Probabilities

Expectation

$\mathbb{E}[X] = \int_\Omega x f(x)dx = \int_\omega xP[X{=}x]dx$

$\mathbb{E}_{Y|X}[Y] = \mathbb{E}_Y[Y|X]$

$\mathbb{E}_{X,Y}[f(X,Y)] = \mathbb{E}_X\mathbb{E}_{Y|X}[f(X,Y)|X]$

$\mathbb{E}_{Y|X}[f(X,Y)|X] = \int_\mathbb{R} f(X,y)p_{Y|X}(y)dy$

Variance & Covariance

$\text{Var}(X) = \mathbb{E}[(X-\mathbb{E}[X])^2] = \mathbb{E}[X^2] - \mathbb{E}[X]^2$

$\text{Var}[aX \pm bY] = a^2\text{Var}[X] + b^2\text{Var}[Y] \pm 2ab\text{Cov}[X,Y] \quad XY iid$

$\text{Cov}(X,Y) = \mathbb{E}[(X-\mathbb{E}[X])(Y-\mathbb{E}[Y])]$

Conditional Probabilities

$P[X|Y] = \frac{P[X,Y]}{P[Y]}$, $P[\overline{X}|Y] = 1 - P[X|Y]$

Distributions

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

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

$\text{Exp}(x|\lambda) = \lambda e^{-\lambda x}$, $\text{Ber}(x|\theta) = \theta^x(1-\theta)^{(1-x)}$

Sigmoid: $\sigma(x) = 1/(1+\exp(-\mathrm{x})))$

Chebyshev & Consistency

$P(|X - \mathbb{E}[X]| \geq \epsilon) \leq \frac{Var(X)}{\epsilon^2}$

$\lim n \to \infty P(|\hat{\mu} - \mu| > \epsilon) = 0$

Cramer Rao lower bound

$\text{Var}[\hat{\theta}] \geq \mathcal{I}_n(\theta)$

$\mathcal{I}_n(\theta) = -\mathbb{E}[\frac{\partial^2 \log[\mathcal{X}_n|\theta]}{\partial\theta^2}]$ $\quad\hat{\theta}$ unbiased

Efficiency of $\hat{\theta}$: $e(\theta_n) = \frac{1}{\text{Var}[\hat{\theta}_n]\mathcal{I}_\backslash(\theta)}$

$e(\theta_n) = 1$ (efficient)

$lim_{n\to\infty} e(\theta_n) = 1$ (asymp. efficient)

Matrix Derivations

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

$\frac{\partial \mathbf{x}^T\mathbf{A}\mathbf{x}}{\partial \mathbf{x}} = (\mathbf{A}+\mathbf{A}^T)\mathbf{x}$

$\frac{\partial}{\partial \mathbf{x}}\mathbf{f}^T\mathbf{g} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}}\mathbf{g} + \mathbf{g}^T\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)^T$

$\mathbf{X}^T\mathbf{X}$: only invertible if none of the Eigenvalue is 0. Inversion instable if ratio from $\mathbf{X}$'s smallest EV to the largest is big.

Optimization

Gradient Descent

$\theta^{new} \leftarrow \theta^{old} - \eta\nabla_\theta\mathcal{L}$

Convergence isn't guaranteed.

Less zigzag by adding momentum:

$\theta^{(l+1)} \leftarrow \theta^{(l)} - \eta\nabla_\theta\mathcal{L} + \mu(\theta^l - \theta^{(l-1)})$

Newton's Method

Use 2nd order derivation. (Hessian)

$\theta^{new} \leftarrow \theta^{old} - \eta(\nabla_\theta\mathcal{L}/\nabla_\theta^2\mathcal{L})$

$H = \nabla_\theta^2\mathcal{L}$ has to be p.d (convex func).

Risks and Losses

Expected Risk

Conditional Expected Risk

$R(f,X) = \int_\mathbb{R} \mathcal{L}(Y, f(X))P(Y|X)dY$

Total Expected Risk $R(f) =$

$= \mathbb{E}_X[R(f,X)] = \int_\mathcal{X} R(f,X)P(X)dX =$

$\int_\mathcal{X}\int_\mathbb{R} \mathcal{L}(Y,f(X))P(X,Y)dXdY$

Empirical Risk

$Z^{train} = (X_1,Y_1),...,(X_n,Y_n)$

$Z^{test} = (X_{n+1},Y_{n+1}),...,(X_{n+m},Y_{n+m})$

Empirical Risk Minimizer $\hat{f}$ s.t.

