# CNN for CV

AI for CV Group
2019

# Week 3-5.
# Classical Machine Learning

# Contents:


## I. Introduction To Machine Learning

A. Supervised Learning
B. Unsupervised Learning

## II. Classical Supervised Learning

C. Linear Regression
D. Logistic Regression
E. Neural Network
F. Back Propagation
G. Regularization
H. SVM

# Contents:


**III. Classical Unsupervised Learning**

I. K-Means

**IV. Concepts & Problems**

J. Training / Validation / Test Set

K. Underfit / Overfit

L. Bias / Variance

M. Gradient Vanishing / Explosion

**V. Other Classical ML Tools**

N. Decision Tree: ID3 / C4.5 / CART

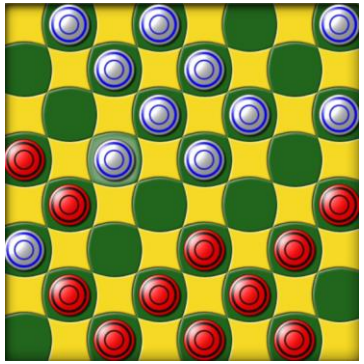
O. Reading Parts: AdaBoost / Haar Feature

# I. Introduction To Machine Learning

# I. Intro To ML

**Arthur Samuel**: Field of study that gives computers the ability to learn without being explicitly programmed.

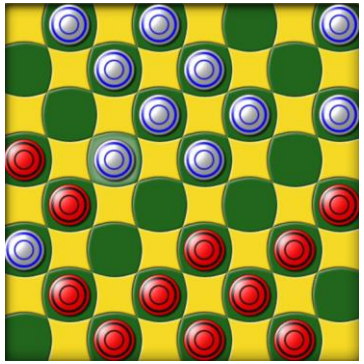
1959



# I. Intro To ML

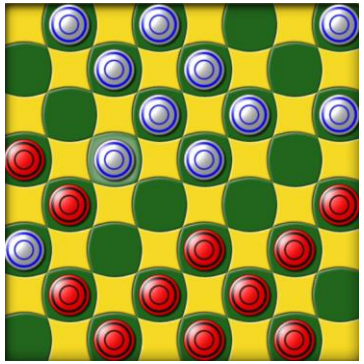
**Arthur Samuel**: Field of study that gives computers the **ability to learn** without being explicitly programmed.

1959



# I. Intro To ML

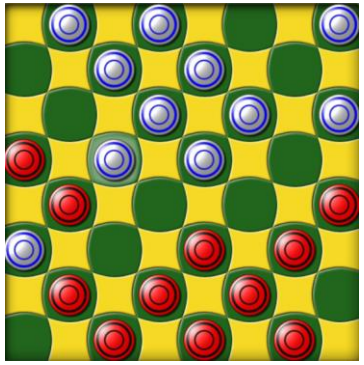
**Arthur Samuel**: Field of study that gives computers the ability to learn without being explicitly programmed. 1959



