Contents

Chapter 1

Empirical Minimization Risk and VC Dimension

Statistics Learning and Deep Learning July 10,2022

Lecture Note

Lecturer: Satyanath Bhat *Scriber: Bin Yu*

1.1 Introduction

In the section, we will give the brief introduction about the statistical learning. And we focus on the classification task and introduce the definition $X \in \mathbb{R}^d \rightarrow$ feature space, $y \rightarrow \{-1, 1\}$, $S_N = \{(x_i, y_i)\}_{i=1}^N$, $H \rightarrow$ hypothesis class collection of classifier, classifier $h : X \rightarrow Y(A)$ where A is the learning algorithm, the goodness of a classifier $L : A \times Y \rightarrow \mathbb{R}^+$, $R(h) = \mathbb{E}_{X,Y}\left(L(h(x), y)\right)$, and $h^\star = argmin_{h \in H} R(h)$

We will introduce the empirical risk minimization

$$\hat{R}_n(h) = \frac{1}{n}\sum_{i}^{n} L(h(x_i), y_i) \tag{1.1}$$

this method will determine the empirical classifier $h_n^\star = argmin \hat{R}_n(h)$

We face a question that we do not know the exact joint probability distribution, that is why we introduce the empirical risk to estimate the classifier. However it will trigger new questions

1. Choice of H. In the sense, the $h^\star$ will be not optimal. Because $h^\star$ depends the hypothesis, and the hypothesis set is finite.
2. $\hat{h}_n^\star \rightarrow_{n\rightarrow\infty} h^\star$, we cannot guarantee whether this estimate can converge to optimal classifier.

In the machine learning community, the $L((h(x), y))$ is the out-sample performance and $L((h(x_i), y_i))$ is the in-sample performance

1.2 Perceptron

In this section, we will introduce the basic idea of perceptron. The figure shows that if we have the labeled data and can find the hyperplane to separate. The red line represents the optimal hyperplane and blue one represents the initial hyperplane. Our goal is to make initial one converge to the optimal solution.

Theorem 1. *Weak law of large numbers: If we have*$X \rightarrow^n_{i.i.d} \{X_i\}_{i=1}$, *then* $\mathbb{E}(x) = \mu$, $\hat{\mu} = \frac{1}{n}\sum_{i=1}^n X_i$

$$\hat{\mu}_n \rightarrow^n \mu$$
$$\forall\ \epsilon\ \delta\ \exists N\ s.t.\ \mathbb{P}(|\hat{\mu}_n - \mu| > \epsilon) < \delta, \forall n \geq N$$

Let us consider the H. Fix $h \in H$

$$\hat{R}_n(h) = \frac{1}{n}\sum_{i=1}^n \mathbb{1}\{h(x_i) \neq y_i\} \tag{1.2}$$

$\forall\ \epsilon\ \delta > 0$

$$\mathbb{P}(|\hat{R}_n(h - R(h))| > \epsilon) < \delta \quad \forall n \geq N \tag{1.3}$$

1.2.1 Uniform convergence

$\forall\ \epsilon\ \delta > 0 \quad \exists N$ s.t.

$$\mathbb{P}(\sup_{h\in H} |\hat{R}_n(h - R(h))| > \epsilon) < \delta \quad \forall n \geq N \tag{1.4}$$

Obviously, the uniform convergence is stronger than weak law of large number. Vapnik and Chervonenkis (2015) shows the detail of uniform convergence. Thus the formula indicates

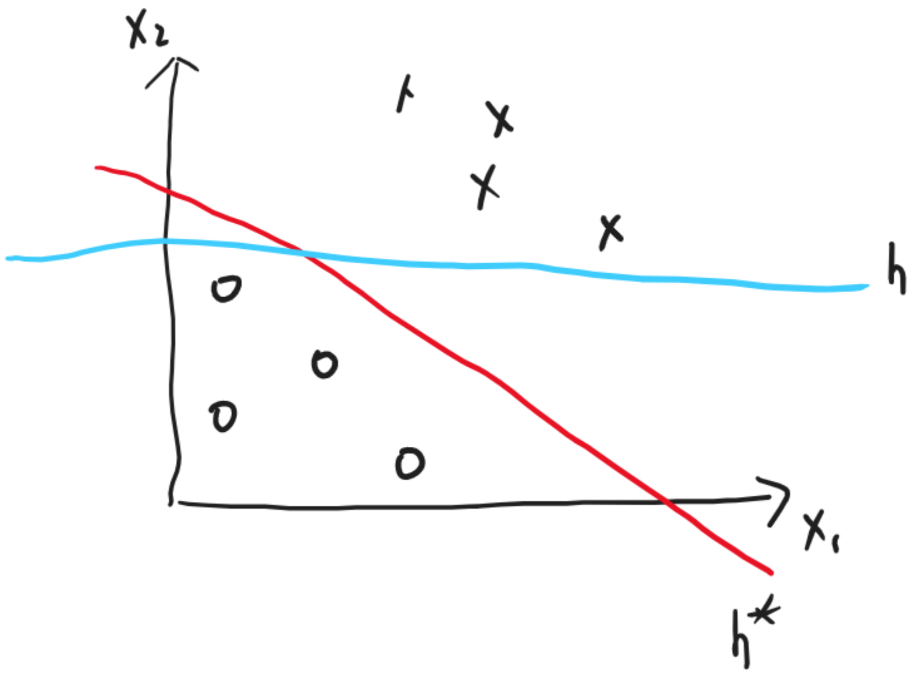

Figure 1.1: Example of Perceptron

- the consistency $\iff$ uniform convergence
- uniform convergence $\rightarrow$ weak law of large number

