Chap 9 Lineer Predicetors.

$\boxed{L_d} := \{h_{w,b} \mid w \in \mathbb{R}^d,\ b \in \mathbb{R}\}$, where

$$\boxed{h_{w,b}(x)} := \langle w, x\rangle + \underbrace{b}_{\text{bias}} = \left(\sum_{i=1}^{d} w_i x_i\right) + b,$$

is called the class of $\boxed{\text{affine functions}}$.

Rmk:

The different hypothesis classes of linear predictors are compositions of a function $\phi : \mathbb{R} \to \mathcal{Y}$ on L_d.

ex

① binary classification : $\phi(x) := \begin{cases} 1, & x > 0 \\ 0, & x = 0 \\ -1, & x < 0 \end{cases}$ this is the $\boxed{\text{sign fun}}$.

② regression : $\phi(x) = x$, the identity fun.

Rmk:

$\boxed{w'} := (b, w_1, \cdots, w_d) \in \mathbb{R}^{d+1}$, $\boxed{x'} := (1, x_1, \cdots, x_d) \in \mathbb{R}^{d+1}$.

Then $h_{w,b}(x) = \langle w, x\rangle + b = \langle w', x'\rangle$, called the $\boxed{\text{homogeneous representation}}$ of $h_{w,b}$, simplifying the representation.

9.1 Halfspaces.

$\boxed{HS_d} = \text{sign} \circ L_d := \{x \mapsto \text{sign}(h_{w,b}(x)) : h_{w,b} \in L_d\}$

called the class of $\boxed{\text{halfspace hypothesis}}$.

Rmk:

In this setting, the realizable case is often called $\boxed{\text{separable}}$. ($\leftrightarrow$ $\boxed{\text{nonseparable}}$).

Note:

Implementing ERM in the nonseparable case w.r.t. 0-1 loss is known to be computationally hard. As a substitute, some people use $\boxed{\text{surrogate loss functions}}$ instead. (as in the logistic regression)

9.1.1 Lineer Programming for the Class of Halfsp.

Linear program (LP):

$$\boxed{\begin{aligned} &\max_{w \in \mathbb{R}^d} \langle u, w\rangle \\ &\text{subject to } Aw \geq v \end{aligned}}$$

, where $A : m \times d$, $v \in \mathbb{R}^m$, $u \in \mathbb{R}^d$ are given and $w \in \mathbb{R}^d$ is to be determined.

Prop

In the realizable case, the ERM problem for halfspaces can be expressed as a linear program.

⟨pf⟩

Goal: Find w s.t. $\text{sign}(\langle w, x_i\rangle) = y_i, \forall i$.

$\iff y_i \langle w, x_i\rangle > 0, \forall i$.

$\because$ separable $\therefore \exists w^*$ s.t. $y_i \langle w^*, x_i\rangle > 0$.

Let $\gamma := \min_i y_i \langle w^*, x_i\rangle$ so that $\gamma \leq y_i \langle w^*, x_i\rangle, \forall i$.

$\Rightarrow y_i \langle w^*/\gamma, x_i\rangle \geq 1, \forall i$.

i.e. $y_i \langle \bar{w}, x_i\rangle \geq 1, \forall i$, w/ $\bar{w} = w^*/\gamma$. (*)

Note that $y_i x_i = y_i \cdot \begin{pmatrix} x_{i1} \\ \vdots \\ x_{id} \end{pmatrix}$

Thus, (*) is $y_i \cdot (x_{i1} \cdots x_{id}) \cdot \bar{w} \geq 1, \forall i$.

Let $A = \begin{pmatrix} y_1 x_{11} & y_1 x_{12} & \cdots & y_1 x_{1d} \\ \vdots & \vdots & & \vdots \\ y_d x_{d1} & y_d x_{d2} & \cdots & y_d x_{dd} \end{pmatrix}$ and $v = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$,

(*) becomes $A\bar{w} \geq v$.

Setting an arbitrary u, say $u = 0$, finding $\bar{w}$ becomes an LP problem. (#)

9.1.2 Perceptron for Halfspaces.

A different approach to implement ERM is the $\boxed{\text{Perceptron algorithm of Rosenblatt}}$.

Batch Perceptron

Input: A training set $(x_1, y_1), \cdots, (x_m, y_m)$

Initialize : $w^{(1)} = (0, \cdots, 0)$

for $t = 1, 2, \cdots$

  if ($\exists i$ s.t. $y_i \langle w^{(t)}, x_i\rangle \leq 0$): do

    $w^{(t+1)} = w^{(t)} + y_i x_i$

  else

    output $w^{(t)}$.

