SVMS

Soft

$$\begin{aligned} \text{minimize} \quad & \mathcal{T}(\mathbf{w},\xi) = \tfrac{1}{2}\mathbf{w}^\top\mathbf{w} + C\textstyle\sum_{i=1}^n \xi_i \\ \text{subject to} \quad & z_i(\mathbf{w}^\top\mathbf{y}_i + w_0) \geq 1-\xi_i \\ & \xi_i \geq 0 \end{aligned}$$

$$g(\mathbf{x}) = \sum_{i=1}^n \alpha_i z_i \underbrace{K(\mathbf{x}_i,\mathbf{x})}_{\text{replaces } \mathbf{y}_i^\top\mathbf{y}}$$

Non Linear Support Vector Machines

$$\begin{aligned} L(\mathbf{w},w_0,\xi,\alpha,\beta) &= \frac{1}{2}\mathbf{w}^\top\mathbf{w} + C\sum_{i=1}^n \xi_i \\ &- \sum_{i=1}^n \alpha_i\left[z_i(\mathbf{w}^\top\mathbf{y}_i + w_0) - 1 + \xi_i\right] - \sum_{i=1}^n \beta_i\xi_i \end{aligned}$$

$$\begin{aligned} \text{maximize} \quad & W(\alpha) = \textstyle\sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i=1}^n\sum_{j=1}^n z_i z_j \alpha_i\alpha_j \mathbf{y}_i^\top\mathbf{y}_j \\ \text{subject to} \quad & \textstyle\sum_{j=1}^n z_j\alpha_j = 0 \wedge \forall i\ C \geq \alpha_i \geq 0 \end{aligned}$$

Structural SVMs (SSVMs)

Reformulated problem with slacks $\xi_i \geq 0$ (margin rescaling):

$$\min_{\mathbf{w},\xi\geq 0} \ \frac{1}{2}\mathbf{w}^\top\mathbf{w} + C\sum_{i=1}^n \xi_i$$

$$\begin{aligned} \text{s.t.} \quad & \mathbf{w}^\top\Psi(z_i,\mathbf{y}_i) - \mathbf{w}^\top\Psi(z,\mathbf{y}_i) \geq \Delta(z,z_i) - \xi_i \quad \forall \mathbf{y}_i \in \mathcal{Y}, z \neq z_i \\ & \Leftrightarrow \mathbf{w}^\top\Psi(z_i,\mathbf{y}_i) - \max_{z\neq z_i}[\Delta(z,z_i) + \mathbf{w}^\top\Psi(z,\mathbf{y}_i)] \geq -\xi_i \quad \forall \mathbf{y}_i \in \mathcal{Y} \end{aligned}$$

ENSEMBLE METHODS

Bagging

1: **for** $b = 1$ to B **do**
2: draw a bootstrap sample $\mathcal{Z}^* = \{(X_1^*, Y_1^*, \ldots, (X_n^*, Y_n^*)\}$ of size n from the training data $\mathcal{Z}$;

 # generation of a random forest tree:

3: **repeat**
4: select m variables at random from p variables ($m \approx \sqrt{p}$);
5: pick the best variable/split-point among the m selected variables;
6: split the node into two daughter nodes;
7: **until** node size n_{min} is reached
8: **end for**
9: **return** Output the ensemble of trees $\{\hat{c}_b(x)\}_{b=1}^B$

Boosting

Input: data $(x_1,y_1),\ldots,(x_n,y_n)$
Output: classifier $\hat{c}_B(x)$

1: **initialize** observation weights $w_i = 1/n$
2: **for** $b = 1$ to B **do**
3: Fit a classifier $c_b(x)$ to the training data using weights w_i;
4: Compute $\epsilon_b \leftarrow \sum_{i=1}^n w_i^{(b)} \mathbb{I}_{\{c_b(x_i)\neq y_i\}} / \sum_{i=1}^n w_i^{(b)}$;
5: Compute $\alpha_b \leftarrow \log\frac{1-\epsilon_b}{\epsilon_b}$;
6: Set $\forall i\ w_i \leftarrow w_i \exp\left(\alpha_b \mathbb{I}_{\{y_i\neq c_b(x_i)\}}\right)$;
7: **end for**
8: **return** $\hat{c}_B(x) = \operatorname{sgn}\left(\sum_{b=1}^B \alpha_b c_b(x)\right)$

$$\begin{aligned} J(F+\alpha c) &= \mathbb{E}\left\{e^{-y(F(x)+\alpha c(x))}\right\} \\ &\approx \mathbb{E}\left\{e^{-yF(x)}(1 - y\alpha c(x) + \tfrac{1}{2}\alpha^2 \underbrace{y^2 c(x)^2}_{=1}\right\} \\ &= \mathbb{E}\left\{e^{-yF(x)}(1 - y\alpha c(x) + \tfrac{1}{2}\alpha^2\right\} \end{aligned}$$

UNSUPERVISED LEARNING

k-Means Algorithm

Require: INPUT: $\mathcal{X} = \{\mathbf{x}_1,\ldots,\mathbf{x}_n\}$
init $\mu_c = \mathbf{x}_c$ for $1 \leq c \leq k$

1: **repeat**
2: Keep prototypes $\mathcal{Y}$ fixed and assign sample vectors $\mathbf{x}$ to nearest prototype

$$c(\mathbf{x}) \in \underset{c\in\{1,\ldots,k\}}{\operatorname{argmin}} \|\mathbf{x} - \mu_c\|^2$$

3: Keep assignments $c(\mathbf{x})$ fixed and estimate prototypes

$$\mu_\alpha = \frac{1}{n_\alpha}\sum_{\mathbf{x}:c(\mathbf{x})=\alpha} \mathbf{x} \quad \text{with} \quad n_\alpha = \#\{\mathbf{x} : c(\mathbf{x}) = \alpha\}$$

4: **until** changes of $c(\mathbf{x})$, $\mathcal{Y}$ vanish
5: **return** $c(\mathbf{x}), \forall \mathbf{x} \in \mathcal{X}$ and the prototypes $\mathcal{Y}$

Neural Network

$$\frac{\partial L_n}{\partial w_{il}^{L+1}} = \delta_i^{L+1} z_l^L \qquad \delta_i^{L+1} = (\hat{y}_i - y_i)\sigma'\left(\sum_{m=1}^{K(L)} w_{im}^{L+1} z_m^L\right)$$

$$\frac{\partial L_n}{\partial w_{ml}^l} = \delta_m^l z_l^{l-1} \qquad \delta_m^l = \left(\sum_{r=1}^{K(l+1)} \delta_r^{l+1} w_{rm}^{l+1}\right) h'\left(\sum_{r=1}^{K(l-1)} w_{mr}^l z_r^{l-1}\right)$$

$$\frac{\partial L_n}{\partial w_{kj}^1} = \delta_k^1 x_j \qquad \delta_m^1 = \left(\sum_{r=1}^{K(2)} \delta_r^2 w_{rm}^2\right) h'\left(\sum_{j=1}^{J} w_{kj}^1 x_j\right)$$

EM

$$\begin{aligned} \frac{\partial}{\partial\mu_c}\left[Q(\theta;\theta^{(j)}) - \lambda\left(\textstyle\sum_{\tilde{c}=1}^k \pi_{\tilde{c}} - 1\right)\right] &= 0 \\ \frac{\partial}{\partial\mu_c}\sum_{\mathbf{x}\in\mathcal{X}}\sum_{\tilde{c}=1}^k \gamma_{\mathbf{x}\tilde{c}} \log\left(\pi_{\tilde{c}} P(\mathbf{x}|\theta_{\tilde{c}})\right) &= 0 \end{aligned}$$