1.3 Uniform Convergence

We review the definition of consistency

$$\begin{gathered}\forall\ \epsilon\ \delta > 0, \exists\ N\ s.t. \\ \mathbb{P}\left[|R(\hat{h}_n^\star) - R(h^\star)| > \epsilon\right] < \delta \quad \forall n \geq N\end{gathered} \tag{1.5}$$

In the first lecture, we hold that the uniform convergence $\iff$ consistency. We will discuss the detail about this argument. It is intuition to show that uniform convergence $\rightarrow$ weak law of large number.

We will prove we cannot hold uniform convergence $\leftarrow$ weak law of large number.

$$\begin{aligned}&R(\hat{h}_n^\star) - R(h^\star) \\ =&R(\hat{h}_n^\star) - \hat{R}(\hat{h}_n^\star) + \hat{R}(\hat{h}_n^\star) - \hat{R}(h_n^\star) + \hat{R}(h_n^\star) - R(h^\star) \\ \leq&R(\hat{h}_n^\star) - \hat{R}(\hat{h}_n^\star) + \hat{R}(h_n^\star) - R(h^\star)\end{aligned}$$

Note that the second term will be negative because $\hat{h}_n^\star$ is the optimal solution of $\hat{R}_n$

$$|R(\hat{h}_n^\star) - R(h^\star)| \leq \sup_{h \in H} 2|\hat{R}(h_n^\star) - R(h^\star)| \tag{1.6}$$

Since the weak law of large number

1.3.1 Original Problem

We like to show that

$$R(\hat{h}_n^\star) \rightarrow R(h^\star) \tag{1.7}$$

However it is difficult to prove that and modify the problem below

$$\begin{gathered}\forall\ \epsilon\ \delta > 0, \exists\ N\ s.t. \\ \mathbb{P}\left[|\hat{R}_n(h) - R(h)| > \epsilon\right] < \delta \quad \forall n \geq N\end{gathered} \tag{1.8}$$

Discuss finite and infinite cases

- $|H| = m < \infty$

  Recall the definition

$$R(h) = \mathbb{E}\left[\mathbb{1}_{h(x)\neq y}\right]$$
$$\hat{R}_n(h) = \frac{1}{n}\sum_i \left(\mathbb{1}_{h(x_i)\neq y_i}\right)$$

  The Hoeffding's inequality imply

$$\mathbb{P}(|\hat{R}_n(h) - R(h)| > \epsilon) < 2\exp -2n\epsilon^2 \tag{1.9}$$

  denote $E_i = \{|\hat{R}_n(h_i) - R(h_i)| > \epsilon\}$ by event

$$\mathbb{P}(\cup_i^m E_i) \leq 2m\exp -2n\epsilon^2 \tag{1.10}$$

- Infinite

$$\mathbb{P}\left(\sup_{h\in H}|\hat{R}_n - R(h)| > \epsilon\right) < C_1\pi(n)\exp -C_2 n\epsilon^2 \tag{1.11}$$

  Where C_1, C_2 are the coefficient and $\pi(n)$ is the growth function

1.4 Ghost Sample

we will show

$$\mathbb{P}\left(\sup_{h\in H}|\hat{R}_n(h) - R(h)|\ > \epsilon\right) \leq 2\mathbb{P}\left(\sup_{h\in H}|R_n(h) - R_n'(h)|\ > \epsilon/2\right) \tag{1.12}$$

where $R_n'(h) = \frac{\sum \mathbb{1}_{h(x_i')\neq y_i'}}{n}$

1.5 Empirical Risk Minimization

- X feature space
- $y \to \{0, 1\}$
- $P_{X,Y}$ joint distribution
- $S_n = | = \{(x_i, y_i)\}_{i=1}^n$
- $h : X \to A$ A is the action. The second definition is $h : X \to \{-1, 1\}$. It means the classifiers is the convex set

- H collection of hypothesis
- $L : A \times y \to \mathbb{R}$ is the loss function
- Risk(out of sample) $R(h) = E_{X,Y}(L(h(x), y))$
- $h^\star = argminR(h)$
- $\hat{R}_n(h) = \frac{1}{n}\sum_i L(h(x_i), y_i)$
- $\hat{h}_n^\star = argmin\hat{R}_n(h)$

we have the argument that

$$\hat{h}_n^\star \to_{n\to\infty} h^\star \tag{1.13}$$

Generally understand the difference between regression and classification task.

$$\begin{aligned} \text{Regression} \quad & \mathbb{E}[y|X = x] \\ \text{Classification} \quad & \mathbb{P}(Y = y|X = x) \end{aligned} \tag{1.14}$$