$\hat{f} \in \arg\min_{f\in\mathcal{C}} \hat{R}(\hat{f}, Z^{train})$

Training error:

$\hat{R}(\hat{f}, Z^{train}) = \frac{1}{n}\sum_{i=1}^n Q(Y_i, \hat{f}(X_i))$

Test error:

$\hat{R}(\hat{f}, Z^{test}) = \frac{1}{m}\sum_{i=n+1}^{n+m} Q(Y_i, \hat{f}(X_i))$

$\hat{R}(\hat{f}, Z^{test}) \neq \mathbb{E}_X[R(f,X)]$

Linear Regression

Data: $Z = (x_i, y_i) \in \mathbb{R}^3 \times \mathbb{R} : 1 \leq i \leq n$

X are iids and Y depends on X.

Model: $\mathbf{Y} = \beta_0 + \sum_{j=1}^d \mathbf{X_j}\beta_j \quad \mathbf{Y} \subset \mathbb{R}$

Introduce $X_0 = 1$ and rewrite

$\mathbf{Y} = \mathbf{X}^T\beta \quad \mathbf{X} \in \mathbb{R}^{(d+1)\times n}, \beta \in \mathbb{R}^{d+1}$

additive Gaussian noise $\epsilon \sim \mathcal{N}(0,\sigma^2)$

$\hat{y} = \mathbf{X}\hat{\beta} + \epsilon$

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

$p(Y|X,\beta,\sigma) \sim \mathcal{N}(Y|X^T\beta, \sigma^2)$

A Regression has Optimum:

$f^*(x) = \mathbb{E}_Y[Y|X=x]$

Linear Regression

Setting: Minimize RSS.

$\mathcal{L} = RSS(\beta) = \sum_{i=1}^n (y_i - x_i^T\beta)^2 =$

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

$X \in \mathbb{R}^{n\times(d+1)}, y \in \mathbb{R}^n, \beta \in \mathbb{R}^{d+1}$

Solution: differentiate w.r.t β

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

Is an orth. projection with lowest variance of all unbiased estimates.

Prediction: $\hat{y} = \mathbf{X}\hat{\beta} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$

Ridge Regression (L2 penalty)

Setting: Penalize the βs

$\mathcal{L} = \sum_{i=1}^n (y_i - x_i^T\beta)^2 + \lambda\sum_{j=1}^d \beta_j^2 =$

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

Solution: differentiate w.r.t β

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

Lasso (L1 penalty)

Setting: seek for a sparse solution

$\mathcal{L} = \sum_{i=1}^n (y_i - x_i^T\beta)^2 + \lambda\sum_{j=1}^d |\beta_j|$

$= (\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta) + \lambda||\beta||_1$

Lasso has no closed form.

Bayesian Linear Regression

Setting: Define a prior over the βs.

e.g. Ridge:

Assume βs distributed with mean 0

$p(\beta|\Lambda) = \mathcal{N}(\beta|\mathbf{0}, \Lambda^{-1}) \propto \exp(-\frac{1}{2}\beta^T\Lambda\beta)$

e.g. Linear Regression:

equivalent to ridge with $\Lambda = \lambda\mathbb{I}$, $\sigma = 1$

Posterior

given observed $\mathbf{X}, \mathbf{y}$, use Baye's theorem to find the posterior

$p(\beta|\mathbf{X}, \mathbf{y}, \Lambda, \sigma) = \mathcal{N}(\mu_\beta, \Sigma_\beta)$

$\mu_\beta = \sigma^2(\mathbf{X}^T\mathbf{X} + \sigma^2\Lambda)^{-1}(X)^T\mathbf{y}$ $\quad\Sigma_\beta = \sigma^2(\mathbf{X}^T\mathbf{X} + \sigma^2\Lambda)^{-1}$

Bayesian Information Criterion (BIC)

$-2log(\hat{p}(X|\hat{\theta}_k, M_k)) + k'logn$ tendency to underfit

Akaike Information Criterion (AIC)

$-2log(\hat{p}(X|\hat{\theta}_k)) + 2k'$, $k' = dim(\theta)$ tendency to select large models (overfit)

Takeuchi Information Criterion (TIC)

$-2log(\hat{p}(X|\hat{\theta}_k)) + 2trace[I_1(\theta_k)J_1^{-1}(\theta_k)]$

reduced to AIC if the true model is an element of the model class.