$P(\mathbf{x}|\theta_c)$ is assumed to be e.g. a (isotropic) Gaussian mixture component!

$$\begin{aligned} \sum_{\mathbf{x}\in\mathcal{X}} \frac{1}{2}\gamma_{\mathbf{x}c}\frac{\partial}{\partial\mu_c}\left(-(\mathbf{x}-\mu_c)^T\Sigma_c^{-1}(\mathbf{x}-\mu_c) - \log 2\pi|\Sigma_c|\right) &= 0 \\ \sum_{\mathbf{x}\in\mathcal{X}} \gamma_{\mathbf{x}c}\Sigma_c^{-1}(\mathbf{x}-\mu_c) &= 0 \end{aligned}$$

$$\boxed{\mu_c = \frac{\sum_{\mathbf{x}\in\mathcal{X}}\gamma_{\mathbf{x}c}\mathbf{x}}{\sum_{\mathbf{x}\in\mathcal{X}}\gamma_{\mathbf{x}c}}}$$

1: **repeat**
2: E-Step

$$\gamma_{\mathbf{x}c} = \frac{P(\mathbf{x}|c,\theta^{(j)})\,P(c|\theta^{(j)})}{P(\mathbf{x}|\theta^{(j)})}$$

3: M-Step

$$\begin{aligned} \mu_c^{(j+1)} &= \frac{\sum_{\mathbf{x}\in\mathcal{X}}\gamma_{\mathbf{x}c}\mathbf{x}}{\sum_{\mathbf{x}\in\mathcal{X}}\gamma_{\mathbf{x}c}} \\ (\sigma_c^2)^{(j+1)} &= \frac{\sum_{\mathbf{x}\in\mathcal{X}}\gamma_{\mathbf{x}c}(\mathbf{x}-\mu_c)^2}{\sum_{\mathbf{x}\in\mathcal{X}}\gamma_{\mathbf{x}c}} \\ \pi_c^{(j+1)} &= \frac{1}{|\mathcal{X}|}\sum_{\mathbf{x}\in\mathcal{X}}\gamma_{\mathbf{x}c} \end{aligned}$$

4: **until** changes small enough

$$\begin{aligned} \mathrm{E}_{\mathbf{M}|\mathbf{x},\tilde{\theta}}[l(\theta)] &= \mathrm{E}_{\mathbf{M}|\mathbf{x},\tilde{\theta}}[l_{\mathbf{M}}(\theta)] - \mathrm{E}_{\mathbf{M}|\mathbf{x},\tilde{\theta}}[\log p(\mathbf{M}|\mathbf{x},\theta)] \\ \Leftrightarrow \quad l(\theta) &= Q(\theta,\tilde{\theta}) - \mathrm{E}_{\mathbf{M}|\mathbf{x},\tilde{\theta}}[\log p(\mathbf{M}|\mathbf{x},\theta)] \end{aligned}$$

ML

- **REGRESSION**
  - Regression Basics
    - Residual Sum of Squares (RSS): $\hat\beta \sim \mathcal{N}(\beta, (\mathbf{X}^T\mathbf{X})^{-1}\sigma^2)$
    - ridge
      - $RSS(\beta) = (\mathbf{y}-\mathbf{X}\beta)^T(\mathbf{y}-\mathbf{X}\beta) + \lambda\beta^T\beta$ — $\hat\beta^{\text{ridge}} = (\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}$
      - $\hat\beta^{\text{ridge}} = \arg\min_\beta \left\{ \sum_{i=1}^n \left(y_i - \beta_0 - \sum_{j=1}^d x_{ij}\beta_j\right)^2 + \lambda\sum_{j=1}^d \beta_j^2 \right\}$ —
        $$\begin{aligned}\mathbf{X}\hat\beta^{\text{ridge}} &= \mathbf{X}(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y} = \mathbf{U}\mathbf{D}(\mathbf{D}^2+\lambda\mathbf{I})^{-1}\mathbf{D}\mathbf{U}^T\mathbf{y},\\ &= \sum_{j=1}^d \mathbf{u}_j \frac{d_j^2}{d_j^2+\lambda}\mathbf{u}_j^T\mathbf{y}\end{aligned}$$
    - bias variance tradeoff
      $$\begin{aligned}&\mathbb{E}_D\mathbb{E}_{X,Y}\left(\hat f(X) - Y\right)^2\\ &= \mathbb{E}_D\mathbb{E}_X\left(\hat f(X) - \mathbb{E}(Y|X)\right)^2 + \mathbb{E}_{X,Y}(Y - \mathbb{E}(Y|X))^2\\ &= \mathbb{E}_X\mathbb{E}_D\left(\hat f(X) - \mathbb{E}_D\hat f(X)\right)^2 && \text{(variance)}\\ &\quad + \mathbb{E}_X\left(\mathbb{E}_D\hat f(X) - \mathbb{E}(Y|X)\right)^2 && \text{(bias)}^2\\ &\quad + \mathbb{E}_{X,Y}(Y - \mathbb{E}(Y|X))^2 && \text{(noise)}\end{aligned}$$
  - Gaussian Processes
    - Gaussian: $\log p(x_i|\theta) = -\frac{1}{2}(x_i-\mu)^T\Sigma^{-1}(x_i-\mu) - \frac{d}{2}\log 2\pi - \frac{1}{2}\log|\Sigma|$
    - Bayesian linear regression
      $$\begin{aligned}p(\beta|\mathbf{X},\mathbf{y},\Lambda) &= \mathcal{N}(\beta|\mu_\beta,\Sigma_\beta)\\ \mu_\beta &= \sigma^2\left(\mathbf{X}^T\mathbf{X}+\sigma^2\Lambda\right)^{-1}\mathbf{X}^T\mathbf{y}\\ \Sigma_\beta &= \sigma^2\left(\mathbf{X}^T\mathbf{X}+\sigma^2\Lambda\right)^{-1}\end{aligned}$$
    - Kernel properties
      - Linear kernel: $k(x,x') = x^T x'$
      - Polynomial kernel: $k(x,x') = (x^T x' + 1)^p$, for $p \in \mathbb{N}$
      - Gaussian (RBF) kernel: $k(x,x') = \exp\left(-\|x-x'\|_2^2/h^2\right)$
      - Sigmoid (tanh) kernel: $k(x,x') = \tanh \kappa x^T x' - b$
    - Prediction
      $$p\left(\begin{bmatrix}\mathbf{y}\\ y_{n+1}\end{bmatrix}\middle| x_{n+1},\mathbf{X},\sigma\right) = \mathcal{N}\left(\begin{bmatrix}\mathbf{y}\\ y_{n+1}\end{bmatrix}\middle| \mathbf{0}, \begin{bmatrix}\mathbf{C}_n & \mathbf{k}\\ \mathbf{k}^T & c\end{bmatrix}\right),$$
      where $\mathbf{C}_n = \mathbf{K} + \sigma^2\mathbf{I}$,
      $c = k(x_{n+1}, x_{n+1}) + \sigma^2$,
      $\mathbf{k} = k(x_{n+1}, \mathbf{X})$,
      and $\mathbf{K} = k(\mathbf{X},\mathbf{X})$.
      $$\begin{aligned}p(y_{n+1}|x_{n+1},\mathbf{X},\mathbf{y}) &= \mathcal{N}\left(y_{n+1}|\mu_{y_{n+1}},\sigma^2_{y_{n+1}}\right)\\ \mu_{y_{n+1}} &= \mathbf{k}^T\mathbf{C}_n^{-1}\mathbf{y}\\ \sigma^2_{y_{n+1}} &= c - \mathbf{k}^T\mathbf{C}_n^{-1}\mathbf{k}\end{aligned}$$
- **DENSITY ESTIMATION**
  - Bayesian Learning: $\sigma_n^2 = \dfrac{\sigma^2\sigma_0^2}{n\sigma_0^2+\sigma^2}$ $\quad \mu_n = \dfrac{n\sigma_0^2}{n\sigma_0^2+\sigma^2}\hat\mu_n + \dfrac{\sigma^2}{n\sigma_0^2+\sigma^2}\mu_0$ — $p(x|\mu) \sim \mathcal{N}(\mu,\sigma^2)$, $p(\mu) \sim \mathcal{N}(\mu_0,\sigma_0^2)$, let $\mathcal{X} = \{x_1,\dots,x_n\}$
  - Nonparametric Density Estimation: $\hat p_n(\mathbf{x}) := \frac{1}{n}\sum_{i=1}^n \frac{1}{V_n}\phi\left(\frac{\mathbf{x}-\mathbf{x}_i}{h_n}\right)$
- **CLASSIFICATION**
  - Bayes Optimal Classifier: $\pi_{\mathcal{O}}p_{\mathcal{O}}(x) \geq \max\left\{(1-d)p(x), \max_z \pi_z p_z(x)\right\}$
  - Design of Linear Discriminant Functions
    - Perceptrons
      - Gradient Descent Procedure: $\boxed{a(k+1) = a(k) - \eta(k)\nabla J(a(k))}$
        - $\frac{\partial}{\partial\eta}J(a(k+1)) = 0 \Rightarrow -\|\nabla J\|^2 + \eta\nabla J^T H\nabla J = 0$
          $\eta^{\text{opt}} = \dfrac{\|\nabla J\|^2}{\nabla J^T H\nabla J}$ optimal learning rate
        - Newton's Rule: $\boxed{a(k+1) = a(k) - H^{-1}\nabla J}$
      - Perceptron Criterion: $\boxed{J_p(a) = \sum_{\tilde x\in\tilde{\mathcal{X}}}(-a^T\tilde x)}$
        $\tilde{\mathcal{X}}$ is the set of misclassified samples.
        - Batch Version with Variable Learning Rate
          - **Require:** initialize $a, \theta, \eta(\cdot)$
          - 1: $k \leftarrow 0$
          - 2: **repeat**
          - 3: $\quad a \leftarrow a + \sum_{\tilde x\in\tilde{\mathcal{X}}}\eta(k)\tilde x$
          - 4: $\quad k \leftarrow k+1$
          - 5: **until** $|\eta(k)\sum_{\tilde x\in\tilde{\mathcal{X}}}\tilde x| < \theta$