And the empirical risk minimization has the consistency property $\forall \epsilon, \delta > 0, \exists N \ s.t.$

$$\mathbb{P}(|R(\hat{h}_n^\star) - R(h^\star)| > \epsilon) < \delta \tag{1.15}$$

1.6 Uniform Convergence

The consistency is equivalent with uniform convergence $\forall \epsilon, \delta > 0, \exists N$ s.t.

$$\mathbb{P}(\sup_{h\in H} |\hat{R}_n(h) - R(h)| > \epsilon) < \delta \tag{1.16}$$

Discuss two cases of hypothesis which is finite and infinite. Let consider the finite case firstly. Our goal is to search for H which satisfies uniform convergence

$$\mathbb{P}(\sup_{h\in H} |\hat{R}_n(h) - R(h)| > \epsilon) < C_1\pi_H(n)\exp^{-C_2 n\epsilon^2} \tag{1.17}$$

where C_1, C_2 are coefficients and $\pi_H(n)$ is the growth function with figure 1.6

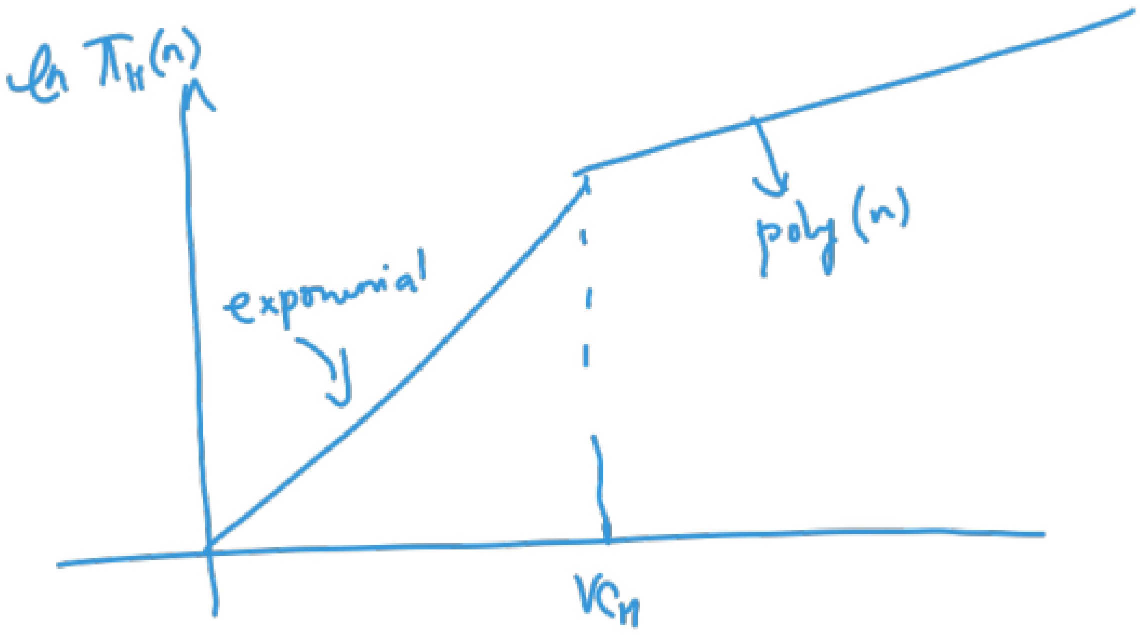

Figure 1.2: VC Dimension

1.6.1 Ghost Sample

We like to use the ghost sample step show that

$$\mathbb{P}(\sup_{h\in H} |\hat{R}_n(h) - R(h)| > \epsilon) < 2\mathbb{P}(\sup_{h\in H} |\hat{R}_n(h) - R'_n(h)| > \frac{\epsilon}{2}) \tag{1.18}$$

where S'_n represents the ghost sample

$$\begin{aligned} R'_n(h) &= \frac{1}{n}\sum L(h(x'_i), y'_i) \\ S'_n &= (x'_i, y'_i) \end{aligned} \tag{1.19}$$

We can prove by the contradiction and assume h_B with the bad hypothesis

$$\mathbb{P}\left(\sup_{h\in H} |\hat{R}_n(h) - R(h)| > \epsilon\right) = \mathbb{P}\left(|\hat{R}_n(h_B) - R(h_B)| > \epsilon\right) \tag{1.20}$$

Approximately

$$\mathbb{P}\left(\sup_{h\in H} |\hat{R}_n(h) - R'_n(h)| > \frac{\epsilon}{2}\right) \geq \mathbb{P}\left(|\hat{R}_n(h_B) - R'_n(h_B)| > \frac{\epsilon}{2}\right) \tag{1.21}$$

we can analysis two event

$$\begin{aligned} C_1 &= \{|\hat{R}_n(h_B) - R(h_B)| > \epsilon\} \\ C_2 &= \{|R'(h_B) - R(h_B)| < \frac{\epsilon}{2}\} \end{aligned} \tag{1.22}$$