Nonlinear Regression

Idea: Feature space transformation

Model: $\mathbf{Y} = f(\mathbf{X}) = \sum_{m=1}^M \beta_m h_m(\mathbf{X})$

Transformation $h_m(\mathbf{X}) : \mathbb{R}^d \to \mathbb{R}$

Cubic Spline

e.g. for d=1 with knots at ξ_1 and ξ_2

$h_1(X) = 1 \quad h_3(X) = X^2 \quad h_5(X) = (X-\xi_1)_+^3$

$h_2(X) = X \quad h_4(X) = X^3 \quad h_6(X) = (X-\xi_2)_+^3$

Wavelets

Functions that measure local properties of the underlying data. Keep the most important ones and get rid of the noise.

Gaussian Process Regression

joint Gaussian over all outputs

$\mathbf{y} = \mathbb{X}\beta + \epsilon \quad \epsilon \sim \mathcal{N}(\epsilon|0, \sigma\mathbb{I}_n)$

We can rewrite the distribution

$P(\begin{bmatrix}\mathbf{y} \\ y_*\end{bmatrix}) = \mathcal{N}(\mathbf{y}|\mathbf{0}, \begin{bmatrix}\mathbf{C_n} & \mathbf{k} \\ \mathbf{k^T} & c\end{bmatrix})$

Such that for prediction:

$p(y_*|\mathbf{x}_*, \mathbf{X}, \mathbf{y}) = \mathcal{N}(y_*|\mu_*, \sigma_*^2)$

$\mu_{y_*} = \mathbf{k}^T\mathbf{C}_n^{-1}\mathbf{y} \quad \mathbf{C}_n = \mathbf{K} + \sigma^2\mathbb{I}$

$\sigma_*^2 = c - \mathbf{k}^T\mathbf{C}_n^{-1}\mathbf{k} \quad c = k(x_*, x_*) + \sigma^2$

$\mathbf{k} = k(x_*, \mathbf{X})$

k is the kernel function.

lengthscale in kernel: how far can we reliably extrapolate

Bias-Variance tradeoff

$\text{Bias}(\hat{f}) = \mathbb{E}[\hat{f}] - f^*$

$\text{Var}(\hat{f}) = \mathbb{E}[(\hat{f} - \mathbb{E}[\hat{f}])^2]$

$|\mathcal{Z}| \downarrow \quad |\mathcal{F}| \uparrow \quad \Rightarrow \quad \text{Var} \uparrow \quad \text{Bias} \downarrow$

$|\mathcal{Z}| \uparrow \quad |\mathcal{F}| \downarrow \quad \Rightarrow \quad \text{Var} \downarrow \quad \text{Bias} \uparrow$

Squared Error Decomposition

$\mathbb{E}_D\mathbb{E}_{X,Y}[(\hat{f}(X) - Y)^2] =$

$\mathbb{E}_{X,Y}[(\mathbb{E}_Y[Y|X] - Y)^2]$ (noise)

$+\mathbb{E}_X\mathbb{E}_D[(\hat{f}_D(X) - \mathbb{E}_D[\hat{f}(X)])^2]$ (var.)

$+\mathbb{E}_X[(\mathbb{E}_D[\hat{f}_D(X)] - \mathbb{E}_Y[Y|X])^2]$ (bias^2)

(can be derivated by vanishing of the crossproducts)

Parametric Density Estimation

Find the most likely parameter of a distribution.

Maximum Likelihood

Likelihood: $P(\mathcal{X}|\theta) = \prod_{i\leq n} p(x_i|\theta)$

Find: $\hat{\theta} \in \arg\max_\theta P(\mathcal{X}|\theta)$

Procedure: solve $\nabla_\theta logP(\mathcal{X}|\theta) = 0$

Efficient & easy to calculate.

Consistent. Converge to best model θ_0 Warning: Overfitting!

Maximum A Posteriori

Assume Knowledge of a prior $P(\theta)$

Find: $\hat{\theta} \in \arg\max_\theta P(\theta|\mathcal{X}) =$

$= \arg\max_\theta P(\mathcal{X}|\theta)P(\theta)$

Solve $\nabla_\theta logP(\mathcal{X}|\theta)P(\theta) = 0$

Bayesian Learning

Prior Knowledge of $p(\theta)$

Find Posterior Density: $p(\theta|\mathcal{X})$

Can be done using Baye's Rules

We can use this Recursively:

$\mathcal{X}^n = \{x_1, \cdots, x_n\}$

$p(\theta|\mathcal{X}^n) = \frac{p(x_n|\theta)p(\theta|\mathcal{X}^{n-1})}{\int p(x_n|\theta)p(\theta|\mathcal{X}^{n-1}d\theta}$ with $p(\theta|\mathcal{X}^0)p(\theta)$

Difficult & needs prior knowledge but better against overfitting.