          Algorithm
    - LDA: $J(w) = \dfrac{\|\tilde m_1 - \tilde m_2\|^2}{\tilde\Sigma_1+\tilde\Sigma_2} = \dfrac{w^T\overbrace{(m_1-m_2)(m_1-m_2)^T}^{=:\Sigma_B}w}{w^T\Sigma_W w}$ — $\boxed{\hat w = \Sigma_W^{-1}(m_1-m_2)}$
- **NUMERICAL ESTIMATION TECHNIQUES**
  - Cross-Validation
    - Initialization: $n\frac{K-1}{K}$
    - v-th step: $\hat f^{-\nu} \in \arg\min_{f\in\mathcal{F}} \frac{1}{|\mathcal{Z}\setminus\mathcal{Z}_\nu|}\sum_{i\notin\mathcal{Z}_\nu}(y_i - f(x_i))^2$
    - Estimation of the prediction error: $\hat{\mathcal{R}}^{\text{cv}} = \frac{1}{n}\sum_{i\leq n}\left(y_i - \hat f^{-\kappa(i)}(x_i)\right)^2$
    - "Leave-one-out" Method: $\hat f^{-i} \in \arg\min_{f\in\mathcal{F}} \frac{1}{n-1}\sum_{i\leq n: i\neq i}(y_j - f(x_j))^2$
  - Bootstrap: $\hat{\mathcal{R}}^{\text{test}}(S(\mathcal{Z})) = \dfrac{1}{B}\sum_{b=1}^B\sum_{z_i\notin\mathcal{Z}^{*b}}\dfrac{\mathbb{I}_{\{c(x_i)\neq y_i\}}}{n-|\mathcal{Z}^{*b}|}$ —
    $$\begin{aligned}\mathrm{P}\{Z_i\in\mathcal{Z}^{*b}\} &= 1-\left(1-\frac{1}{n}\right)^n\\ &\xrightarrow{n\to\infty} 1-\frac{1}{e} = 0.632.\end{aligned}$$
    - .632 bootstrap: $\hat{\mathcal{R}}^{(.632)} = 0.368\cdot\hat{\mathcal{R}}^{*} + 0.632\cdot\hat{\mathcal{R}}^{(1)}$
  - Jackknife: $\text{bias}^{\text{JK}} := (n-1)(\tilde S_n - \hat S_n)$ with $\tilde S_n = \frac{1}{n}\sum_{i=1}^n \hat S_{n-1}^{(-i)}$ — $\hat S^{\text{JK}} = \hat S_n - \text{bias}^{\text{JK}}$

ML

REGRESSION

Regression Basics

Residual Sum of Squares (RSS)

$$\hat{\beta} \sim \mathcal{N}(\beta, (\mathbf{X}^T\mathbf{X})^{-1}\sigma^2)$$

ridge

$$RSS(\beta) = (\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta) + \lambda\beta^T\beta$$

$$\hat{\beta}^{\text{ridge}} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}$$

$$\hat{\beta}^{\text{ridge}} = \arg\min_{\beta}\left\{\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{d} x_{ij}\beta_j\right)^2 + \lambda\sum_{j=1}^{d}\beta_j^2\right\}$$

$$\begin{aligned}\mathbf{X}\hat{\beta}^{\text{ridge}} &= \mathbf{X}(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y} = \mathbf{U}\mathbf{D}(\mathbf{D}^2 + \lambda\mathbf{I})^{-1}\mathbf{D}\mathbf{U}^T\mathbf{y} \\ &= \sum_{j=1}^{d}\mathbf{u}_j\frac{d_j^2}{d_j^2+\lambda}\mathbf{u}_j^T\mathbf{y}\end{aligned}$$

bias variance tradeoff

$$\begin{aligned}&\mathbb{E}_D\mathbb{E}_{X,Y}\left(\hat{f}(X) - Y\right)^2 \\ &= \mathbb{E}_D\mathbb{E}_X\left(\hat{f}(X) - \mathbb{E}(Y|X)\right)^2 + \mathbb{E}_{X,Y}(Y - \mathbb{E}(Y|X))^2 \\ &= \mathbb{E}_X\mathbb{E}_D\left(\hat{f}(X) - \mathbb{E}_D\hat{f}(X)\right)^2 && \text{(variance)} \\ &+ \mathbb{E}_X\left(\mathbb{E}_D\hat{f}(X) - \mathbb{E}(Y|X)\right)^2 && \text{(bias)}^2 \\ &+ \mathbb{E}_{X,Y}(Y - \mathbb{E}(Y|X))^2 && \text{(noise)}\end{aligned}$$