Idea:
The update of the Perceptron guides the sol'n to be more correct on the ith example.:

$$y_i\langle w^{(t+1)}, x_i\rangle = y_i\langle w^{(t)} + y_i x_i, x_i\rangle$$
$$= y_i\langle w^{(t)}, x_i\rangle + \|x_i\|^2.$$

Thm 9.1 (Perceptron under separable case).

$(x_1,y_1),\cdots,(x_m,y_m)$: separable.

$B := \min\{\|w\| : y_i\langle w, x_i\rangle \geq 1, \forall i\in[m]\}$.

$R := \max_i \|x_i\|$.

Then the Perceptron algorithm stops after at most $(RB)^2$ interations.

(when it stops, we have $y_i\langle w^{(t)}, x_i\rangle > 0$, $\forall i\in[m]$).

9.1.3 The VC dim of Halfspaces.

Thm 9.2 (See P122 for proof).

The VC dimension of the class of homogeneous halfsp. in $\mathbb{R}^d$ is d.

Thm 9.3 (See P122 for proof)

The VC dimension of the class of nonhomogeneous half sp. in $\mathbb{R}^d$ is $d+1$.

9.2 Linear Regression.

$\mathcal{H}_{reg} = L_d = \{x \mapsto \langle w, x\rangle + b : w\in\mathbb{R}^d, b\in\mathbb{R}\}$.

One common way is to use the squared-loss fun.

In this case, the empirical risk is called the Mean Squared Error

Another option: absolute value loss functi'n

Note: This can be solved by LP. (Exercise !).

Rmk.:

Regression problems cannot be analyzed using VC dimension. Rigorous means will be introduced later in the book to analyze regression problems.

9.2.1 Least Squares.

Least squares is the algorithm for solving ERM problems for the hypothesis class of linear regression predictors w.r.t. squared loss.

Goal:

Solve $\underset{w}{\operatorname{argmin}} L_S(h_w) = \underset{w}{\operatorname{argmin}} \frac{1}{m}\sum_{i=1}^m (\langle w, x_i\rangle - y_i)^2$.

Taking gradient, we have

$$\nabla_w L_S(h_w) = (Aw - b)\cdot\frac{2}{m}, \text{ where}$$

$$A = \left(\sum_{i=1}^m x_i x_i^T\right) \text{ and } b = \sum_{i=1}^m y_i x_i$$

$$= \begin{pmatrix} | & & | \\ x_1 & \cdots & x_m \\ | & & | \end{pmatrix}\begin{pmatrix} | & & | \\ x_1 & \cdots & x_m \\ | & & | \end{pmatrix}^T \qquad = \begin{pmatrix} | & & | \\ x_1 & \cdots & x_m \\ | & & | \end{pmatrix}\begin{pmatrix} y_1 \\ \vdots \\ y_m \end{pmatrix}.$$

Setting the gradient to 0, we are solving

$$Aw = b.$$

(Note: $L_S(h_w)$ is convex, as a fun. of w, and thus every local min. is a global min.)

Prop

W/ A and b as above, the equation $Aw = b$ always has a sol'n.

<Pf>

$\because$ A is real symmetric

$\therefore$ We can write it as the eigen decomp.:

$A = VDV^T$, where V: orthogonal and D: diag.

Define D^+ as a $d\times d$ diagonal matrix w/

$$D^+_{ii} = \begin{cases} 1/D_{ii} & \text{if } D_{ii}\neq 0 \\ 0 & \text{o.w.} \end{cases}$$

Define

$A^+ = VD^+V^T$, and $\hat{w} = A^+ b$. (to be claimed as a sol'n to $Aw=b$.)

Then $A\hat{w} = VDV^TVD^+V^Tb = VDD^+V^Tb$

$$= \sum_{i: D_{ii}\neq 0} \langle b, v_i\rangle v_i, \text{ where } V = \begin{pmatrix} | & & | \\ v_1 & \cdots & v_d \\ | & & | \end{pmatrix}.$$

Thus, $A\hat{w}$ is the orthogonal proj. of b onto $R(A)$.

(since v_i w/ $D_{ii}\neq 0$ are the e-vec. w/ nonzero e-val.)

$X := \begin{pmatrix} | & & | \\ x_1 & \cdots & x_m \\ | & & | \end{pmatrix}$. (so $A = XX^T$).

