MLT

WEEKS 1-4

① Best-Fit line : $\max_{w: \|w\|^2=1} w^T C w$; where $C = 1/n \sum x_i x_i^T$

↳ Residue might have signal Replace x_i with $x_i - (x_i^T w_1) w_1$

② To reduce time complexity of PCA :

① Compute eigendecomposition of $X^T X$

↳ eigenvectors $\{\beta_1, \dots, \beta_k\}$ corresponding to eigenvalues $\{n\lambda_1, \dots, n\lambda_k\}$

② Set $\alpha_k = \frac{\beta_k}{\sqrt{n\lambda_k}}$ $\forall k$

③ $w_k = X\alpha_k$ $\forall k$

③ Kernel Functions :

→ polynomial : $k(x_1, x_2) = (x_1^T x_2 + 1)^p$, for some $p \geq 1$ and $x \in \mathbb{R}^d$

→ gaussian : $k(x_1, x_2) = \exp\left(\frac{-\|x - x'\|^2}{2\sigma^2}\right)$, for some $\sigma > 0$

→ linear : $k(x_1, x_2) = x_1^T x_2$

④ Kernel Functions : Mercer's Theorem

↳ $k: \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is valid kernel iff:

① k is symmetric i.e., $k(x, x') = k(x', x)$

② The matrix $K \in \mathbb{R}^{n \times n}$ where $k_{ij} = k(x_i, x_j)$ is positive semi-definite

⑤ Kernel PCA $k: \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$

① Compute $K \in \mathbb{R}^{n \times n}$, where $k_{ij} = k(x_i, x_j)$ $\forall i, j$

② Compute $\{\beta_1, \dots, \beta_l\}$ eigenvectors and $\{n\lambda_1, \dots, n\lambda_l\}$ eigenvalues of K and normalise to get $\alpha_u = \frac{\beta_u}{\sqrt{n\lambda_u}}$

③ Compute $\sum_{j=1}^{n} \alpha_{kj} K_{ij}$ $\forall k$

$\underset{\in \mathbb{R}^d}{x_i} \to \left[\sum_{j=1}^{n} \alpha_{1j} K_{ij}, \sum_{j=1}^{n} \alpha_{2j} K_{ij}, \dots, \sum_{j=1}^{n} \alpha_{lj} K_{ij}\right]$

⑥ K-means Algorithm

① Initialize by assigning datapoints to k-clusters.

Loss Function : $\sum_{i=1}^{n} \|x_i - \mu_{z_i}\|_2^2$

② Compute means : $\forall k$ $\mu_k = \frac{\sum x_i \mathbb{1}(z_i^t = k)}{\sum \mathbb{1}(z_i^t = k)}$

③ Reassign : $\forall i$ $z_i^{t+1} = \arg\min_k \|x_i - \mu_k^t\|_2^2$

K-means++

↳ initialize μ_i randomly then assign scores based on how "Far" the datapoints are from the selected mean. $S(x) = \min_{j=1,\dots,l} \|x - \mu_j\|^2$ $\forall x$

⑦ Gaussian Mixture Models

↳ Steps :

① Pick which mixture a datapoint comes from.

$P(z_i = l) = \pi_l$, where $\sum_{i=1}^{k} \pi_i = 1$ and $0 \leq \pi_i \leq 1$ $\forall i$

② Generate a datapoint from that mixture.

$x_i \sim N(\mu_{z_i}, \sigma_{z_i}^2)$

↳ Parameters to be estimated :

→ $\pi = [\pi_1, \pi_2, \dots, \pi_k]$

→ (μ_k, σ_k^2) $\forall k$

total : $2k + k - 1 = 3k - 1$

(8) E-M algorithm (For GMM)

↳ Steps :

(1) Initialise $\theta^0 = \begin{Bmatrix} \mu_1^0, \dots, \mu_K^0 \\ \sigma_1^{2\,0}, \dots, \sigma_K^{2\,0} \\ \pi_1^0, \dots, \pi_K^0 \end{Bmatrix}$

(2) Until convergence ($\|\theta^{t+1} - \theta^t\| \le \varepsilon$):

