Binary Linear Classifier

Dataset $\mathcal{D}=\{(\underline{x_1},y_1),(\underline{x_2},y_2),\ldots,(\underline{x_N},y_N)\}$, $x_i\in\mathbb{R}^d$, $y\in\{-1,+1\}$

Weight Vector $\underline{w}=(w_1,w_2,\ldots,w_d)\in\mathbb{R}^d$ Constant $b\in\mathbb{R}$ Given Input $\underline{x}=(x_1,x_2,\ldots,x_d)\in\mathbb{R}^d$

Compute $\sum_{i=1}^{d} w_i x_i > b \longrightarrow \hat{y}=+1$ $\quad \sum_{i=1}^{d} w_i x_i < b \longrightarrow \hat{y}=-1$

Model Parameters $\Omega=\{\underline{w},b\}$

Training

Loss Function $E_{in}(\Omega)=$ Average # Of Miss Classified Points $=\frac{1}{N}\sum_{i=1}^{N}\mathbb{1}\{y_i\neq\hat{y}_i\}$ — Indicator Function; y_i (True Label) $\hat{y}_i$ (Predicted Label)

Optimal Model Parameters $\Omega^*=\underset{\Omega}{\operatorname{argmin}}\,E_{in}(\Omega)$

$\underline{x}=(x_0=1,x_1,x_2,\ldots,x_d)\in\mathbb{R}^{d+1}$

$\underline{w}=(w_0=-b,w_1,w_2,\ldots,w_d)\in\mathbb{R}^{d+1}$

$\underline{w}^T\underline{x}=\sum_{i=0}^{d}w_ix_i=w_0x_0+\sum_{i=1}^{d}w_ix_i \;\begin{matrix}>0, & \hat{y}=+1\\ <0, & \hat{y}=-1\end{matrix}$ Decision Rule $\hat{y}=h_{\underline{w}}(\underline{x})=\mathrm{sign}(\underline{w}^T\underline{x})$

Perceptron Learning Algorithm (Learning Rate Of 1)

Input: Linearly Separable Dataset $\mathcal{D}$, Output: $\underline{w}\in\mathbb{R}^{d+1}$, $E_{in}(\underline{w})=0$

Initialize $\underline{w}$ in an arbitrary fashion, $\underline{w}=(w_0=0,w_1=0,\ldots,w_d=0)\in\mathbb{R}^{d+1}$

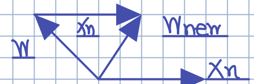
Step 1: Check If $E_{in}(\underline{w})=0$, If YES, Output $\underline{w}$

Step 2: Let $(\underline{x_n},y_n)$ be a misclassified point in $\mathcal{D}$, If $(\underline{x_n},y_n)$ is on the boundary, treat it as misclassified

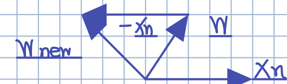
If $y_n=+1$ then $\underline{w}\longleftarrow\underline{w}+\underline{x_n}$ $\quad$ If $y_n=-1$ then $\underline{w}\longleftarrow\underline{w}-\underline{x_n}$ $\quad$ Go To Step 1

Geometric Intuition

$(\underline{x_n},y_n)$ is misclassified, $y_n=+1$, $\hat{y}_n=-1$, $\underline{w}^T\underline{x_n}\leq 0$



$y_n=-1$, $\hat{y}_n=+1$, $\underline{w}^T\underline{x_n}\geq 0$



Pocket Algorithm

Idea: At each iteration, keep aside the "best weight vector", Run PLA sufficiently many iterations

Linear Regression

Data Matrix $(N\times(d+1))$ $\mathcal{X}=\begin{bmatrix}\underline{x_1}^T\\ \vdots\\ \underline{x_N}^T\end{bmatrix}$ Observation Vector $(N\times 1)$ $\underline{y}=\begin{bmatrix}y_1\\ \vdots\\ y_N\end{bmatrix}$ Weight Vector $((d+1)\times 1)$ $\underline{w}=\begin{bmatrix}w_0\\ \vdots\\ w_d\end{bmatrix}$ Prediction Vector $(N\times 1)$ $\hat{\underline{y}}=\begin{bmatrix}\hat{y}_1\\ \vdots\\ \hat{y}_N\end{bmatrix}$