Numerical Est. Techinques

Setting: Estimate $\hat{f}(x) \in \mathcal{F}$ with minimal prediction error.

K-Fold Cross Validation

Initialisation (split training set):

$\mathcal{Z} = \mathcal{Z}_1 \bigcup \mathcal{Z}_2 \bigcup \cdots \bigcup \mathcal{Z}_K, \mathcal{Z}_\mu \bigcap \mathcal{Z}_\nu = \emptyset$

with map $\kappa : \{1, \cdots, n\} \to \{, \cdots, K\}$

$|\mathcal{Z}_k| \approx n\frac{K-1}{K}$

Learning:

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

Validation:

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

tendance to Underfit

Leave-one-out: $K = n$ (unbiased but Var can be large $\leftarrow$ corr. datasets)

Bootstrapping

Bootstrap samples: $\mathcal{Z}^* = \{\mathcal{Z}_1^*, \cdots \mathcal{Z}_n^*\}$ each data point in $\mathcal{Z}_i^*$ was randomly drawn from $\mathcal{Z}$ with replacement.

e_0 Estimator: the error rate for the test data (data that wasn't selected by the bootstrap) is assumed to be the error estimate (e.g. for classification):

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

Jackknife

Estimate of an Estimator $\hat{S}_n$'s Bias.

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

$\text{bias}^{JK} = (n-1)(\tilde{S}_n - \hat{S}_m)$

$\tilde{S}_n = \frac{1}{n}\sum_{i=1}^n \hat{S}_{n-1}^{(-i)}$ avg. LOO Estimator.

Debiased est. can have big variance!

Bootstrap **Debiased**

$\overline{S} = 2\hat{S} - \frac{1}{B}\sum_b \hat{S}^*(b)$

Classification

group points in classes $1, \cdots, k, \mathcal{D}, \mathcal{O}$

$\mathcal{D}$: doubt class, $\mathcal{O}$: outliers.

Data: $\mathcal{Z} = \{z_i = (x_i, y_i) : 1 \leq i \leq n\}$ Assume we know $p_y(x) = P[X = x|Y = y]$

Found: classifier $\hat{c} : \mathcal{X} \to \mathcal{Y} := \{1, \cdots, \mathcal{D}\}$

Error: $\hat{R}(\hat{c}|\mathcal{Z}) = \sum_{(x_i,y_i)\in\mathcal{Z}} \mathbb{I}_{\{\hat{c}(x_i)\neq y_i\}}$

Expected Error:

$\mathcal{R}(\hat{c}) = \sum_{y\leq k} P[y]\mathbb{E}_{x|y}[\mathbb{I}_{\{\hat{c}(x_i)\neq y_i\}}|Y = y]$

(add term from $\mathcal{D}$)

Loss Functions

0-1 Loss: $L^{0-1}(y,z) = \begin{cases} 0 & \text{if}(z = y) \\ 1 & \text{if}(z \neq y) \end{cases}$

Exponential Loss:

$L^{exp}(y,z) = \exp(-(2y-1)(2z-1))$

Logistic Loss:

$L^{log}(y,z) = \ln(1 + \exp((2y-1)(2z-1)))$

Hinge Loss:

Favors sparsity. Used in SVM

$L^{hinge}(y,z) = \max\{0, 1-(2y-1)(2z-1)\}$

Bayes Optimal Classifier

Minimizes total risk for 0-1 Loss

$\hat{c}(x) = \begin{cases} y & \text{if} p(y|x) = \max_{z\leq k} p(z|x) > 1 - d \\ \mathcal{D} & \text{if} p(y|x) < 1 - d \forall y \end{cases}$

Generalize to other loss functions

Discriminant Functions

Functions $g_k(x) \quad 1 \leq k \leq K$

Decide: $g_y(x) > g_z(x) \forall z \neq y \Rightarrow$ chose y

Const factor doesn't change decision.

$g_k(x) = P[y|x] \propto P[x|y]P[y] \Rightarrow$

$g_k(x) = lnP[x|y] + lnP[y] = lnP[x|y] + \pi_y$

implements an opt. Baye classifier.

