_u \cdot p_i^T)^2 \\ &= \sum_{i=1, \dots, d} \argmin_{P_i} \sum_{u \in S_{*,i}} (r_{ui} - q_u \cdot p_i^T)^2 \end{aligned}$$

$\hat{r}_{ui} = q_u \cdot p_i^T$
 q_u row u of Q
 p_i row i of P

Where $S_{*,i} = \{u | (u, i) \in S\}$ are all users who have rated item i .

$$\begin{aligned} Q^{(t+1)} &= \argmin_Q f(P^{(t+1)}, Q) \\ &= \sum_{u=1, \dots, n} \argmin_{q_u} \sum_{i \in S_{u,*}} (r_{ui} - q_u \cdot p_i^T)^2 \end{aligned}$$

Where $S_{u,*} = \{i | (u, i) \in S\}$ are all items for user u .

Closed Form Solution: (ordinary LS regression)

$$\begin{aligned} p_i^T &= \left(\frac{1}{|S_{*,i}|} \sum_{u \in S_{*,i}} q_u^T q_u \right)^{-1} \cdot \frac{1}{|S_{*,i}|} \sum_{u \in S_{*,i}} q_u^T r_{ui} \\ q_u^T &= \left(\frac{1}{|S_{u,*}|} \sum_{i \in S_{u,*}} p_i^T p_i \right)^{-1} \cdot \frac{1}{|S_{u,*}|} \sum_{i \in S_{u,*}} p_i^T r_{ui} \end{aligned}$$

For L_2 -
 Regularization add
 $+\lambda \cdot I_{d \times d}$ like in
 normal equation

3. Repeat step 2 until convergence.

The solution is simple to implement and efficient on sparse data. However, it is only an approximation and heavily depends on the initialization.

User, item & overall bias b_u, b_i, b :

$$\min_{P, Q} \sum_{(u,i) \in S} (r_{ui} - (q_u \cdot p_i^T + b_u + b_i + b))^2$$

11.3 Matrix Factorization with SGD

1. Pick random user u and a random item i with rating r_{ui} (batch size 1)
2. Compute the gradients $\frac{\partial \mathcal{L}}{\partial q_u}$ and $\frac{\partial \mathcal{L}}{\partial p_i}$
3. Update Rule: $q_u \leftarrow q_u - \tau \frac{\partial \mathcal{L}}{\partial q_u}, \quad p_i \leftarrow p_i - \tau \frac{\partial \mathcal{L}}{\partial p_i}$

Objective Function: $\mathcal{L} := \sum_{(u,i) \in S} (r_{ui} - q_u \cdot p_i^T)^2$

$$\begin{aligned} e_{ui} &\leftarrow r_{ui} - q_u \cdot p_i^T \\ q_u &\leftarrow q_u - 2\tau(e_{ui} p_i) \\ p_i &\leftarrow p_i - 2\tau(e_{ui} q_u) \end{aligned}$$

Objective Function: $\mathcal{L} + \text{Bias} + \text{Regularization}$:

$$\begin{aligned} e_{ui} &\leftarrow r_{ui} - (q_u \cdot p_i^T + b_u + b_i + b) \\ q_u &\leftarrow q_u - 2\tau(e_{ui} p_i - \lambda_1 q_u) \\ p_i &\leftarrow p_i - 2\tau(e_{ui} q_u - \lambda_2 p_i) \end{aligned}$$

$$\begin{aligned} b_u &\leftarrow b_u + 2\tau e_{ui} \\ b_i &\leftarrow b_i + 2\tau e_{ui} \\ b &= \frac{1}{|S|} \sum_{(u,i) \in S} r_{ui} \end{aligned}$$

11.4 Regularization

$$\min_{P, Q} \sum_{(u,i) \in S} (r_{ui} - q_u \cdot p_i^T)^2 + \left[\lambda_1 \sum_u \|q_u\|^2 + \lambda_2 \sum_i \|p_i\|^2 \right]$$

L2 regularization:

- Tries to shrink the parameter vector equally.
- Large values are highly penalized.
- It is unlikely that any component will be exactly 0.

L1 regularization:

- Enforces sparsity of the parameter vector.
- More intuitive that sparse input data comes from a sparse signal. Less latent factors to store.

11.5 Non-Negative Matrix Factorization

Task: Factorize non-negative A in non-negative Q and P , i.e. $A \approx Q \cdot P^T$

Formally:

- Given $A \in \mathbb{R}^{n \times d}$ with $A_{ij} \geq 0$ and integer k , find $P \in \mathbb{R}^{n \times k}, Q \in \mathbb{R}^{k \times d}$ such that $\|A - Q \cdot P^T\|_F$ is minimized subject to $Q \geq 0$ and $P \geq 0$

$$\min_{P \geq 0, Q \geq 0} \|A - Q \cdot P^T\|_F$$

11.6 Neighbor Graph Methods

Matrix factorization methods (PCA, SVD) preserve the global structure but may lose the local structure. NG methods preserve the neighborhood of each point.

General Procedure:

1. Construct neighbor graph of high-dim data.
2. Initialize points randomly in low-dim space.
3. Optimize coordinates in low-dim space s.t. similarities align (between low- & high-dim data)



t-Distributed Stochastic Neighbor Embedding (t-SNE):

- Is SotA for visualizing high-dim data
- Cluster sizes/ distances may be misleading
- Not good for more than output 3-dim.
- Reduced "Crowding Problem"

High-dim similarities for input \mathbf{x}_i :

Prob. of choosing \mathbf{x}_i when choosing according to the similarity to \mathbf{x}_j

$$p_{j|i} = \frac{\exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma_i^2}\right)}{\sum_{k \neq i} \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_k\|^2}{2\sigma_i^2}\right)} \quad p_{ii} = 0$$

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$$

Set σ_i such that perplexity is constant (fixed # of neigh.)