Gaussian Processes

Gaussian

$$\log p(x_i|\theta) = -\frac{1}{2}(x_i - \mu)^T\Sigma^{-1}(x_i - \mu) - \frac{d}{2}\log 2\pi - \frac{1}{2}\log|\Sigma|$$

Bayesian linear regression

$$\begin{aligned}p(\beta|\mathbf{X},\mathbf{y},\Lambda) &= \mathcal{N}(\beta|\mu_\beta, \Sigma_\beta) \\ \mu_\beta &= \sigma^2\left(\mathbf{X}^T\mathbf{X} + \sigma^2\Lambda\right)^{-1}\mathbf{X}^T\mathbf{y} \\ \Sigma_\beta &= \sigma^2\left(\mathbf{X}^T\mathbf{X} + \sigma^2\Lambda\right)^{-1}\end{aligned}$$

Kernel properties

Linear kernel:	$k(x,x') = x^Tx'$
Polynomial kernel:	$k(x,x') = (x^Tx' + 1)^p$, for $p \in \mathbb{N}$
Gaussian (RBF) kernel:	$k(x,x') = \exp(-\|x - x'\|_2^2/h^2)$
Sigmoid (tanh) kernel:	$k(x,x') = \tanh \kappa x^Tx' - b$

Prediction

$$p\left(\begin{bmatrix}\mathbf{y} \\ y_{n+1}\end{bmatrix}\middle| x_{n+1}, \mathbf{X}, \sigma\right) = \mathcal{N}\left(\begin{bmatrix}\mathbf{y} \\ y_{n+1}\end{bmatrix}\middle| 0, \begin{bmatrix}\mathbf{C}_n & \mathbf{k} \\ \mathbf{k}^T & c\end{bmatrix}\right),$$

where $\mathbf{C}_n = \mathbf{K} + \sigma^2\mathbf{I}$,
$c = k(x_{n+1}, x_{n+1}) + \sigma^2$,
$\mathbf{k} = k(x_{n+1}, \mathbf{X})$,
and $\mathbf{K} = k(\mathbf{X}, \mathbf{X})$.

$$\begin{aligned}p(y_{n+1}|x_{n+1},\mathbf{X},\mathbf{y}) &= \mathcal{N}\left(y_{n+1}|\mu_{y_{n+1}}, \sigma^2_{y_{n+1}}\right) \\ \mu_{y_{n+1}} &= \mathbf{k}^T\mathbf{C}_n^{-1}\mathbf{y} \\ \sigma^2_{y_{n+1}} &= c - \mathbf{k}^T\mathbf{C}_n^{-1}\mathbf{k}\end{aligned}$$

DENSITY ESTIMATION

Bayesian Learning

$$\sigma_n^2 = \frac{\sigma^2\sigma_0^2}{n\sigma_0^2 + \sigma^2} \qquad \mu_n = \frac{n\sigma_0^2}{n\sigma_0^2 + \sigma^2}\hat{\mu}_n + \frac{\sigma^2}{n\sigma_0^2 + \sigma^2}\mu_0$$

$p(x|\mu) \sim \mathcal{N}(\mu, \sigma^2)$, $p(\mu) \sim \mathcal{N}(\mu_0, \sigma_0^2)$, let $\mathcal{X} = \{x_1, \ldots, x_n$

Nonparametric Density Estimation

$$\hat{p}_n(\mathbf{x}) := \frac{1}{n}\sum_{i=1}^{n}\frac{1}{V_n}\phi\left(\frac{\mathbf{x} - \mathbf{x}_i}{h_n}\right)$$

CLASSIFICATION

Bayes Optimal Classifier

$$\pi_O p_O(x) \geq \max\left\{(1 - d)p(x), \max_z \pi_z p_z(x)\right\}$$

Design of Linear Discriminant Functions

Perceptrons

Gradient Descent Procedure

$$\boxed{a(k+1) = a(k) - \eta(k)\nabla J(a(k))}$$

$$\begin{aligned}\frac{\partial}{\partial\eta}J(a(k+1)) &= 0 \quad \Rightarrow \quad -\|\nabla J\|^2 + \eta\nabla J^T H\nabla J = 0 \\ \eta^{\text{opt}} &= \frac{\|\nabla J\|^2}{\nabla J^T H\nabla J} \quad \text{optimal learning rate}\end{aligned}$$

Newton's Rule: $\boxed{a(k+1) = a(k) - H^{-1}\nabla J}$

Perceptron Criterion: $\boxed{J_p(a) = \sum_{\tilde{x}\in\tilde{\mathcal{X}}}(-a^T\tilde{x})}$

$\tilde{\mathcal{X}}$ is the set of misclassified samples.

Batch Version with Variable Learning Rate

Require: initialize $a, \theta, \eta(\cdot)$
1: $k \leftarrow 0$
2: **repeat**
3: $\quad a \leftarrow a + \sum_{\tilde{x}\in\tilde{\mathcal{X}}}\eta(k)\tilde{x}$
4: $\quad k \leftarrow k + 1$
5: **until** $|\eta(k)\sum_{\tilde{x}\in\tilde{\mathcal{X}}}\tilde{x}| < \theta$

Algorithm

LDA

$$J(w) = \frac{\|\tilde{m}_1 - \tilde{m}_2\|^2}{\tilde{\Sigma}_1 + \tilde{\Sigma}_2} = \frac{w^T\overbrace{(m_1 - m_2)(m_1 - m_2)^T}^{=:\Sigma_B}w}{w^T\Sigma_W w}$$

$$\boxed{\hat{w} = \Sigma_W^{-1}(m_1 - m_2)}$$

NUMERICAL ESTIMATION TECHNIQUES

Cross-Validation

Initialization $\quad n\frac{K-1}{K}$

v-th step

$$\hat{f}^{-\nu} \in \arg\min_{f\in\mathcal{F}}\frac{1}{|\mathcal{Z}\setminus\mathcal{Z}_\nu|}\sum_{i\notin\mathcal{Z}_\nu}(y_i - f(x_i))^2$$

Estimation of the prediction error

$$\hat{\mathcal{R}}^{\text{cv}} = \frac{1}{n}\sum_{i\leq n}\left(y_i - \hat{f}^{-\kappa(i)}(x_i)\right)^2$$

"Leave-one-out" Method

$$\hat{f}^{-i} \in \arg\min_{f\in\mathcal{F}}\frac{1}{n-1}\sum_{i\leq n: i\neq i}(y_j - f(x_j))^2$$

Bootstrap

$$\hat{\mathcal{R}}^{\text{test}}(S(\mathcal{Z})) = \frac{1}{B}\sum_{b=1}^{B}\sum_{z_i\notin\mathcal{Z}^{*b}}\frac{\mathbb{I}_{\{c(x_i)\neq y_i\}}}{n - |\mathcal{Z}^{*b}|}$$

$$\mathbf{P}\{Z_i \in \mathcal{Z}^{*b}\} = 1 - \left(1 - \frac{1}{n}\right)^n \xrightarrow{n\to\infty} 1 - \frac{1}{e} = 0.632.$$

.632 bootstrap: $\hat{\mathcal{R}}^{(.632)} = 0.368\cdot\hat{\mathcal{R}}^{*} + 0.632\cdot\hat{\mathcal{R}}^{(1)}$