Decision Surface of Discriminant

Solve: $g_{k_1}(x) - g_{k_2}(x) = 0$ Special case with Gaussian classes:
if $\Sigma_y = \Sigma \Rightarrow$ linear decision surface
$g_k(x) = w^T(x - x_0)$ $\quad w = \Sigma^{-1}(\mu_1 - \mu_2)$
$x_0 = \frac{1}{2}(\mu_1 + \mu_2) - \frac{\sigma^2(\mu_1-\mu_2)}{(\mu_1-\mu_2)^T\Sigma^{-1}(\mu_1-\mu_2)} \log\frac{\pi_1}{\pi_2}$

Linear Classifier

optimal for Gaussian with equal cov.
Stat. simplicity & comput. efficiency.
$g(x) = a^T\tilde{x}$ $\quad a = (w_0, w)^T, \tilde{x} = (1, x)^T$
$a^T\tilde{x}_i > 0 \Rightarrow y_i = 1$ $\quad a^T\tilde{x}_i < 0 \Rightarrow y_i = 2$
Normalization: $\tilde{x}_i \rightarrow -\tilde{x}_i$ if $y_i = 2$
Find a: $a^T\tilde{x} > 0$ (linearly separable)
Learning w. Gradient Descent:
$a(k+1) = a(k) - \eta(k)\nabla J(a(k))$
$J(.)$: cost function $\eta(.)$: learning rate
Newton's rule (opt. grad descent):
$a(k+1) = a(k) - H^{-1}\nabla J$ $\quad H = \frac{\partial^2 J}{\partial a_i \partial a_j}$

Perceptron Criterion

$J_P(a) = \sum_{\tilde{x}\in\tilde{\mathcal{X}}}(-a^T\tilde{x})$
$\mathcal{X}$set of misclassified samples.
$\Rightarrow a(k+1) = a(k) + \eta(k)\sum\sum_{\tilde{x}\in\tilde{\mathcal{X}}}\tilde{x}$ Converges if data separable.

WINNOW Algorithm

Performs better when many dimensions are irrelevant. Search for 2 weight vectors a^+, a^- (for each class). If a point is misclassified:
$a_i^+ \leftarrow \alpha^{+\tilde{x}_i}a_i^+, a_i^- \leftarrow \alpha^{-\tilde{x}_i}a_i^-$ (class 1 err.)
$a_i^+ \leftarrow \alpha^{-\tilde{x}_i}a_i^+, a_i^- \leftarrow \alpha^{+\tilde{x}_i}a_i^-$ (class 2 err.)
Exponential update.

Fisher's Linear Discr. Analysis

Maximize distance of the means of the projected classes to find projection plane separating them best.
proj mean: $\tilde{\mu}_\alpha = \frac{1}{n_\alpha}\sum_{x\in\mathcal{X}_\alpha} w^T x = w^T\mu_\alpha$
Dist of proj means: $|w^T(\mu_1 - \mu_2)|$ Classes proj. cov: $\tilde{\Sigma}_1 + \tilde{\Sigma}_2 = w^T(\Sigma_1 + \Sigma_2)w$
Fishers Criterion:
$J(w) = \frac{\|\tilde{\mu}_1 - \tilde{\mu}_2\|^2}{\tilde{\Sigma}_1 + \tilde{\Sigma}_2} = \frac{w^T(\mu_1-\mu_2)(\mu_1-\mu_2)^T w}{w^T(\Sigma_1+\Sigma_2)w}$
Fishers Crit for Multiple Classes:
$J(W) = \frac{|W^T S_B W|}{W^T S_W W}$
$S_B = \sum_{i=1}^k n_k(\mu_k - \mu)(\mu_k - \mu)^T$
$S_W = \sum_{i=1}^k \sum_{x\in\mathcal{D}_i}(x - \mu_i)(x - \mu_i)^T$

Linear Discriminant for Multiclasses

Reformulate as $(k-1)$ "class α - not class α"dichotomie. But some area are ambiguous

Support Vector Machine (SVM)