Note $\operatorname{rank}(A) = \operatorname{rank}(XX^T) = d - \operatorname{nullity}(XX^T)$
$= d - \operatorname{nullity}(X^T) = \operatorname{rank}(X^T) = \operatorname{rank}(X)$ and
$R(A) \subseteq R(X)$. Thus, $R(A) = R(X)$.

$\because b \in R(X) \therefore A\hat{w} = b$. (#)

9.3 Logistic Regression.

In logistic regression, we learn a family of functions $h : \mathbb{R}^d \to [0,1]$. Mainly used for classification task.

We compose L_d w/ a $\boxed{\text{sigmoid function}}$ (meaning "S-shaped" function).

In $\boxed{\text{logistic regression}}$, we use

$$\boxed{\phi_{sig}(z) = \frac{1}{1+e^{-z}}}, \text{ called the } \boxed{\text{logistic function}}.$$

$$\boxed{\mathcal{H}_{sig}} := \phi_{sig} \circ L_d = \{ x \mapsto \phi_{sig}(\langle w, x \rangle) : w \in \mathbb{R}^d \}.$$

Note:

① When $\langle w, x \rangle$ is very positive, $\phi_{sig}(\langle w,x\rangle) \approx 1$.

② " " " " negative, " " ≈ 0.

③ " " " " close to 0, " " $\approx 1/2$.

Thus, $\mathcal{H}_{sig}$ is solving classification problem via a probabilistic approach.

The loss function used by logistic regression is

$$\boxed{\ell(h_w(x,y)) = \log(1 + e^{-y\langle w,x\rangle})}.$$

Thus, given a training set $S = (x_1,y_1), \cdots, (x_m,y_m)$, we are solving (if we use ERM paradigm).

$$\boxed{\underset{w \in \mathbb{R}^d}{\operatorname{argmin}} \frac{1}{m} \sum_{i=1}^{m} \log(1 + e^{-y_i \langle w, x_i \rangle})}.$$

Rmk:

① $\because \log(1+e^{-y\langle w,x\rangle})$, the loss functn, is convex
$\therefore$ ERM is easy to implement. (via gradient descent).

② Implementing ERM on logistic regression is equivalent to finding Maximum Likelihood Estimator (to be introduced in Chap 24).

Chap 10 Boosting.

$\boxed{Q}$: Can an efficient week learner be "boosted" into an efficient strong learner?

$\boxed{\text{AdaBoost}}$ (= Adaptive Boosting) will be introduced here.

Rmk:

AdaBoost has been used successfully to detect faces in images.

10.1 Weak Learnability.

Def $\mathcal{H}$: a hypothesis class.

(1) A learning algorithm A is a $\boxed{\gamma\text{-weak-learner}}$ for $\mathcal{H}$ if

$\exists\, m_{\mathcal{H}} : (0,1) \to \mathbb{N}$ s.t. $\forall\, \delta \in (0,1)$, dist. $\mathcal{D}$ over $\mathcal{X}$, and labelling $f : \mathcal{X} \to \{\pm 1\}$,

if realizability holds w.r.t. $\mathcal{H}, \mathcal{D}, f$, then

$L_{(\mathcal{D},f)}(h) \le \frac{1}{2} - \gamma$ w/ prob. $\ge 1-\delta$ over $m \ge m_{\mathcal{H}}(\delta)$ iid samples generated by $\mathcal{D}$ and labelled by f,

where h is the returned hypothesis by the algorithm.

It seems that $h \notin \mathcal{H}$ is also allowed.

(2) $\mathcal{H}$ is $\boxed{\gamma\text{-weak-learnable}}$ if it has a γ-weak-learner.

Rmk:

① $L_{(\mathcal{D},f)}(h) \le \frac{1}{2} - \gamma$ means the hypothesis returned is only a little (quantified by γ) better than random guess.

② The potential advantage of week learning is that maybe there is an efficient algorithm for it. This motivates the opening question in this Chap.

Note: (Possible approach to construct week learner).
For given $\mathcal{H}$, choose a simple B and consider ERM_B instead of $ERM_{\mathcal{H}}$.