Low-dim similarities for parameters \mathbf{y}_i : (t-Distr.)

$$q_{ij} = \frac{(1 + \|\mathbf{y}_i - \mathbf{y}_j\|^2)^{-1}}{\sum_k \sum_{l \neq k} (1 + \|\mathbf{y}_k - \mathbf{y}_l\|^2)^{-1}} \quad q_{ii} = 0$$

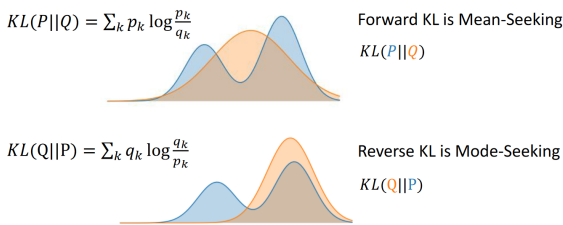
Change \mathbf{y}_i s.t. $p_{ij} \approx q_{ij}$: (min. their KL divergence)

$$\min_{\mathbf{y}_i} KL(P||Q) = \sum_i \sum_{j \neq i} p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

$$\frac{\partial KL(P||Q)}{\partial y_s} = 4 \sum_j (y_s - y_j)(p_{sj} - q_{sj}) (1 + \|y_s - y_j\|^2)^{-1}$$

KL Divergence:

$$KL(P||Q) \geq 0 \quad \forall P, Q \quad KL(P||Q) = 0 \text{ iff } P = Q$$



11.7 Auto-Encoders

Neural network that finds a compact representation of non-linear data by learning to reconstruct its input, i.e. $f_{dec}(f_{enc}(\mathbf{X})) = \mathbf{X}W_{enc}W_{dec} \approx \mathbf{X}$. The bottleneck layer has fewer neurons ($L \ll D$). Using an Autoencoder with lin. activations is like PCA/ SVD.

Objective: $\min_W \frac{1}{N} \sum_{i=1}^N \|f(\mathbf{x}_i, \mathbf{W}) - \mathbf{x}_i\|^2$

12 Clustering

Group objects into clusters based on their similarity.

12.1 K-Means

Distance Measures:

Manhattan: $\|\mathbf{x}_i - \mathbf{x}_j\|_1 = \sum_d |\mathbf{x}_{id} - \mathbf{x}_{jd}|$

Euclidean: $\|\mathbf{x}_i - \mathbf{x}_j\|_2 = \sqrt{\sum_d (\mathbf{x}_{id} - \mathbf{x}_{jd})^2}$

Mahalanobis: $d = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^T \Sigma^{-1} (\mathbf{x}_i - \mathbf{x}_j)}$

Objective Function:

$$J(\mathbf{X}, \mathbf{Z}, \boldsymbol{\mu}) = \sum_{i=1}^N \sum_{k=1}^K \mathbf{z}_{ik} \|\mathbf{x}_i - \boldsymbol{\mu}_k\|_2^2$$

with one-hot encoding $\mathbf{z}_i \in \{0,1\}^K$ and centroids $\boldsymbol{\mu}_k \in \mathbb{R}^D$. L1-norm would lead to K-Medians.

Alternating Optimization with Lloyd's algorithm:

1. Initialize the centroids $\boldsymbol{\mu} = \{\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_K\}$
2. Update cluster indicators (solve $\min_z J(\mathbf{X}, \mathbf{Z}, \boldsymbol{\mu})$)

$$\mathbf{z}_{ik} = \begin{cases} 1, & \text{if } k = \arg \min_j \|\mathbf{x}_i - \boldsymbol{\mu}_j\|_2^2 \\ 0, & \text{else} \end{cases}$$

3. Update centroids (solve $\min_{\boldsymbol{\mu}} J(\mathbf{X}, \mathbf{Z}, \boldsymbol{\mu})$)

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{i=1}^N \mathbf{z}_{ik} \mathbf{x}_i \quad \text{with} \quad N_k = \sum_{i=1}^N \mathbf{z}_{ik}$$

For L_1 : $\boldsymbol{\mu}_{kd} = \text{median}\{\mathbf{x}_{id} \text{ such that } \mathbf{z}_{ik} = 1\}$

4. If objective $J(\mathbf{X}, \mathbf{Z}, \boldsymbol{\mu})$ has not converged \rightarrow Step 2

Initialization of Centroids with K-means++:

1. Choose the first centroid $\boldsymbol{\mu}_1$ uniformly at random among the data points.
2. For each point \mathbf{x}_i compute the distance $D_i^2 = \|\mathbf{x}_i - \boldsymbol{\mu}_1\|_2^2$.
3. Sample the next centroid $\boldsymbol{\mu}_k$ from $\{\mathbf{x}_i\}$ with probability proportional to D_i^2 .
4. Recompute the distances $D_i^2 = \min\{\|\mathbf{x}_i - \boldsymbol{\mu}_1\|_2^2, \dots, \|\mathbf{x}_i - \boldsymbol{\mu}_k\|_2^2\}$.
5. Continue steps 3 and 4 until K initial centroids have been chosen.

Limitations:

Modeling Issues:

- Underlying assumptions are not explicit.
- Quality depends on the distance measure a lot.
- Can't detect clusters w/ overlapping convex hulls.
- Sensitivity to outliers.
- No uncertainty measure.

Algorithmic Issues:

- Extreme sensitivity to initialization.

12.2 Gaussian Mixture Model (GMM)

$$p(\mathbf{x}, \mathbf{z} | \boldsymbol{\theta}) = p(\mathbf{x} | \mathbf{z}, \boldsymbol{\theta}) \cdot p(\mathbf{z} | \boldsymbol{\theta})$$

Cluster Prior: $p(\mathbf{z} | \boldsymbol{\pi}) = \text{Cat}(\boldsymbol{\pi})$

Per-class distr.: $p(\mathbf{x} | \mathbf{z}_k = 1, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$

The variables \mathbf{z} never observed \rightarrow latent variables.

