1 Regression

Linear Regression

Error: $\hat{R}(w) = \sum_{i=1}^n (y_i - w^T x_i)^2 = \|Xw - y\|_2^2$
Closed form: $w^* = (X^T X)^{-1} X^T y$
Gradient: $\nabla_w \hat{R}(w) = 2X^T(Xw - y)$

Ridge regression

Error: $\hat{R}(w) = \sum_{i=1}^{n} (y_i - w^T x_i)^2 + \lambda \|w\|_2^2$
Closed form: $w^* = (X^T X + \lambda I)^{-1} X^T y$
Grad: $\nabla_w \hat{R}(w) = -2\sum_{i=1}^n (y_i - w^T x_i) \cdot x_i + 2\lambda w$
Feat. Sel.: $Xw^* = \sum_{j=1}^d u_j \frac{\sigma_j^2}{\sigma_j^2 + \lambda} u_j^T y$, $X = U\Sigma V^T$

Combination of Regression Models:

bias$[\hat{f}(x)] = \frac{1}{B}\sum_{i=1}^B$ bias$[\hat{f}_i(x)]$
$\mathbb{V}[\hat{f}(x)] \approx \frac{\sigma^2}{B}$, assuming small covariances and similar variances

RSS Estimator

Distribution of estimator: $\hat{\beta} \sim \mathcal{N}(\beta, (X^T X)^{-1}\sigma^2)$. **Unbiasedness:** $\mathbb{E}[\hat{\beta}] = \mathbb{E}[(X^T X)^{-1} X^T y] = (X^T X)^{-1} X^T \mathbb{E}[X\beta + \epsilon] = (X^T X)^{-1}(X^T X)\beta + X^T \mathbb{E}[\epsilon] = \beta + 0$ **Variance of** $a^T\hat{\beta}$: $\mathbb{V}(a^T (X^T X)^{-1} X^T (X\beta + \epsilon)) = \mathbb{V}(a^T\beta) + \mathbb{E}(a^T (X^T X)^{-1} X^T \epsilon\epsilon^T X (X^T X)^{-1} a) = \sigma^2 a^T (X^T X)^{-1} a$

Gauss-Markov Theorem

For any linear estimator $\tilde{\theta} = c^T \mathbf{y}$ that is unbiased for $a^T\beta$ it holds: $\mathbb{V}(a^T\hat{\beta}) \le \mathbb{V}(c^T\mathbf{y})$
Proof: Let $c^T\mathbf{y} = a^T\hat{\beta} + a^T\mathbf{Dy} = a^T((\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T + \mathbf{D})\mathbf{y}$ be an unbiased estimator of $a^T\beta$; then it follow $a^T\mathbf{DX}\beta = 0$ which implies $\mathbf{DX} = 0$.
$\mathbb{V}(c^T\mathbf{y}) = \mathbb{E}[(c^T\mathbf{y})^2] - \mathbb{E}(c^T\mathbf{y})^2 = c^T(\mathbb{E}\mathbf{yy}^T - \mathbb{E}\mathbf{y}\mathbb{E}\mathbf{y}^T)c = \sigma^2 c^T c = \sigma^2\left(a^T((\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T + \mathbf{D})(\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1} + \mathbf{D}^T)a\right)$
$= \sigma^2\left(a^T(\mathbf{X}^T\mathbf{X})^{-1}a + \mathbf{DD}^T a\right) = \mathbb{V}(a^T\hat{\beta}) + a^T\mathbf{DD^T}a \ge \mathbb{V}(a^T\hat{\beta})$ (note: $\mathbf{DD^T}$ is PSD)

Bias vs. Variance

$\mathbb{E}_D\mathbb{E}_{X,Y}\left(\hat{f}(X) - Y\right)^2 =$
$\mathbb{E}_D\mathbb{E}_X\left(\hat{f}(X) - \mathbb{E}(Y|X)\right)^2 + \mathbb{E}_{X,Y}(Y - \mathbb{E}(Y|X))^2$
$= \mathbb{E}_X\mathbb{E}_D\left(\hat{f}(X) - \mathbb{E}_D(\hat{f}(X))\right)^2$ (variance)
$+ \mathbb{E}_X\left(\mathbb{E}_D(\hat{f}(X)) - \mathbb{E}(Y|X)\right)^2$ (bias^2)
$+ \mathbb{E}_{X,Y}(Y - \mathbb{E}(Y|X))^2$ (noise)
High bias can cause an algorithm to miss the relevant relations between features and target outputs (underfitting).
High variance can cause overfitting: modeling the random noise in the training data, rather than the intended outputs.

Curse of Dimensionality

To obtain a reliable estimate at a given regularity, the required number of samples grows exponentially with the dimension of the sample space.