→ Expectation step : $\lambda^{t+1} = \arg\max_{\lambda} \text{modified\_log}(\theta^t, \lambda)$

$$\lambda_k^{t+1} = \frac{\left(\frac{1}{\sqrt{2\pi}\,\sigma_k} \exp\left(\frac{-(x_i - \mu_k)^2}{2\sigma_k^2}\right)\right) \cdot \pi_k}{\sum_{l=1}^{K} \left(\frac{1}{\sqrt{2\pi}\,\sigma_l} \exp\left(\frac{-(x_i - \mu_k)^2}{2\sigma_k^2}\right) \cdot \pi_l\right)} \quad \forall k$$

→ Maximisation step : $\theta^{t+1} = \arg\max_{\theta} \text{modified\_log}(\theta, \lambda^{t+1})$

$$\mu_k^{t+1} = \frac{\sum_{i=1}^{n} \lambda_k^i x_i}{\sum_{i=1}^{n} \lambda_k^i} \qquad \sigma_k^{2\,t+1} = \frac{\sum_{i=1}^{n} \lambda_k^i (x_i - \mu_k^{t+1})^2}{\sum_{i=1}^{n} \lambda_k^i} \qquad \pi_k^{t+1} = \frac{\sum_{i=1}^{n} \lambda_k^i}{n} \quad \forall k$$

WEEKS 5-8

(9) Regression

→ Given $\{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$, where $x_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$

$$\min_{w \in \mathbb{R}^d} \sum_{i=1}^{n} (w^T x_i - y_i)^2$$

$$\min_{W \in \mathbb{R}^d} \|W^T X - Y\|_2^2 \qquad X = \underset{d \times n}{\begin{bmatrix} | & | & & | \\ x_1 & x_2 & \dots & x_n \\ | & | & & | \end{bmatrix}} \qquad W = \underset{d \times 1}{\begin{bmatrix} w_1 \\ \vdots \\ w_d \end{bmatrix}} \qquad Y = \underset{n \times 1}{\begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}}$$

$$W^* = (XX^T)^+ (XY)$$

→ $X^T W^*$ is the projection of labels (Y) onto the subspace spanned by the Features.

(10) Gradient Descent For Regression : $W^{t+1} = W^t - \eta[(XX^T)W^t - XY]$

→ Stochastic Gradient if n is large :

↳ Sample k datapoints uniformly at random

↳ After T rounds , use $W^* = \frac{1}{T} \sum_{t=1}^{T} W^t$ i.e., average of all W^t

(11) Kernel Regression

→ $W^* = X\alpha^*$, where $\alpha^* = (X^T X)^{-1} Y$

→ in the case of kernel mapping :

$$W^* = \phi(X)\alpha^*$$

→ $y_{test} = W^* \phi(x_{test})$

$y_{test} = \sum_{i=1}^{n} \alpha_i^* \, k(x_i, x_{test})$ $\qquad \hat{Y} = K^T \alpha^*$ $\qquad$ where $K = (X^T X + 1)^p$ or some other kernel Function

(12) ML estimator For Linear Regression

→ $P(y|x) = w^T x + \varepsilon$, where $\varepsilon \sim N(0, \sigma^2)$

$y|x \sim N(w^T x, \sigma^2)$

$E[\|\hat{W}_{ML} - W\|^2] = \sigma^2 \cdot \text{trace}((XX^T)^+)$

→ eigenvalues of $XX^T = \{\lambda_1, \dots, \lambda_d\}$

eigenvalues of $(XX^T)^{-1} = \{1/\lambda_1, \dots, 1/\lambda_d\}$

→ MSE of $\hat{W}_{ML} = \sigma^2 \left(\sum_{i=1}^{d} 1/\lambda_i \right)$

(13) Cross validation

→ Using the estimator : $\hat{W} = (XX^T + \lambda I)^{-1} XY$

↳ this has lesser MSE than $\hat{W}_{ML}$

because $\text{trace}((XX^T + \lambda I)^{-1}) = \sum_{i=1}^{n} \frac{1}{\lambda_i + \lambda}$

→ K-Fold cross validation : leave one fold for test and train on the rest for optimal λ.