GMM Likelihood: (Mixture of K Gaussians)

$$\begin{aligned} p(\mathbf{x} | \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) &= \sum_{k=1}^K p(z_k = 1 | \boldsymbol{\pi}) p(\mathbf{x} | z_k = 1, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \\ &= \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \end{aligned}$$

Inference/ Responsibility/ Posterior:

$$\begin{aligned} p(z_{ik} = 1 | \mathbf{x}_i, \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) &= \frac{p(z_{ik} = 1 | \boldsymbol{\pi}) p(\mathbf{x}_i | z_{ik} = 1, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{p(\mathbf{x}_i | \boldsymbol{\pi}, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)} \\ &= \frac{\pi_k \mathcal{N}(\mathbf{x}_i | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_i | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} = \gamma(z_{ik} = 1) \end{aligned}$$

The latent variables are independent: $\gamma(\mathbf{Z}) = \prod_{i=1}^N \gamma(\mathbf{z}_i)$

$\gamma(z_{ik})$ = Responsibility of component k for observation i

EM Algorithm for GMM:

1. Initialize model parameters $\{\boldsymbol{\pi}^{(0)}, \boldsymbol{\mu}_1^{(0)}, \dots, \boldsymbol{\mu}_K^{(0)}, \boldsymbol{\Sigma}_1^{(0)}, \dots, \boldsymbol{\Sigma}_K^{(0)}\}$.
2. **E step.** Evaluate the responsibilities

$$\gamma_t(z_{ik}) = \frac{\pi_k^{(t)} \mathcal{N}(\mathbf{x}_i | \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)})}{\sum_{j=1}^K \pi_j^{(t)} \mathcal{N}(\mathbf{x}_i | \boldsymbol{\mu}_j^{(t)}, \boldsymbol{\Sigma}_j^{(t)})}$$

3. **M step.** Re-estimate the parameters

$$\begin{aligned} \boldsymbol{\mu}_k^{(t+1)} &= \frac{1}{N_k} \sum_{i=1}^N \gamma_t(z_{ik}) \mathbf{x}_i \\ \boldsymbol{\Sigma}_k^{(t+1)} &= \frac{1}{N_k} \sum_{i=1}^N \gamma_t(z_{ik}) (\mathbf{x}_i - \boldsymbol{\mu}_k^{(t+1)}) (\mathbf{x}_i - \boldsymbol{\mu}_k^{(t+1)})^T \\ \pi_k^{(t+1)} &= \frac{N_k}{N} \quad \text{where } N_k = \sum_{i=1}^N \gamma_t(z_{ik}). \end{aligned}$$

4. Iterate steps 2 & 3 until $\mathbb{E}_{\mathbf{Z} \sim \gamma_t(\mathbf{Z})} [\log p(\mathbf{X}, \mathbf{Z} | \boldsymbol{\pi}^{(t)}, \boldsymbol{\mu}^{(t)}, \boldsymbol{\Sigma}^{(t)})]$ converges

12.3 Expectation-Maximization Algorithm

Used when it is intractable to optimize the log-likelihood $\log p(\mathbf{X} | \boldsymbol{\theta})$ due to latent variables \mathbf{Z} . Easier to optimize the joint probability $\log p(\mathbf{X}, \mathbf{Z} | \boldsymbol{\theta})$.

- **E step** - evaluate the posterior $\gamma_t(\mathbf{Z}) = p(\mathbf{Z} | \mathbf{X}, \boldsymbol{\theta}^{(t)})$.
- **M step** - maximize the expected joint log-likelihood $\log p(\mathbf{X}, \mathbf{Z} | \boldsymbol{\theta})$ w.r.t. $\boldsymbol{\theta}$ under the current beliefs $\gamma_t(\mathbf{Z})$

$$\boldsymbol{\theta}^{(t+1)} = \arg \max_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{Z} \sim \gamma_t(\mathbf{Z})} [\log p(\mathbf{X}, \mathbf{Z} | \boldsymbol{\theta})]$$

$$\log p(\mathbf{X} | \boldsymbol{\theta}) = \mathcal{L}(\mathbf{q}, \boldsymbol{\theta}) + \mathbb{KL}(\mathbf{q} || p(\cdot | \mathbf{X}, \boldsymbol{\theta}))$$

After the E-step we have $\log p(\mathbf{X} | \boldsymbol{\theta}) = \mathcal{L}(\mathbf{q}, \boldsymbol{\theta})$. So if we now perform the M-step and maximize $\mathcal{L}(\mathbf{q}, \boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$, either

- $\mathcal{L}(\mathbf{q}, \boldsymbol{\theta})$ does not change, we are at a local maximum of $\log p(\mathbf{X} | \boldsymbol{\theta})$ and EM converges
- or $\mathcal{L}(\mathbf{q}, \boldsymbol{\theta})$ "pushes" up against $\log p(\mathbf{X} | \boldsymbol{\theta})$.

$\mathcal{L}(\mathbf{q}, \boldsymbol{\theta})$ is a lower-bound on $\log p(\mathbf{X} | \boldsymbol{\theta})$ for any \mathbf{q} , because $\mathcal{L}(\mathbf{q}, \boldsymbol{\theta})$ is maximal when the \mathbb{KL} divergence is 0, which happens when $q(\mathbf{Z}) = \gamma(\mathbf{Z}) = p(\mathbf{Z} | \mathbf{X}, \boldsymbol{\theta})$.

$$\log p(\mathbf{X} | \boldsymbol{\theta}) = \mathcal{L}(\mathbf{q}, \boldsymbol{\theta}) = \mathbb{E}_{\mathbf{Z} \sim q} \left[\log \frac{p(\mathbf{X}, \mathbf{Z} | \boldsymbol{\theta})}{q(\mathbf{Z})} \right]$$

E-Step:

Use Bayes rule as in the EM Algorithm for GMM.

M-Step:

1. Rewrite the expected data log-likelihood $\mathcal{L}(\gamma_t, \boldsymbol{\theta})$:

$$\mathbb{E}_{\mathbf{Z} \sim \gamma_t} [\log p(\mathbf{X}, \mathbf{Z} | \boldsymbol{\theta})] = \sum_{i=1}^N \sum_{k=1}^K \gamma(z_{ik}) \log p(\mathbf{x}_i, z_{ik} | \boldsymbol{\theta})$$

2. Absorb all terms constant w.r.t. $\boldsymbol{\theta}$.
3. Compute derivative w.r.t. $\boldsymbol{\theta}$ and find the root.

12.4 Choosing Number of Clusters

Heuristic Methods

Elbow/Knee Heuristic: Plot within-cluster sum of squared distances or likelihood for varying K . The optimal K is at the knee.