In $R'_n(h_B)$, the random source is S_n, S'_n. And in $R(h_B)$, the random source is S_n

Proposition 2.

$$\begin{aligned}\mathbb{P}(f(x,y) \geq a) &= \mathbb{E}_y \mathbb{P}_{X|Y}(f(x,y) \geq a|y) \\ \mathbb{E}[X] &= \mathbb{E}[\mathbb{E}[X|Y]]\end{aligned} \tag{1.23}$$

continue to discuss the consistency

$$\begin{aligned}\mathbb{P}\left(|\hat{R}_n(h_B) - R'_n(h_B)| > \frac{\epsilon}{2}\right) &\geq \mathbb{P}(C_1 \cap C_2) \\ &= \mathbb{E}[1_{C_1} 1_{C_2}] \\ &= \mathbb{E}_{S_n}[1_{C_1} \mathbb{E}_{S'_n|S_n}[1_{C_2}]] \\ &= \mathbb{P}(|R'_n(h_B) - R(h_B)| < \frac{\epsilon}{2}|S_n) \\ &\geq 1 - 4\frac{Var(R'_n(h_B))}{\epsilon} \\ &\geq 1 - \frac{1}{n\epsilon^2} \\ &\geq \frac{1}{2}\end{aligned} \tag{1.24}$$

Go back this proabblility

$$\begin{aligned}\mathbb{P}\left(\sup_{h\in H} |\hat{R}_n(h) - R'_n(h)| > \frac{\epsilon}{2}\right) &\geq \frac{1}{2}\mathbb{P}(C_1) \\ &= \frac{1}{2}\mathbb{P}\left[|\hat{R}_n(h_B) - R(h_B)| > \epsilon\right] \\ &= \frac{1}{2}\mathbb{P}\left(\sup_{h\in H} |\hat{R}_n(h) - R_n(h)| > \epsilon\right)\end{aligned} \tag{1.25}$$

1.6.2 Symmetrization

$$\mathbb{P}\left(\sup_{h\in H} |\hat{R}_n(h) - R'_n(h)| > \frac{\epsilon}{2}\right) = \mathbb{P}\left[\sup_{h\in H} |\frac{1}{n}\sum\left(\mathbb{1}_{\{h(x_i)\neq y_i\}} - \mathbb{1}_{\{h(x'_i)\neq y'_i\}}\right)| > \frac{\epsilon}{2}\right] \tag{1.26}$$

Furthermore

$$\begin{aligned}&\mathbb{P}\left[\sup_{h\in H} |\frac{1}{n}\sum \sigma_i\left(\mathbb{1}_{\{h(x_i)\neq y_i\}} - \mathbb{1}_{\{h(x'_i)\neq y'_i\}}\right)| > \frac{\epsilon}{2}\right] \\ \leq\ &\mathbb{P}\left[\sup_{h\in H} |\frac{1}{n}\sum \sigma_i \mathbb{1}_{\{h(x_i)\neq y_i\}}| > \frac{\epsilon}{4} \cup \sup_{h\in H} |\frac{1}{n}\sum \sigma_i \mathbb{1}_{\{h(x'_i)\neq y'_i\}}| > \frac{\epsilon}{4}\right] \\ &\leq 2\mathbb{P}\left[\sup_{h\in H} |\frac{1}{n}\sum \sigma_i \mathbb{1}_{\{h(x_i)\neq y_i\}}| > \frac{\epsilon}{4}\right]\end{aligned} \tag{1.27}$$