Ridge Parametric to nonparametric

Ansatz: $w = \sum_i \alpha_i x$
$w^* = \underset{w}{\text{argmin}} \sum_i (w^T x_i - y_i)^2 + \lambda\|w\|_2^2 =$
$\text{argmin}_{\alpha_{1:n}} \sum_{i=1}^n (\sum_{j=1}^n \alpha_j x_j^T x_i - y_i)^2 + \lambda \sum_i \sum_j \alpha_i \alpha_j (x_i^T x_j)$
$= \text{argmin}_{\alpha_{1:n}} \sum_{i=1}^n (\alpha^T K_i - y_i)^2 + \lambda \alpha^T K \alpha$
$= \text{argmin}_{\alpha} \|\alpha^T K - y\|_2^2 + \lambda \alpha^T K \alpha$
Closed form: $\alpha^* = (K + \lambda I)^{-1} y$
Prediction: $y^* = w^{*T} x = \sum_{i=1}^n \alpha_i^* k(x_i, x)$

2 Gaussian Processes

Gaussian Process

$p\left(\begin{bmatrix}\mathbf{y}\\ y^*\end{bmatrix} | x^*, \mathbf{X}, \sigma\right) = \mathcal{N}\left(\begin{bmatrix}\mathbf{y}\\ y^*\end{bmatrix} | \mathbf{0}, \begin{bmatrix}\mathbf{C_n} & \mathbf{k}\\ \mathbf{k}^T & c\end{bmatrix}\right)$
with $\mathbf{C_n} = \mathbf{K} + \sigma^2\mathbf{I}, c = k(x_{n+1}, x_{n+1}) + \sigma^2$, $\mathbf{k} = k(x_{n+1}, \mathbf{X}), \mathbf{K} = k(\mathbf{X}, \mathbf{X})$
$p(y^*|x^*, X, y) = \mathcal{N}(y^*|\mu, \sigma^2)$
with $\mu = k^T C_n^{-1} y, \sigma^2 = c - k^T C_n^{-1} k$

GP Hyperparameter Optimization

Log-likelihood:
$l(Y|\theta) = -\frac{n}{2}\log(2\pi) - \frac{1}{2}\log|C_n| - \frac{1}{2}Y^T C_n^{-1} Y$
Set of hyperparameters θ determine parameters C_n. Gradient descent: $\nabla_{\theta_i} l(Y|\theta) = -\frac{1}{2}tr(C_n^{-1}\frac{\partial C_n}{\partial\theta_i}) + \frac{1}{2}Y^T C_n^{-1}\frac{\partial C_n}{\partial\theta_i}C_n^{-1}Y$

Kernels

$k_1(x,y) + k_2(x,y)$, $k_1(x,y)\cdot k_2(x,y)$, $c \cdot k_1(x,y)$ for $c > 0$, $f(k_1(x,y))$, where f is exponential/polynomial with positive coefficents
$k(x,y) = \phi(x)^T\phi(y)$, for some ϕ ass. with k

3 Bayesian Methods

MLE

$\theta^* = \text{argmax}_\theta P(y|x,\theta)$
$= \text{argmax}_\theta \prod_{i=1}^n P(y_i|x_i,\theta)$ (iid)
$= \text{argmax}_\theta \sum_{i=1}^n logP(y_i|x_i,\theta)$

MAP

$w^* = \underset{w}{\text{argmax}} P(w|x,y) = \underset{w}{\text{argmax}} \frac{P(w|x)P(y|x,w)}{P(y|x)}$
$= \underset{w}{\text{argmax}} logP(w) + \sum_i logP(y_i|x_i,w) + const.$

MLE = MAP

$n \to \infty$ or prior is uniformly distr.

4 Numerical Estimates Methods

Cross Validation/LOO

1. Split the data in K subsets. $D = D_1 \cup \ldots \cup D_k$ $\kappa : [1,n] \to [1,k]$ denotes subset (x_i, y_i) is element of
2. Train model $\hat{f}^{-v}(x)$ to $K-1$ subsets. Validate with not used subset.
$\hat{f}^{-v} \in \arg\min_{f\in F} \frac{1}{|D/D_v|}\sum i \notin Z_v (y_i - f(x_i))^2$
3. Pred. error $\hat{R}_{CV} = \frac{1}{n}\sum_{i\le n}(y_i - \hat{f}^{-\kappa(i)}(x_i))^2$

Problem with LOO: unbiased, but high variance.

Bootstrapping

1. Resample data with replacement $D_1, \ldots, D_k$.
2. Train model
3. $S = \frac{1}{B}\sum_{b\le B} S(D_b)$ (mean)

$\sigma^2(S) = \frac{1}{B-1}\sum_{b\le B}(S(D_b) - S)^2$ (variance)
Bootstrap works if the deviation between empirical and bootstrap estimator converges in probability to the deviation between true parameter value and the empirical estimator.