$\hat{y}_i=\underline{x_i}^T\underline{w}$

$\hat{\underline{y}}=\mathcal{X}\underline{w}$

$y_i=x_{i0}w_0+x_{i1}w_1+\ldots+x_{id}w_d$

Loss Function $E_{in}(\underline{w})=\|\underline{y}-\hat{\underline{y}}\|^2=\|\underline{y}-\mathcal{X}\underline{w}\|^2$

$\underline{w}^*=\underset{\underline{w}\in\mathbb{R}^{d+1}}{\operatorname{argmin}}\,E_{in}(\underline{w})=(\mathcal{X}^T\mathcal{X})^{-1}\mathcal{X}^T\underline{y}$

$\downarrow$

Least Square Solution

Regularized Least Square

$\underset{\underline{w}\in\mathbb{R}^{d+1}}{\operatorname{argmin}}\{\|\underline{y}-\mathcal{X}\underline{w}\|^2+\lambda\|\underline{w}\|^2\}=\underline{w}_\lambda^*$ — Regularization Coefficient

$\underline{w}_\lambda^*=(\mathcal{X}^T\mathcal{X}+\lambda)^{-1}\mathcal{X}^T\underline{y}$

Polynomial Fitting, $x\longrightarrow\underline{z}=(1,x,x^2,\ldots,x^d)$

$\nabla f(\underline{w})=\left[\frac{\partial f(\underline{w})}{\partial w_1}\;\cdots\;\frac{\partial f(\underline{w})}{\partial w_d}\right]^T\in\mathbb{R}^d$ $\quad\nabla_{\underline{w}}(\underline{w}^TA\underline{v})=A\underline{v}$ $\quad\nabla_{\underline{w}}(\underline{v}^TA\underline{w})=(\underline{v}^TA)^T=A^T\underline{v}$ $\quad\nabla_{\underline{w}}(\underline{w}^TA\underline{w})=(A+A^T)\underline{w}$

Logistic Regression

$P_r(y=1|\underline{x})=\theta(s)=\theta(\underline{w}^T\underline{x})$

Sigmoid Activation Function

$\theta(s)=\frac{e^s}{1+e^s}$ (Non Linear)

$\hat{P}_{\underline{w}}(1|\underline{x})=\frac{e^{\underline{w}^T\underline{x}}}{1+e^{\underline{w}^T\underline{x}}}$ $\quad\hat{P}_{\underline{w}}(-1|\underline{x})=\frac{e^{-\underline{w}^T\underline{x}}}{1+e^{-\underline{w}^T\underline{x}}}$ $\quad\hat{P}_{\underline{w}}(y|\underline{x})=\frac{e^{y\underline{w}^T\underline{x}}}{1+e^{y\underline{w}^T\underline{x}}}$

Input $\mathcal{D}$, Model Parameter $\underline{w}\in\mathbb{R}^{d+1}$, Output $[\hat{P}_{\underline{w}}(1|\underline{x}),\hat{P}_{\underline{w}}(-1|\underline{x})]$

Log Loss Function, $-\log\hat{P}_{\underline{w}}(y|\underline{x})$, $e_n(\underline{w})=-\log\hat{P}_{\underline{w}}(y_n|\underline{x_n})=\log(1+e^{-y_n\underline{w}^T\underline{x_n}})$

$E_{in}(\underline{w})=\frac{1}{N}\sum_{n=1}^{N}e_n(\underline{w})=\frac{1}{N}\sum_{n=1}^{N}\log(1+e^{-y_n\underline{w}^T\underline{x_n}})$ $\quad\underline{w}^*=\underset{\underline{w}\in\mathbb{R}^{d+1}}{\operatorname{argmin}}\,E_{in}(\underline{w})$ $\quad\underline{w}_\lambda^*=\underset{\underline{w}\in\mathbb{R}^{d+1}}{\operatorname{argmin}}\,(E_{in}(\underline{w})+\lambda\|\underline{w}\|^2)$