Jackknife

$$\text{bias}^{\text{JK}} := (n-1)(\tilde{S}_n - \hat{S}_n) \text{ with } \tilde{S}_n = \frac{1}{n}\sum_{i=1}^{n}\hat{S}_{n-1}^{(-i)}$$

$$\hat{S}^{\text{JK}} = \hat{S}_n - \text{bias}^{\text{JK}}$$

SVMS

Soft

$$\begin{aligned} \text{minimize} \quad & \mathcal{T}(\mathbf{w}, \xi) = \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + C\textstyle\sum_{i=1}^n \xi_i \\ \text{subject to} \quad & z_i(\mathbf{w}^T\mathbf{y}_i + w_0) \geq 1 - \xi_i \\ & \xi_i \geq 0 \end{aligned}$$

$$g(\mathbf{x}) = \sum_{i=1}^n \alpha_i z_i \underbrace{K(\mathbf{x}_i, \mathbf{x})}_{\text{replaces } \mathbf{y}_i^T\mathbf{y}}$$

Non Linear Support Vector Machines

$$\begin{aligned} L(\mathbf{w}, w_0, \xi, \alpha, \beta) &= \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^n \xi_i \\ &- \sum_{i=1}^n \alpha_i \left[z_i(\mathbf{w}^T\mathbf{y}_i + w_0) - 1 + \xi_i\right] - \sum_{i=1}^n \beta_i \xi_i \end{aligned}$$

maximize $W(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i=1}^n \sum_{j=1}^n z_i z_j \alpha_i \alpha_j \mathbf{y}_i^T\mathbf{y}_j$

subject to $\sum_{j=1}^n z_j \alpha_j = 0 \wedge \forall i\ C \geq \alpha_i \geq 0$

Structural SVMs (SSVMs)

Reformulated problem with slacks $\xi_i \geq 0$ (margin rescaling):

$$\min_{\mathbf{w}, \xi \geq 0} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^n \xi_i$$

$$\text{s.t.} \quad \mathbf{w}^T\Psi(z_i, \mathbf{y}_i) - \mathbf{w}^T\Psi(z, \mathbf{y}_i) \geq \Delta(z, z_i) - \xi_i \quad \forall \mathbf{y}_i \in \mathcal{Y}, z \neq z_i$$

$$\Leftrightarrow \mathbf{w}^T\Psi(z_i, \mathbf{y}_i) - \max_{z \neq z_i}[\Delta(z, z_i) + \mathbf{w}^T\Psi(z, \mathbf{y}_i)] \geq -\xi_i \quad \forall \mathbf{y}_i \in \mathcal{Y}$$

ENSEMBLE METHODS

Bagging

1: **for** $b = 1$ to B **do**
2: draw a bootstrap sample $\mathcal{Z}^* = \{(X_1^*, Y_1^*), \ldots, (X_n^*, Y_n^*)\}$ of size n from the training data $\mathcal{Z}$;
 # generation of a random forest tree:
3: **repeat**
4: select m variables at random from p variables ($m \approx \sqrt{p}$);
5: pick the best variable/split-point among the m selected variables;
6: split the node into two daughter nodes;
7: **until** node size n_{min} is reached
8: **end for**
9: **return** Output the ensemble of trees $\{\hat{c}_b(x)\}_{b=1}^B$

Boosting

Input: data $(x_1, y_1), \ldots, (x_n, y_n)$
Output: classifier $\hat{c}_B(x)$

1: **initialize** observation weights $w_i = 1/n$
2: **for** $b = 1$ to B **do**
3: Fit a classifier $c_b(x)$ to the training data using weights w_i;
4: Compute $\epsilon_b \leftarrow \sum_{i=1}^n w_i^{(b)} \mathbb{I}_{\{c_b(x_i) \neq y_i\}} / \sum_{i=1}^n w_i^{(b)}$;
5: Compute $\alpha_b \leftarrow \log \frac{1-\epsilon_b}{\epsilon_b}$;
6: Set $\forall i\ w_i \leftarrow w_i \exp\left(\alpha_b \mathbb{I}_{\{y_i \neq c_b(x_i)\}}\right)$;
7: **end for**
8: **return** $\hat{c}_B(x) = \mathrm{sgn}\left(\sum_{b=1}^B \alpha_b c_b(x)\right)$

$$\begin{aligned} J(F + \alpha c) &= \mathbb{E}\left\{e^{-y(F(x) + \alpha c(x))}\right\} \\ &\approx \mathbb{E}\left\{e^{-yF(x)}(1 - y\alpha c(x) + \tfrac{1}{2}\alpha^2 \underbrace{y^2 c(x)^2}_{=1})\right\} \\ &= \mathbb{E}\left\{e^{-yF(x)}(1 - y\alpha c(x) + \tfrac{1}{2}\alpha^2)\right\} \end{aligned}$$

k-Means Algorithm

Require: INPUT: $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$
init $\mu_c = \mathbf{x}_c$ for $1 \leq c \leq k$
1: **repeat**
2: Keep prototypes $\mathcal{Y}$ fixed and assign sample vectors $\mathbf{x}$ to nearest prototype

$$c(\mathbf{x}) \in \underset{c \in \{1, \ldots, k\}}{\mathrm{argmin}} \|\mathbf{x} - \mu_c\|^2$$

3: Keep assignments $c(\mathbf{x})$ fixed and estimate prototypes

$$\mu_\alpha = \frac{1}{n_\alpha} \sum_{\mathbf{x}: c(\mathbf{x}) = \alpha} \mathbf{x} \quad \text{with} \quad n_\alpha = \#\{\mathbf{x} : c(\mathbf{x}) = \alpha\}$$

4: **until** changes of $c(\mathbf{x}), \mathcal{Y}$ vanish
5: **return** $c(\mathbf{x}), \forall \mathbf{x} \in \mathcal{X}$ and the prototypes $\mathcal{Y}$

Neural Network

$$\frac{\partial L_n}{\partial w_{ik}^{L+1}} = \delta_i^{L+1} z_k^L \qquad \delta_i^{L+1} = (\hat{y}_i - y_i)\sigma'\left(\sum_{\alpha=1}^{K(L)} w_{i\alpha}^{L+1} z_\alpha^L\right)$$

$$\frac{\partial L_n}{\partial w_{mk}^l} = \delta_m^l z_k^{l-1} \qquad \delta_m^l = \left(\sum_{r=1}^{K(l+1)} \delta_r^{l+1} w_{rm}^{l+1}\right) h'\left(\sum_{r=1}^{K(l-1)} w_{mr}^l z_r^{l-1}\right)$$

$$\frac{\partial L_n}{\partial w_{kj}^1} = \delta_k^1 x_j \qquad \delta_m^1 = \left(\sum_{r=1}^{K(2)} \delta_r^2 w_{rm}^2\right) h'\left(\sum_{j=1}^J w_{kj}^1 x_j\right)$$

UNSUPERVISED LEARNING

$$\frac{\partial}{\partial \mu_c}\left[Q(\theta; \theta^{(j)}) - \lambda\left(\textstyle\sum_{\bar{c}=1}^k \pi_{\bar{c}} - 1\right)\right] = 0$$

$$\frac{\partial}{\partial \mu_c} \underbrace{\sum_{\mathbf{x} \in \mathcal{X}} \sum_{\bar{c}=1}^k \gamma_{\mathbf{x}\bar{c}} \log\left(\pi_{\bar{c}} P(\mathbf{x}|\theta_{\bar{c}})\right)}_{Q(\theta; \theta^{(j)})} = 0$$