Example:

$\mathcal{X} := \mathbb{R}$

$\mathcal{H} := \{\boxed{\text{3-piece classifiers}}\} = \{ h_{\theta_1,\theta_2,b} : \theta_1, \theta_2 \in \mathbb{R}, \theta_1 < \theta_2, b \in \{\pm 1\} \}$

where $h_{\theta_1,\theta_2,b}(x) = \begin{cases} +b & \text{if } x < \theta_1 \text{ or } x > \theta_2 \\ -b & \text{if } \theta_1 \le x \le \theta_2. \end{cases}$

$\mathcal{B} := \{\boxed{\text{decision stumps}}\} = \{x \mapsto \mathrm{sign}(x-\theta)\cdot b : \theta \in \mathbb{R}, b \in \{\pm 1\}\}$.

Prop

$ERM_{\mathcal{B}}$ is a γ-weak-learner for $\mathcal{H}$, for $\gamma = 1/12$.

<Pf>

Not hard. See P133, upper part. #

10.1.1 Efficient Implementation of ERM for Decision Stumps.

↳ SKIPPED

10.2 AdaBoost.

AdaBoost is an algorithm having access to a weak learner and finds a hypothesis w/ low empirical risk.

The following is the pseudocode.

AdaBoost (for binary classification).

Input:

training set : $S = (x_1, y_1), \cdots, (x_m, y_m)$.

weak learner : WL

number of rounds : T.

Initialize: $D^{(1)} = (\frac{1}{m}, \cdots, \frac{1}{m})$. (a dist. on $x_1, \cdots, x_m$).

for $t = 1, \cdots, T$:

invoke weak learner : $h_t = WL(D^{(t)}, S)$.

compute : $\varepsilon_t = \sum_{i=1}^{m} D_i^{(t)} \cdot \mathbb{1}_{[y_i \neq h_t(x_i)]}$

let $w_t = \frac{1}{2}\log\left(\frac{1}{\varepsilon_t} - 1\right)$ (the smaller ε_t, the larger w_t).

update the dist. $D_i^{(t+1)} = \dfrac{D_i^{(t)} \exp(-w_t y_i h_t(x_i))}{\sum_{j=1}^{m} D_j^{(t)} \exp(-w_t y_j h_t(x_j))}$ for $i = 1, \cdots, m$.

Output:

the hypothesis $h_s(x) = \mathrm{sign}\left(\sum_{t=1}^{T} w_t h_t(x)\right)$.

Rmk:

① The smaller ε_t is, the larger w_t is.

i.e. if h_t is more correct, then it has more weight.

② If x_i is wrongly classified, then

$-w_t y_i h_t(x_i) = w_t > 0$. $\Rightarrow D_i^{(t+1)}$ exceptionally large.

$\Rightarrow x_i$ is taken care of more at the next round.

On the other hand, if x_i : correctly classified,

then $D_i^{(t+1)}$: exceptionally small. $\Rightarrow x_i$: less taken care of next round.

Other than the above intuitions, we also have the following theoretical guarantee:

Thm 10.2

S : training set.

Suppose at each iteration of AdaBoost, the weak learner returns a hypothesis w/ $\varepsilon_t \leq 1/2 - \gamma$.

Then

$$L_S(h_s) = \frac{1}{m}\sum_{i=1}^{m} \mathbb{1}_{[h_s(x_i) \neq y_i]} \leq e^{-2\gamma^2 T}.$$

Rmk:

By weak learnability, $\varepsilon_t \leq \frac{1}{2} - \gamma$ may fail w/ prob. at most δ.

By union bd, the assumption in Thm 10.2 holds w/ prob. $\geq 1 - T\delta$.

10.3 Linear Comb. of Base Hypotheses.

$\mathcal{B}$: a hypothesis class ($\mathcal{B}$ stands for base).

$T \in \mathbb{N}$.

Define

$$\boxed{L(\mathcal{B}, T)} = \left\{x \mapsto \mathrm{sign}\left(\sum_{t=1}^{T} w_t h_t(x)\right) : w \in \mathbb{R}^T, h_t \in \mathcal{B}, \forall t\right\}$$

In this sec, we estimate (bound) the VCdim of $L(\mathcal{B}, T)$ in terms of ① the VCdim of $\mathcal{B}$ and ② T.

10.3.1 VC dim of $L(\mathcal{B}, T)$.

Lemma 10.3

$\mathcal{B}$: a base hypothesis class

Assume both T and $\mathrm{VCdim}(\mathcal{B}) \geq 3$.

Then

$$\mathrm{VCdim}(L(\mathcal{B}, T)) \leq T \cdot (\mathrm{VCdim}(\mathcal{B}) + 1) \cdot (3\log(T(\mathrm{VCdim}(\mathcal{B}) + 1)) + 2)$$

Rmk:

Thm 10.2 allows us to use AdaBoost to reduce empirical risk while Lemma 10.3 guarantees the true risk not far from empirical risk.