5 Classification

0/1 loss

$L^{0-1}(y, c(x)) = \mathbb{1}_{[y \neq c(x)]}$

Bayesian Decision Theory

Est. cond. dist: $P(y|x,w) = Ber(\sigma(w^T x))$
Action set: $\mathcal{A} = \{+1, -1\}$
Cost fn: $C(y,a) = \begin{cases} c_{FP} \text{ , if } y = -1 \text{ and } a = +1 \\ c_{FN} \text{ , if } y = +1 \text{ and } a = -1 \\ 0 \text{ , otherwise} \end{cases}$
The action that minimizes the expected cost is:
$C_+ = \mathbb{E}_y[C(y,+1)|x] = P(y = +1|x)\cdot 0 + (P(y = -1)|x)\cdot c_{FP}$
$C_- = \mathbb{E}_y[C(y,-1)|x] = P(y = +1|x)\cdot c_{FN} + P(y = -1|x)\cdot 0$
Predict +1 if $C_+ \le C_-$

6 Design of Discriminant

Stochastic Gradient Descent

1. Start arbitrary $w_0 \in \mathbb{R}^d$
2. For t do: Pick $(x_1, y_1) \in_{u.a.r.} D$

$w_{t+1} = w_t - \eta_t \nabla l(w_t, x_1, y_1)$

Newton optimization

$w_{t+1} = w_t - H^{-1}\nabla l(w_t)$, where $H = \nabla^2 l(w_t)$

Perceptron

SGD + Perceptron loss ($\max\{0, -y_i w^T x_i\}$)
Theorem: If D is linearly seperable ⇒ Perceptron will obtain a linear seperator.

Fishers LDA

$\hat{w} = \arg\max_w \frac{\mathbf{w}^T\mathbf{S}_T\mathbf{w}}{\mathbf{w}^T\mathbf{S}_W\mathbf{w}} = \frac{\sigma_{between}}{\sigma_{within}} \propto \mathbf{S}_W^{-1}(\overline{x}_1 - \overline{x}_2)$
$\mathbf{S}_B = (\overline{x_1} - \overline{x_2})(\overline{x_1} - \overline{x_2})^T$
$\mathbf{S}_W = \sum_{j=1}^{\text{class1}}(x_{1j} - \overline{x_1})^2 + \sum_{j=1}^{\text{class2}}(x_{2j} - \overline{x_2})^2$
new point x_0 class 1 if $(\overline{x}_1 - \overline{x}_2)^T\mathbf{S}_W^{-1}x_0 \ge \hat{m} = \frac{1}{2}(\overline{x}_1 - \overline{x}_2)^T\mathbf{S}_W^{-1}(\overline{x}_1 + \overline{x}_2)$), class 2 otherwise

7 SVM

Primal, constrained:
$\min_w w^\top w + C\sum_{i=1}^n \xi_i$,
s.t. $y_i w^\top x_i \ge 1 - \xi_i, \xi_i \ge 0$
Primal, unconstrained (hinge loss):
$\min_w w^\top w + C\sum_{i=1}^n \max(0, 1 - y_i w^\top x_i)$
Dual:
$\max_\alpha \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j x_i^\top x_j$,
s.t. $0 \le \alpha_i \le C$
Dual to primal: $w^* = \sum_{i=1}^n \alpha_i^* y_i x_i$, $\alpha_i > 0$.
Error: $\hat{R}(w) = \sum_{i=1}^n \max\{0, 1 - y_i w^T x_i\} + \lambda\|w\|_2^2$
$\nabla_w \hat{R}(w) = \left\{ \begin{array}{l} -X^T y + 2\lambda w \text{ , if } y_i w^T x_i < 1 \\ 2\lambda w\text{, otherwise} \end{array} \right\}$

8 Non-linear SVM

Multiclass SVM

$\min_{w,\eta\ge 0} \frac{1}{2}w^T w + C\sum_i \xi_i$
s.t. $\forall y_i \in Y : (w_{z_i}^T y_i) - max_{z\neq z_i}(w_z^T y_i) \ge 1 - \xi_i$

Structured SVM

$\min_{w,\eta} \frac{\lambda}{2}\|w\|^2 + \frac{1}{n}\sum_{i=1}^n \eta_i$, $\eta \ge H_i(w)\forall i$, where $H_i(w) = \max_{y\in Y(x_i)} L(y_i, y) - w^T(\phi(x_i, y_i) - \phi(x_i, y))$

9 Ensemble method

Random Forest

for b=1:B do:
draw a bootstrap sample D_b
repeat until node size<n_{min}:
1. select m features from p features
2. pick the best variable and split-point
3. Split the node accordingly

return the forest $\{\hat{c}_b(x)\}_{b=1}^B$

Adaboost

Initialize weights $w_i = 1/n$
for b=1:B do:
1. Fit classifier $c_b(x)$ with weights w_i
2. Compute error $\epsilon_b = \sum_i w_i^{(b)}\mathbb{1}_{[c_b(x_i)\neq y_i]} / \sum_i w_i^{(b)}$
3. Compute coeff. $\alpha_b = log(\frac{1-\epsilon_b}{\epsilon_b})$
4. Update weights $w_i = w_i \exp(\alpha_b \mathbb{1}_{[y_i \neq c_b(x_i)]})$

Return $\hat{c}_B(x) = \text{sign}\left(\sum_{b=1}^B \alpha_b c_b(x)\right)$
Loss: Exponential loss function
Model: Additive logistic regression
Bayesian approach (assumes posteriors)
Newtonlike updates (Gradient Descent)

Bagging

for $b = 1$ to B **do**:
1. Z^{*b} = b-th bootstrap sample from Z
2. Construct classifier c_b based on Z^{*b}

return ensemble class. $\hat{c}_B(x) = sgn(\sum_{i=1}^{B} c_i(x))$

Works: Covariance small (different subset for training), Variance small (similar behaviour of weak learners), biases weakly affected.