Gap Statistic: Compare within-cluster variation to uniform data. Choose smallest K such that gap statistic is within one std. dev. of gap at $K + 1$.

Silhouette: Per point difference of within-cluster mean distance to points in the closest other cluster. Maximize this for all points.

Choosing Number of Clusters: Probabilistic Methods

Needs generative model that defines the data likelihood $\hat{L} = p(\mathbf{X} | \hat{\mathbf{Z}}, \hat{\boldsymbol{\theta}})$ (with optimal parameters $\hat{\mathbf{Z}}, \hat{\boldsymbol{\theta}}$).

- **Bayesian information criterion (BIC):**
Approximate model likelihood $p(\mathbf{X} | \text{model})$. Balances number of parameters vs. likelihood, $\text{BIC} = M \log N - 2 \log \hat{L}$.
- **Akaike information criterion (AIC):**
Estimate information lost by given model, $\text{AIC} = 2M - 2 \log \hat{L}$
- Here M is the number of free parameters (e.g. $K \cdot (D + D^2 + 1)$ for GMM) and N is the number of samples

12.5 Hierarchical Clustering

- **Agglomerative:** Bottom-up, merge cluster pairs iteratively

- **Divisive:** Top-down, split data recursively

Cluster pairs are chosen by minimum cluster distance.

Linkage criterion defines cluster distance from sample distance:

- **Complete-linkage clustering:** Maximum sample distance
- **Single-linkage clustering:** Minimum sample distance
- **Unweighted average linkage clustering (UPGMA):** Mean sample distance
- **Centroid linkage clustering (UPGMC):** Centroid distance
- **Weighted average linkage clustering (WPGMA):** Distance defined recursively as average of distances in previous level (before merging)

13 Advanced Topics

Considerations beyond Accuracy:

Privacy, Security, Fairness, Explainability, Accountability

13.1 Differential Privacy

Model Inversion: Attacks recover information about the training data from the trained model.

Randomized Response: Introducing randomization to provide plausible deniability.

1. Flip a coin. If it lands tails answer truthfully.
2. Else flip another coin. If it lands tails, answer yes, else no.

Unbiased Estimator: $\mu = \frac{1}{4}(1 - \hat{\mu}) + \frac{3}{4}\hat{\mu}$ where $\hat{\mu}$ is the MLE.

Definition Differential Privacy:

A randomized mechanism $\mathcal{M}_f : \mathcal{X} \rightarrow \mathcal{Y}^2$ is ϵ -differentially private if for all neighboring inputs $X \simeq X'$ and for all sets of outputs $Y \subseteq \mathcal{Y}$ we have:³

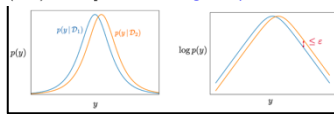
$$\mathbb{P}[\mathcal{M}_f(X) \in Y] \leq e^\epsilon \mathbb{P}[\mathcal{M}_f(X') \in Y]$$

For any possible set of outputs we have

$$e^{-\epsilon} \leq \frac{\mathbb{P}[\mathcal{M}_f(X) \in Y]}{\mathbb{P}[\mathcal{M}_f(X') \in Y]} \leq e^\epsilon$$

$\mathcal{M}_f(X') \in Y$ is the probability that the mechanism's output for changed inputs is in Y .

$$\Rightarrow |\log(\cdot) - \log(\cdot)| \leq \epsilon$$



Laplace Mechanism:

- Define the global sensitivity of a function $f : \mathcal{X} \rightarrow \mathbb{R}^d$ as $\Delta_p = \sup_{X \simeq X'} \|f(X) - f(X')\|_p$
- Δ_p measures the magnitude by which a single instance can change the output of the function in the worst case
- Output perturbation with Laplace Noise:
 - A curator holds data $X = (x_1, \dots, x_n) \in \mathcal{X}$ about n individuals
 - The curator computes the function $f(X)$
 - They sample i.i.d. Laplace noise $Z \sim \text{Lap}(0, \frac{\Delta_1}{\epsilon})^d$ Dimensions d are independent.
 - They reveal the noisy value $f(X) + Z$

$$\mathcal{M}_f(X) \sim p_X(y) \sim \text{Lap}\left(y|f(X), \frac{\Delta_1}{\epsilon}\right)^d = \prod_{i=1}^d \frac{\epsilon}{2\Delta_1} e^{-\frac{\epsilon}{\Delta_1}|y_i - f(X)_i|}$$

Perturbation Techniques:

Perturb input, weights, objective or gradients.

Fundamental properties of DP:

- Robustness to post-processing. Operations after applying the DP-mechanism preserve DP.
- Composition: If the data is composed of different DP datasets, the overall data is also DP.
- Group privacy: If \mathcal{M} is ϵ -DP w.r.t. $X \simeq X'$, then \mathcal{M} is $(t\epsilon)$ -DP for t changed instances ($X \simeq_t X'$).

Federated Learning: Learning a model without any centralized entity having access to all the data. \rightarrow Send weights from server to user \rightarrow Compute loss and gradients locally \rightarrow Send updates weights + noise back.

13.2 Algorithmic Fairness

Causes of Bias:

- Tainted training data: algorithm maintains the existing bias caused by human bias.
- Skewed sample: initial predictions influence future observations; selection bias.
- Proxies: some features may be highly correlated to excluded (& legally protected) features.
- Sample size disparity: models tend to fit larger groups first.
- Limited features: features may be less informative or reliable for minority groups.

Notions of Fairness:

- Group fairness: treat all groups equally.
- Individual fairness: treat similar examples similarly.
- Counterfactual fairness: uses tools from causal inference. "What would the decision be if the individual belonged to a different group?"

Fairness through Unawareness:

Use $R = r(X)$ instead of $R = r(X, A)$. Does not work due to correlations with of features in X with A .

First Fairness Criterion: Independence

Binary predictor R indep. of sensitive features A : $R \perp A$

For all groups a, b : $P_a\{R = 1\} = P_b\{R = 1\}$

or relaxed: $|P_a\{R = 1\} - P_b\{R = 1\}| \leq \epsilon$

+ Legal support.

- Rules out the optimal predictor when base rates are different across groups.