$P(\mathbf{x}|\theta_c)$ is assumed to be e.g. a (isotropic) Gaussian mixture component!

$$\sum_{\mathbf{x} \in \mathcal{X}} \frac{1}{2}\gamma_{\mathbf{x}c} \frac{\partial}{\partial \mu_c}\left(-(\mathbf{x} - \mu_c)^T \Sigma_c^{-1}(\mathbf{x} - \mu_c) - \log 2\pi|\Sigma_c|\right) = 0$$

$$\sum_{\mathbf{x} \in \mathcal{X}} \gamma_{\mathbf{x}c} \Sigma_c^{-1}(\mathbf{x} - \mu_c) = 0$$

$$\boxed{\mu_c = \frac{\sum_{\mathbf{x} \in \mathcal{X}} \gamma_{\mathbf{x}c} \mathbf{x}}{\sum_{\mathbf{x} \in \mathcal{X}} \gamma_{\mathbf{x}c}}}$$

1: **repeat**
2: **E-Step**

$$\gamma_{\mathbf{x}c} = \frac{P(\mathbf{x}|c, \theta^{(j)}) P(c|\theta^{(j)})}{P(\mathbf{x}|\theta^{(j)})}$$

3: **M-Step**

$$\begin{aligned} \mu_c^{(j+1)} &= \frac{\sum_{\mathbf{x} \in \mathcal{X}} \gamma_{\mathbf{x}c} \mathbf{x}}{\sum_{\mathbf{x} \in \mathcal{X}} \gamma_{\mathbf{x}c}} \\ (\sigma_c^2)^{(j+1)} &= \frac{\sum_{\mathbf{x} \in \mathcal{X}} \gamma_{\mathbf{x}c}(\mathbf{x} - \mu_c)^2}{\sum_{\mathbf{x} \in \mathcal{X}} \gamma_{\mathbf{x}c}} \\ \pi_c^{(j+1)} &= \frac{1}{|\mathcal{X}|} \sum_{\mathbf{x} \in \mathcal{X}} \gamma_{\mathbf{x}c} \end{aligned}$$

4: **until** changes small enough

EM

$$\begin{aligned} \mathbb{E}_{\mathbf{M}|\mathbf{x}, \tilde{\theta}}[l(\theta)] &= \mathbb{E}_{\mathbf{M}|\mathbf{x}, \tilde{\theta}}[l_{\mathbf{M}}(\theta)] - \mathbb{E}_{\mathbf{M}|\mathbf{x}, \tilde{\theta}}[\log p(\mathbf{M}|\mathbf{x}, \theta)] \\ \Leftrightarrow \quad l(\theta) &= Q(\theta, \tilde{\theta}) - \mathbb{E}_{\mathbf{M}|\mathbf{x}, \tilde{\theta}}[\log p(\mathbf{M}|\mathbf{x}, \theta)] \end{aligned}$$

矩阵求导：$\frac{\partial a^T\beta}{\partial\beta}=a$　$\frac{\partial \beta^TA\beta}{\partial\beta}=2A\beta$　$\frac{\partial\beta^Ta}{\partial\beta}=a$　$\frac{\partial a^TXb}{\partial X}=ab^T$　$\frac{\partial a^TX^Tb}{\partial X}=ba^T$

ML

REGRESSION

Regression Basics

- Residual Sum of Squares (RSS): $\hat\beta\sim\mathcal{N}(\beta,(\mathbf{X}^T\mathbf{X})^{-1}\sigma^2)$
- ridge
  - $RSS(\beta)=(\mathbf{y}-\mathbf{X}\beta)^T(\mathbf{y}-\mathbf{X}\beta)+\lambda\beta^T\beta$ — $\hat\beta^{ridge}=(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}$
  - $\hat\beta^{ridge}=\arg\min_\beta\left\{\sum_{i=1}^n\left(y_i-\beta_0-\sum_{j=1}^d x_{ij}\beta_j\right)^2+\lambda\sum_{j=1}^d\beta_j^2\right\}$ —
    $$\mathbf{X}\hat\beta^{ridge}=\mathbf{X}(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}=\mathbf{U}\mathbf{D}(\mathbf{D}^2+\lambda\mathbf{I})^{-1}\mathbf{D}\mathbf{U}^T\mathbf{y}=\sum_{j=1}^d\mathbf{u}_j\frac{d_j^2}{d_j^2+\lambda}\mathbf{u}_j^T\mathbf{y}$$
- bias variance tradeoff
  $$\begin{aligned}&\mathbb{E}_D\mathbb{E}_{X,Y}\left(\hat f(X)-Y\right)^2\\&=\mathbb{E}_D\mathbb{E}_X\left(\hat f(X)-\mathbb{E}(Y|X)\right)^2+\mathbb{E}_{X,Y}(Y-\mathbb{E}(Y|X))^2\\&=\mathbb{E}_X\mathbb{E}_D\left(\hat f(X)-\mathbb{E}_D\hat f(X)\right)^2 &&\text{(variance)}\\&+\mathbb{E}_X\left(\mathbb{E}_D\hat f(X)-\mathbb{E}(Y|X)\right)^2 &&\text{(bias)}^2\\&+\mathbb{E}_{X,Y}(Y-\mathbb{E}(Y|X))^2 &&\text{(noise)}\end{aligned}$$

Gaussian Processes

- Gaussian: $\log p(x_i|\theta)=-\frac{1}{2}(x_i-\mu)^T\Sigma^{-1}(x_i-\mu)-\frac{d}{2}\log 2\pi-\frac{1}{2}\log|\Sigma|$
- Bayesian linear regression
  $$\begin{aligned}p(\beta|\mathbf{X},\mathbf{y},\Lambda)&=\mathcal{N}(\beta|\mu_\beta,\Sigma_\beta)\\\mu_\beta&=\sigma^2\left(\mathbf{X}^T\mathbf{X}+\sigma^2\Lambda\right)^{-1}\mathbf{X}^T\mathbf{y}\\\Sigma_\beta&=\sigma^2\left(\mathbf{X}^T\mathbf{X}+\sigma^2\Lambda\right)^{-1}\end{aligned}$$
- Kernel properties
  - Linear kernel: $k(x,x')=x^Tx'$
  - Polynomial kernel: $k(x,x')=(x^Tx'+1)^p$, for $p\in\mathbb{N}$
  - Gaussian (RBF) kernel: $k(x,x')=\exp\left(-\|x-x'\|_2^2/h^2\right)$
  - Sigmoid (tanh) kernel: $k(x,x')=\tanh\kappa x^Tx'-b$
- Prediction
  $$p\left(\begin{bmatrix}\mathbf{y}\\y_{n+1}\end{bmatrix}\Big|x_{n+1},\mathbf{X},\sigma\right)=\mathcal{N}\left(\begin{bmatrix}\mathbf{y}\\y_{n+1}\end{bmatrix}\Big|\mathbf{0},\begin{bmatrix}\mathbf{C}_n&\mathbf{k}\\\mathbf{k}^T&c\end{bmatrix}\right),$$
  where $\mathbf{C}_n=\mathbf{K}+\sigma^2\mathbf{I}$, $c=k(x_{n+1},x_{n+1})+\sigma^2$, $\mathbf{k}=k(x_{n+1},\mathbf{X})$, and $\mathbf{K}=k(\mathbf{X},\mathbf{X})$.
  $$\begin{aligned}p(y_{n+1}|x_{n+1},\mathbf{X},\mathbf{y})&=\mathcal{N}\left(y_{n+1}|\mu_{y_{n+1}},\sigma^2_{y_{n+1}}\right)\\\mu_{y_{n+1}}&=\mathbf{k}^T\mathbf{C}_n^{-1}\mathbf{y}\\\sigma^2_{y_{n+1}}&=c-\mathbf{k}^T\mathbf{C}_n^{-1}\mathbf{k}\end{aligned}$$