(14) Bayesian Modelling For Linear Regression

→ Likelihood: $y|x \sim N(W^T x, 1)$

Prior: $W \sim N(0, \gamma^2 I)$ → $\begin{bmatrix} \gamma^2 & & 0 \\ & \ddots & \\ 0 & & \gamma^2 \end{bmatrix}_{d \times d}$

→ $\hat{W}_{MAP} = (XX^T + \frac{1}{\gamma^2} I)^{-1} XY$

(15) Ridge Regression: $\hat{W}_R = \underset{W}{\arg\min} \sum_{i=1}^{n} (W^T x_i - y_i)^2 + \lambda \|W\|_2^2$, where $\|W\|_2^2 = \sum_i^n (W_i)^2$ $\hat{W}_R = (XX^T + \lambda I)^{-1} XY$

(16) Lasso Regression: $\hat{W}_L = \underset{W}{\arg\min} \sum_{i=1}^{n} (W^T x_i - y_i)^2 + \lambda \|W\|_1$, where $\|W\|_1 = \sum_i^n |W_i|$

(17) K-NN Algorithm

→ $\hat{y}_{test} = \text{majority}(y_1^*, y_2^*, \ldots, y_k^*)$, where $\{y_1^*, y_2^*, \ldots, y_k^*\}$ correspond to $\{x_1^*, x_2^*, \ldots, x_k^*\}$ → k-nearest neighbours to x_{test}.

→ choosing k: cross-validate

(18) Decision Trees

→ "Impurity" for a set of labels $\{y_1, \ldots, y_n\}$:

$\text{Entropy}(\{y_1, \ldots, y_n\}) = \text{Entropy}(p) = -(p \cdot \log(p) + (1-p) \cdot \log(1-p))$

→ Goodness of a question $f_k \le \theta$

$\text{Information Gain}(f_k, \theta) = \text{Entropy}(D) - [\gamma \, \text{Entropy}(D_{yes}) + (1-\gamma)\, \text{Entropy}(D_{no})]$

, where $\gamma = \frac{|D_{yes}|}{|D|}$

$-(0.3 \log(0.3) + 0.7 \log(0.7))$

$-0.1569 - 0.1084$

(19) Classification Modelling
① Generative: $P(x, y)$
② Discriminative: $P(y|x)$

(20) Generative model-based

→ # of parameters to estimate: $(2)^d - 1$

Assuming features are orthogonal: $2d + 1$

→ steps of algorithm:

① $P(y=1) = p$

② $P(x = [f_1, f_2, \ldots, f_d] | y) = \prod_{i=1}^{d} (p_i^y)^{f_i} (1 - p_i^y)^{1-f_i}$

(21) Naive Bayes Algorithm

→ Data: $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$, where $x \in \{0,1\}^d$, $y \in \{0,1\}$

ML estimates:

→ $\hat{p} = \frac{1}{n} \sum_{i=1}^{n} y_i$ → $\hat{p}_j^y = \dfrac{\sum_{i=1}^{n} \mathbb{1}(f_j^i = 1, y_i = y)}{\sum_{i=1}^{n} \mathbb{1}(y_i = y)}$

→ Prediction:

$P(y_{test} = 1 | x_{test}) \propto P(x_{test} | y_{test} = 1) \cdot P(y_{test} = 1)$

$P(y_{test} = 1 | x_{test}) \propto \left[\prod_{j=1}^{d} (\hat{p}_j^1)^{f_j} (1 - \hat{p}_j^1)^{(1-f_j)}\right] \cdot \hat{p}$

$P(y_{test} = 0 | x_{test}) \propto \left[\prod_{j=1}^{d} (\hat{p}_j^0)^{f_j} (1 - \hat{p}_j^0)^{(1-f_j)}\right] \cdot (1 - \hat{p})$

(22) Decision Function of Naive Bayes:

→ $y_{test} = 1$ if $W^T x_{test} + b \ge 0$, where $w_i = \log\left(\dfrac{\hat{p}_i^1 (1 - \hat{p}_i^0)}{\hat{p}_i^0 (1 - \hat{p}_i^1)}\right)$ and $b = \log\left(\dfrac{(1 - \hat{p}_i^1)}{(1 - \hat{p}_i^0)}\right) + \log\left(\dfrac{\hat{p}}{(1 - \hat{p})}\right)$

23 Gaussian Naive Bayes

→ Data : $\{(x_1,y_1),\dots,(x_n,y_n)\}$ $x \in \mathbb{R}^d$, $y \in \{0,1\}$

$x|y=1 \sim N(\mu_1, \Sigma)$ $x|y=0 \sim N(\mu_0, \Sigma)$

ML estimates :

→ $\hat{p} = \frac{1}{n}\sum_{i=1}^{n} y_i$ → $\hat{\mu}_0 = \frac{\sum_{i=1}^{n} \mathbb{1}(y=0)\cdot x_i}{\sum_{i=1}^{n} \mathbb{1}(y=0)}$ } → same for $\hat{\mu}_1$ → $\hat{\Sigma} = \frac{1}{n}\sum_{i=1}^{n}(x_i-\hat{\mu}_{y_i})(x_i-\hat{\mu}_{y_i})^T$

→ Prediction :

$y_{test}=1$ if $P(x_{test};\hat{\mu}_1,\hat{\Sigma})\cdot\hat{p} \geq P(x_{test};\hat{\mu}_0,\hat{\Sigma})\cdot(1-\hat{p})$

$\exp(-(x_{test}-\hat{\mu}_1)^T\hat{\Sigma}(x_{test}-\hat{\mu}_1))\cdot\hat{p} \geq \exp(-(x_{test}-\hat{\mu}_0)^T\hat{\Sigma}(x_{test}-\hat{\mu}_0))\ (1-\hat{p})$

$\Rightarrow (\hat{\mu}_1-\hat{\mu}_0)^T\hat{\Sigma}^{-1}x_{test} + \hat{\mu}_0^T\hat{\Sigma}^{-1}\hat{\mu}_0 - \hat{\mu}_1^T\hat{\Sigma}^{-1}\hat{\mu}_1 + \log\left(\frac{(1-\hat{p})}{\hat{p}}\right) \geq 0$

WEEKS 9-12

24 Perceptron Learning Algorithm

→ Data: $\{(x_1,y_1),\dots,(x_n,y_n)\}$; $x_i \in \mathbb{R}^d$, $y_i \in \{\pm 1\}$

Initialize : $W^0 = 0 \in \mathbb{R}^d$

Steps :

If $\text{sign}(W^{t^T}x_i) = y_i$, do nothing

Else $W^{t+1} = W^t + x_i y_i$

→ Assumptions :

① Linear Seperability with γ - margin :

$(W^T x_i)y_i \geq \gamma \ \forall i$, where $\gamma > 0$

② Radius :

$\|x_i\|_2 \leq R \ \forall i$, where $R > 0$

③ $\|W^*\| = 1$

→ $l \leq \frac{R^2}{\gamma^2}$, where l = # of mistakes , R= radius that bounds all points.

γ = margin that seperates the data .

25 Logistic Regression

→ Data : $\{(x_1,y_1),\dots,(x_n,y_n)\}$; $x_i \in \mathbb{R}^d$, $y_i \in \{0,1\}$

Model : $P(y=1|x) = g(W^T x)$, where $g(z) = \frac{1}{1+e^{-z}}$

→ Use gradient ascent .

$W_{t+1} = W_t + \eta_t \nabla \log(W)$

$W_{t+1} = W_t + \eta_t\left[\sum_{i=1}^{n} x_i + (y_i - g(W_t^T x_i))\right]$

26 Maximising Perceptron Margin

→ $\text{width}(W) = \min_z \frac{1}{2}\|x-z\|^2$ s.t. $W^T z = 1$, where $W^T x = -1$

$\text{width}(W) = \frac{2}{\|W\|^2}$; maximising width : $\max_W \frac{2}{\|W\|^2}$ s.t $(W^T x_i)y_i \geq 1 \ \forall i$