Second Criterion: Separation

R independent of A , given target Y : $R \perp A \mid Y$

Enforce equal TP and FP rates for all groups a, b :

$P_a(R = 1 \mid Y = 1) = P_b(R = 1 \mid Y = 1)$ true positive (TP)

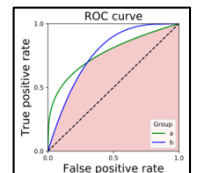
$P_a(R = 1 \mid Y = 0) = P_b(R = 1 \mid Y = 0)$ false positive (FP)

Enforce only TP-rate \rightarrow Equality of opportunity

+ Optimal predictor not ruled out.

+ Penalizes laziness.

- May not close gap between groups.



Third Criterion: Sufficiency

Labels Y independent of A , given prediction R : $Y \perp A \mid R$

Enforce equal target rates given equal scores for a, b :

$P_a\{Y = 1 \mid R = r\} = P_b\{Y = 1 \mid R = r\}$

+ Satisfied by Bayes' optimal classifier.

+ No need to see A for predicting Y when is R given.

+ Equal chance of success $Y = 1$ given acceptance $R = 1$

- May not close gaps between the groups.

14 Probability Distributions

14.1 Gaussian

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

$$\text{With } \beta = \frac{1}{\sigma^2}: \quad \mathcal{N}(x | \mu, \beta^{-1}) = \frac{\beta}{\sqrt{2\pi}} e^{-\frac{\beta}{2}(x-\mu)^2}$$

$$\sum_{i=1}^N \log \mathcal{N}(x_i | \mu_i, \beta^{-1}) = \frac{N}{2} \log \beta - \frac{N}{2} \log(2\pi) - \frac{\beta}{2} \sum_{i=1}^N (\mu_i - x_i)^2$$

$$\mathbb{E}[x] = \mu \quad \text{Var}(x) = \sigma^2$$

Multivariate:

$$\mathcal{N}(\mathbf{x} | \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{\frac{D}{2}} |\boldsymbol{\Sigma}|^{\frac{1}{2}}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x}-\boldsymbol{\mu})}$$

$$\log \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}, \boldsymbol{\Sigma}) = -\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x}-\boldsymbol{\mu}) - \frac{D}{2} \log 2\pi - \frac{1}{2} \log(\det \boldsymbol{\Sigma})$$

$$\mathbb{E}[\mathbf{x}] = \boldsymbol{\mu} \quad \boldsymbol{\Sigma} = \mathbb{E}[\mathbf{x}\mathbf{x}^T] - \mathbb{E}[\mathbf{x}] \mathbb{E}[\mathbf{x}]^T \quad (\mathbf{x} \text{ is a vector})$$

$$\text{For matrix } \mathbf{X}: \quad \boldsymbol{\Sigma} = \mathbb{E}[\mathbf{X}^T \mathbf{X}] - \mathbb{E}[\mathbf{X}]^T \mathbb{E}[\mathbf{X}]$$

Sum of two Gaussians is a Gaussian

$$z = x + y \quad \text{with } x \sim \mathcal{N}(\mu_x, \sigma_x^2) \text{ and } y \sim \mathcal{N}(\mu_y, \sigma_y^2)$$

$$\Rightarrow z \sim \mathcal{N}(\mu_x + \mu_y, \sigma_x^2 + \sigma_y^2)$$

14.2 Bernoulli

$$\text{Ber}(x | \theta) = \theta^x (1 - \theta)^{1-x}$$

$$\log(\text{Ber}(x | \theta)) = x \log \theta + (1 - x) \log(1 - \theta)$$

$$\mathbb{E}[x] = \theta \quad \text{Var}(x) = p(1 - p)$$

14.1 Binomial

$$\text{Bin}(X = k | N, \theta) = \binom{N}{k} \theta^k (1 - \theta)^{N-k}$$

$$\log \text{Bin}(k | N, \theta) = \log \binom{N}{k} + x \log \theta + (N - k) \log(1 - \theta)$$

$$\mathbb{E}[x] = N\theta \quad \text{Var}(x) = Np(1 - p)$$

14.1 Beta

$$\text{Beta}(x | a, b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} x^{a-1} (1-x)^{b-1}$$

$$\log \text{Beta}(x | a, b) \propto (a-1) \log x + (b-1) \log(1-x)$$

$$\mathbb{E}[x] = \frac{a}{a+b} \quad \text{Var}(x) = \frac{ab}{(a+b)^2(a+b+1)}$$

14.1 Gamma

$$\text{Gam}(x | a, b) = \frac{1}{\Gamma(a)} b^a x^{a-1} e^{-bx}$$

$$\log \text{Gam}(x | a, b) \propto (a-1) \log x - bx$$

$$\mathbb{E}[x] = \frac{a}{b} \quad \text{Var}(x) = \frac{a}{b^2}$$

14.1 Laplace

$$\text{Lap}(x | \mu, b) = \frac{1}{2b} e^{-\frac{|x-\mu|}{b}}$$

$$\log \text{Lap}(x | \mu, b) = -\log 2b - \frac{|x-\mu|}{b}$$

$$\mathbb{E}[x] = \mu \quad \text{Var}(x) = 2b^2$$

14.1 Others

Distribution	PDF or PMF	Mean	Variance
<i>Bernoulli</i> (p)	$\begin{cases} p, & \text{if } x = 1 \\ 1-p, & \text{if } x = 0. \end{cases}$	p	$p(1-p)$
<i>Binomial</i> (n, p)	$\binom{n}{k} p^k (1-p)^{n-k}$ for $0 \leq k \leq n$	np	npq
<i>Geometric</i> (p)	$p(1-p)^{k-1}$ for $k = 1, 2, \dots$	$\frac{1}{p}$	$\frac{1-p}{p^2}$
<i>Poisson</i> (λ)	$e^{-\lambda} \lambda^k / k!$ for $k = 1, 2, \dots$	λ	λ
<i>Uniform</i> (a, b)	$\frac{1}{b-a} \quad \forall x \in (a, b)$	$\frac{a+b}{2}$	$\frac{(b-a)^2}{12}$
<i>Gaussian</i> (μ, σ^2)	$\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$	μ	σ^2
<i>Exponential</i> (λ)	$\lambda e^{-\lambda x} \quad x \geq 0, \lambda > 0$	$\frac{1}{\lambda}$	$\frac{1}{\lambda^2}$