Maximum Likelihood Viewpoint

$P_r(y_1y_2\ldots y_N|\underline{x_1}\underline{x_2}\ldots\underline{x_N})=\prod_{n=1}^{N}P_r(y_n|\underline{x_n})$ Independent Event

$\underline{w}^*=\underset{\underline{w}\in\mathbb{R}^{d+1}}{\operatorname{argmax}}\prod_{n=1}^{N}P_r(y_n|\underline{x_n})=\underset{\underline{w}\in\mathbb{R}^{d+1}}{\operatorname{argmax}}\log\left(\prod_{n=1}^{N}P_r(y_n|\underline{x_n})\right)=\underset{\underline{w}\in\mathbb{R}^{d+1}}{\operatorname{argmax}}\sum_{n=1}^{N}\log(P_r(y_n|\underline{x_n}))=\underset{\underline{w}\in\mathbb{R}^{d+1}}{\operatorname{argmin}}\,E_{in}(\underline{w})$

Cross Entropy Viewpoint

$S=\{S_1,S_2,\ldots,S_M\}$, $P=(p(S_1),p(S_2),\ldots,p(S_M))$, $Q=(q(S_1),q(S_2),\ldots,q(S_M))$

$CE(P,Q)=-\sum_{i=1}^{M}p(S_i)\log q(S_i)$ $\quad e_n(\underline{w})=-\{\mathbb{1}\{y_n=+1\}\log\hat{P}_{\underline{w}}(1|\underline{x_n})+\mathbb{1}\{y_n=-1\}\log\hat{P}_{\underline{w}}(-1|\underline{x_n})\}$

$P_n=(\mathbb{1}\{y_n=+1\},\mathbb{1}\{y_n=-1\})$ $\quad Q_n=(\log\hat{P}_{\underline{w}}(1|\underline{x_n}),\log\hat{P}_{\underline{w}}(-1|\underline{x_n}))$ $\quad E_{in}(\underline{w})=\frac{1}{N}\sum_{n=1}^{N}CE(P_n,\hat{P}_{\underline{w}}(\underline{x_n}))$

Knowledge Distillation

Transfer knowledge from a large model to a smaller one

Training For Logistic Regression (No Analytical Solution) (Gradient Descent)

If $x<x^*$, $f(x)$ is decreasing, $f'(x)<0$

$x>x^*$, $f(x)$ is increasing, $f'(x)>0$

1. Initialize $x=x_0$
2. If $f'(x)\approx 0$, stop and output x
3. If $f'(x)>0$, $x=x-\varepsilon$
4. If $f'(x)<0$, $x=x+\varepsilon$
5. Go to step 2

ε (Step Size) (eg 10^{-3})

$x_1 = x_0 + \delta u \quad \|u\| = 1 \quad u^* = \underset{u \in \mathbb{R}^2}{\arg\min} f(x_0 + \delta u)$ Taylor Approximation $f(x_0 + \delta u) \approx f(x_0) + \delta u^T \nabla f(x_0)$

$u^* = \frac{-\nabla f(x_0)}{\|f(x_0)\|}$, $x_t = x_{t-1} + \delta_t u^*$ $\quad x_t = x_{t-1} - \varepsilon_t \nabla f(x_{t-1})$ $\quad (\text{Step Size})_t \propto \varepsilon_t \|\nabla f(x_{t-1})\|$

Gradient Descent Algorithm Initialize x_0 in some arbitrary fashion, $t = 0, 1, 2, \ldots$, Compute $g_t = \nabla f(x_t)$

Select Direction $u_t = -g_t$, Update $x_{t+1} = x_t + \varepsilon_t u_t$, Continue until $\|\nabla f(x_t)\| \approx 0$

