Supervised Classification

Boosting

AdaBoosting

- **Weight Assignment:** $w_{i,0} = \frac{1}{n}$
- **Error Rate:** $r_j = \frac{\sum_{\substack{i=1 \\ \hat{y}_j^{(i)} \neq y^{(i)}}}^{N} w^{(i)}}{\sum_{i=1}^{N} w^{(i)}}$
- Predictor Weight: $\alpha_j = \eta \ln\left(\frac{1-r_j}{r_j}\right)$

Hard SVM

- $y(x) = w^T\phi(x) + b = 0$
- **Margin:** $\frac{1}{\|w\|}$
- **Objective:** $\min_{w,b} \frac{1}{2}\|w\|^2$
- **Discriminant Function:** $f(x) = w^T\phi(x) + b$
- **Support Vectors:** $y_i(w^T\phi(x_i) + b) = 1$
- **Polynomial Kernel:** $K(x,y) = (1 + \langle x, y\rangle)^d$
- **Gaussian RBF Kernel:** $K(x,y) = \exp\left(-\frac{\|x-y\|^2}{2\sigma^2}\right)$

Soft SVM

- **Objective:** $\min_{w,b,\xi} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi_i$
- **Constraints:** $y_i(w^T\phi(x_i) + b) \geq 1 - \xi_i$
- **Slack Variables:** $\xi_i \geq 0$
- **Lagrangian:** $\mathcal{L}(w,b,\xi,\alpha,\beta) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i(y_i(w^T\phi(x_i) + b) - 1 + \xi_i) - \sum_{i=1}^{N}\beta_i\xi_i$
- **KKT Conditions:** $\alpha_i \geq 0, \beta_i \geq 0, \alpha_i(y_i(w^T\phi(x_i) + b) - 1 + \xi_i) = 0, \beta_i\xi_i = 0$
- **Dual Problem:** $\max_\alpha \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j K(x_i,x_j)$
- **Predictor:** $f(x) = \sum_{i=1}^{N}\alpha_i y_i K(x,x_i) + b$

Dimensionality Reduction

Curse of Dimensionality

- **Volume:** $V_d(r) = r^d$
- **Ratio:** $ratio = \frac{V_{crust}}{V_{S_1}} = \frac{V_{S_1} - V_{crust}}{V_{S_1}}$
- **Vol Eqn:** $V = \frac{r^D \cdot \pi^{D/2}}{\rho(D/2+1)}$
- **ratio** $= 1 - (1 - \frac{\epsilon}{r})^D$

Feature Selection

- **Embedded:** L1: L1: $\|\mathbf{w}\|_1 = \sum_{j=0}^{M}|w_j|$
- **Wrappers:** Recursive Feature Elimination using Greedy Search
- **Feature Extraction:** PCA, LDA

PCA:

- **Mean:** $\mu = \frac{1}{N}\sum_{i=1}^{N}x_i$
- **Covariance:** $\Sigma = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)(x_i - \mu)^T$
- **Eigendecomposition:** $\Sigma = W\Lambda W^T$
- **Sorting:** $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_M$
- **Projection:** $z = W^T x$
- **Reconstruction:** $x = Wz + \mu$

MDS:

- **Distance Matrix:** $D = \{d_{ij}\}$
- **Gram Matrix:** $G = -\frac{1}{2}HDH$
- **Eigendecomposition:** $G = V\Lambda V^T$
- **Projection:** $Z = V\Lambda^{1/2}$
- **Reconstruction:** $D = \{d_{ij}\}$

0.1 ISOMAP

- **Shortest Path:** $d_{ij} = \min_{p_{ij}}\sum_{k=1}^{L_{ij}-1}\|x_{p_{ij}(k)} - x_{p_{ij}(k+1)}\|$
- **Time Complexity:** $O(N^3)$

LLE

- Finding a set of weights $W \in \Re^{D\times d}$ that minimizes the reconstruction error
- $x_i = \sum_{j=1}^{K} w_{ij}x_{i(j)}$

t-SNE

- **Objective:** $KL(P\|Q) = \sum_i\sum_j p_{ij}\log\frac{p_{ij}}{q_{ij}}$
- **Perplexity:** $\mathrm{Perp}(P_i) = 2^{H(P_i)}$
- **Symmetric SNE:** $p_{ij} = \frac{p_{i|j} + p_{j|i}}{2N}$
- **Gradient:** $\frac{\partial C}{\partial y_i} = 4\sum_j(p_{ij} - q_{ij})(y_i - y_j)(1 + \|y_i - y_j\|^2)^{-1}$
- **Time Complexity:** $O(N^2)$

Clustering

K-Means

1. Centroid Assignment: $u^{(i)} = \arg\min_j \|x^{(i)} - \mu_j\|^2$
2. Requires **Scaling the Data**
3. Convergence if Assignments do not change

Cluster Validity Metrics

Internal Criteria

1. **Silhouette Coefficient:** $s = \frac{1}{N}\sum_{i=1}^{N}\frac{b_i - a_i}{\max(a_i,b_i)}$

External Criteria

1. **Rand Index:**
   (a) **a** is the number of pairs of elements in X that are in the same subset in C and in the same subset in D.
   (b) b is the number of pairs of elements in X that are in different subset in C and in different subset in D.
   (c) c is the number of pairs of elements in X that are in the same subset in C and in different subset in D.
   (d) d is the number of pairs of elements in X that are in different subset in C and in the same subset in D.