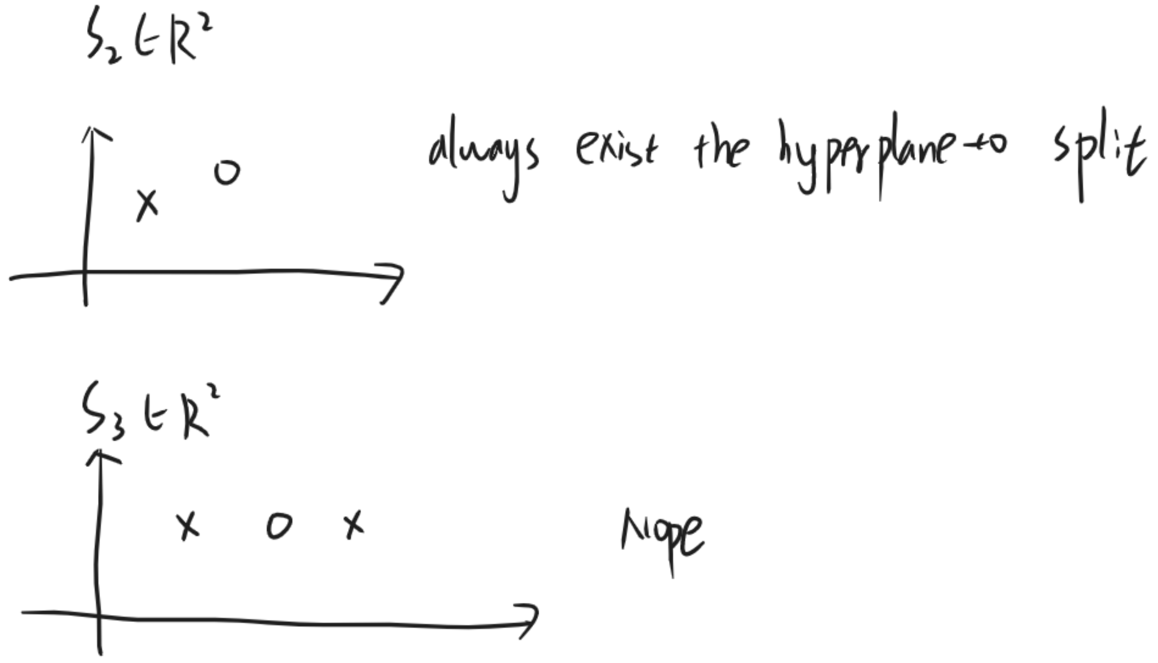

Figure 1.3: $S_2 \in \mathbb{R}^2$

where

$$\sigma_i = \begin{cases} -1 & w.p.\frac{1}{2} \\ 1 & w.p.\frac{1}{2} \end{cases} \tag{1.28}$$

Conditioning on S_n

$$\begin{aligned} \mathbb{P}\left[\sup_{h\in H} |\frac{1}{n}\sum \sigma_i \mathbb{1}_{\{h(x_i)\neq y_i\}}| > \frac{\epsilon}{4}\right] &= \mathbb{E}\left[\mathbb{P}\left[\sup_{h\in H} |\frac{1}{n}\sum \sigma_i \mathbb{1}_{\{h(x_i)\neq y_i\}}| > \frac{\epsilon}{4}\right] |S_n\right] \\ &\leq \mathbb{E}_{S_n}\left[2\pi_H(S_n)\right] \exp^{-\frac{-2n\epsilon^2}{32}} \\ &\leq 2\pi_H(n) \exp^{-\frac{-2n\epsilon^2}{32}} \end{aligned} \tag{1.29}$$

where

$$\begin{aligned} \pi_H(n) &= \max_{S_n} \pi_H(S_n) \\ \pi_H(n) &= \begin{cases} 2^n & N \leq d_{VC} \\ polynomial(n) & n > d_{VC} \end{cases} \end{aligned} \tag{1.30}$$

why we need the σ_i random variable. It is the sub sample operation. Looking at the figure 1.6.2, intuition idea is if we have $S_3 \in \mathbb{R}^2 \times \mathbb{R}$. Always we can find maximal possible $2^{n=3}$ hyperplanes to shattered the datapoints. That's why the changing point of segment function is $d_{VC} = x^n$

1.7 VC dimension

Remark 3. *$d_{VC}(H) = \infty$, learning may not be feasible*

We introduce the steps to compute $d_{VC}(H)$

1. Suppose K points $\exists S_k$ such that $\pi_H(s_k) = 2^k$. It could be shattered
2. Suppose $d_{VC} = k$, $\forall S_{k+1}$ could not be shattered.

Example 4. *Consider 3 points. If the 3 points samples as the triangle shape. It could be shattered. Otherwise the 3 points are collinear which might be not shattered. However, $\forall S_4 \in \mathbb{R}^2 \times \mathbb{R}$. Meanwhile, there are three different cases. the points are collinear, one point is internal to the triangle and external to the triangle. The intuition of VC dimension is we could find maximal $d_{VC} \leq 2^k$ hyperplanes to be shattered in S_k.*

1.8 Sauer–Shelah Lemma

We will discuss

$$\pi_H(n) = \sum_{i=0}^{d_{VC}} \binom{n}{i}, \quad n \geq d_{VC} \tag{1.31}$$

we can prove this formula by function $B(n,k)$

Definition 5. *B(n,k)=number ways to label n points with k and a breakpoint*

Chapter 2

Neural Network and Deep Learning

2.1 Preliminaries

2.1.1 Hyperplane

The $y = mx + c$ is one of definition of line. It will be limited in the further. We will introduce another definition in the $\mathbb{R}^n$. If one line pass the original point in the $\mathbb{R}^2$. The function will be $w^T x = 0 \leftrightarrow w_1 x_1 + w_2 x_2 = 0$. If the line do not pass the original point. Then we will introduce the derivative. Function $f : \mathbb{R} \to \mathbb{R}$

$$f'(x) = \lim_{h \to 0} \left[\frac{f(x+h) - f(x)}{h} \right] \tag{2.1}$$

Then we want to approximate x linearly. We can show below equation by Taylor expansion

$$\begin{aligned} f(x+h) &= f(x) + hf'(x) + o(h) \\ f(x+h) - f(x) - hf'(x) &= o(h) \\ \lim_{h \to 0} \frac{f(x+h) - f(x) - hf'(x)}{h} &= 0 \end{aligned}$$