Generalize Perceptron with margin and kernel. Find plane that maximizes margin m s.t.
$z_i g(\mathbf{y}) = z_i(\mathbf{w}^T\mathbf{y} + w_0) \geq m \quad \forall \mathbf{y}_i \in \mathcal{Y}$
$z_i \in \{-1, +1\} \quad \mathbf{y_i} = \phi(\mathbf{x_i})$
Vectors $\mathbf{y}_i$ are the support vectors
Functional Margin Problem:
minimizes $\|\mathbf{w}\|$ for m=1: $L(\mathbf{w}, w_0, \alpha) =$
$= \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^n \alpha_i[z_i(\mathbf{w}^T\mathbf{y}_i + w_0) - 1]$
where αs are Lagrange multipliers.
$\frac{\partial L}{\partial w}=0$ and $\frac{\partial L}{\partial w_0}=0$ give us constraints
$\mathbf{w} = \sum_{i=1}^n \alpha_i z_i \mathbf{y_i}$ $\quad 0 = \sum_{i=1}^n \alpha_i z_i$
Replacing these in $L(\mathbf{w}, w_0, \alpha)$ we get
$\tilde{L}(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i,j=1}^n \alpha_i\alpha_j z_i z_j \mathbf{y_i}^T\mathbf{y_j}$
with $\alpha_i \geq 0$ and $\sum_{i=1}^n \alpha_i z_i = 0$
This is the dual representation. The optimal hyperplane is given by
$\mathbf{w}^* = \sum_{i=1}^n \alpha_i^* z_i \mathbf{y_i}$
$w_0^* = -\frac{1}{2}(\min_{z_i=1}\mathbf{w}^{*T}\mathbf{y_i} + \max_{z_i=-1}\mathbf{w}^{*T}\mathbf{y_i})$
where α maximize the dual problem.
Only Support Vectors ($\alpha_i \neq 0$) contribute to the evaluation.
Optimal Margin: $\mathbf{w}^T\mathbf{w} = \sum_{i\in SV}\alpha_i^*$
Discrim.: $g^*(\mathbf{x}) = \sum_{i\in SV} z_i\alpha_i\mathbf{y_i}^T\mathbf{y_i} + w_0^*$
class = sign$(\mathbf{y}^T\mathbf{w}^* + w_0^*)$

Soft Margin SVM

Introduce slack to relax constraints
$z_i(\mathbf{w}^T\mathbf{y}_i + w_0) \geq m(1 - \xi)$
$L(\mathbf{w}, w_0, \xi, \alpha, \beta) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^n \xi_i -$
$-\sum_{i=1}^n \alpha_i[z_i(\mathbf{w}^T\mathbf{y}_i + w_0) - 1 + \xi_i]$
$-\sum_{i=1}^n \beta_i\xi_i$
C controls margin maximization vs. constraint violation
Dual Problem same than usual SVM but with suppl. constr.: $\alpha_i \leq C$

Non-Linear SVM

use kernel in discriminant funct: $g(\mathbf{x}) = \sum_{i=1}^n \alpha_i z_i K(\mathbf{x_i}, \mathbf{x})$
E.g solve the XOR Problem with:
$K(x, y) = (1 + x_1y_1 + x_2y_2)^2$

Multiclass SVM

$\forall$class $z \in \{1, 2, \cdots, M\}$ we introduce $\mathbf{w}_z$ and define the margin m s.t.:
$(\mathbf{w}_{z_i}^T\mathbf{y}_i + w_{z_i,0}) - max_{z\neq Z_i}(\mathbf{w}_z^T\mathbf{y}_i + w_{z,0}) \geq 0 \quad \forall \mathbf{y}_i \in \mathcal{Y}$

Structured SVM

Each sample $\mathbf{y}$ is assigned to a structured output label z
Output Space Representation:
joint feature map: $\psi(z, \mathbf{y})$
scoring function: $f_\mathbf{w}(z, \mathbf{y}) = \mathbf{w}^T\psi(\mathbf{z}, \mathbf{y})$
Classify: $\hat{z} = h(\mathbf{y}) \arg\max_{z\in\mathcal{K}} f_{\mathbf{w}(z,\mathbf{y})}$