DBSCAN

- **Core Point:** $N_\epsilon(x) \geq \mathrm{minPts}$
- **Border Point:** $N_\epsilon(x) < \mathrm{minPts}$, but x is in the ϵ-neighborhood of a core point
- **Noise Point:** Neither core nor border
- **Time Complexity:** $O(N\log N)$

Heirarchical Clustering

Agglomerative

- **Single Linkage:** $d(C_i,C_j) = \min_{x\in C_i, y\in C_j}\|x - y\|$
- **Complete Linkage:** $d(C_i,C_j) = \max_{x\in C_i, y\in C_j}\|x - y\|$
- **Average Linkage:** $d(C_i,C_j) = \frac{1}{|C_i||C_j|}\sum_{x\in C_i}\sum_{y\in C_j}\|x - y\|$

Distance Metrics

1. **Euclidean:** $\|x - y\|_2 = \sqrt{\sum_{i=1}^{D}(x_i - y_i)^2}$
2. **City-Block:** $\|x-y\|_1 = \sum_{i=1}^{D}|x_i - y_i|$
3. **Mahalanobis:** $\|x - y\|_M = \sqrt{(x-y)^T M(x-y)}$ where M is the covariance matrix
4. **Cosine:** $\cos(x,y) = \frac{x^T y}{\|x\|_2\|y\|_2}$

Neural Networks

Activation Functions

1. **Heaviside:** $H(x) = \begin{cases} 0 & x < 0 \\ 1 & x \geq 0 \end{cases}$
2. **Linear:** $f(x) = x$
3. **Sigmoid:** $f(x) = \frac{1}{1+e^{-x}}$
4. **Tanh:** $f(x) = \frac{\exp(x)-\exp(-x)}{\exp(x)+\exp(-x)}$
5. **ReLU:** $f(x) = \max(0,x)$
6. **Leaky ReLU:** $f(x) = \max(\alpha\cdot x, x)$

Neural Networks

Activation Functions

1. **Heaviside:** $H(x) = \begin{cases} 0 & x < 0 \\ 1 & x \geq 0 \end{cases}$
2. **Linear:** $f(x) = x$
3. **Sigmoid:** $f(x) = \frac{1}{1+e^{-x}}$
4. **Tanh:** $f(x) = \frac{\exp(x)-\exp(-x)}{\exp(x)+\exp(-x)}$
5. **ReLU:** $f(x) = \max(0,x)$
6. **Leaky ReLU:** $f(x) = \max(\alpha\cdot x, x)$

7. **Softmax:** $f(x)_i = \frac{\exp(x_i)}{\sum_{j=1}^{K}\exp(x_j)}$
8. **Exponential Linear Unit:** $f(x) = \begin{cases} x & x \geq 0 \\ \alpha(\exp(x) - 1) & x < 0 \end{cases}$
9. **Softplus:** $f(x) = \log(1 + \exp(x))$

Backpropagation

1. **Forward Pass:** $z^{(l)} = W^{(l)}a^{(l-1)} + b^{(l)}$, $a^{(l)} = f^{(l)}(z^{(l)})$
2. **Backward Pass:** $\delta^{(L)} = \nabla_a J \odot f^{(L)\prime}(z^{(L)})$, $\delta^{(l)} = (W^{(l+1)})^T\delta^{(l+1)} \odot f^{(l)\prime}(z^{(l)})$
3. **Weight Update:** $\nabla_{W^{(l)}}J = \delta^{(l)}(a^{(l-1)})^T$, $\nabla_{b^{(l)}}J = \delta^{(l)}$

Optimizers

Gradient Descent

1. **Batch Gradient Descent:** $\theta = \theta - \eta\nabla_\theta J(\theta)$
2. **Stochastic Gradient Descent:** $\theta = \theta - \eta\nabla_\theta J(\theta; x^{(i)}; y^{(i)})$
3. **Mini-Batch Gradient Descent:** $\theta = \theta - \eta\nabla_\theta J(\theta; x^{(i:i+n)}; y^{(i:i+n)})$
4. Where $J(\theta)$ is the loss function, and θ are the parameters

Momentum

1. **Update:** $v = \gamma v + \eta \nabla_\theta J(\theta)$, $\theta = \theta - v$
2. **Nesterov Momentum:** $v = \gamma v + \eta \nabla_\theta J(\theta - \gamma v)$, $\theta = \theta - v$
3. **Adam:** $m = \beta_1 m + (1 - \beta_1) \nabla_\theta J(\theta)$, $v = \beta_2 v + (1 - \beta_2)(\nabla_\theta J(\theta))^2$, $\theta = \theta - \eta \frac{m}{\sqrt{v} + \epsilon}$