**Tom Mitchell**: Well-posed Learning Problem: A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E. 1998

# I. Intro To ML

**Arthur Samuel**: Field of study that gives computers the ability to learn without being explicitly programmed. 1959



1. A solution exists; 2. Solution is unique; 3. Solution's behavior is continuous.

**Tom Mitchell**: Well-posed Learning Problem: A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E. 1998

# I. Intro To ML

**ML Algorithms**:

- Supervised Learning
- Unsupervised Learning

**Others**:

- Reinforcement Learning
- Recommender Systems

# I. Intro To ML

**ML Algorithms**:

- Supervised Learning
- Unsupervised Learning

**Others**:

- Reinforcement Learning
- Recommender Systems

# I. Intro To ML

**ML Algorithms**:

- Supervised Learning
- Unsupervised Learning

**Others**:

- Reinforcement Learning
- Recommender Systems

# I. Intro To ML

## A. Supervised Learning:

# I. Intro To ML

## A. Supervised Learning:

**Who supervises who?**

**Who is supervised by whom?**

# I. Intro To ML

## A. Supervised Learning:

**Who supervises who?**

**A “teacher” supervises “me”.**

**Who is supervised by whom?**

**“I” am supervised by a “teacher”.**

# I. Intro To ML

## A. Supervised Learning:

**Who supervises who?**

**A “teacher” supervises “me”.**

**A “standard” supervises the “system”.**

**Who is supervised by whom?**

**“I” am supervised by a “teacher”.**

**The “system” is supervised by the “standard”.**

# I. Intro To ML

## A. Supervised Learning:

**There is an answer as our standard!**

**Who supervises who?**

**A "teacher" supervises "me".**

**A "standard" supervises the "system".**

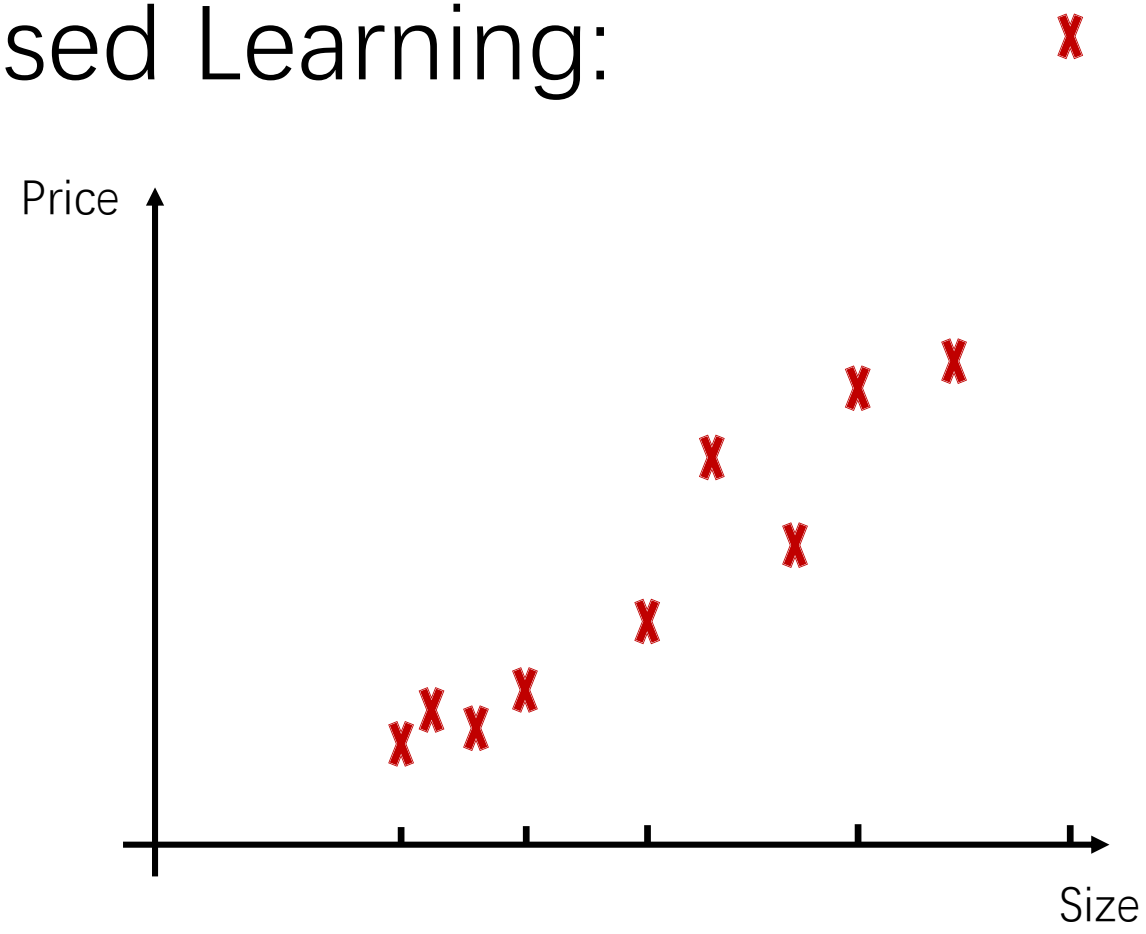