DENSITY ESTIMATION

- Bayesian Learning: $\sigma_n^2=\frac{\sigma^2\sigma_0^2}{n\sigma_0^2+\sigma^2}$　$\mu_n=\frac{n\sigma_0^2}{n\sigma_0^2+\sigma^2}\hat\mu_n+\frac{\sigma^2}{n\sigma_0^2+\sigma^2}\mu_0$ — $p(x|\mu)\sim\mathcal{N}(\mu,\sigma^2),p(\mu)\sim\mathcal{N}(\mu_0,\sigma_0^2)$, let $\mathcal{X}=\{x_1,\ldots,x_n\}$
- Nonparametric Density Estimation: $\hat p_n(\mathbf{x}):=\frac{1}{n}\sum_{i=1}^n\frac{1}{V_n}\phi\left(\frac{\mathbf{x}-\mathbf{x}_i}{h_n}\right)$

CLASSIFICATION

- Bayes Optimal Classifier: $\pi_\mathcal{O}p_\mathcal{O}(x)\geq\max\left\{(1-d)p(x),\max_z\pi_zp_z(x)\right\}$
- Design of Linear Discriminant Functions
  - Perceptrons
    - Gradient Descent Procedure: $\boxed{a(k+1)=a(k)-\eta(k)\nabla J(a(k))}$
      - $\frac{\partial}{\partial\eta}J(a(k+1))=0\Rightarrow-\|\nabla J\|^2+\eta\nabla J^TH\nabla J=0$
      - $\eta^{opt}=\frac{\|\nabla J\|^2}{\nabla J^TH\nabla J}$ optimal learning rate
      - Newton's Rule: $\boxed{a(k+1)=a(k)-H^{-1}\nabla J}$
    - Perceptron Criterion: $\boxed{J_p(a)=\sum_{\tilde x\in\tilde{\mathcal{X}}}(-a^T\tilde x)}$ — $\tilde{\mathcal{X}}$ is the set of misclassified samples.
      - Batch Version with Variable Learning Rate
        - **Require:** initialize $a,\theta,\eta(\cdot)$
        - 1: $k\leftarrow 0$
        - 2: **repeat**
        - 3: $a\leftarrow a+\sum_{\tilde x\in\tilde{\mathcal{X}}}\eta(k)\tilde x$
        - 4: $k\leftarrow k+1$
        - 5: **until** $|\eta(k)\sum_{\tilde x\in\tilde{\mathcal{X}}}\tilde x|<\theta$
      - Algorithm
  - LDA: $J(w)=\frac{\|\tilde m_1-\tilde m_2\|^2}{\tilde\Sigma_1+\tilde\Sigma_2}=\frac{w^T\overbrace{(m_1-m_2)(m_1-m_2)^T}^{=:\Sigma_B}w}{w^T\Sigma_Ww}$ — $\boxed{\hat w=\Sigma_W^{-1}(m_1-m_2)}$

NUMERICAL ESTIMATION TECHNIQUES

- Cross-Validation
  - Initialization: $n\frac{K-1}{K}$
  - v-th step: $\hat f^{-\nu}\in\arg\min_{f\in\mathcal{F}}\frac{1}{|\mathcal{Z}\setminus\mathcal{Z}_\nu|}\sum_{i\notin\mathcal{Z}_\nu}(y_i-f(x_i))^2$
  - Estimation of the prediction error: $\hat{\mathcal{R}}^{cv}=\frac{1}{n}\sum_{i\leq n}\left(y_i-\hat f^{-\kappa(i)}(x_i)\right)^2$
  - "Leave-one-out" Method: $\hat f^{-i}\in\arg\min_{f\in\mathcal{F}}\frac{1}{n-1}\sum_{j\leq n:i\neq j}(y_j-f(x_j))^2$
- Bootstrap
  - $\hat{\mathcal{R}}^{test}(\mathcal{S}(\mathcal{Z}))=\frac{1}{B}\sum_{b=1}^B\sum_{z_i\notin\mathcal{Z}^{*b}}\frac{\mathbb{I}_{\{c(x_i)\neq y_i\}}}{n-|\mathcal{Z}^{*b}|}$ — $P\{Z_i\in\mathcal{Z}^{*b}\}=1-\left(1-\frac{1}{n}\right)^n\xrightarrow{n\to\infty}1-\frac{1}{e}=0.632.$
  - .632 bootstrap: $\hat{\mathcal{R}}^{(.632)}=0.368\cdot\hat{\mathcal{R}}^{*}+0.632\cdot\hat{\mathcal{R}}^{(1)}$
- Jackknife: $\text{bias}^{JK}:=(n-1)(\tilde S_n-\hat S_n)$ with $\tilde S_n=\frac{1}{n}\sum_{i=1}^n\hat S_{n-1}^{(-i)}$ — $\hat S^{JK}=\hat S_n-\text{bias}^{JK}$

SVMS

Soft

minimize $\quad \mathcal{T}(\mathbf{w},\xi) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{n}\xi_i$

subject to $\quad z_i(\mathbf{w}^T\mathbf{y}_i + w_0) \geq 1 - \xi_i$

$\quad \xi_i \geq 0$

$$g(\mathbf{x}) = \sum_{i=1}^{n}\alpha_i z_i \underbrace{K(\mathbf{x}_i,\mathbf{x})}_{\text{replaces } \mathbf{y}_i^T\mathbf{y}}$$

Non Linear Support Vector Machines

$$\begin{aligned} L(\mathbf{w}, w_0, \xi, \alpha, \beta) &= \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{n}\xi_i \\ &- \sum_{i=1}^{n}\alpha_i\left[z_i(\mathbf{w}^T\mathbf{y}_i + w_0) - 1 + \xi_i\right] - \sum_{i=1}^{n}\beta_i\xi_i \end{aligned}$$

maximize $\quad W(\alpha) = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} z_i z_j \alpha_i \alpha_j \mathbf{y}_i^T\mathbf{y}_j$

subject to $\quad \sum_{j=1}^{n} z_j\alpha_j = 0 \wedge \forall i\ C \geq \alpha_i \geq 0$

Structural SVMs (SSVMs)

Reformulated problem with slacks $\xi_i \geq 0$ (margin rescaling):

$$\min_{\mathbf{w},\xi\geq 0} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{n}\xi_i$$

s.t. $\mathbf{w}^T\Psi(z_i,\mathbf{y}_i) - \mathbf{w}^T\Psi(z,\mathbf{y}_i) \geq \Delta(z,z_i) - \xi_i \quad \forall \mathbf{y}_i \in \mathcal{Y}, z \neq z_i$

$\Leftrightarrow \mathbf{w}^T\Psi(z_i,\mathbf{y}_i) - \max_{z\neq z_i}[\Delta(z,z_i) + \mathbf{w}^T\Psi(z,\mathbf{y}_i)] \geq -\xi_i \quad \forall \mathbf{y}_i \in \mathcal{Y}$