Kernels

Similarity based reasoning
Gram Matrix $K = (K(\mathbf{x}_i, \mathbf{x}_j)) \quad 1 \leq i, j \leq n$
$K(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^T\phi(\mathbf{x}')$ $\quad K(\mathbf{x}, \mathbf{x}') = K(\mathbf{x}', \mathbf{x})$
$K(\mathbf{x}, \mathbf{x}')$ pos.semi-def. (all EV ≥ 0)
If K_1 & K_2 are kernels K is too:
$K(\mathbf{x}, \mathbf{x}') = K_1(\mathbf{x}, \mathbf{x}')K_2(\mathbf{x}, \mathbf{x}')$
$K(\mathbf{x}, \mathbf{x}') = \alpha K_1(\mathbf{x}, \mathbf{x}') + \beta K_2(\mathbf{x}, \mathbf{x}')$
$K(\mathbf{x}, \mathbf{x}') = K_1(h(\mathbf{x}), h(\mathbf{x}'))$ $\quad h: \mathcal{X} \rightarrow \mathcal{X}$
$K(\mathbf{x}, \mathbf{x}') = h(K_1(\mathbf{x}, \mathbf{x}'))$ $\quad h$: poly/exp
Kernel Function Examples:
$K(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T\mathbf{x}'$ $\quad K(\mathbf{x}, \mathbf{x}') = (\mathbf{x}^T\mathbf{x}' + 1)^p$
RBF(Gauss): $K(\mathbf{x}, \mathbf{x}') = \exp(-\|\mathbf{x} - \mathbf{x}'\|_2^2/h^2)$
Sigmoid: $K(\mathbf{x}, \mathbf{x}') = \tanh(\alpha\mathbf{x}^T\mathbf{x}' + c)$
not p.s-d eg: $\mathbf{x} = [1, -1], \mathbf{x}' = [-1, 2]$

Ensemble Methods

Combining Regressors

set of estimators: $\hat{f}_1(x), \cdots, \hat{f}_B(x)$ simple average: $\hat{f}(x) = \frac{1}{B}\sum_{i=1}^B \hat{f}_i(x)$
Bias$[\hat{f}(x)] = \frac{1}{B}\sum_{i=1}^B$ Bias$[f_i(x)]$
Var$[\hat{f}(x)] \approx \frac{\sigma}{B}$ if the estimators are uncorrelated.

Combining Classifiers

Input: classifiers $c_1(x), \cdots, c_B(x)$
Infer $\hat{c}_B(x) = sgn(\sum_{b=1}^B \alpha_b c_b(x))$
with weights $\{\alpha_b\}_{b=1}^B$
Requires diversity of the classifiers.

Bagging

Train on bootstrapped subsets.
Sample: $\mathcal{Z} = \{(x_1, y_1), \cdots (x_n, y_n)\}$
$\mathcal{Z}^*$: chose i.i.d from $\mathcal{Z}$ w. replacement

Random Forest (Bagging strategy)

Collection of uncorr. decision trees. Partition data space recursively. Grow the tree sufficiently deep to reduce bias. Prediction with voting.

Boosting

Combine uncorr. weak learners in sequence. (Weak to avoid overfitting).
Coeff. of $\hat{c}_{b+1}$ depend on $\hat{c}_b$'s results
AdaBoost (minimizes exp. loss)
Init: $\mathcal{X} = \{(x_1, y_1), \cdots, (x_n, y_n)\}, w_i^{(1)} = \frac{1}{n}$
Fit $\hat{c}_b(x)$ to $\mathcal{X}$ weighted by $w^{(b)}$
$\epsilon_b = \sum_{i=1}^n w_i^{(b)}\mathbb{I}_{\{c_b(x_i)\neq y_i\}} / \sum_{i=1}^n w_i^{(b)}$
$\alpha_b = \log\frac{1-\epsilon_b}{\epsilon_b}$
$w_i^{(b+1)} = w_i^{(b)}\exp(\alpha_i\mathbb{I}_{\{\hat{c_b}(x_i)\neq y_i\}})$
return $\hat{c}_B(x) = \text{sgn}(\sum_{b=1}^B \alpha_b c_b(x))$
best approx. at log-odds ratio.

Neural Networks

Multi Layer Perceptron

$\{x_j\}_{j=1}^J$ input, $\{y_i\}_{i=1}^I$ output
$\{z_k^l\}_{k=1}^{K(l)}$ hidden nodes in layer l $1 \leq l \leq L$
$w_m k^l$ weights from z_k^{l-1} to z_m^l
$w_i k^{L+1}$ weights from z_k^L to output y_i
$z_k^l = h(a_k^l) = h(\sum_{m=1}^{K(l-1)} w_{km}^l z_m^{l-1})$
$y_i = \sigma(a_i^{L+1}) = h(\sum_{m=1}^{K(l-1)} w_{im}^{L+1} z_m^L)$
$\mathcal{L}(\hat{y}(\mathbf{W}, \mathbf{X}), y) = \sum_{n=1}^N \mathcal{L}_n(\hat{y}(\mathbf{W}, \mathbf{X}_n), Y_n)$
$L = 0$ or $h(a) = a \Rightarrow$ multiple lin. reg.
Layers $\Rightarrow$ generaliz. & simplicity.
Model data generating mechanism.