Bag. aggr. pred.: $h_B(x) = E_{D'\sim D}[h_{D'}(x)]$

Ideal aggr. pred.: $h_A(x) = E_{D\sim P(x,y)}[h_D(x)]$

$E_D[L(y,h_D(x))] = E_D[(y-h_D(x))^2] = E_D[y^2] - 2E_D[y \cdot h_D(x)] + E_D[h_D(x)^2] = y^2 - 2y \cdot E_D[h_D(x)] + E_D[h_D(x)^2] \geq y^2 - 2y \cdot E_D[h_D(x)] + E_D[h_D(x)]^2 = y^2 - 2y \cdot h_A(x) + h_A(x)^2 = (y - h_A(x))^2 = L(y, h_A(x))$

Bias↓&Var.↓: Use complex decision tree (bias↓), ensemble mult. decision trees (var↓)

10 Unsupervised Learning

Histogram

$H = (H_1, .., H_k)$ with $H_i = \#\{x \in S | x \in I_j\}$ with I_j = k pairwise distinct subintervals. Histogram as density estimation: $\widetilde{H} = \frac{1}{n}(H_1, ..H_k)$

Parzen

$\hat{p}_n = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{V_n}\phi(\frac{x-x_i}{h_n})$ where $\int \phi(x)dx = 1$ Problems: 1) V_0 too small - noisy, V_0 too big: oversmoothed 2) Different behavior of the data distribution may require different strategies in different parts of the feature space.

$\int \frac{1}{N}\sum_{i=1}^{N}\phi(\frac{|x-x_i|}{h})dx_i = \frac{1}{N}\frac{1}{V}\sum_{i=1}^{N}\int\phi(\frac{|x-x_i|}{h})dx_i = \frac{1}{VN} \cdot VN = 1$

K-NN

$\hat{p}_n = \frac{1}{V_k}$ volume with k neighbours

error rate of 1-NN classifier is bounded by twice the Bayes error rate

K-means

$L(\mu) = \sum_{i=1}^{n} \min_{j\in\{1...k\}} \|x_i - \mu_y\|_2^2$

11 Mixture Model

Latent variable

We denote the latent variable indicating the component the point is sampled from by Z, which takes on values in $\{1, ..., k\}$. (γ_j)

E-step: Posterior probabilities

$\gamma_j^t(x_i) = P(Z = j | x_i, \theta_t) = \frac{P(x_i|Z=j,\theta_t)P(Z=j|\theta_t)}{P(x_i;\theta_t)} = \frac{\pi_j \mathcal{N}(\mathbf{x}_i|\boldsymbol{\mu}_j,\Sigma_j)}{\sum_{k=1}^{K}\pi_k\mathcal{N}(\mathbf{x}_i|\boldsymbol{\mu}_k,\Sigma_k)}$

M-step: maximizing expected log likelihood

$\mathbb{E}_{\gamma^t}[\log P(\mathcal{D};\theta)] = \mathbb{E}_{\gamma^t}[\log \prod_{i=1}^{n} P(x_i, z_i; \theta)] = \sum_{i=1}^{n} \mathbb{E}_{\gamma^t}[\log P(x_i, z_i; \theta)] = \sum_{i=1}^{n}\sum_{j=1}^{k}\gamma_j^t(x_i)\log(P(x_i|z_i = j;\theta)P(z_i = j;\theta))$

$\theta_{t+1} = \underset{\theta}{\operatorname{argmax}}\,\mathbb{E}_{\gamma^t}[\log P(\mathcal{D};\theta)]$

$\boldsymbol{\mu}_j := \frac{\sum_{i=1}^{N}\gamma_j(x_i)\mathbf{x}_i}{\sum_{i=1}^{N}\gamma_j(x_i)}$, $\Sigma_j = \frac{\sum_{i=1}^{N}\gamma_j(x_i)(\mathbf{x}_i-\boldsymbol{\mu}_j)(\mathbf{x}_j-\boldsymbol{\mu}_j)^T}{\sum_{i=1}^{N}\gamma_j(x_i)}$

$\pi_j = \frac{1}{N}\sum_{i=1}^{N}\gamma_j(x_i)$

Gaussian Mixture Models

$p(\mathbf{x}) = \sum_{k=1}^{K}\pi_k\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k,\Sigma_k)$

Assignment variable: $\mathbf{z}_k \in \{0,1\}$, $\sum_{k=1}^{K}\mathbf{z}_k = 1$, $\Pr(\mathbf{z}_k = 1) = \boldsymbol{\pi}_k \Leftrightarrow p(\mathbf{z}) = \prod_{k=1}^{K}\boldsymbol{\pi}_k^{\mathbf{z}_k}$, π_k = mixing prop. of cluster k

Complete data distribution:

$p(\mathbf{x},\mathbf{z}) = \prod_{k=1}^{K}\left(\boldsymbol{\pi}_k\mathcal{N}(\boldsymbol{\mu}_k,\Sigma_k)\right)^{\mathbf{z}_k}$

Likelihood of observed data (iid) $\mathbf{X} = [\mathbf{x_1}, .., \mathbf{x_N}]$: $p(\mathbf{X}|\boldsymbol{\pi},\boldsymbol{\mu},\Sigma) = \prod_{n=1}^{N}p(\mathbf{x}_n|\boldsymbol{\pi},\boldsymbol{\mu},\Sigma) = \prod_{n=1}^{N}\sum_{k=1}^{K}\pi_k\mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k,\Sigma_k)$

Log-likelihood: $\log p(\mathbf{X}|\boldsymbol{\pi},\boldsymbol{\mu},\Sigma) = \sum_{i=1}^{N}\log\left(\sum_{k=1}^{K}\pi_k\mathcal{N}(\mathbf{x}_i|\boldsymbol{\mu}_k,\Sigma_k)\right)$

12 Time series

Markov Model

Markov assumption: $P(Y_t|Y_{1:t-1}) = P(Y_t|Y_{t-1})$

Stationarity assumption:
$P(Y_{t+1} = y_1|Y_t = y_2) = P(Y_t = y_1|Y_{t-1} = y_2)$

Product rule:
$P(Y_t, ..., Y_1) = P(Y_t|Y_{t-1}, ..., Y_1) \cdot ... \cdot P(Y_1)$

Sum rule:
$P(Y_{t+2}|Y_{1:t}) = \sum_{Y_{t+1}^i} P(Y_{t+2}Y_{t+1}^1|Y_{1:t})$

Hidden Markov Model

triplet $M = (\Sigma, Q, \Theta)$

Σ symbols, Q states, $\Theta = (A, E)$ transition and emission, $e_k(b)$ emission prob. $x_k \in Q, b \in \Sigma$

Forward/Backward - Alternative

Goal: $P(x_t|s) \propto P(x_t, s) = P(s_{t+1:n}|x_t)P(x_t, s_{1:k})$

Evaluation (Forward/Backward)

Transition A and emission E known. Sequence s given.
Wanted: prob that s is generated by HMM.

Forward:
Wanted: $f_l(s_t) = P(x_t = l, s_{1:t})$
$f_l(s_{t+1}) = e_l(s_{t+1})\sum_k f_k(s_t)a_{k,l}$,
$f_l(s_1) = \pi_l e_l(s_1)\forall l \in Q$

Backward:
Wanted: $b_l(s_t) = P(s_{t+1:n}|x_t = l)$
$b_l(s_t) = \sum_k e_k(s_{t+1})b_k(s_{t+1})a_{l,k}$,
$b_l(s_n) = 1\forall l \in Q$

Complexity in time: $\mathcal{O}(|\Sigma|^2 \cdot T)$

Decoding (Viterbi)

Given: Observation sequence $O = \{O_1O_2 \dots O_T\}$, $a_{ij} = P(q_{t+1} = S_j|q_t = S_i)$, $b_j(k) = P(v_k \text{at } t|q_t = S_j)$

Wanted: most likely path $Q = \{q_1, q_2, \dots q_T\}$

$\delta_t(i)$ best score along single path, at a time t, which accounts for the first t observations and ends in S_i

$\delta_t(j) = max_{1\leq i\leq N}[\delta_{t-1}(i)a_{ij}]b_j(O_t)$

$\phi_t(j) = argmax_{1\leq i\leq N}[\delta_{t-1}(i)a_{ij}]$

Time: $\mathcal{O}(|S|^2 \cdot T)$ Space $\mathcal{O}(|S| \cdot T)$

Decoding (Viterbi) - Alternative

Transition $a_{i,j} = P(x_{t+1} = j|x_t = i)$ and emission $e_l(s_t) = P(s_t|x_t = l)$ known. Sequence s given.

Wanted: Most likely path x responsible for the sequence.

$v_l(s_{t+1}) = e_l(s_{t+1})\max_k(v_k(s_t)a_{k,l})$

$v_l(s_1) = \pi_l e_l(s_1)\forall l \in Q$

Time: $\mathcal{O}(|\Sigma|^2 \cdot T)$, Space: $\mathcal{O}(|\Sigma| \cdot T)$

Learning (Baum-Welch)

Know: Set of sequences $s^1, ..., s^m$
Wanted: max transition A and emission E

E-step I: Compute all $f_k(s_t^j)$ (forward-algo.) & $b_k(s_t^j)$ (backward algo.)

E-step II: Compute A_{kl}, $E_k(b)$ for all states and symbols

$A_{kl} = \sum_{j=1}^{m}\frac{1}{P(\mathbf{s}^j)}\sum_{t=1}^{n}f_k^j(s_t^j)a_{kl}e_l(s_{t+1}^j)b_l^j(s_{t+1}^j)$

$E_k(b) = \sum_{j=1}^{m}\frac{1}{P(\mathbf{s}^j)}\sum_{t|S_t^j=b}^{n}f_k^j(s_t^j)b_k^j(s_t^j)$

M-step: Compute param. estimates a_{kl}, $e_k(b)$

$a_{kl} = \frac{A_{kl}}{\sum_{i=1}^{n}A_{ki}}$, $e_k(b) = \frac{E_k(b)}{\sum_{b'}E_k(b')}$

Complexity: $\mathcal{O}(|\Sigma|^2)$ in storage (space).