ENSEMBLE METHODS

Bagging

for $b = 1$ to B **do**
 draw a bootstrap sample $\mathcal{Z}^* = \{(X_1^*, Y_1^*, \ldots, (X_n^*, Y_n^*)\}$ of size n from the training data $\mathcal{Z}$;
 // generation of a random forest tree:
 repeat
  select m variables at random from p variables ($m \approx \sqrt{p}$);
  pick the best variable/split-point among the m selected variables;
  split the node into two daughter nodes;
 until node size n_{min} is reached
end for
return Output the ensemble of trees $\{\hat{c}_b(x)\}_{b=1}^{B}$

Boosting

Input: data $(x_1,y_1),\ldots,(x_n,y_n)$
Output: classifier $\hat{c}_B(x)$

initialize observation weights $w_i = 1/n$
for $b = 1$ to B **do**
 Fit a classifier $c_b(x)$ to the training data using weights w_i;
 Compute $\epsilon_b \leftarrow \sum_{i=1}^{n} w_i^{(b)}\mathbb{I}_{\{c_b(x_i)\neq y_i\}} / \sum_{i=1}^{n} w_i^{(b)}$;
 Compute $\alpha_b \leftarrow \log\frac{1-\epsilon_b}{\epsilon_b}$;
 Set $\forall i\ w_i \leftarrow w_i \exp\left(\alpha_b \mathbb{I}_{\{y_i\neq c_b(x_i)\}}\right)$;
end for
return $\hat{c}_B(x) = \operatorname{sgn}\left(\sum_{b=1}^{B}\alpha_b c_b(x)\right)$

$$\begin{aligned} J(F+\alpha c) &= \mathbb{E}\left\{e^{-y(F(x)+\alpha c(x))}\right\} \\ &\approx \mathbb{E}\left\{e^{-yF(x)}(1 - y\alpha c(x) + \tfrac{1}{2}\alpha^2\underbrace{y^2 c(x)^2}_{=1})\right\} \\ &= \mathbb{E}\left\{e^{-yF(x)}(1 - y\alpha c(x) + \tfrac{1}{2}\alpha^2)\right\} \end{aligned}$$

UNSUPERVISED LEARNING

k-Means Algorithm

Require: INPUT: $\mathcal{X} = \{\mathbf{x}_1,\ldots,\mathbf{x}_n\}$
 init $\mu_c = \mathbf{x}_c$ for $1 \leq c \leq k$
repeat
 Keep prototypes $\mathcal{Y}$ fixed and assign sample vectors $\mathbf{x}$ to nearest prototype

$$c(\mathbf{x}) \in \operatorname*{argmin}_{c\in\{1,\ldots,k\}} \|\mathbf{x} - \mu_c\|^2$$

 Keep assignments $c(\mathbf{x})$ fixed and estimate prototypes

$$\mu_\alpha = \frac{1}{n_\alpha}\sum_{\mathbf{x}:c(\mathbf{x})=\alpha}\mathbf{x} \quad \text{with} \quad n_\alpha = \#\{\mathbf{x} : c(\mathbf{x}) = \alpha\}$$

until changes of $c(\mathbf{x}), \mathcal{Y}$ vanish
return $c(\mathbf{x}), \forall \mathbf{x} \in \mathcal{X}$ and the prototypes $\mathcal{Y}$

Neural Network

$$\frac{\partial L_n}{\partial w_{il}^{L+1}} = \delta_i^{L+1} z_l^L \qquad \delta_i^{L+1} = (\hat{y}_i - y_i)\sigma'\left(\sum_{m=1}^{K(L)} w_{im}^{L+1} z_m^L\right)$$

$$\frac{\partial L_n}{\partial w_{ml}^{l}} = \delta_m^l z_l^{l-1} \qquad \delta_m^l = \left(\sum_{i=1}^{K(l+1)} \delta_i^{l+1} w_{im}^{l+1}\right)h'\left(\sum_{r=1}^{K(l-1)} w_{mr}^l z_r^{l-1}\right)$$

$$\frac{\partial L_n}{\partial w_{kj}^{1}} = \delta_k^1 x_j \qquad \delta_m^1 = \left(\sum_{i=1}^{K(2)} \delta_i^2 w_{im}^2\right)h'\left(\sum_{j=1}^{J} w_{kj}^1 x_j\right)$$

EM

$$\frac{\partial}{\partial\mu_c}\left[Q(\theta;\theta^{(j)}) - \lambda\left(\textstyle\sum_{\hat{c}=1}^{k}\pi_{\hat{c}} - 1\right)\right] = 0$$

$$\frac{\partial}{\partial\mu_c}\sum_{\mathbf{x}\in\mathcal{X}}\sum_{\hat{c}=1}^{k}\gamma_{\mathbf{x}\hat{c}}\log\left(\pi_{\hat{c}}P(\mathbf{x}|\theta_{\hat{c}})\right) = 0$$

$P(\mathbf{x}|\theta_c)$ is assumed to be e.g. a (isotropic) Gaussian mixture component!

$$\sum_{\mathbf{x}\in\mathcal{X}}\frac{1}{2}\gamma_{\mathbf{x}c}\frac{\partial}{\partial\mu_c}\left(-(\mathbf{x}-\mu_c)^T\Sigma_c^{-1}(\mathbf{x}-\mu_c) - \log 2\pi|\Sigma_c|\right) = 0$$

$$\sum_{\mathbf{x}\in\mathcal{X}}\gamma_{\mathbf{x}c}\Sigma_c^{-1}(\mathbf{x}-\mu_c) = 0$$

$$\boxed{\mu_c = \frac{\sum_{\mathbf{x}\in\mathcal{X}}\gamma_{\mathbf{x}c}\mathbf{x}}{\sum_{\mathbf{x}\in\mathcal{X}}\gamma_{\mathbf{x}c}}}$$

repeat
 E-Step

$$\gamma_{\mathbf{x}c} = \frac{P\left(\mathbf{x}|c,\theta^{(j)}\right)P\left(c|\theta^{(j)}\right)}{P\left(\mathbf{x}|\theta^{(j)}\right)}$$

 M-Step

$$\mu_c^{(j+1)} = \frac{\sum_{\mathbf{x}\in\mathcal{X}}\gamma_{\mathbf{x}c}\mathbf{x}}{\sum_{\mathbf{x}\in\mathcal{X}}\gamma_{\mathbf{x}c}}$$

$$(\sigma_c^2)^{(j+1)} = \frac{\sum_{\mathbf{x}\in\mathcal{X}}\gamma_{\mathbf{x}c}(\mathbf{x}-\mu_c)^2}{\sum_{\mathbf{x}\in\mathcal{X}}\gamma_{\mathbf{x}c}}$$

$$\pi_c^{(j+1)} = \frac{1}{|\mathcal{X}|}\sum_{\mathbf{x}\in\mathcal{X}}\gamma_{\mathbf{x}c}$$

until changes small enough

$$\begin{aligned} \mathbb{E}_{\mathbf{M}|\mathbf{x},\hat{\theta}}[l(\theta)] &= \mathbb{E}_{\mathbf{M}|\mathbf{x},\hat{\theta}}[l_{\mathbf{M}}(\theta)] - \mathbb{E}_{\mathbf{M}|\mathbf{x},\hat{\theta}}[\log p(\mathbf{M}|\mathbf{x},\theta)] \\ \Leftrightarrow \quad l(\theta) &= Q(\theta,\hat{\theta}) - \mathbb{E}_{\mathbf{M}|\mathbf{x},\hat{\theta}}[\log p(\mathbf{M}|\mathbf{x},\theta)] \end{aligned}$$