Backpropagation

Effic. evaluation of loss derivative:
$\frac{\partial\mathcal{L}_n}{\partial w_{ik}^{L+1}} = \delta_i^{L+1}z_k^L$ $\quad \frac{\partial\mathcal{L}_n}{\partial w_{mk}^l} = \delta_m^l z_k^{l-1}$
$\delta_i^{L+1} = (\hat{y}_i - y_i)\sigma'(\sum_{m=1}^{K(L)} w_{im}^{L+1} z_m^L)$
$\delta_m^l = (\sum_{r=1}^{K(l+1)}\delta_r^{l+1}w_{rm}^{l+1})\cdot$
$h'(\sum_{r=1}^{K(l-1)} w_{mr}^l z_r^{l-1})$
$w_{ij}^l \leftarrow w_{ij}^l + \eta\delta_i^l z_j^{(l-1)}$

Regularization

Avoid overfitting on complex nets.
Early Stopping separate data into train/error/validation sets.
Drop Out Combine thinned nets with removed nodes.
Bayesian priors on w's

Autoencoder

Data compression purposes, Output should reproduce input.$\Rightarrow$ PCA

Convolutional Neural Network

Modelling invariance. Convolutional Layers (filters on a region) & Pooling Layers (aggregate nodes together).

Unsupervised Learning

Histograms

$p_i = \frac{n_i}{N\Delta_i}$ $\quad n \leq N$ in bin i of size Δ_i
Not scaling to multiple dimensions.
$K \simeq NP$ $\quad P \simeq p(x)V \Rightarrow p(x) = \frac{K}{NV}$
K #samples in region of volume V, P probability of falling in it.

Kernel Density Estimator

Fix V and determine K.
Gaussian Kernel: $\phi(u) = \frac{\exp(-\frac{1}{2}\|x\|^2)}{\sqrt{2\pi}}$
Result in a smoother density model
$p(x) = \frac{1}{N}\sum_{n=1}^N \frac{1}{(2\pi h^2)^{D/2}}\exp(-\frac{\|x - x_n\|^2}{2h^2})$
We can chose any other kernel ϕ with
$\phi(u) \geq 0 \quad \int\phi(u)du = 1$

K-Nearest Neighbors

Fix K and find V
$\hat{p}(x) = \frac{1}{V_k(x)}, v_k(x)$ minimal volume around x containing k neighbors.
Classifier: classify x by the majority of the vote of its k-NN.
1-NN Error Rate the 1-NN error rate P is always $P^* \leq P \leq 2P^*$ where P^* is the error rate of the Bayes rule. $\Rightarrow$ as k goes to infinity kNN becomes optimal
KNN not optimal if class densities are very different.

Mixture Models

Gaussian Mixture

EM-Algorithm

Latent Variable: unknown data $\rightarrow$ What cluster generated each sample?
EM does ML for unknown parameters.
Latent var. $M_{\mathbf{x}c} = \begin{cases} 1 & \text{c generated x} \\ 0 & \text{else} \end{cases}$
$P(\mathcal{X}, M|\theta) = \prod_{x\in\mathcal{X}}\prod_{c=1}^k(\pi_c P(\mathbf{x}|\theta_c))^{M_{\mathbf{x}c}}$

E-Step

$\gamma_{\mathbf{x}c} = \mathbb{E}[M_{\mathbf{x}c}|\mathcal{X}, \theta^{(j)}] = \frac{P(\mathbf{x}|c,\theta^{(j)})P(c|\theta^{(j)})}{P(\mathbf{x}|\theta^{(j)})}$

M-Step

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

k-Means

identify clusters of data.
Given $\mathcal{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$
Find $c(.)$ and $\mathcal{Y}$ minimizing
$\mathcal{R}^k m(c, \mathcal{Y}) = \sum_{x\in\mathcal{X}}\|x - \mu_{c(x)}\|^2$ Assign to nearest cluster. Recompute all clusters and repeat. Also called **hard EM**. Special case of GMM w. uniform prior and diag. covariance ($\rightarrow 0$).

Extras

Taylorreihe: $\sum_{n=0}^\infty \frac{f^{(n)}(a)}{n!}(x - a)^n$
Convex/Concav: $f'' \geq 0$ or $f'' \leq 0$
LinAlg: $X_{-i}Y_{-j}^T = XY^T - x_iy_i^T$