10.4 AdaBoost for Face Detection.

see P140 - 141 or the paper by Viola and Jones for details.

Chap 11 Model Selection and Validation.

11.1 Model Selection using SRM.

Given $\mathcal{H}_1, \mathcal{H}_2, \mathcal{H}_3, \cdots$, a countable seq. of hypothesis classes.

Assume each $\mathcal{H}_d$ enjoys the UC property w/

$$\boxed{m^{UC}_{\mathcal{H}_d}(\varepsilon,\delta) \le \frac{g(d)\log(1/\delta)}{\varepsilon^2}}$$, where

$g: \mathbb{N} \to \mathbb{R}$: some monotone increasing fun.

Rmk:

① For binary classification (see Thm 6.8), take $g(d) = C_2(D+1)$, where $D = \text{VCdim}(\mathcal{H}_d)$.

② For AdaBoost, see Thm 10.3.

Taking $w(d) = \frac{6}{\pi^2}\cdot\frac{1}{d^2}$, Thm 7.4 implies

$$\boxed{L_{\mathcal{D}}(h) \le L_S(h) + \sqrt{\frac{g(d)\cdot(\log(1/\delta) + 2\log(d) + \log(\pi^2/6))}{m}}}$$

Recall that SRM will search for "d" and "$h\in\mathcal{H}_d$" that minimizes the RHS. In this sense, SRM is doing model selection.

Issue:

In many practical situations, the bd on the RHS is pessimistic and not useable. !!

11.2 Validation.

11.2.1 Hold-out Set.

Let $V = (x_1,y_1), \cdots, (x_{m_v}, y_{m_v})$ be m_v fresh examples sampled w.r.t. $\mathcal{D}$. (v: validation).

We have:

Thm 11.1

$h\in\mathcal{H}$, some predictor.

Assume the loss fun. is in $[0,1]$.

Then, $\forall \delta\in(0,1)$,

$$|L_V(h) - L_{\mathcal{D}}(h)| \le \sqrt{\frac{\log(2/\delta)}{2m_v}}$$

w/ prob. $\ge 1-\delta$ over the choices of a validation set V of size m_v.

<Pf>

Recall the Hoeffding's ineq.:

Lemma 4.5 (Hoeffding's ineq.)

$\theta_1,\cdots,\theta_m$: iid w/ $E[\theta_i] = \mu$ and $P[a\le\theta_i\le b] = 1$.

Then, $\forall \varepsilon > 0$,

$$P\left[\left|\frac{1}{m}\sum_{i=1}^m \theta_i - \mu\right| > \varepsilon\right] \le 2\cdot\exp(-2m\varepsilon^2/(b-a)^2).$$

Take $\theta_i = \ell(h,(x_i,y_i))$, where $(x_i,y_i)\sim\mathcal{D}$.

Then $\mu = L_{\mathcal{D}}(h)$ and $\frac{1}{m}\sum_{i=1}^{m_v}\theta_i = L_V(h)$.

Solving $2\exp(-2m_v\varepsilon^2) = \delta$ for ε, the result follows. #

Rmk:

① Bound in Thm 11.1 does not depend on the algorithm to construct h and is often tighter than that given by SRM.

② Price we pay is that we need additional fresh examples other than the training set.

In practice, V is constructed by holding out part of the whole examples. Thus, the validation set is also called a hold-out set.

11.2.2 Validation for Model Selection.

Validation can be used for model selection as follows:

Assume h_i is the returned hypothesis from $\mathcal{H}_i$, $i = 1,\cdots,r$. Denote $\mathcal{H} = \{h_1,\cdots,h_r\}$.

Then we have:

Thm 11.2

$\mathcal{H} = \{h_1,\cdots,h_r\}$, an arbitrary set of predictors.

Assume the loss fun. is in $[0,1]$.

V: a validation set of size m_v, sampled indep. of $\mathcal{H}$.

Then, w/ prob. $\ge 1-\delta$ over the choice of V,

$$|L_{\mathcal{D}}(h) - L_V(h)| \le \sqrt{\frac{\log(2|\mathcal{H}|/\delta)}{2m_v}}, \forall h\in\mathcal{H}.$$

<Pf>

Use Hoeffding and union bd. #

Rmk: (Gist)

Too many models may result in overfitting.