13 Neural Network

Backpropagation

For each unit j on the output layer:
- Compute error signal: $\delta_j = \ell_j'(f_j)$
- For each unit i on layer L: $\frac{\partial}{\partial w_{j,i}} = \delta_j v_i$

For each unit j on hidden layer $l = \{L-1, .., 1\}$:
- Error signal: $\delta_j = \phi'(z_j)\sum_{i\in Layer_{l+1}} w_{i,j}\delta_i$
- For each unit i on layer $l-1$: $\frac{\partial}{\partial w_{j,i}} = \delta_j v_i$

14 Appendix

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

$f(x) = \frac{1}{\sqrt{(2\pi)^d\det\Sigma}}e^{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)}$, $\mathcal{N}(x|\mu,\Sigma)$

Condition number: $\kappa(A) = \frac{\sigma_{max}(A)}{\sigma_{min}(A)}$

Calculus

- Part.: $\int u(x)v'(x)dx = u(x)v(x) - \int v(x)u'(x)dx$
- Chain r.: $\frac{f(y)}{g(x)} = \frac{dz}{dx}\Big|_{x=x_0} = \frac{dz}{dy}\Big|_{z=g(x_0)} \cdot \frac{dy}{dx}\Big|_{x=x_0}$
- $\frac{\partial}{\partial\mathbf{x}}(\mathbf{b}^\top\mathbf{x}) = \frac{\partial}{\partial\mathbf{x}}(\mathbf{x}^\top\mathbf{b}) = \mathbf{b}$ • $\frac{\partial}{\partial\mathbf{x}}(\mathbf{x}^\top\mathbf{x}) = 2\mathbf{x}$
- $\frac{\partial}{\partial\mathbf{x}}(\mathbf{x}^\top\mathbf{A}\mathbf{x}) = (\mathbf{A}^\top + \mathbf{A})\mathbf{x} =^{\text{if A sym.}} 2\mathbf{A}\mathbf{x}$
- $\frac{\partial}{\partial\mathbf{x}}(\mathbf{b}^\top\mathbf{A}\mathbf{x}) = \mathbf{A}^\top\mathbf{b}$ • $\frac{\partial}{\partial\mathbf{X}}(\mathbf{c}^\top\mathbf{X}\mathbf{b}) = \mathbf{c}\mathbf{b}^\top$
- $\frac{\partial}{\partial\mathbf{X}}(\mathbf{c}^\top\mathbf{X}^\top\mathbf{b}) = \mathbf{b}\mathbf{c}^\top$ • $\frac{\partial}{\partial\mathbf{x}}(\|\mathbf{x}-\mathbf{b}\|_2) = \frac{\mathbf{x}-\mathbf{b}}{\|\mathbf{x}-\mathbf{b}\|_2}$
- $\frac{\partial}{\partial\mathbf{x}}(\|\mathbf{x}\|_2^2) = \frac{\partial}{\partial\mathbf{x}}(\|\mathbf{x}^\top\mathbf{x}\|_2) = 2\mathbf{x}$ • $\frac{\partial}{\partial\mathbf{X}}(\|\mathbf{X}\|_F^2) = 2\mathbf{X}$
- $\text{sigmoid}(x) = \sigma(x) = \frac{1}{1+\exp(-x)}$
- $\nabla\text{sigmoid}(x) = \text{sigmoid}(x)(1-\text{sigmoid}(x))$
- $\nabla\tanh(x) = 1 - \tanh^2(x)$

Probability / Statistics

Sum Rule $P(X = x_i) = \sum_{j=1}^{J}p(X = x_i, Y = y_i)$

$\forall y \in Y: \sum_{x\in X}P(x|y) = 1$ (property for any fixed y)

Product rule $P(X, Y) = P(Y|X)P(X)$

Independence $P(X, Y) = P(X)P(Y)$

Bayes' Rule $P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}\frac{P(X|Y)P(Y)}{\sum_{i=1}^{k}P(X|Y_i)P(Y_i)}$

Conditional independence $X \perp Y|Z$
$P(X, Y|Z) = P(X|Z)P(Y|Z)$
$P(X|Z, Y) = P(X|Z)$

$P(x_1, \dots, x_n) = \prod_{i=1}^{n}P(x_i)$ (iff IID)

MGF $\mathbf{M}_X(t) = \mathbb{E}[e^{\mathbf{t}^T\mathbf{X}}]$, $\mathbf{X} = (X_1, .., X_n)$

Conj. prior if $p(\theta|X)$ is from the same distribution family as $p(\theta)$, then the prior distribution $p(\theta)$ is called conjugate to $p(X|\theta)$. Gamma conjugate to Exponential, normal conjugate to normal. Show $p(\theta|X) \sim p(X|\theta)p(\theta)$.