Linear Regression $\nabla E_n(w_t) = \frac{1}{N}\sum_{n=1}^{N} \nabla e_n(w)$ $\quad \nabla_w e_n(w) = 2(w^T x_n - y_n)x_n$ $\quad w_{t+1} = w_t - \varepsilon_t \frac{2}{N}\sum_{n=1}^{N}(w^T x_n - y_n)x_n$ $\quad O(Nd)$

w_t will converge to w_{LS}^*

Stochastic Gradient Descent $w_{t+1} = w_t - \varepsilon_t \nabla e_n(w_t)$ $\quad n \in \text{Uniform}\{1, 2, \ldots, N\}$

Mini Batch Gradient Descent Draw M examples at random without replacement $w_{t+1} = w_t - \frac{\varepsilon_t}{M}\sum_{n=1}^{M} \nabla e_{(n)}(w_t)$

SGD Update Rule $w_{k+1} = w_k + \varepsilon_k \frac{y_n x_n}{1 + e^{y_n w_k^T x_n}}$ (Logistic Regression)

Multiple Classes Logistic Regression $\Omega = \{w(1), w(2), w(3)\}$ $\quad [\hat{P}_\Omega(1|x), \hat{P}_\Omega(2|x), \hat{P}_\Omega(3|x)]$

$\hat{P}_\Omega(i|x) = \frac{e^{w^T(i)x}}{e^{w^T(1)x} + e^{w^T(2)x} + e^{w^T(3)x}}$ $\quad e_n(\Omega) = -\log \hat{P}_\Omega(y_n|x_n)$ $\quad w(1) - w(2) = w$

$\Omega_{k+1} = \Omega_k - \varepsilon_k \nabla_{\Omega_k} e_n(\Omega_k)$

SGD + Momentum Velocity Vector $v_0 = 0$, $g_t = \nabla e_n(w_t)$, $v_t = -\varepsilon_t g_t + \mu v_{t-1}$, $w_{t+1} = w_t + v_t$ (Momentum Coefficient: μ)

Nestrov Momentum $v_t = \mu v_{t-1} - \varepsilon_t \nabla e_n(w_t + \mu v_{t-1})$, $w_{t+1} = w_t + v_t$ $\quad$ GD: $O(1/k)$, Nestrov: $O(1/k^2)$

AdaGrad $g_t = \nabla e_n(w_t) \in \mathbb{R}^d$, $v_t = v_{t-1} + g_t \odot g_t$, $v_t(j) = \sum_{q=1}^{t} g_q^2(j)$ $\quad w_{t+1}(j) = w_t(j) - \frac{\varepsilon_t}{\sqrt{v_t(j)}} g_t(j)$

RMS — PROP $v_t = \rho v_{t-1} + (1-\rho) g(t) \odot g(t)$ $\quad w_{t+1} = w_t - \frac{\varepsilon_t}{\sqrt{v_t}} \odot g(t)$

ADAM $v_t = \mu v_{t-1} - \frac{\varepsilon_t}{\sqrt{v_t}} \odot g(t)$ $\quad w_{t+1} = w_t + v_t$

For Classification, $\theta \rightarrow$ Sigmoid, For Regression, $\theta \rightarrow$ Identity Function $\quad \hat{y}_1 = OR(h_1, h_2)$ $\quad \hat{y}_2 = AND(h_1, h_2)$

$OR(h_1, h_2) = \text{sign}(h_1 + h_2 + 1.5)$ $\quad AND(h_1, h_2) = \text{sign}(h_1 + h_2 - 1.5)$ $\quad XOR(h_1, h_2) = AND(\hat{y}_1, \hat{y}_2)$ $\quad (M - 0.5)$

Layer 1, Implement c_1, c_2, c_3 to output h_1, h_2, h_3 $\quad$ Layer 2, Implement y_1, y_2, y_3 $\quad$ Layer 3, $y = OR(y_1, y_2, y_3)$

Neural Network Input Layer $l = 0$, Hidden Layers $1 \le l \le L-1$, Output Layer L, $w_{ij}^{(l)}$ weight between node i in $l-1$ and j in l