**Who is supervised by whom?**

**"I" am supervised by a "teacher".**

**The "system" is supervised by the "standard".**

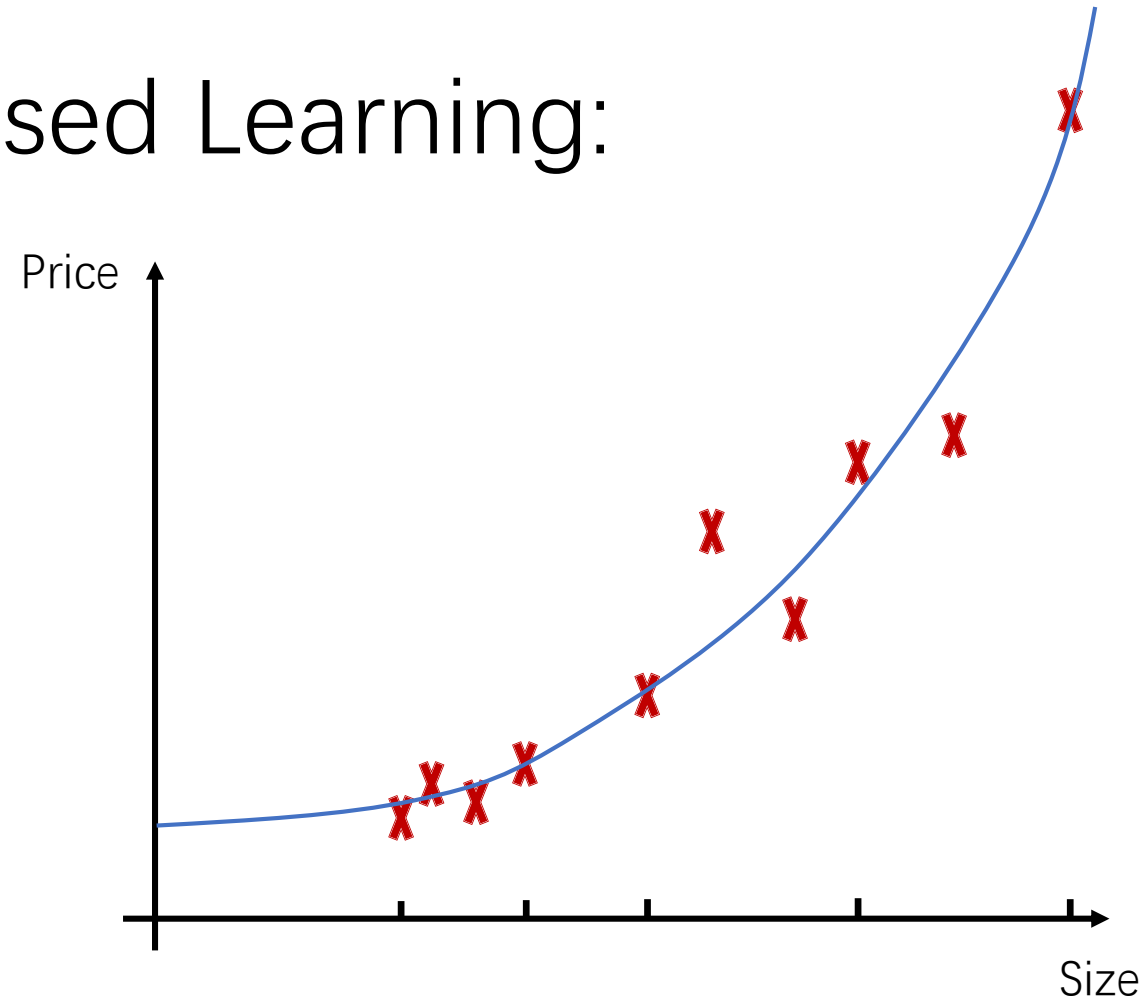
# I. Intro To ML

## A. Supervised Learning:



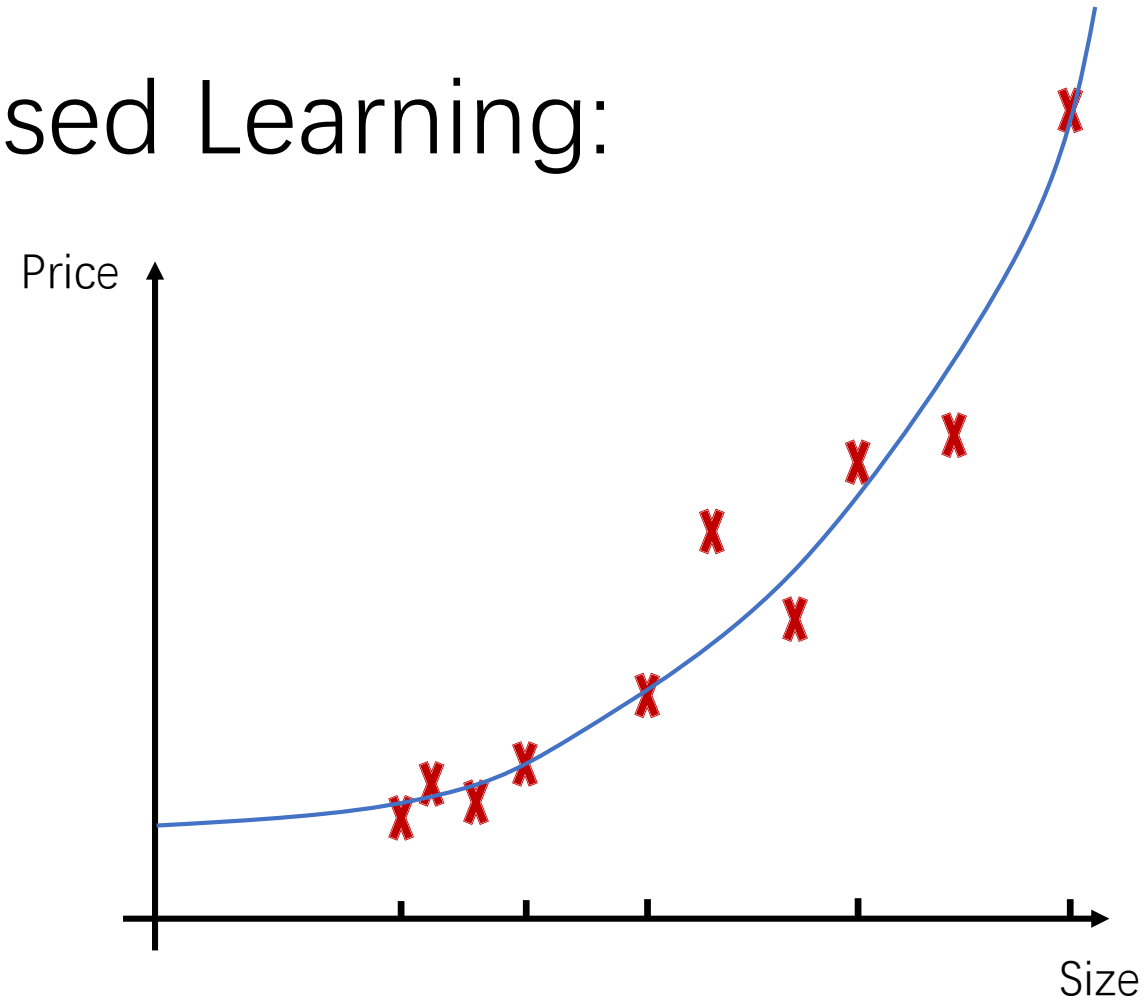
# I. Intro To ML

## A. Supervised Learning:



# I. Intro To ML

## A. Supervised Learning:




**Regression**

# I. Intro To ML

## A. Supervised Learning:

**Regression**

**Object Detection**

**Keypoint Detection**

**Prediction (age, charm, counting)**

**.....**

# I. Intro To ML

## A. Supervised Learning:

**Regression**

**Object Detection**

**Keypoint Detection**

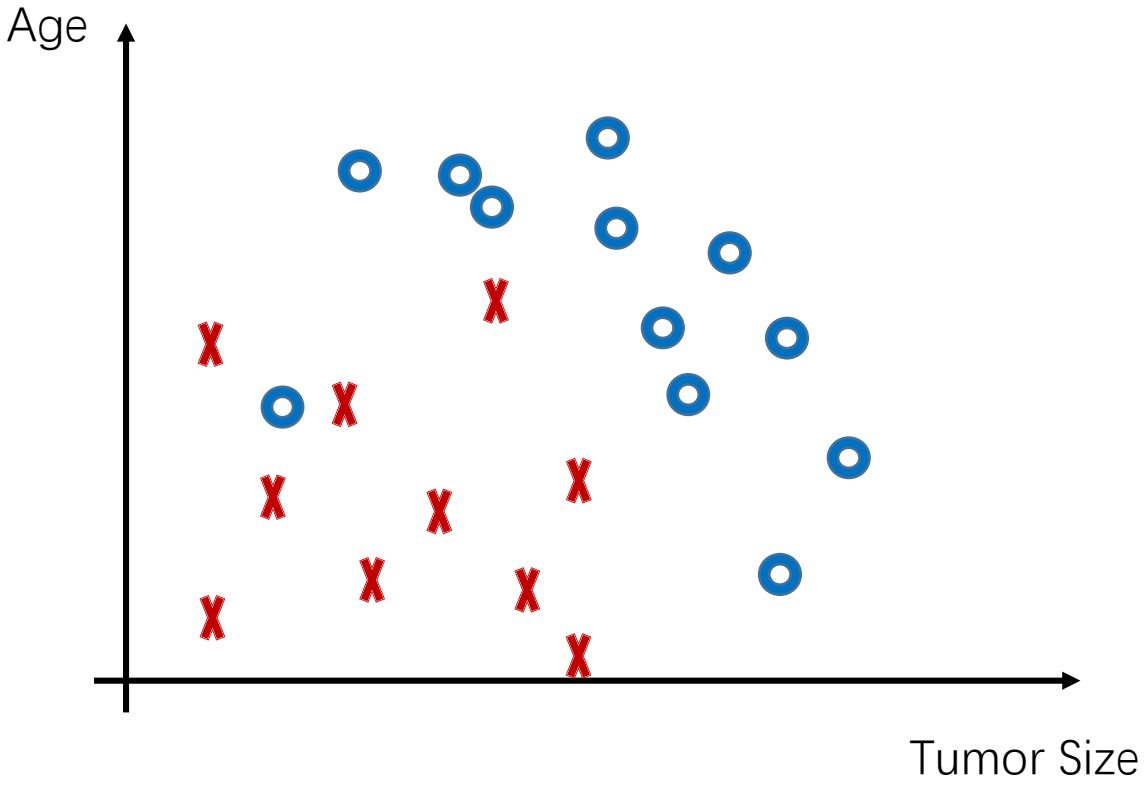
**Prediction (age, charm, counting)**

**.....**

**Give out numbers**

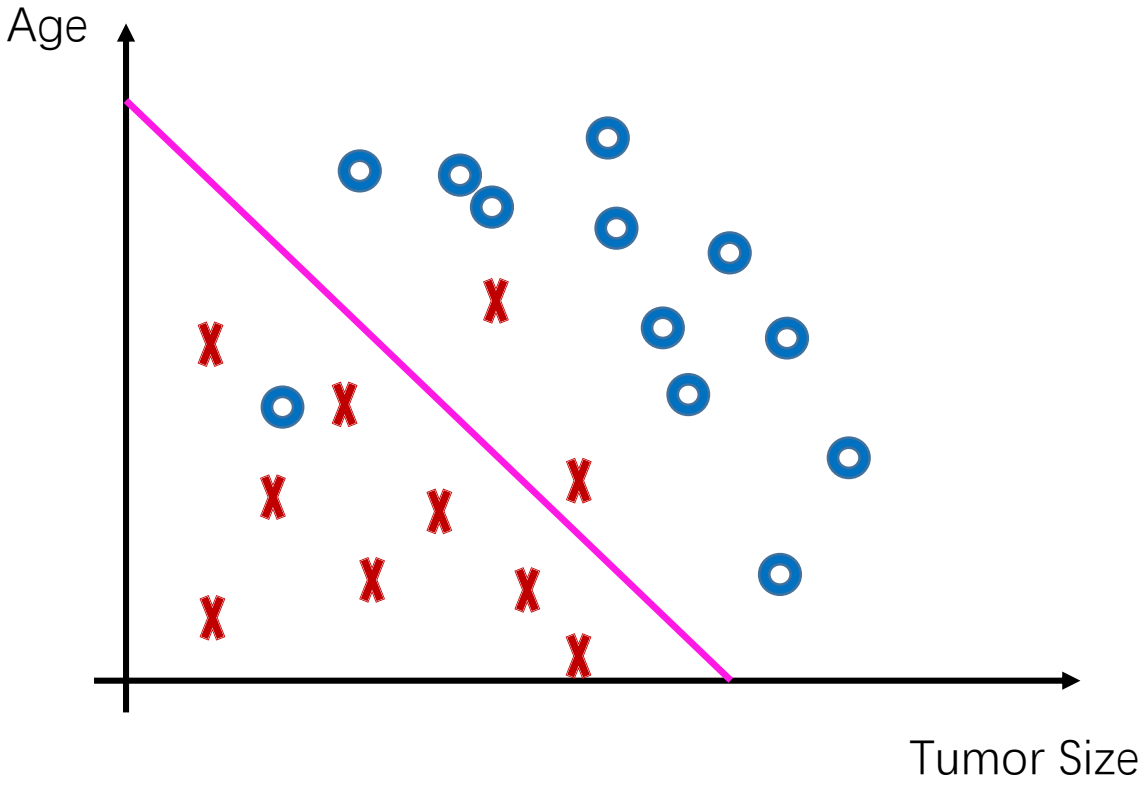
# I. Intro To ML

## A. Supervised Learning:



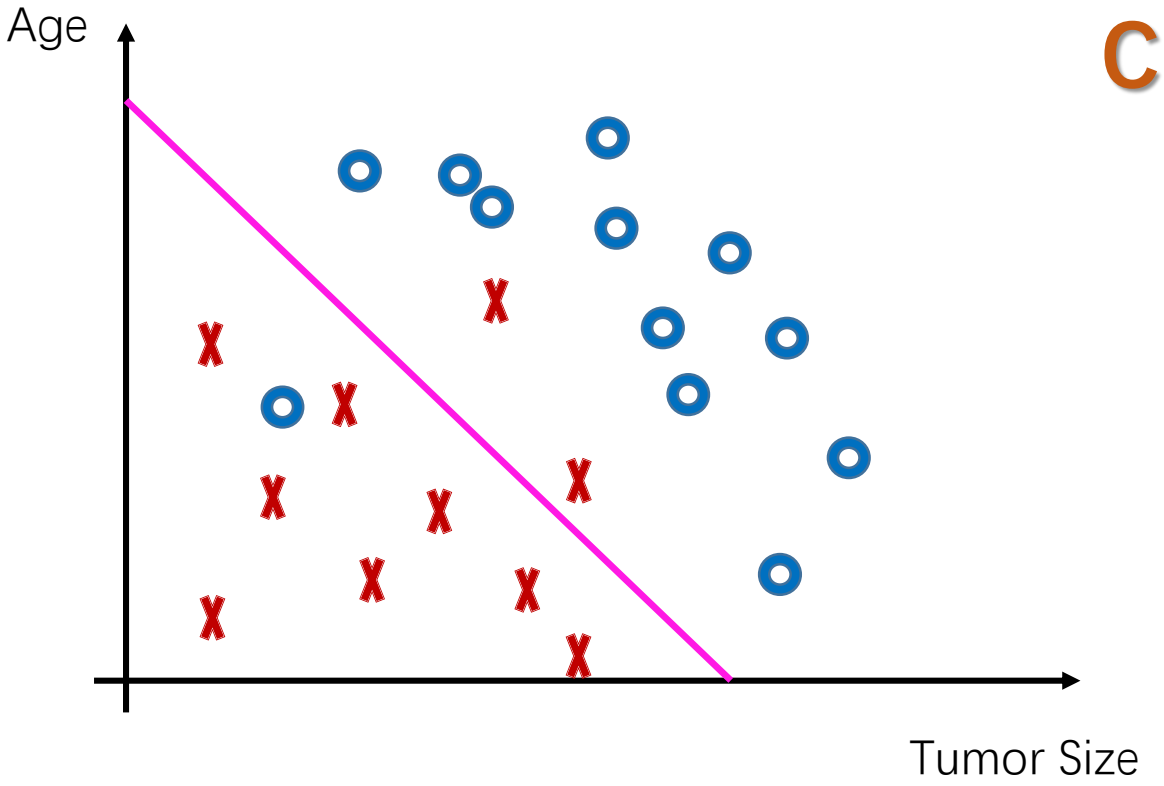
# I. Intro To ML

## A. Supervised Learning:



# I. Intro To ML

## A. Supervised Learning:




**Classification**

# I. Intro To ML

## A. Supervised Learning:

### Classification

Image/Video Classification

Segmentation

.....

# I. Intro To ML

## B. Unsupervised Learning:

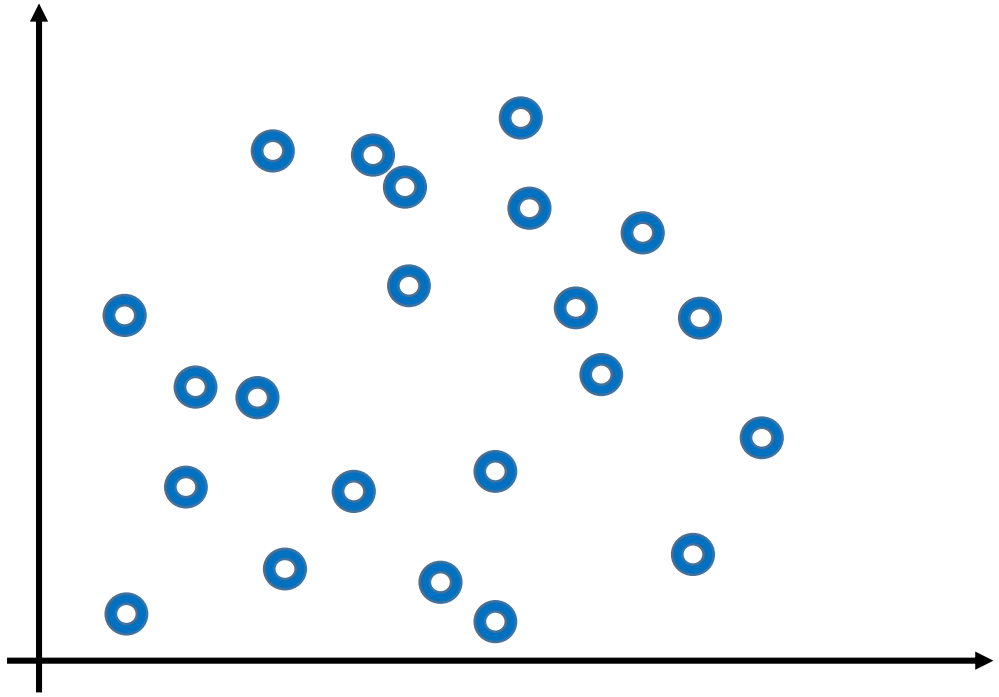
# I. Intro To ML

## B. Unsupervised Learning:

**There is no answers as our standards!**

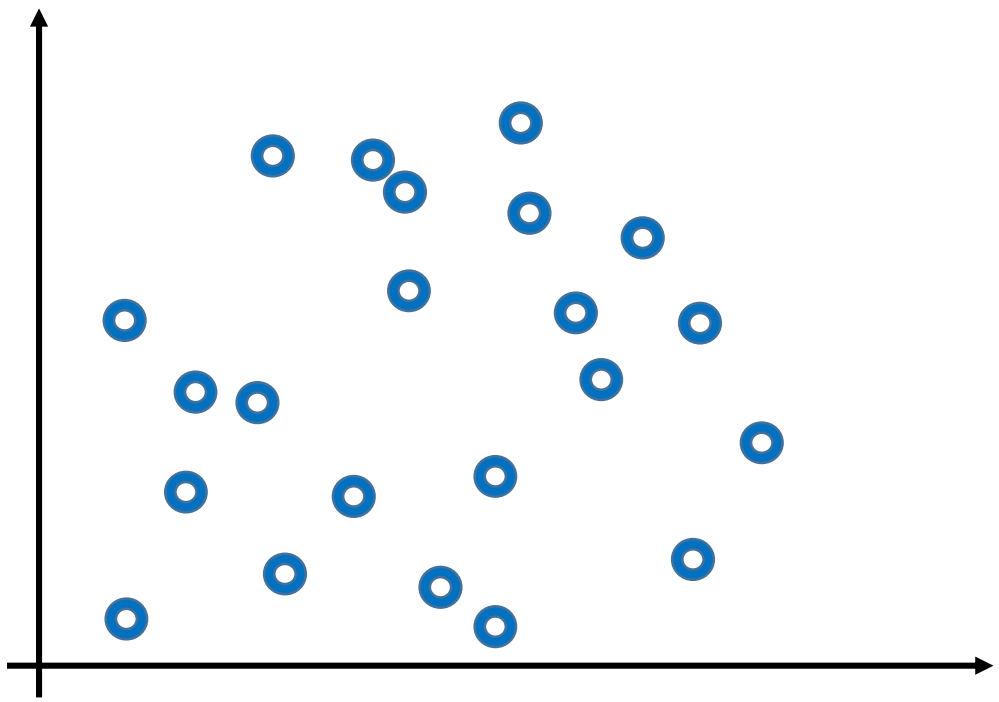
# I. Intro To ML

## B. Unsupervised Learning:



# I. Intro To ML

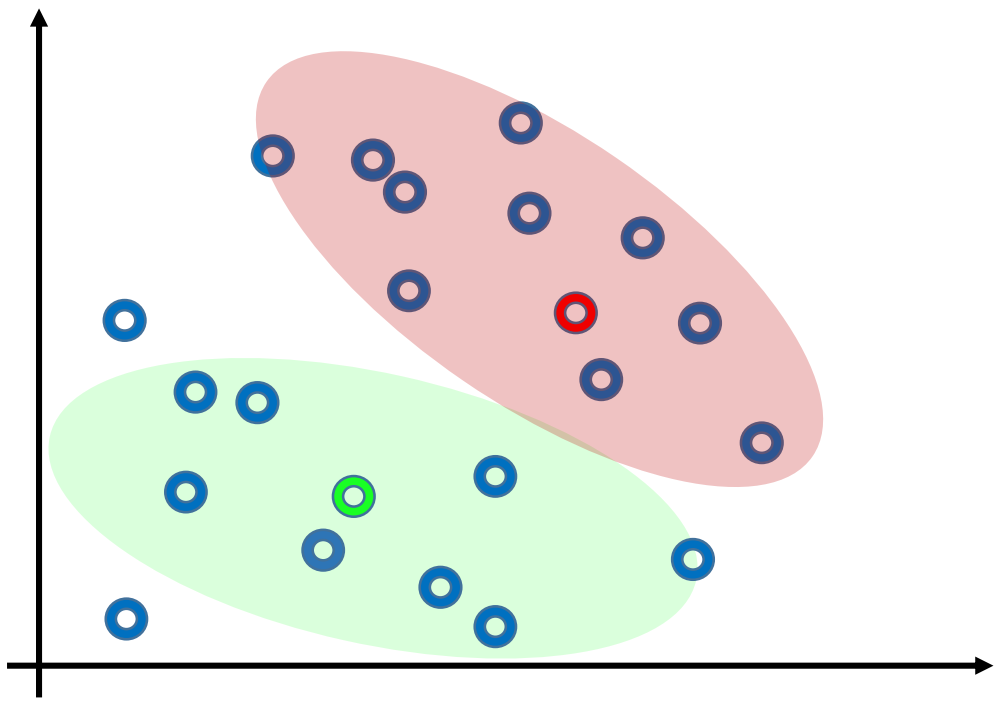
## B. Unsupervised Learning:



**Clustering**

# I. Intro To ML

## B. Unsupervised Learning:



**Clustering**

# I. Intro To ML

## B. Unsupervised Learning:

**Q: How to type a video within 1,000,000,000 videos in a short time?**

# I. Intro To ML

## B. Unsupervised Learning:

**Q: How to type a video within 1,000,000,000 videos in a short time?**

**A: That's classification. We build a model to classify the video.**

# I. Intro To ML

## B. Unsupervised Learning:

**Q: How to type a video within 1,000,000,000 videos in a short time?**

**A: That's classification. We build a model to classify the video.** ?

# I. Intro To ML

## B. Unsupervised Learning:

**Q: How to type a video within 1,000,000,000 videos in a short time?**

**A: That's classification. We build a model to classify the video.**

**?**

**P: 1. Who labels the data?**

**2. Videos are changing. We'll be training all the rest time in the universe.**

# I. Intro To ML

## B. Unsupervised Learning:

**Q: How to type a video within 1,000,000,000 videos in a short time?**

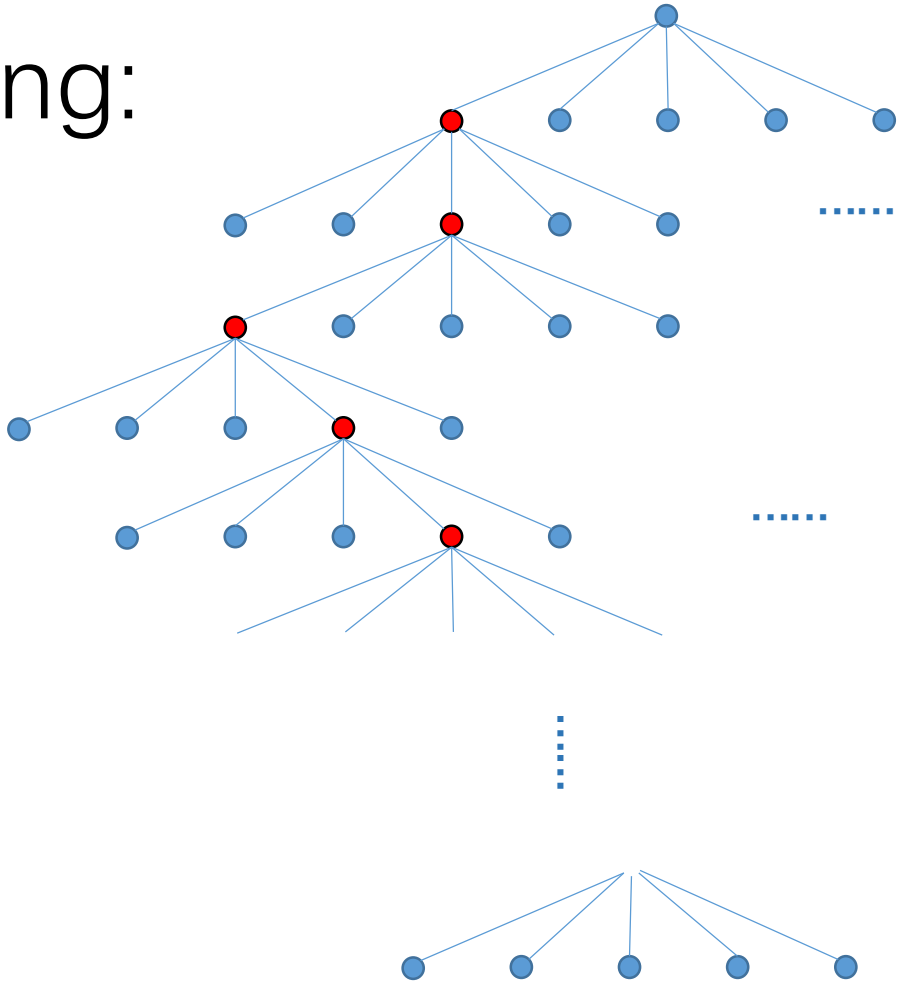
**A: Clustering**

# I. Intro To ML

## B. Unsupervised Learning:

Q: How to type a video within 1,000,000,000 videos in a short time?

A: Clustering per level.
Root the center.
Recurse.



# II. Supervised Learning

# II. Supervised Learning

## C. Linear Regression:

# II. Supervised Learning

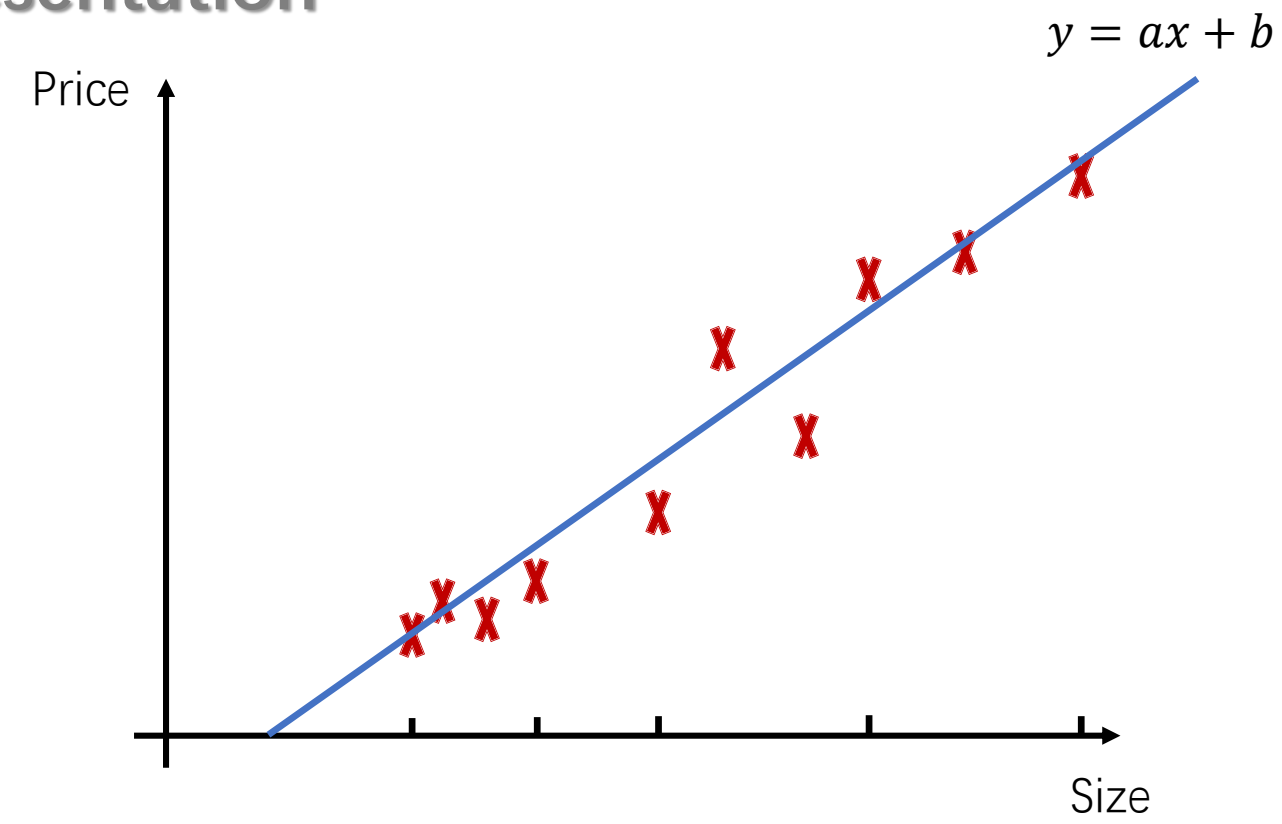
## C. Linear Regression:

**Regression: Fit A Line**

# II. Supervised Learning

## C. Linear Regression:

### C1. Representation



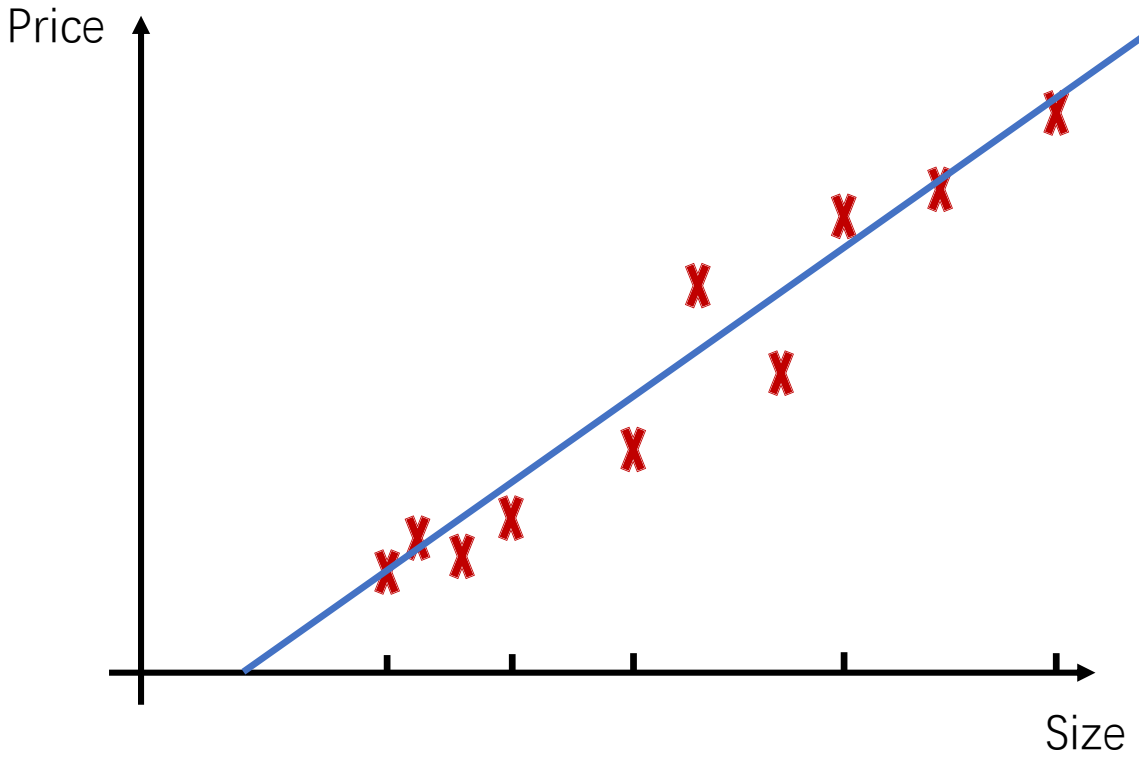
# II. Supervised Learning

## C. Linear Regression:

### C1. Representation

**What shape the line?**

$y = ax + b$



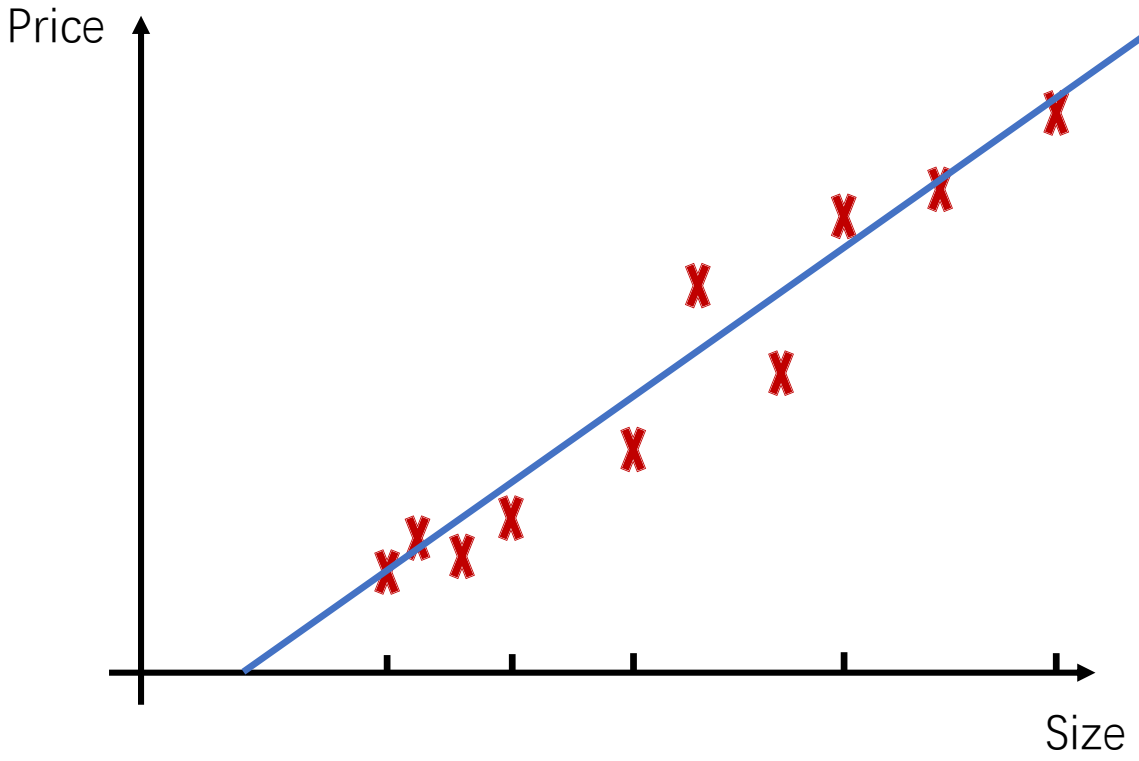
# II. Supervised Learning

## C. Linear Regression:

### C1. Representation

**What shape the line?**

$y = ax + b$



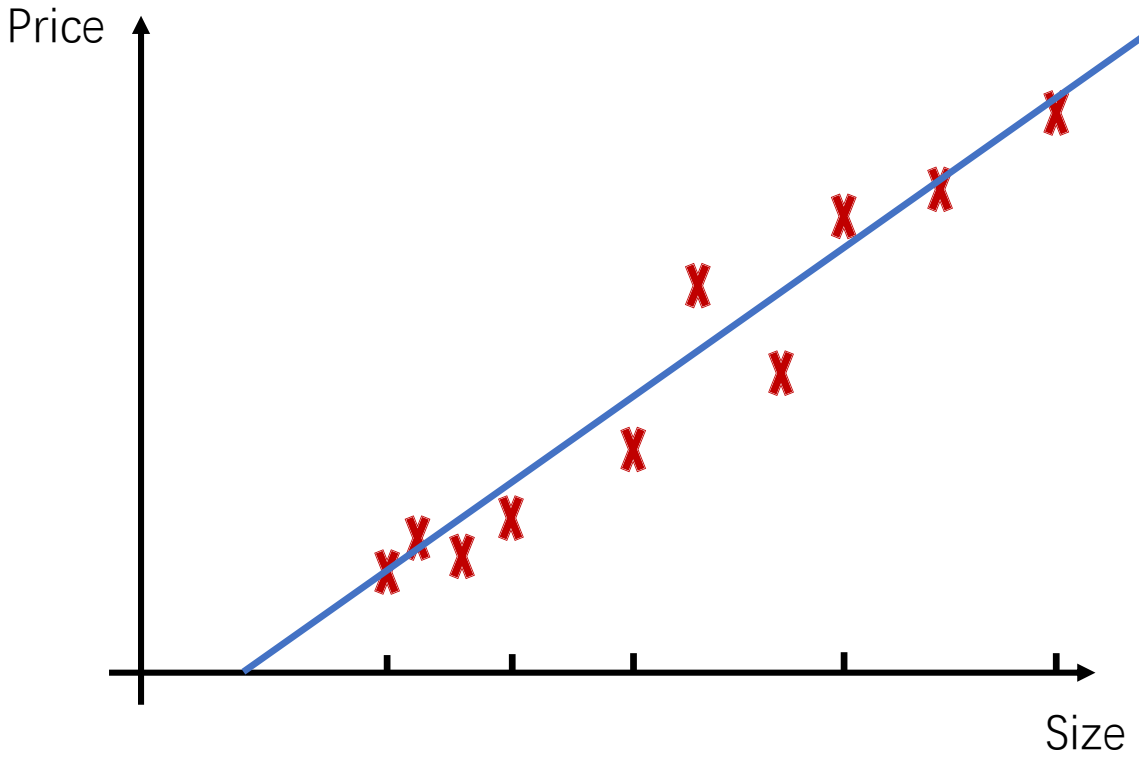
# II. Supervised Learning

## C. Linear Regression:

### C1. Representation

**Let's give a & b a name:**

$y = ax + b$



# II. Supervised Learning

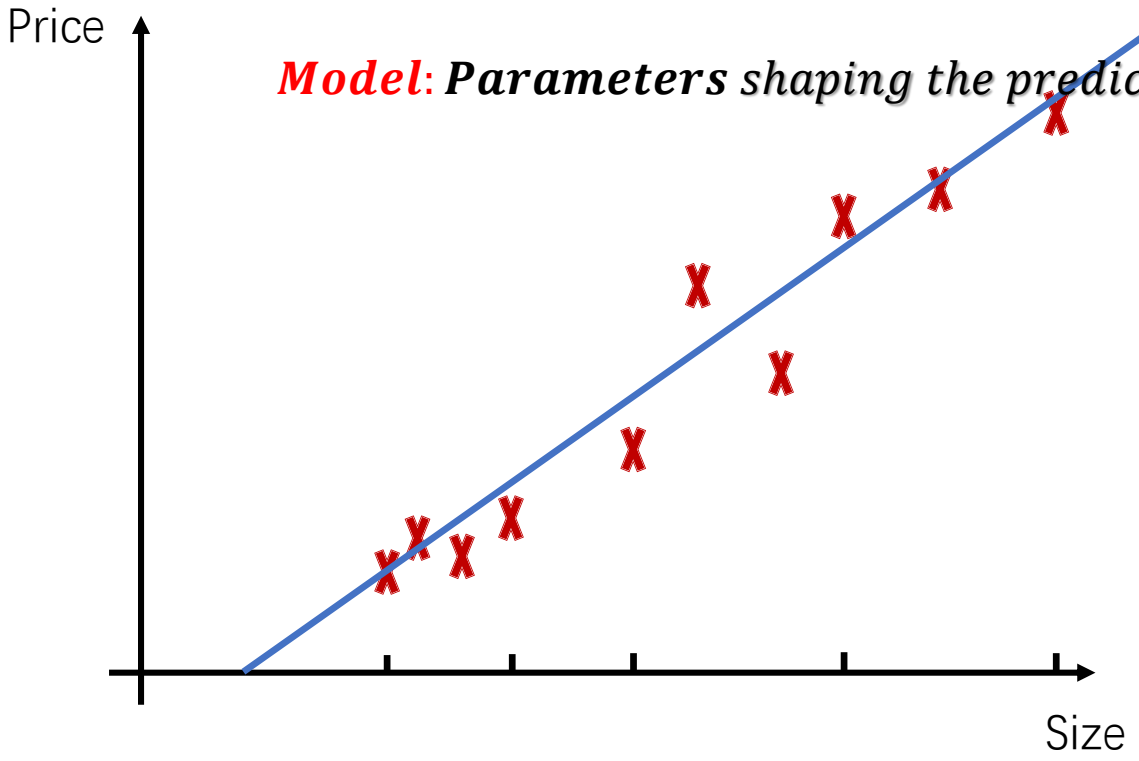
## C. Linear Regression:

### C1. Representation

**Let's give a & b a name: MODEL**

$y = ax + b$

***Model*: *Parameters shaping the predictions are called MODEL***



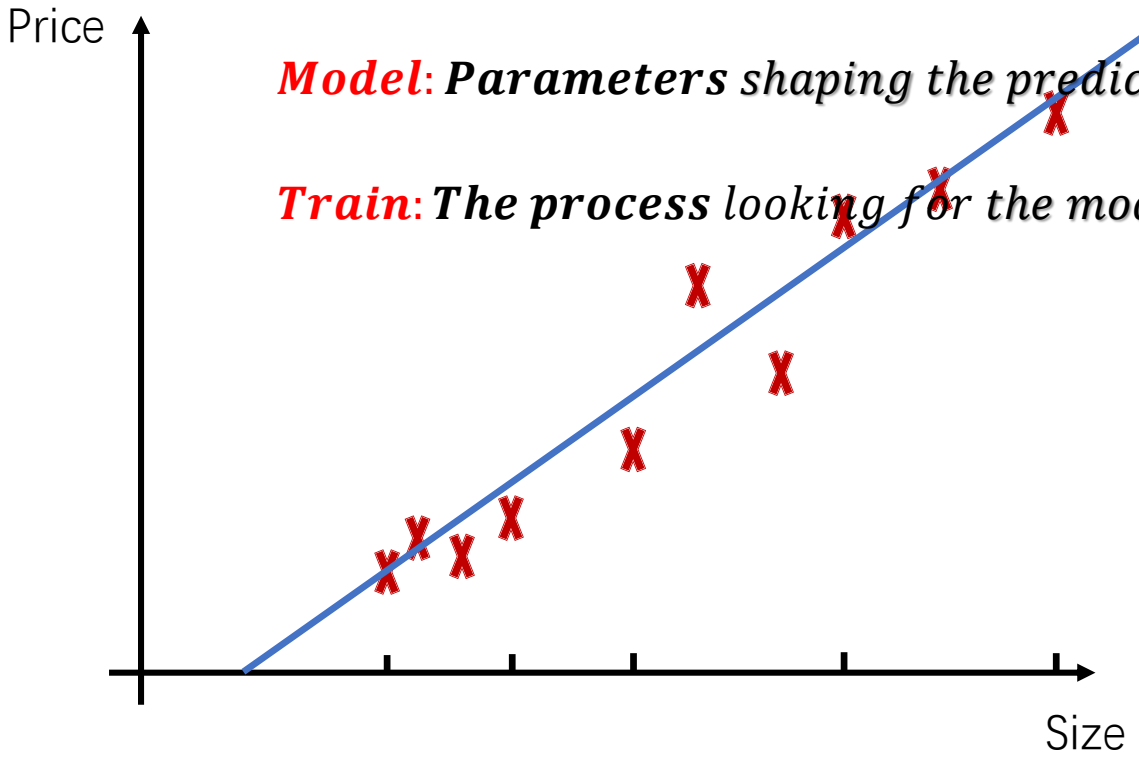
# II. Supervised Learning

## C. Linear Regression:

### C1. Representation

**Let's give a & b a name: MODEL**

$y = ax + b$




***Model***: ***Parameters*** *shaping the predictions* ***are called MODEL***

***Train***: ***The process*** *looking for the model* ($a$ & $b$ *here*).

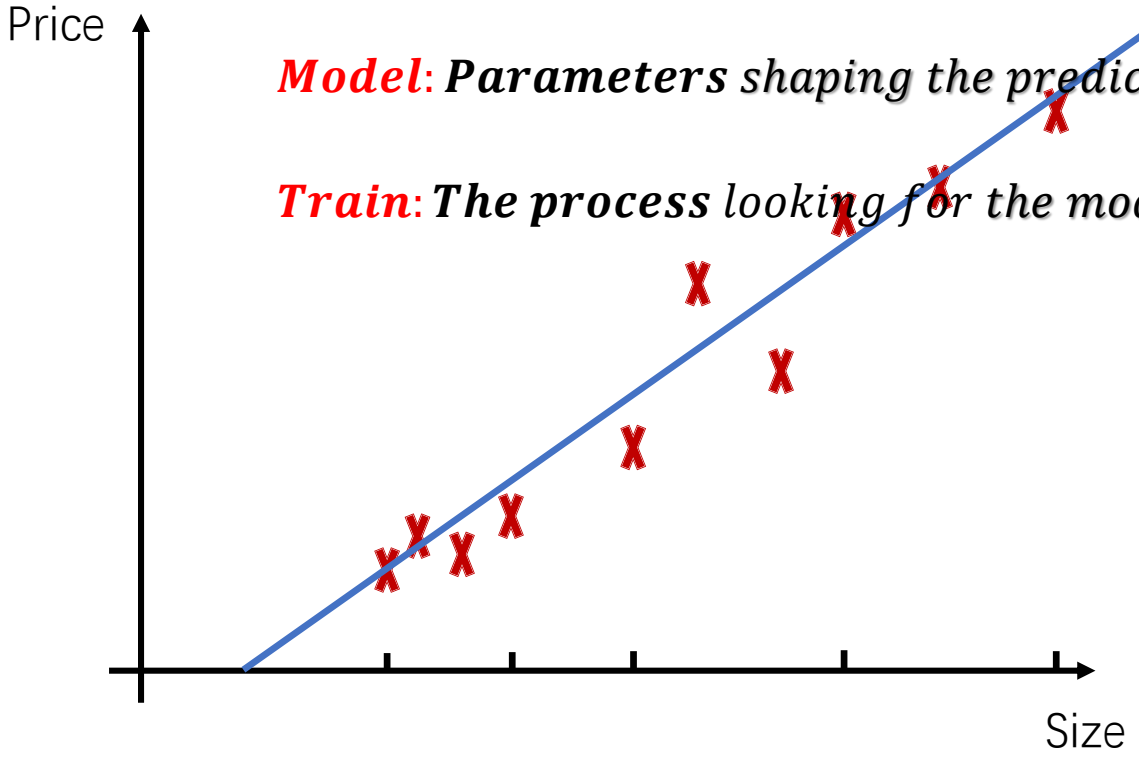
# II. Supervised Learning

## C. Linear Regression:

### C1. Representation

**For generalizing purpose:**

$h_\theta(x) = \theta_0 + \theta_1 x$




***Model***: ***Parameters*** *shaping the predictions* ***are called MODEL***

***Train***: ***The process*** *looking for the model* ($a$ & $b$ *here*).

# II. Supervised Learning

## C. Linear Regression:

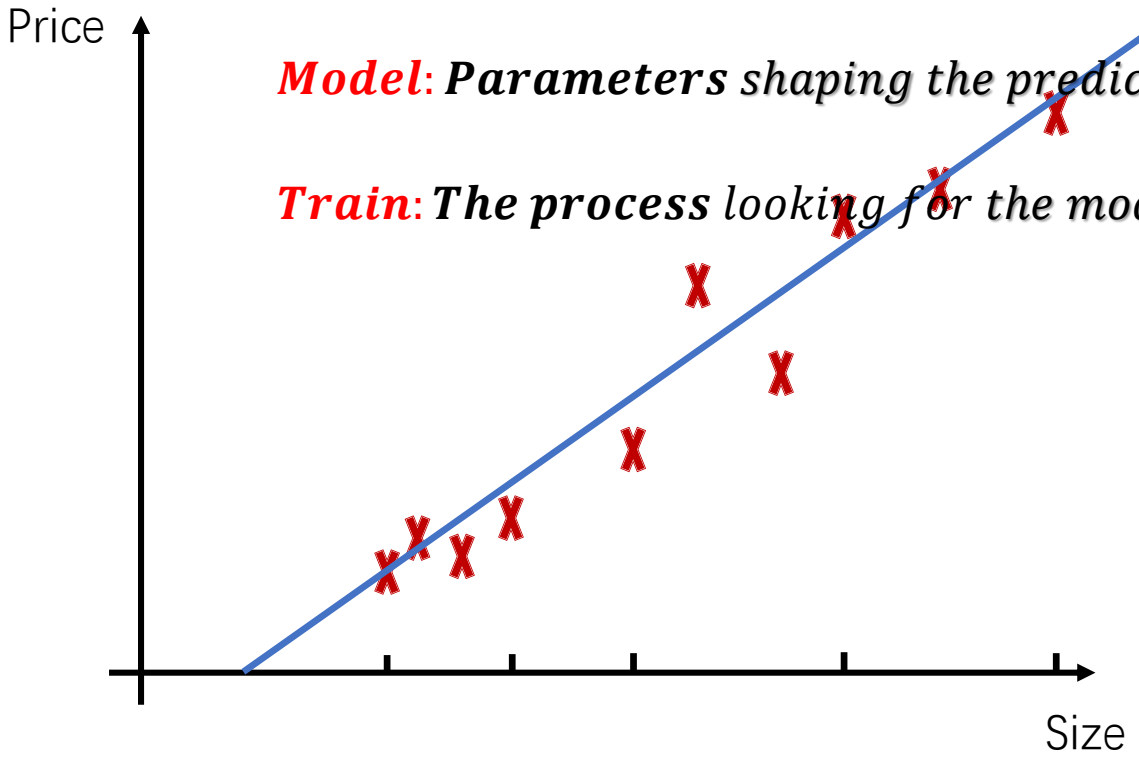
### C1. Representation

For generalizing purpose:

Hypotheses $h_\theta(x) = \theta_0 + \theta_1 x$

***Model***: ***Parameters*** *shaping the predictions* ***are called MODEL***

***Train***: ***The process*** *looking for the model* ($a$ & $b$ *here*).



# II. Supervised Learning

## C. Linear Regression:

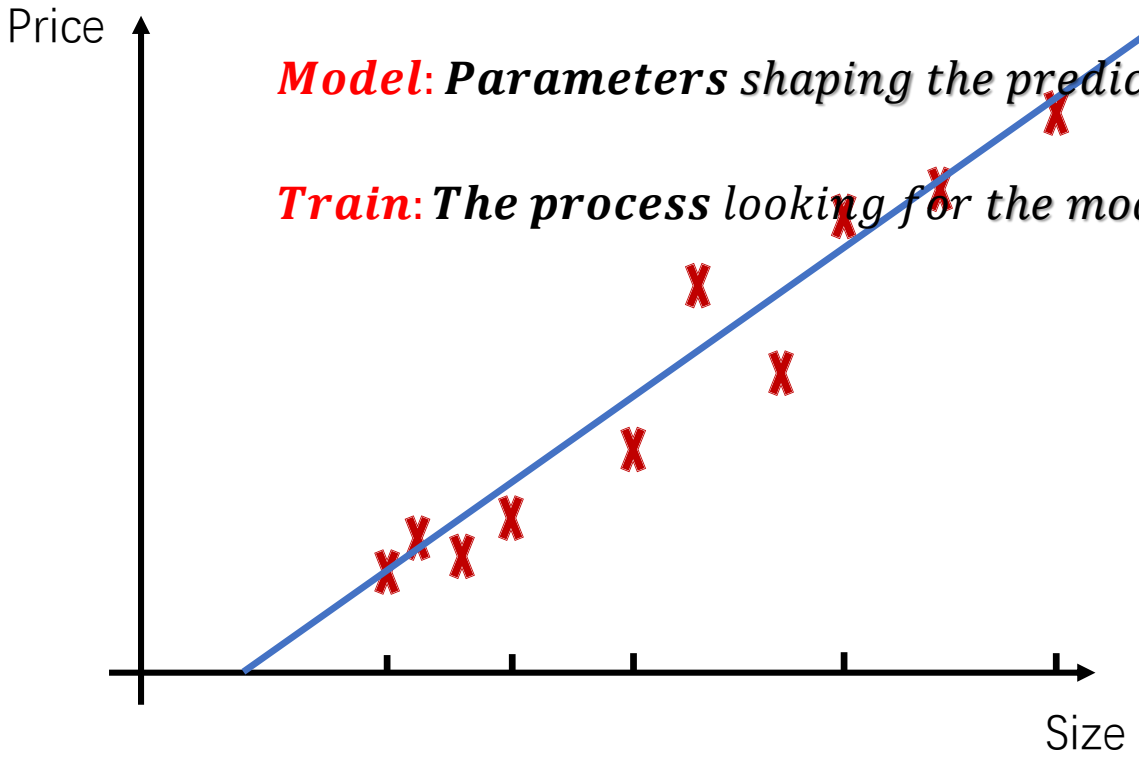
### C1. Representation

**For generalizing purpose:**

$h_\theta(\boldsymbol{x}) = \theta_0 + \theta_1 \boldsymbol{x}$

***Model***: ***Parameters*** *shaping the predictions* ***are called MODEL***

***Train***: ***The process*** *looking for the model (a & b here).*



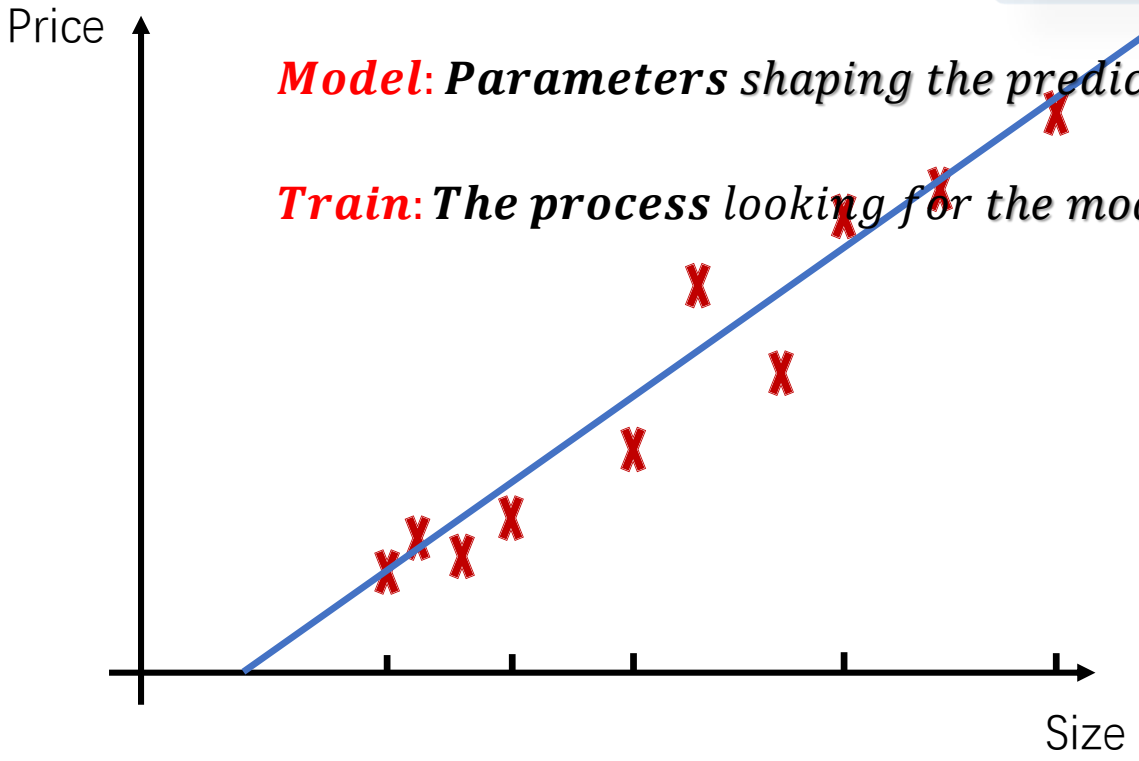
# II. Supervised Learning

## C. Linear Regression:

### C1. Representation

For generalizing purpose:

$$h_\theta(x^i) = \theta_0 + \theta_1 x^i = \boldsymbol{\theta x^i}$$




***Model***: ***Parameters*** *shaping the predictions* ***are called MODEL***

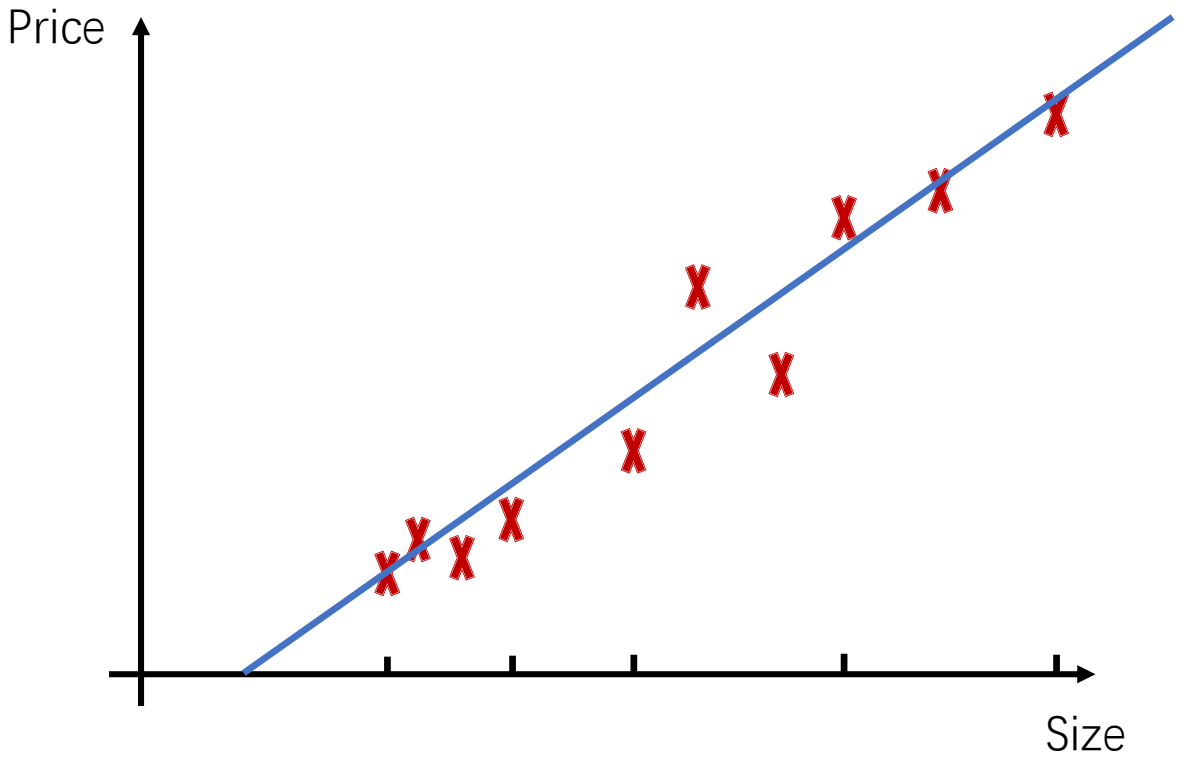
***Train***: ***The process*** *looking for the model (a & b here).*