Expectation

$\mathbb{E}[X] = \int_{-\infty}^{\infty}xp(x)dx$
$\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y]$, if X & Y indep.
$\text{Var}(X) = \int_x(x-\mathbb{E}[X])^2p(x)dx$
$\text{Var}(X) = \mathbb{E}[X^2] - (\mathbb{E}[X])^2$
$\text{Cov}[X, Y] = \mathbb{E}(X - \mathbb{E}X)(Y - \mathbb{E}Y)$
$\text{Cov}[X, Y] = \int_x\int_y p(x, y)(x-\mu_x)(y-\mu_y)dxdy$

Jensen's inequality

X:random variable & φ:convex function $\rightarrow$ $\varphi(\mathbb{E}[X]) \leq \mathbb{E}[\varphi(X)]$

15 Model Selection

Bootstrapping

Sample creation with replacement. Calculate mean and var of it and average.

$\overline{S} = \frac{1}{B}\sum S(Z*)$

$\sigma^2(S) = \frac{1}{B-1}(S(Z^*) - \overline{S})^2$

Bootstrapping works if for $n \to \infty$ the error of empirical&bootstrap is the same as real&empiricial

Probability for a sample not to appear in set: $(1-\frac{1}{n})^n$. Goes to $\frac{1}{e}$ for $n \to \infty$

Multiplicity N sample to choose k times with replacement: $\binom{N-1+k}{k}$. In bootstrapping $N = k$

15.1 Jackknife

Method for debiasing at the price of variance $\overline{\theta}_{\text{Jack}} = \frac{1}{n}\sum_{i=1}^{n}(\overline{\theta}_i)$ (leave out ith sample, estimate and average)

16 Ensemble Methods

Use combination of simple hypotheses (weak learners) to create one strong learner.
strong learners: minimum error is below some $\delta < 0.5$
weak learner: maximum error is below 0.5

$$f(x) = \sum_{i=1}^{n} \beta_i h_i(x) \tag{1}$$

Bagging: train weak learners on bootstrapped sets with equal weights.
Boosting: train on all data, but reweigh misclassified samples higher.

Decision Trees

Stumps: partition linearly along 1 axis
$h(x) = sign(ax_i - t)$
Decision Tree: recursive tree of stumps, leaves have labels. To train, either label if leaf's data is pure enough, or split data based on score.

Ada Boost

Effectively minimize exponential loss.
$f^*(x) = \arg\min_{f \in F}\sum_{i=1}^{n}\exp(-y_i f(x_i))$
Train m weak learners, greedily selecting each one

$$(\beta_i, h_i) = \arg\min_{\beta,h}\sum_{i=1}^{n}\exp(-y_i(f_{i-1}(x_j) + \beta h(x_j)))$$

$c_b(x)$ trained with w_i

$$\epsilon_b = \sum_{i}^{n} \frac{w_i^b}{\sum_i^n w_i^b} I_{c(x_i)\neq y_i}$$

$\alpha_b = log\frac{1-\epsilon_b}{\epsilon_b}$

$w_i^{b+1} = w_i^b \cdot exp(\alpha_b I_{y_i \neq c_b(x_i)})$

Exponential loss function
Additive logistic regression
Bayesian approached (assumes posteriors)
Newtonlike updates (Gradient Descent)
If previous classifier bad, next has heigh weight

17 Generative Methods

Discriminative - estimate $P(y|x)$ - conditional.
Generative - estimate $P(y,x)$ - joint, model data generation.

Naive Bayes

All features independent.
$P(y|x) = \frac{1}{Z}P(y)P(x|y), Z = \sum_y P(y)P(x|y)$
$y = \arg\max_{y'} P(y'|x) = \arg\max_{y'} \hat{P}(y')\prod_{i=1}^{d}\hat{P}(x_i|y')$

Discriminant Function
$f(x) = \log(\frac{P(y=1|x)}{P(y==1|x)}), y = sign(f(x))$

Fischer's Linear Discriminant Analysis (LDA)

Idea: project high dimensional data on one axis.
Complexity: $\mathcal{O}(d^2 n$ with d number of classifiers
$c = 2, p = 0.5, \hat{\Sigma}_- = \hat{\Sigma}_+ = \hat{\Sigma}$
$y = sign(w^\top x + w_0)$
$w = \hat{\Sigma}^{-1}(\hat{\mu}_+ - \hat{\mu}_-)$
$w_0 = \frac{1}{2}(\hat{\mu}_-^\top \Sigma^{-1}\hat{\mu}_- - \hat{\mu}_+^\top\Sigma^{-1}\hat{\mu}_+)$

18 Unsupervised Learning

Parzen

$\hat{p}_n = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{V_n}\phi(\frac{x-x_i}{h_n})$ where $\int\phi(x)dx = 1$

K-NN

$\hat{p}_n = \frac{1}{V_k}$ volume with k neighbours

K-means

$L(\mu) = \sum_{i=1}^{n}\min_{j\in\{1...k\}}\|x_i - \mu_y\|_2^2$

Lloyd's Heuristic:
(1) assign each x_i to closest cluster
(2) recalculate means of clusters.
Iteration over (repeated till stable):
Step 1: $\text{argmin}_c\|x - \mu_c\|^2$