Thus

$$\lim_{h \to 0} \frac{||f(x+h) - f(x) - hT||}{||h||} = 0 \tag{2.2}$$

where $T : \mathbb{R}^n \to \mathbb{R}^m$ is total derivative

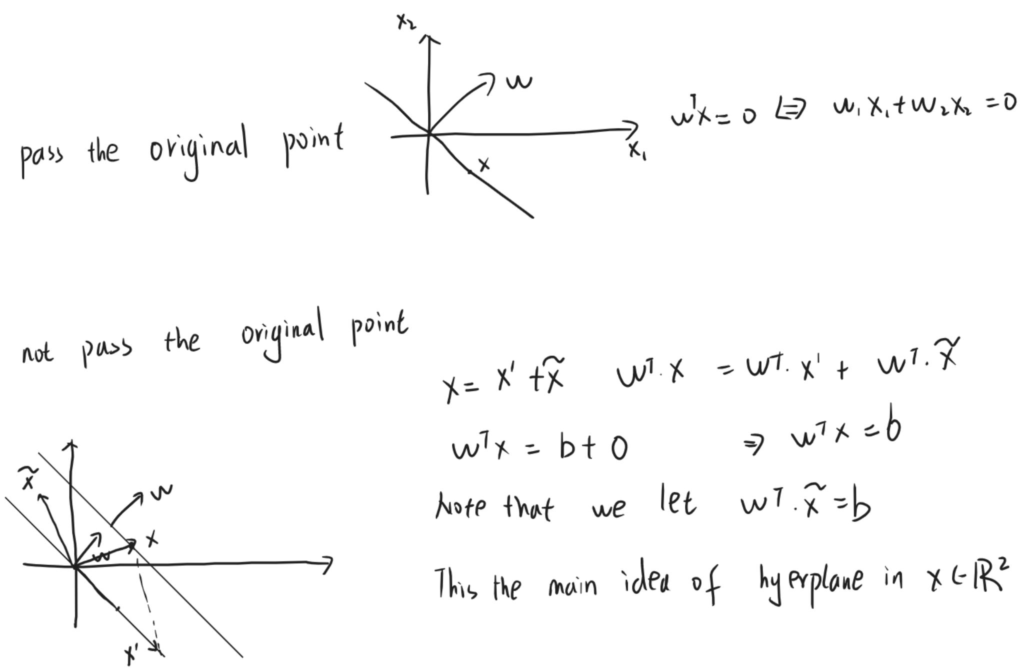
pass the original point
x2
w
x1
x
w^T x = 0 ⟺ w1x1 + w2x2 = 0
not pass the original point
x = x' + x̃   w^T·x = w^T·x' + w^T·x̃
w^T x = b + 0   ⇒ w^T x = b
Note that we let w^T·x̃ = b
This the main idea of hyperplane in x ∈ ℝ²
x̃
w
x
x'

Figure 2.1: Hyperplane

Example 6. *Function $f : \mathbb{R}^n \to \mathbb{R}^m$, the T_f will be*

$$\begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \cdots & \cdots & \cdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

And the chain rules is the important in the machine learning.

$$\begin{aligned} f : \mathbb{R} \to \mathbb{R}, g : \mathbb{R} \to \mathbb{R} \quad m(x) = f(g(x)) \\ \lim_{h \to 0} \frac{m(x+h) - m(x)}{h} = f'(g(x))g'(x) \end{aligned} \tag{2.3}$$

we will extend the chain rule into $f : \mathbb{R}^n \to \mathbb{R}^k$ and $g : \mathbb{R}^m \to \mathbb{R}^n$

$$T_{h_{(k \times m)}} = T_{f_{(k \times n)}} T_{g_{(n \times m)}} \tag{2.4}$$

2.2 Feedforward Neural Network

We introduce feed-forward neural network. Here we consider the linear regression we have