15 Derivatives

Derivative w.r.t. column vector:

$$\frac{\partial \mathbf{a}^T \mathbf{x}}{\partial \mathbf{x}} = \frac{\partial \mathbf{x}^T \mathbf{a}}{\partial \mathbf{x}} = \mathbf{a}$$

$$\frac{\partial \mathbf{A} \mathbf{x}}{\partial \mathbf{x}} = \mathbf{x} \quad \frac{\partial \mathbf{x}^T \mathbf{A}}{\partial \mathbf{x}} = \mathbf{A}^T$$

$$\frac{\partial \mathbf{x}^T \mathbf{A} \mathbf{x}}{\partial \mathbf{x}} = \begin{cases} 2\mathbf{A} \mathbf{x}, & \text{A is symmetric} \\ (\mathbf{A} + \mathbf{A}^T) \mathbf{x}^T, & \text{otherwise} \end{cases}$$

Derivative w.r.t. matrix:

$$\frac{\partial \mathbf{a}^T \mathbf{X}^{-1} \mathbf{b}}{\partial \mathbf{X}} = -\mathbf{X}^{-T} \mathbf{b} \mathbf{a}^T \mathbf{X}^{-T}$$

$$\frac{\partial \log |\det \mathbf{X}|}{\partial \mathbf{X}} = \mathbf{X}^{-T} \quad \text{and} \quad \log \det \mathbf{X} = -\log \det \mathbf{X}^{-1}$$

$$\frac{\partial \text{tr}(\mathbf{A} \mathbf{B})}{\partial \mathbf{A}} = \mathbf{B}^T$$

Likelihood	Model parameters	Conjugate prior distribution	Prior hyperparameters	Posterior hyperparameters ^[note 1]	Interpretation of hyperparameters	Posterior predictive ^[note 2]
Bernoulli	<i>p</i> (probability)	Beta	<i>α</i> , <i>β</i>	$\alpha + \sum_{i=1}^n x_i, \beta + n - \sum_{i=1}^n x_i$	<i>α</i> successes, <i>β</i> failures ^[note 3]	$p(\tilde{x} = 1) = \frac{\alpha'}{\alpha' + \beta'}$
Binomial	<i>p</i> (probability)	Beta	<i>α</i> , <i>β</i>	$\alpha + \sum_{i=1}^n x_i, \beta + \sum_{i=1}^n N_i - \sum_{i=1}^n x_i$	<i>α</i> successes, <i>β</i> failures ^[note 3]	BetaBin(<i>̄x</i> <i>α'</i> , <i>β'</i>) (beta-binomial)
Negative binomial with known failure number, <i>r</i>	<i>p</i> (probability)	Beta	<i>α</i> , <i>β</i>	$\alpha + \sum_{i=1}^n x_i, \beta + r n$	<i>α</i> total successes, <i>β</i> failures ^[note 3] (i.e., $\frac{\beta}{r}$ experiments, assuming <i>r</i> stays fixed)	BetaNegBin(<i>̄x</i> <i>α'</i> , <i>β'</i>) (beta-negative binomial)
Poisson	<i>λ</i> (rate)	Gamma	<i>k</i> , <i>θ</i>	$k + \sum_{i=1}^n x_i, \frac{\theta}{n\theta + 1}$	<i>k</i> total occurrences in $\frac{1}{\theta}$ intervals	$\text{NB}\left(\tilde{x} \mid k', \frac{\theta'}{\theta' + 1}\right)$ (negative binomial)
			<i>α</i> , <i>β</i> ^[note 4]	$\alpha + \sum_{i=1}^n x_i, \beta + n$	<i>α</i> total occurrences in <i>β</i> intervals	$\text{NB}\left(\tilde{x} \mid \alpha', \frac{1}{1 + \beta'}\right)$ (negative binomial)
Categorical	p (probability vector), <i>K</i> (number of categories; i.e., size of p)	Dirichlet	α	α + (<i>c</i> ₁ , . . . , <i>c</i> _k), where <i>c</i> _{<i>i</i>} is the number of observations in category <i>i</i>	<i>α</i> _{<i>i</i>} occurrences of category <i>i</i> ^[note 3]	$p(\tilde{x} = i) = \frac{\alpha_i'}{\sum_i \alpha_i'}$ $= \frac{\alpha_i + c_i}{\sum_i \alpha_i + n}$
Multinomial	p (probability vector), <i>k</i> (number of categories; i.e., size of p)	Dirichlet	α	α + $\sum_{i=1}^n \mathbf{x}_i$	<i>α</i> _{<i>i</i>} occurrences of category <i>i</i> ^[note 3]	DirMult(̄x α') (Dirichlet-multinomial)
Hypergeometric with known total population size, <i>N</i>	<i>M</i> (number of target members)	Beta-binomial ^[4]	<i>n</i> = <i>N</i> , <i>α</i> , <i>β</i>	$\alpha + \sum_{i=1}^n x_i, \beta + \sum_{i=1}^n N_i - \sum_{i=1}^n x_i$	<i>α</i> successes, <i>β</i> failures ^[note 3]	
Geometric	<i>ρ</i> ₀ (probability)	Beta	<i>α</i> , <i>β</i>	$\alpha + n, \beta + \sum_{i=1}^n x_i$	<i>α</i> experiments, <i>β</i> total failures ^[note 3]	