$d^{(l)}$ Number of nodes excluding bias node in layer l, $x_j^{(l)}$ Output from node j in layer l, $s_j^{(l)}$ Input to node j in layer l

$$S^{(l)} = \begin{bmatrix} s_1^{(l)} \\ \vdots \\ s_{d^{(l)}}^{(l)} \end{bmatrix} \quad \theta(S^{(l)}) = \begin{bmatrix} \theta(s_1^{(l)}) \\ \vdots \\ \theta(s_{d^{(l)}}^{(l)}) \end{bmatrix} \quad x^{(l)} = \begin{bmatrix} 1 \\ \\ s^{(l)} \end{bmatrix} \quad W^{(l)} = \left\{ w_{i,j}^{(l)} \right\} \begin{matrix} \text{Row } 0 \le i \le d^{(l-1)} \\ \text{Column } 1 \le j \le d^{(l)} \end{matrix}$$

Forward Propagation Algorithm

Input $x^{(0)} = [1 \; x_1 \; x_2 \; \cdots \; x_d]^T$

for $l = 1, 2, \ldots, L$ do

$S^{(l)} = [W^{(l)}]^T x^{(l-1)}$

$x^{(l)} = \theta(S^{(l)})$

$\sum_{l=1}^{L}(d^{(l-1)} + 1)d^{(l)} + \sum_{l=1}^{L} d^{(l)}$

Output $x^{(L)}$

Backward Propagation Algorithm $\quad$ Backward Message

Input (x, y), $\Omega = \{w^{(1)}, \ldots, w^{(L)}\}$ Output $\delta^{(l)}$ for $l = 1, 2, \ldots, L$

$\frac{\partial e}{\partial w_{i,j}^{(l)}} \quad \forall i, j, l$

1. Run forward propagation to compute $\{S^{(l)}, x^{(l)}\}$ and $e(\Omega) = g(x^{(L)}, y)$

2. $\delta^{(L)} = \nabla_{x^{(L)}} g(x^{(L)}, y) \odot \theta'(S^{(L)})$

3. For $l = L-1, \ldots, 1$

$\delta^{(l)} = [W^{(l+1)} \delta^{(l+1)}]_1^{d^{(l)}} \odot \theta'(S^{(l)})$

$\frac{\partial e}{\partial w_{i,j}^{(l)}} = x_i^{(l-1)} \delta_j^{(l)}$ $\quad w_{i,j}^{(l)} \leftarrow w_{i,j}^{(l)} - \varepsilon_k \frac{\partial e}{\partial w_{i,j}^{(l)}}$

Linear Regression Minimizer Weights

$E_{in}(w) = \frac{1}{N}\|y - Xw\|^2 = \frac{1}{N}(y - Xw)^T(y - Xw)$

$= \frac{1}{N}[y^T y - y^T X w - w^T X^T y + w^T X^T X w]$

$\frac{\partial E_{in}(w)}{\partial w} = \frac{1}{N}[-X^T y - X^T y + (X^T X + X^T X)w]$

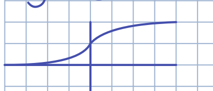
$0 = \frac{1}{N}[2X^T X w - 2X^T y]$ $\quad X^T X w = X^T y$

$w = (X^T X)^{-1} X^T y$

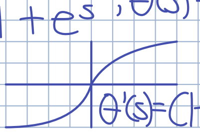
Matrix Inverse $\text{Adj}(A) = \text{Cofactor}(A)^T$

$\frac{1}{\det(A)} \text{Adj}(A)$ $\quad \text{Sign}(x) \begin{cases} -1, x \le 0 \\ 1, x > 0 \end{cases}$

Sigmoid $\theta(s) = \frac{e^s}{1 + e^s}$, $\theta'(s) = \theta(s)(1 - \theta(s))$



$\tanh(s) = \frac{e^s - e^{-s}}{e^s + e^{-s}}$



$\theta'(s) = (1 - \tanh^2(s))$

ReLU $\theta(s) = \max(0, s)$

Leaky ReLU $\theta(s) = \begin{cases} s, s \ge 0 \\ \alpha s, s < 0 \end{cases}$

Data Augmentation

Weights Initialization