→ main problem : $\min_W \frac{\|W\|^2}{2}$ s.t. $(W^T x_i)\cdot y_i \geq 1 \ \forall i$

$W^* = XY\alpha = \sum_{i=1}^{n} \alpha_i (x_i y_i)$

Dual problem : $\max_{\alpha \geq 0} \ \alpha^T \mathbf{1} - \frac{1}{2}(XY\alpha)^T(XY\alpha)$

27 Support Vector Machine

→ Complimentary Slackness : if $\alpha_i > 0 \Rightarrow (W^{*T}x_i)y_i = 1$

→ Prediction :

$y_{test} = W^{*T}x_{test} = \sum_i \alpha_i^* \cdot y_i \cdot (x_i^T \cdot x_{test})$

Kernel : $y_{test} = W^{*T}\phi(x_{test}) = \sum_i \alpha_i^* \cdot y_i \cdot k(x_i^T, x_{test})$

28 Soft-Margin SVM

→ $\min_{w,\varepsilon} \frac{1}{2}\|W\|^2 + C\sum_i \varepsilon_i$ s.t. $(W^T x_i)\cdot y_i + \varepsilon_i \geq 1$, $\varepsilon_i \geq 0 \ \forall i$

→ Dual Problem: $\max_{0\leq\alpha\leq C} \alpha^T 1 - \frac{1}{2}(XY\alpha)^T(XY\alpha)$

→ Complimentary Slackness:

Dual:

① $\alpha_i^* = 0 \quad \Rightarrow \quad W^{*T} x_i y_i \geq 1$

② $\alpha_i^* \in (0, C) \quad \Rightarrow \quad W^{*T} x_i y_i = 1$

③ $\alpha_i^* = C \quad \Rightarrow \quad W^{*T} x_i y_i \leq 1$

29 Bagging

→ $D_1, D_2, \dots, D_m$

$\downarrow \ \downarrow \ \dots \ \downarrow$

$h_1 \ \ h_2 \ \dots \ h_m$

$h_i : \mathbb{R}^d \to \{\pm 1\}$

$h^*(x) = \text{sign}\left(\frac{1}{m}\sum_{i=1}^{m} h_i(x)\right)$

→ Creating m-different datasets with bootstrapping:

sampling with replacent.

P(a point doesn't get picked) = $1 - 1/n$

P(a point appears in any bag) = $1 - (1 - 1/n)^n \approx 67\%$

30 Boosting

→ $S = \{(x_1, y_1), \dots, (x_n, y_n)\}$ $\quad x_i \in \mathbb{R}^d \quad y_i \in \{\pm 1\}$

$D_0(i) = 1/n$ → weights

→ Algorithm:

For $t = 1, \dots, T$:

① h_t = Input (S, D_t) to a weak learner

② $\alpha_t = \ln \sqrt{\frac{1 - err(h_t)}{err(h_t)}}$

③ $\tilde{D}_{t+1}(i) = \begin{cases} D_t(i)\cdot e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i \\ D_t(i)\cdot e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \end{cases}$

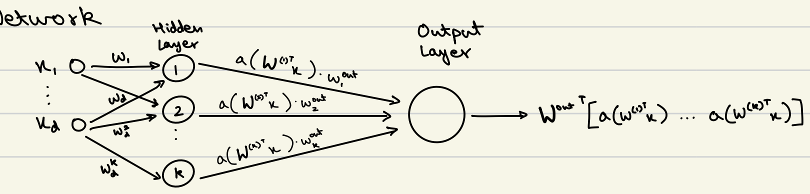
④ $D_{t+1}(i) = \frac{\tilde{D}_{t+1}(i)}{\sum_j \tilde{D}_{t+1}(j)}$

Prediction:

$h^*(x) = \text{sign}\left(\sum_t \alpha_t h_t(x)\right)$

31 Neural Network

→



Parameters: $\{w^{(1)}, \dots, w^{(k)}\}$ $w^{(i)} \in \mathbb{R}^d$, $w^{out} \in \mathbb{R}^k$

Prediction:

$\hat{y} = \sum_{i=1}^{k} w_i^{out}\, a(w^{(i)T} x)$

activation function