Likelihood	Model parameters	Conjugate prior distribution	Prior hyperparameters	Posterior hyperparameters ^[note 1]	Interpretation of hyperparameters	Posterior predictive ^[note 5]
Normal with known variance <i>σ</i> ²	<i>μ</i> (mean)	Normal	<i>μ</i> ₀ , <i>σ</i> ₀ ²	$\frac{1}{\frac{1}{\sigma_0^2} + \frac{n}{\sigma^2}} \left(\frac{\mu_0}{\sigma_0^2} + \frac{\sum_{i=1}^n x_i}{\sigma^2} \right), \left(\frac{1}{\sigma_0^2} + \frac{n}{\sigma^2} \right)^{-1}$	mean was estimated from observations with total precision (sum of all individual precisions) 1 / <i>σ</i> ₀ ² and with sample mean <i>μ</i> ₀	$\mathcal{N}(\tilde{x} \mu_0', \sigma_0'^2 + \sigma^2)$ ^[5]
Normal with known precision <i>τ</i>	<i>μ</i> (mean)	Normal	<i>μ</i> ₀ , <i>τ</i> ₀	$\frac{\tau_0 \mu_0 + \tau \sum_{i=1}^n x_i}{\tau_0 + n\tau}, \tau_0 + n\tau$	mean was estimated from observations with total precision (sum of all individual precisions) τ ₀ and with sample mean <i>μ</i> ₀	$\mathcal{N}\left(\tilde{x} \mid \mu_0', \frac{1}{\tau_0} + \frac{1}{\tau}\right)$ ^[5]
Normal with known mean <i>μ</i>	<i>σ</i> ² (variance)	Inverse gamma	<i>α</i> , <i>β</i> ^[note 6]	$\alpha + \frac{n}{2}, \beta + \frac{\sum_{i=1}^n (x_i - \mu)^2}{2}$	variance was estimated from 2α observations with sample variance <i>β</i> / <i>α</i> (i.e. with sum of squared deviations 2β , where deviations are from known mean <i>μ</i>)	$t_{2\alpha'}(\tilde{x} \mu, \sigma^2 = \beta' / \alpha')$ ^[5]
Normal with known mean <i>μ</i>	<i>σ</i> ² (variance)	Scaled inverse chi-squared	<i>ν</i> , <i>σ</i> ₀ ²	$\nu + n, \frac{\nu \sigma_0^2 + \sum_{i=1}^n (x_i - \mu)^2}{\nu + n}$	variance was estimated from <i>ν</i> observations with sample variance <i>σ</i> ₀ ²	$t_{\nu'}(\tilde{x} \mu, \sigma_0'^2)$ ^[5]
Normal with known mean <i>μ</i>	<i>τ</i> (precision)	Gamma	<i>α</i> , <i>β</i> ^[note 4]	$\alpha + \frac{n}{2}, \beta + \frac{\sum_{i=1}^n (x_i - \mu)^2}{2}$	precision was estimated from 2α observations with sample variance <i>β</i> / <i>α</i> (i.e. with sum of squared deviations 2β , where deviations are from known mean <i>μ</i>)	$t_{2\alpha'}(\tilde{x} \mid \mu, \sigma^2 = \beta' / \alpha')$ ^[5]
Normal ^[note 7]	<i>μ</i> and <i>σ</i> ² Assuming exchangeability	Normal-inverse gamma	<i>μ</i> ₀ , <i>ν</i> , <i>α</i> , <i>β</i>	$\frac{\nu \mu_0 + n \bar{x}}{\nu + n}, \nu + n, \alpha + \frac{n}{2}, \beta + \frac{1}{2} \sum_{i=1}^n (x_i - \bar{x})^2 + \frac{n\nu}{\nu + n} \frac{(\bar{x} - \mu_0)^2}{2}$ <ul style="list-style-type: none"> <i>̄x</i> is the sample mean	mean was estimated from <i>ν</i> observations with sample mean <i>μ</i> ₀ ; variance was estimated from 2α observations with sample mean <i>μ</i> ₀ and sum of squared deviations 2β	$t_{2\alpha'}\left(\tilde{x} \mid \mu', \frac{\beta'(\nu' + 1)}{\nu' \alpha'}\right)$ ^[5]
Normal	<i>μ</i> and <i>τ</i> Assuming exchangeability	Normal-gamma	<i>μ</i> ₀ , <i>ν</i> , <i>α</i> , <i>β</i>	$\frac{\nu \mu_0 + n \bar{x}}{\nu + n}, \nu + n, \alpha + \frac{n}{2}, \beta + \frac{1}{2} \sum_{i=1}^n (x_i - \bar{x})^2 + \frac{n\nu}{\nu + n} \frac{(\bar{x} - \mu_0)^2}{2}$ <ul style="list-style-type: none"> <i>̄x</i> is the sample mean	mean was estimated from <i>ν</i> observations with sample mean <i>μ</i> ₀ , and precision was estimated from 2α observations with sample mean <i>μ</i> ₀ and sum of squared deviations 2β	$t_{2\alpha'}\left(\tilde{x} \mid \mu', \frac{\beta'(\nu' + 1)}{\alpha' \nu'}\right)$ ^[5]
Multivariate normal with known covariance matrix Σ	μ (mean vector)	Multivariate normal	μ ₀ , Σ ₀	$(\mathbf{\Sigma}_0^{-1} + n \mathbf{\Sigma}^{-1})^{-1} (\mathbf{\Sigma}_0^{-1} \mu_0 + n \mathbf{\Sigma}^{-1} \bar{\mathbf{x}}), (\mathbf{\Sigma}_0^{-1} + n \mathbf{\Sigma}^{-1})^{-1}$ <ul style="list-style-type: none"> ̄x is the sample mean	mean was estimated from observations with total precision (sum of all individual precisions) Σ ₀ ^{−1} and with sample mean <i>μ</i> ₀	$\mathcal{N}(\bar{\mathbf{x}} \mid \mu_0', \mathbf{\Sigma}_0' + \mathbf{\Sigma})$ ^[5]
Multivariate normal with known precision matrix Λ	μ (mean vector)	Multivariate normal	μ ₀ , Λ ₀	$(\mathbf{\Lambda}_0 + n \mathbf{\Lambda})^{-1} (\mathbf{\Lambda}_0 \mu_0 + n \mathbf{\Lambda} \bar{\mathbf{x}}), (\mathbf{\Lambda}_0 + n \mathbf{\Lambda})$ <ul style="list-style-type: none"> ̄x is the sample mean	mean was estimated from observations with total precision (sum of all individual precisions) Λ ₀ and with sample mean <i>μ</i> ₀	$\mathcal{N}\left(\bar{\mathbf{x}} \mid \mu_0', (\mathbf{\Lambda}_0'^{-1} + \mathbf{\Lambda}^{-1})^{-1}\right)$ ^[5]