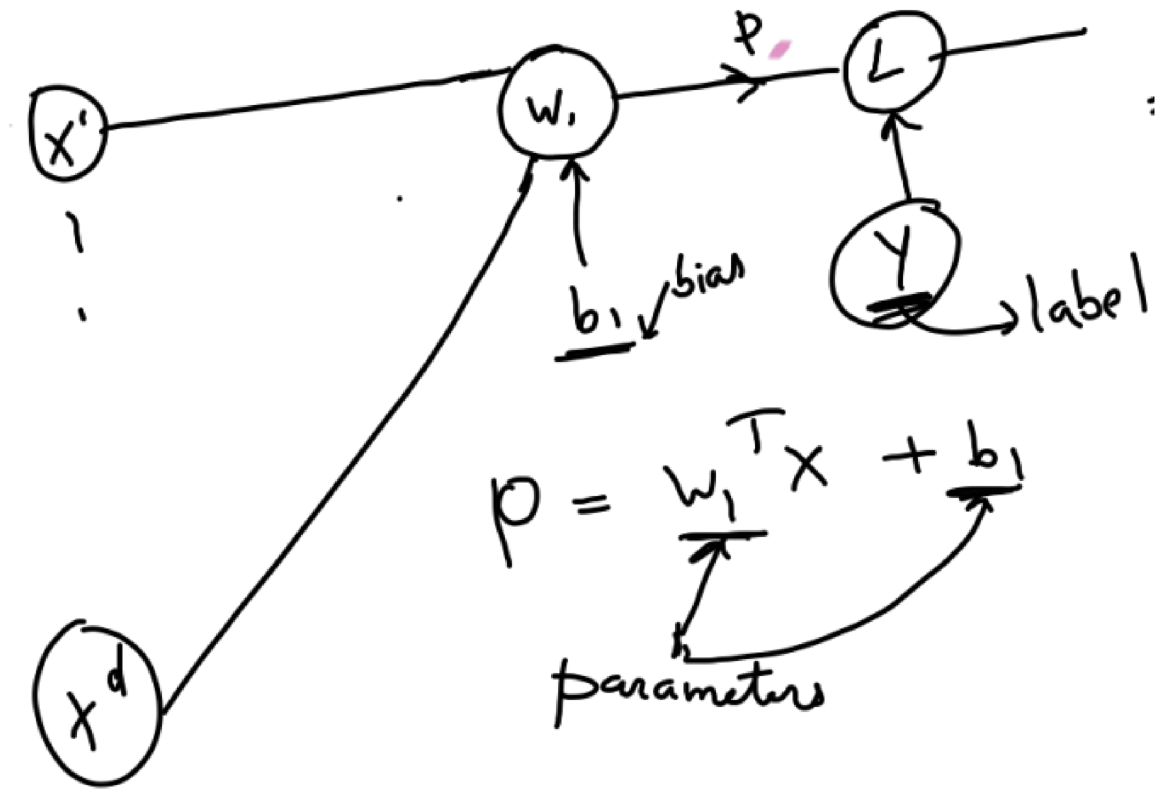

Figure 2.2: Hyperplane

$S_n = \{(X_i, y_i)\}_i^n \in \mathbb{R}^d \times \mathbb{R}$. w is the weight, b is the bias, and y is the label

$$\begin{aligned} p &= w^T x + b \\ L &= \frac{1}{n}\sum(wx_i + b - y_i)^2 \end{aligned} \tag{2.5}$$

And we will use the gradient descent to optimize the loss function

$$\begin{aligned} w_{new} &= w_{old} - \eta \bigtriangledown_w L \\ b_{new} &= b_{old} - \eta \bigtriangledown_b L \end{aligned} \tag{2.6}$$

we will repeat till convergence (L does not change and be less than error). Regarding the gradient descent, the batch and stochastic are two different variant. The stochastic gradient descent is to random draw the subsample to compute the gradient from the original sample. The batch gradient descent is to compute all gradient over the original sample. That's the basic idea of optimization method. However, it will trigger two question

- How to makes prediction?
- how to do gradient descent?

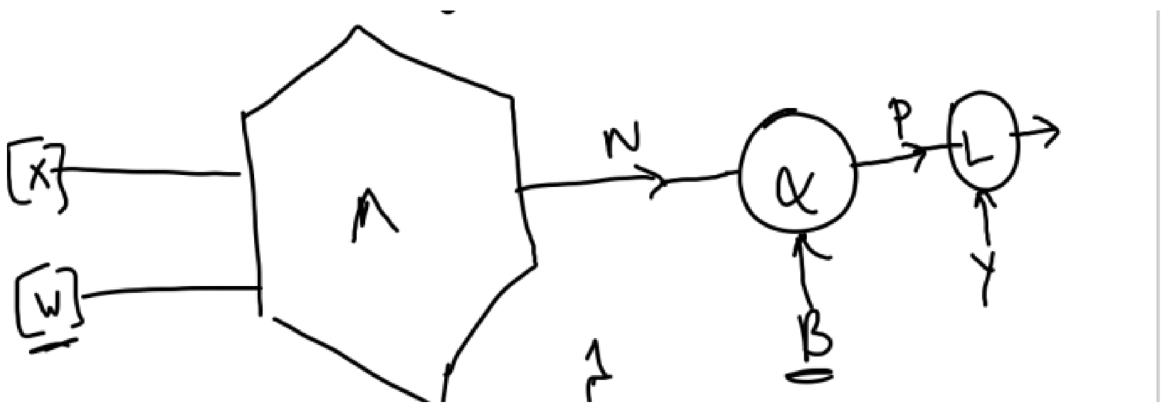

Figure 2.3: Linear Regression

We have

$$X = \begin{bmatrix} x_{11} & x_{12} & ... & x_{1d} \\ x_{21} & x_{22} & ... & x_{2d} \\ ... & & & \\ x_{n1} & x_{n2} & ... & x_{nd} \end{bmatrix} \qquad w = \begin{bmatrix} w_{11} \\ w_{21} \\ ... \\ w_{d1} \end{bmatrix} \qquad N = \begin{bmatrix} N_1 \\ N_2 \\ ... \\ N_n \end{bmatrix}$$

The definition of N and α shows

$$\begin{aligned} N &= Xw \\ \alpha &: \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n \end{aligned} \tag{2.7}$$

The structure of linear regression imply this

$$P = \begin{bmatrix} N_1 + B \\ N_2 + B \\ ... \\ N_n + B \end{bmatrix} \qquad L = \sum_{i=1}^{n} (P_i - y_i)^2 \tag{2.8}$$