Step 2: $\mu_\alpha = \frac{1}{n_\alpha}\sum\vec{x}$

19 Neural Networks

Learning features

Parameterize the feature maps and optimize over the parameters:
$w^* = \underset{w,\Theta}{\text{argmin}}\sum_{i=1}^{n} l(y_i, \sum_{j=1}^{m} w_j \Phi(x_i, \Theta_j))$

20 Hidden-Markov model

State only depends on previous state.
Always given: sequence of symbols $\vec{s} = \{s_1, s_2, \ldots s_n\}$

Evaluation (Forward & Backward)

Known: $a_{ij}, e_k(s_t)$
Wanted: $P(X = x_i|S = s_t)$

$$f_l(s_{t+1}) = e_l(s_{t+1})\sum f_k(s_t)a_{kl} \tag{2}$$

$$b_l(s_t) = e_l(s_t)\sum b_k(s_{t+1})a_{lk} \tag{3}$$

$$P(\vec{s}) = \sum_k f_k(s_n)a_k \cdot \text{ end} \tag{4}$$

$$P(x_{l,t}|\vec{s}) = \frac{f_l(s_t)b_l(s_t)}{P(\vec{s})} \tag{5}$$

Complexity in time: $\mathcal{O}(|S|^2 \cdot T)$

20.1 Learning (Baum-Welch)

Known: only sequence and sequence space Θ
Wanted: $a_{ij}, e_k(s_t)$ & most likely path $\vec{x} = \{x_1, x_2, \ldots x_n\}$
E-step I: $f_k(s_t), b_k(s_t)$ by forward & backward algorithm
E-step II:

$$P(X_t = x_k, X_{t+1} = x_l|\vec{s}, \Theta) = \tag{6}$$

$$\frac{1}{P(\vec{s})}f_k(s_t)a_{kl}e_l(s_{t+1})b_l(s_{t+1}) \tag{7}$$

$$A_{kl} = \sum_{j=1}^{m}\sum_{t=1}^{n} P(X_t = x_k, X_{t+1} = x_l|\vec{s}, \Theta) \tag{8}$$

M-step :

$$a_{kl} = \frac{A_{kl}}{\sum_i^n A_{ki}} \text{ and } e_k(b) = \frac{E_k(b)}{\sum_{b'}E_k(b')} \tag{9}$$

Complexity: $\mathcal{O}(|S|^2)$ in storage (space)

Reformulating the perceptron

Ansatz: $w = \sum_{j=1}^{n}\alpha_j y_j x_j$
$\min_{w\in\mathbb{R}^d}\sum_{i=1}^{n}\max[0, -y_i w^T x_i]$
$= \min_{\alpha_{1:n}}\sum_{i=1}^{n}\max[0, -y_i(\sum_{j=1}^{n}\alpha_j y_j x_j)^T x_i]$
$= \min_{\alpha_{1:n}}\sum_{i=1}^{n}\max[0, -\sum_{j=1}^{n}\alpha_j y_i y_j x_i^T x_j]$

Kernelized Perceptron

1. Initialize $\alpha_1 = \ldots = \alpha_n = 0$
2. For t do
Pick data $(x_i, y_i) \in_{u.a.r} D$
Predict $\hat{y} = sign(\sum_{j=1}^{n}\alpha_j y_j k(x_j, x_i))$
If $\hat{y} \neq y_i$ set $\alpha_i = \alpha_i + \eta_t$

Regularization

The error term L and the regularization C with regularization parameter λ: $\min_w L(w) + \lambda C(w)$
L1-regularization for number of features
L2-regularization for the length of w

Convex

g(x) is convex
$\Leftrightarrow x_1, x_2 \in \mathbb{R}, \lambda \in [0,1]:$
$g(\lambda x_1) + (1 - \lambda x_2) \leq \lambda g(x_1) + (1-\lambda)g(x_2) \Leftrightarrow g''(x) > 0$

Parametric to nonparametric linear regression

Ansatz: $w = \sum_i \alpha_i x$
Parametric: $w^* = \underset{w}{\text{argmin}}\sum_i(*Tx_i - y_i)^2 + \lambda\|w\|_2^2$
$= \underset{\alpha_{1:n}}{\text{argmin}}\sum_{i=1}^{n}(\sum_{j=1}^{n}\alpha_j x_j^T x_i - y_i)^2 + \lambda\sum_i\sum_j\alpha_i\alpha_j(x_i^T x_j)$
$= \underset{\alpha_{1:n}}{\text{argmin}}\sum_{i=1}^{n}(\alpha^T K_i - y_i)^2 + \lambda\alpha^T K\alpha$
$= \underset{\alpha}{\text{argmin}}\|\alpha^T K - y\|_2^2 + \lambda\alpha^T K\alpha$
Closed form: $\alpha^* = (K + \lambda I)^{-1}y$
Prediction: $y^* = w^{*T}x = \sum_{i=1}^{n}\alpha_i^* k(x_i, x)$