Multivariate normal with known mean μ	Σ (covariance matrix)	Inverse-Wishart	<i>ν</i> , Ψ	$n + \nu, \mathbf{\Psi} + \sum_{i=1}^n (\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^T$	covariance matrix was estimated from <i>ν</i> observations with sum of pairwise deviation products Ψ	$t_{\nu' - p + 1}\left(\bar{\mathbf{x}} \mu', \frac{1}{\nu' - p + 1} \mathbf{\Psi}'\right)$ ^[5]
Multivariate normal with known mean μ	Λ (precision matrix)	Wishart	<i>ν</i> , V	$n + \nu, \left(\mathbf{V}^{-1} + \sum_{i=1}^n (\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^T\right)^{-1}$	covariance matrix was estimated from <i>ν</i> observations with sum of pairwise deviation products V ^{−1}	$t_{\nu' - p + 1}\left(\bar{\mathbf{x}} \mid \mu', \frac{1}{\nu' - p + 1} \mathbf{V}'^{-1}\right)$ ^[5]
Multivariate normal	μ (mean vector) and Σ (covariance matrix)	normal-inverse-Wishart	μ ₀ , <i>κ</i> ₀ , <i>ν</i> ₀ , Ψ	$\frac{\kappa_0 \mu_0 + n \bar{\mathbf{x}}}{\kappa_0 + n}, \kappa_0 + n, \nu_0 + n, \mathbf{\Psi} + \mathbf{C} + \frac{\kappa_0 n}{\kappa_0 + n} (\bar{\mathbf{x}} - \mu_0)(\bar{\mathbf{x}} - \mu_0)^T$ <ul style="list-style-type: none"> ̄x is the sample mean C = $\sum_{i=1}^n (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T$	mean was estimated from <i>κ</i> ₀ observations with sample mean <i>μ</i> ₀ ; covariance matrix was estimated from <i>ν</i> ₀ observations with sample mean <i>μ</i> ₀ and with sum of pairwise deviation products Ψ = <i>ν</i> ₀ Σ ₀	$t_{\nu_0' - p + 1}\left(\bar{\mathbf{x}} \mu_0', \frac{\kappa_0' + 1}{\kappa_0'(\nu_0' - p + 1)} \mathbf{\Psi}'\right)$ ^[5]
				$\frac{\kappa_0 \mu_0 + n \bar{\mathbf{x}}}{\kappa_0 + n}, \kappa_0 + n, \nu_0 + n, \left(\mathbf{V}^{-1} + \mathbf{C} + \frac{\kappa_0 n}{\kappa_0 + n} (\bar{\mathbf{x}} - \mu_0)(\bar{\mathbf{x}} - \mu_0)^T\right)^{-1}$ <ul style="list-style-type: none"> ̄x is the sample mean C = $\sum_{i=1}^n (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T$	mean was estimated from <i>κ</i> ₀ observations with sample mean <i>μ</i> ₀ ; covariance matrix was estimated from <i>ν</i> ₀ observations with sample mean <i>μ</i> ₀ and with sum of pairwise deviation products V ^{−1}	$t_{\nu_0' - p + 1}\left(\bar{\mathbf{x}} \mid \mu_0', \frac{\kappa_0' + 1}{\kappa_0'(\nu_0' - p + 1)} \mathbf{V}'^{-1}\right)$ ^[5]
Uniform	<i>U</i> (0, <i>θ</i>)	Pareto	<i>x</i> _m , <i>k</i>	max { <i>x</i> ₁ , . . . , <i>x</i> _{<i>n</i>} , <i>x</i> _m }, <i>k</i> + <i>n</i>	<i>k</i> observations with maximum value <i>x</i> _m	
Pareto with known minimum <i>x</i> _m	<i>k</i> (shape)	Gamma	<i>α</i> , <i>β</i>	$\alpha + n, \beta + \sum_{i=1}^n \ln \frac{x_i}{x_m}$	<i>α</i> observations with sum <i>β</i> of the order of magnitude of each observation (i.e. the logarithm of the ratio of each observation to the minimum <i>x</i> _m)	
Weibull with known shape <i>β</i>	<i>θ</i> (scale)	Inverse gamma ^[4]	<i>a</i> , <i>b</i>	$a + n, b + \sum_{i=1}^n x_i^\beta$	<i>a</i> observations with sum <i>b</i> of the <i>β</i> th power of each observation	
Log-normal	Same as for the normal distribution after exponentiating the data					
Exponential	<i>λ</i> (rate)	Gamma	<i>α</i> , <i>β</i> ^[note 4]	$\alpha + n, \beta + \sum_{i=1}^n x_i$	<i>α</i> − 1 observations that sum to <i>β</i> ^[6]	Lomax(<i>̄x</i> <i>β'</i> , <i>α'</i>) (Lomax distribution)
Gamma with known shape <i>α</i>	<i>β</i> (rate)	Gamma	<i>α</i> ₀ , <i>β</i> ₀	$\alpha_0 + n\alpha, \beta_0 + \sum_{i=1}^n x_i$	<i>α</i> ₀ / <i>α</i> observations with sum <i>β</i> ₀	CG(̄x <i>α</i> ₀ ['] , <i>β</i> ₀ [']) = <i>β'</i> (̄x <i>α</i> ₀ ['] , 1 , <i>β</i> ₀ [']) ^[note 8]
Inverse Gamma with known shape <i>α</i>	<i>β</i> (inverse scale)	Gamma	<i>α</i> ₀ , <i>β</i> ₀	$\alpha_0 + n\alpha, \beta_0 + \sum_{i=1}^n \frac{1}{x_i}$	<i>α</i> ₀ / <i>α</i> observations with sum <i>β</i> ₀	
Gamma with known rate <i>β</i>	<i>α</i> (shape)	$\propto \frac{a^{\alpha-1} \beta^{\alpha c}}{\Gamma(\alpha)^b}$	<i>a</i> , <i>b</i> , <i>c</i>	$a \prod_{i=1}^n x_i, b + n, c + n$	<i>b</i> or <i>c</i> observations (<i>b</i> for estimating <i>α</i> , <i>c</i> for estimating <i>β</i>) with product <i>a</i>	