Then we can take the derivative to L over P

$$\begin{aligned} \bigtriangledown_P L &= \begin{bmatrix} \frac{\partial L}{\partial P_1} & \frac{\partial L}{\partial P_2} & ... & \frac{\partial L}{\partial P_n} \end{bmatrix} \\ &= \begin{bmatrix} 2(P_1 - y_1) & 2(P_2 - y_2) & ... & 2(P_n - y_n) \end{bmatrix} \end{aligned} \tag{2.9}$$

This is the back propagation. And we apply the chain rule into the $\bigtriangledown_P L$

$$\begin{aligned} \frac{\partial L}{\partial N_i} &= \frac{\partial L}{\partial P_i} \frac{\partial P_i}{\partial N_i} \\ &= \frac{\partial L}{\partial P_i} \times 1 \\ \frac{\partial L}{\partial N} &= \begin{bmatrix} \frac{\partial L}{\partial P_1} & \frac{\partial L}{\partial P_2} & ... & \frac{\partial L}{\partial P_n} \end{bmatrix} \\ \frac{\partial L}{\partial B} &= \sum \frac{\partial L}{\partial P_i} \frac{\partial P_i}{\partial B} \\ \frac{\partial L}{\partial B} &= \sum \frac{\partial L}{\partial P_i} \end{aligned}$$

Then we need to consider two kinds of derivative $\frac{\partial L}{\partial w}$ and $\frac{\partial L}{\partial x}$. we will introduce the function n. We can get $\frac{\partial L}{\partial x_{11}} = \frac{\partial L}{\partial n_{11}} \frac{\partial n_{11}}{\partial x_{11}} + \frac{\partial L}{\partial n_{12}} \frac{\partial n_{12}}{\partial x_{11}}$

2.3 Convollutional Neural Networks

It is very popular in the computer vision and image recognition. And we need to understand the froward loss and backward loss. The common image is consist of the red green and blue, three basic color. The size of image is $m * n * 3$ (think the RGB three channels). We will introduce the window/filter A and trainiing sample x

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \qquad x = \begin{bmatrix} x_{11} & x_{12} & ... & x_{m3} \\ ... & & & \\ x_{m1} & x_{m2} & ... & x_{mn} \end{bmatrix}$$

For the simplicity, each x_{ij} is the entry not the vector (ignore the RGB setting). The modified matrix shows

$$B = \begin{bmatrix} \sum_{i=0,j=0}^{i=3,j=3} a_{i,j} * x_{i,j} & \sum_{i=1,j=0}^{i=3,j=3} a_{i_1,j} * x_{i,j} & ... & \sum_{i=m,j=0}^{i=m,j=3} a_{i,j} * x_{i,j} \end{bmatrix}$$

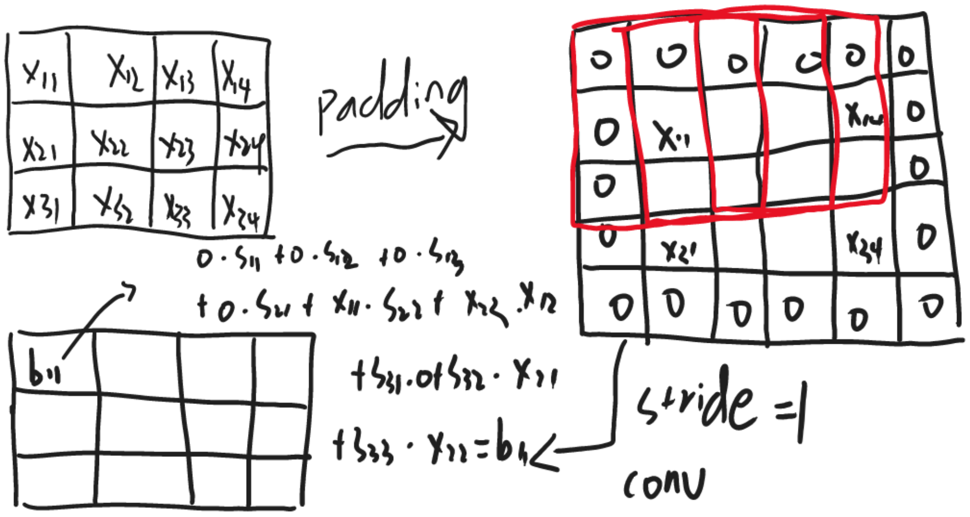

Figure 2.4: Conventional Process

The conventional process could be visualized in the below figure. We simplify the conventional process and introduce the underlying concept. Normally, we use 3×3 filter. And the stride is the filter move distance. In this instance, stride equals 1. Then we will resize the original image from $m \times n$ to $m + 2 \times n + 2$. Then we need do the flatten operations to transform the matrix to vector. The next step is similar in the previous lecture, we can use different neural network structure. All CNN contents is from this book Weidman (2019)

Bibliography

Vapnik, Vladimir N, A Ya Chervonenkis. 2015. On the uniform convergence of relative frequencies of events to their probabilities. *Measures of complexity*. Springer, 11–30.

Weidman, Seth. 2019. *Deep learning from scratch: building with python from first principles*. O'Reilly Media.