# II. Supervised Learning

## C. Linear Regression:

### C2. How to choose $\theta$-Cost Function

$$h_\theta(x^i) = \theta_0 + \theta_1 x^i = \boldsymbol{\theta x^i}$$

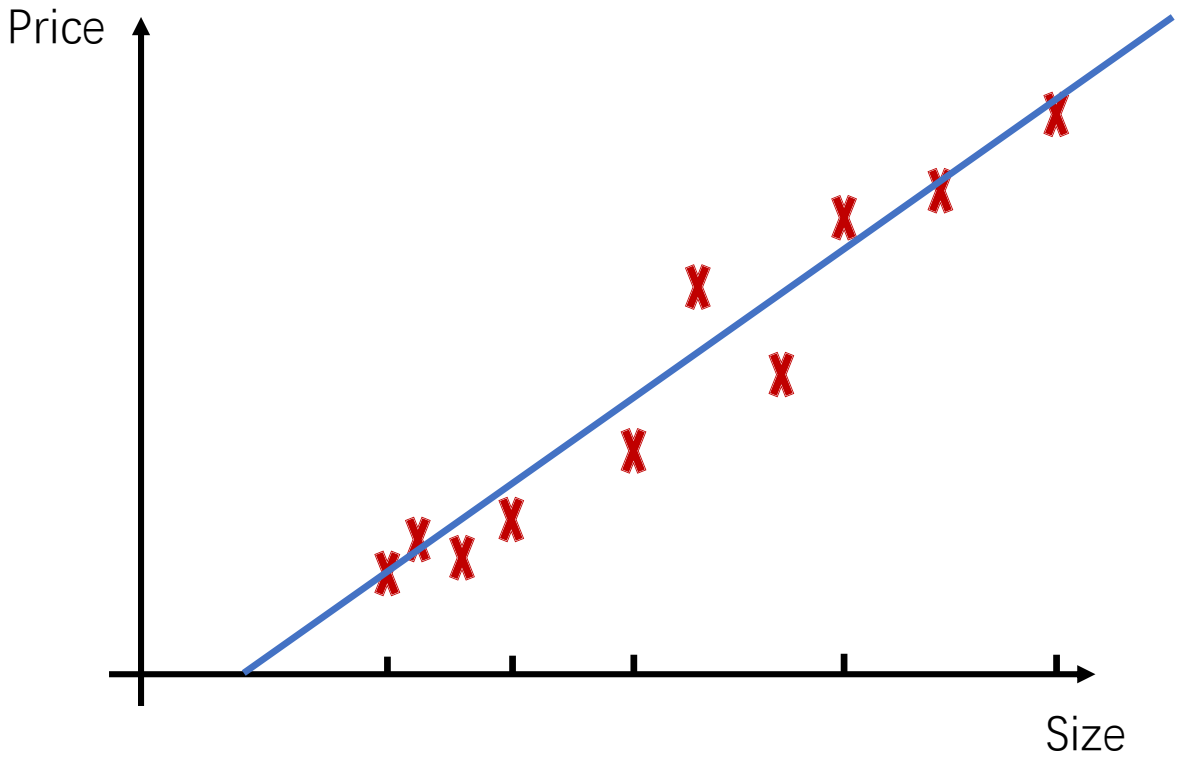


# II. Supervised Learning

## C. Linear Regression:

### C2. How to choose $\theta$-Cost Function

$$h_\theta(x^i) = \theta_0 + \theta_1 x^i = \boldsymbol{\theta x^i}$$




**Cost:**

Difference between hypotheses and reality

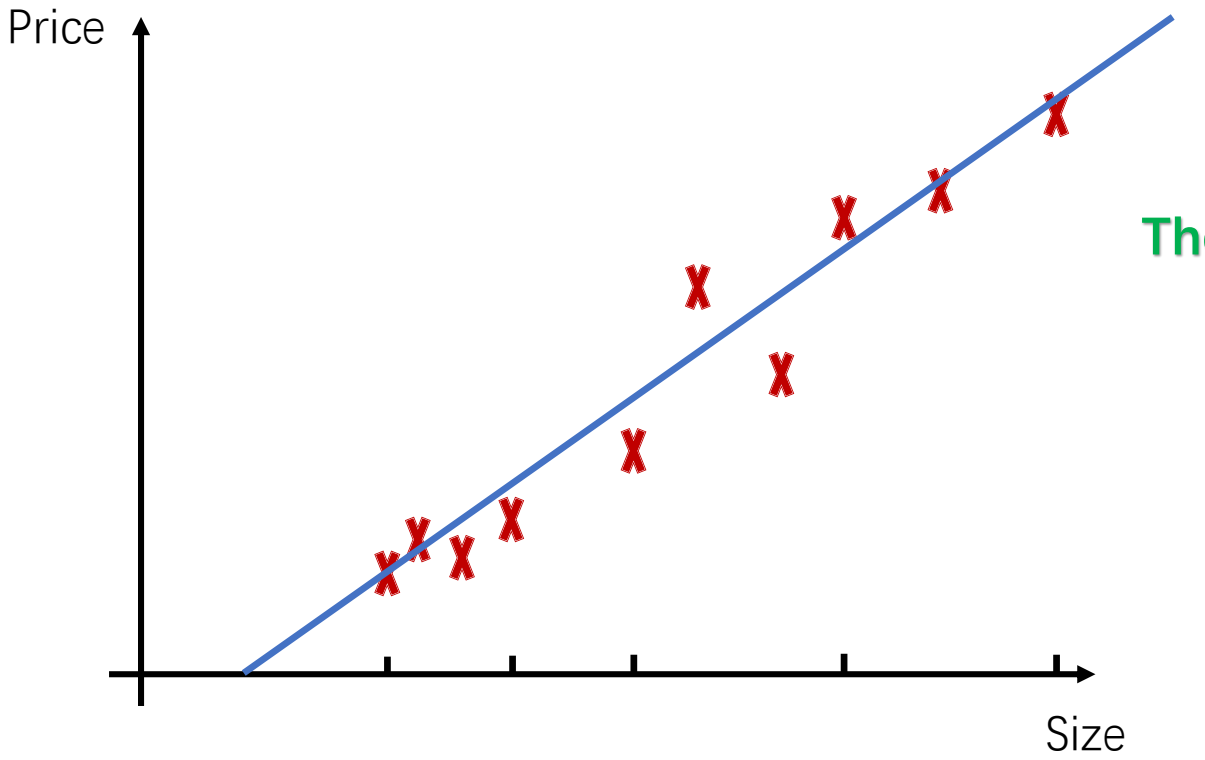
**Label data**

# II. Supervised Learning

## C. Linear Regression:

### C2. How to choose $\theta$-Cost Function

$$h_\theta(x^i) = \theta_0 + \theta_1 x^i = \boldsymbol{\theta x^i}$$




**Cost:**

Difference between hypotheses and reality

Label data

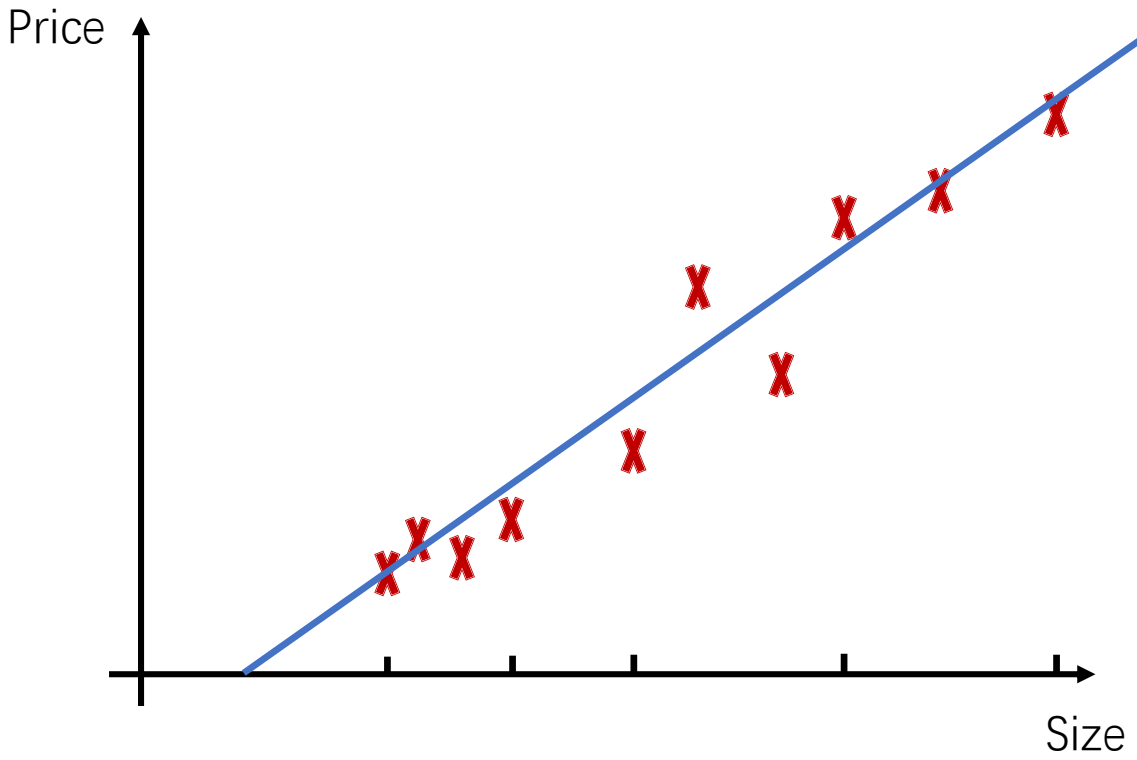
**The less the difference/cost, the better the model**

# II. Supervised Learning

## C. Linear Regression:

### C2. How to choose $\theta$-Cost Function

$$h_\theta(x^i) = \theta_0 + \theta_1 x^i = \boldsymbol{\theta x^i}$$




**Cost:**

Difference between hypotheses and reality

Label data

**The less the difference/cost, the better the model**

$$Cost = J(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^i) - y^i)^2$$

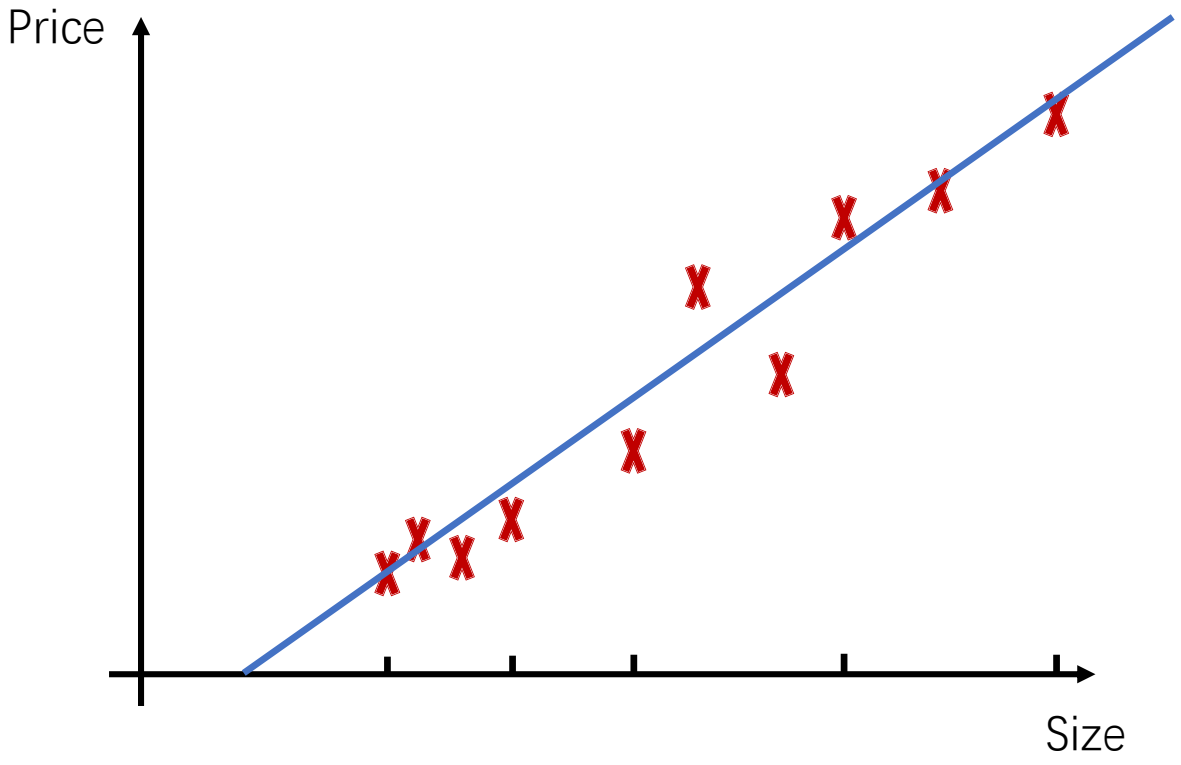
$$\underset{\theta_0,\theta_1}{minimize} J(\theta_0, \theta_1)$$

Squared Error Function

# II. Supervised Learning

## C. Linear Regression:

### C2. How to choose $\theta$-Cost Function




**Hypothesis:** $h_\theta(x^i) = \theta_0 + \theta_1 x^i = \boldsymbol{\theta x^i}$

**Parameters:** $\theta_0, \theta_1$

**Cost Func:** $J(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^i) - y^i)^2$

**Goal:** $\underset{\theta_0,\theta_1}{minimize} J(\theta_0, \theta_1)$

# II. Supervised Learning

## C. Linear Regression:

### C3. How to calculate-Gradient Descent

**Gradient Descent Algo:**

$$\begin{aligned}&\textit{while not converge}\ \{\\&\quad \theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta) \qquad \textit{for } j = 0 \textit{ and } j = 1\\&\}\end{aligned}$$

$$\begin{aligned}&\text{temp0} := \theta_0 - \alpha \tfrac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)\\&\text{temp1} := \theta_1 - \alpha \tfrac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)\\&\theta_0 := \text{temp0}\\&\theta_1 := \text{temp1}\end{aligned}$$

**Hypothesis:** $h_\theta(x^i) = \theta_0 + \theta_1 x^i = \boldsymbol{\theta x^i}$

**Parameters:** $\theta_0, \theta_1$

**Cost Func:** $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^i) - y^i)^2$

**Goal:** $\underset{\theta_0, \theta_1}{minimize} J(\theta_0, \theta_1)$

# II. Supervised Learning

## C. Linear Regression:

### C3. How to calculate-Gradient Descent

**Gradient Descent Algo:**

$$while\ not\ converge\ \{$$

$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta) \qquad for\ j = 0\ and\ j = 1$$

$$\}$$

$$\frac{\partial}{\partial \theta_0} J(\theta_0) =$$

$$\frac{\partial}{\partial \theta_1} J(\theta_1) =$$

**Hypothesis:** $h_\theta(x^i) = \theta_0 + \theta_1 x^i = \boldsymbol{\theta x^i}$

**Parameters:** $\theta_0, \theta_1$

**Cost Func:** $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^i) - y^i)^2$

**Goal:** $\underset{\theta_0, \theta_1}{minimize} J(\theta_0, \theta_1)$

# II. Supervised Learning

## C. Linear Regression:

### C3. How to calculate-Gradient Descent

**Gradient Descent Algo:**

$$\begin{aligned}&\textit{while not converge}\ \{\\&\qquad \theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta) \qquad \textit{for } j = 0 \textit{ and } j = 1\\&\}\end{aligned}$$

$$\frac{\partial}{\partial \theta_0} J(\theta_0) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^i) - y^i)$$

$$\frac{\partial}{\partial \theta_1} J(\theta_1) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^i) - y^i) \cdot x^i$$

**Hypothesis:** $h_\theta(x^i) = \theta_0 + \theta_1 x^i = \boldsymbol{\theta x^i}$

**Parameters:** $\theta_0, \theta_1$

**Cost Func:** $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^i) - y^i)^2$

**Goal:** $\underset{\theta_0, \theta_1}{minimize} J(\theta_0, \theta_1)$

# II. Supervised Learning

Explain

## C. Linear Regression:

### C3. Illustration-Gradient Descent

**Hypothesis:** $h_\theta(x^i) = \theta_0 + \theta_1 x^i = \boldsymbol{\theta x^i}$

**Parameters:** $\theta_0, \theta_1$

**Cost Func:** $J(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^i) - y^i)^2$

**Goal:** $\underset{\theta_0,\theta_1}{minimize} J(\theta_0, \theta_1)$

$J(\theta_j)$: ***Quadratic Function***

$\frac{d}{d\theta_j}J(\theta_j)$: ***Slope***

$\theta_j = \theta_j - \alpha\frac{\partial}{\partial\theta_j}J(\theta)$: ***Gradient Descent***

$\theta_1 > extreme$

$\theta_1 < extreme$

# II. Supervised Learning

Explain

## C. Linear Regression:

### C3. Illustration-Gradient Descent

**Hypothesis:** $h_\theta(x^i) = \theta_0 + \theta_1 x^i = \boldsymbol{\theta x^i}$

**Parameters:** $\theta_0, \theta_1$

**Cost Func:** $J(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^i) - y^i)^2$

**Goal:** $\underset{\theta_0,\theta_1}{minimize} J(\theta_0, \theta_1)$

$J(\theta_j)$: ***Quadratic Function***

$\frac{d}{d\theta_j}J(\theta_j)$: ***Slope***

$\theta_j = \theta_j - \alpha\frac{\partial}{\partial\theta_j}J(\theta)$: ***Gradient Descent***

$\alpha$: *too small*

$\alpha$: *too big*

# II. Supervised Learning

Explain

## C. Linear Regression:

### C3. Illustration-Gradient Descent

$J(\theta_j)$: ***Quadratic Function***

$\frac{d}{d\theta_j}J(\theta_j)$: ***Slope***

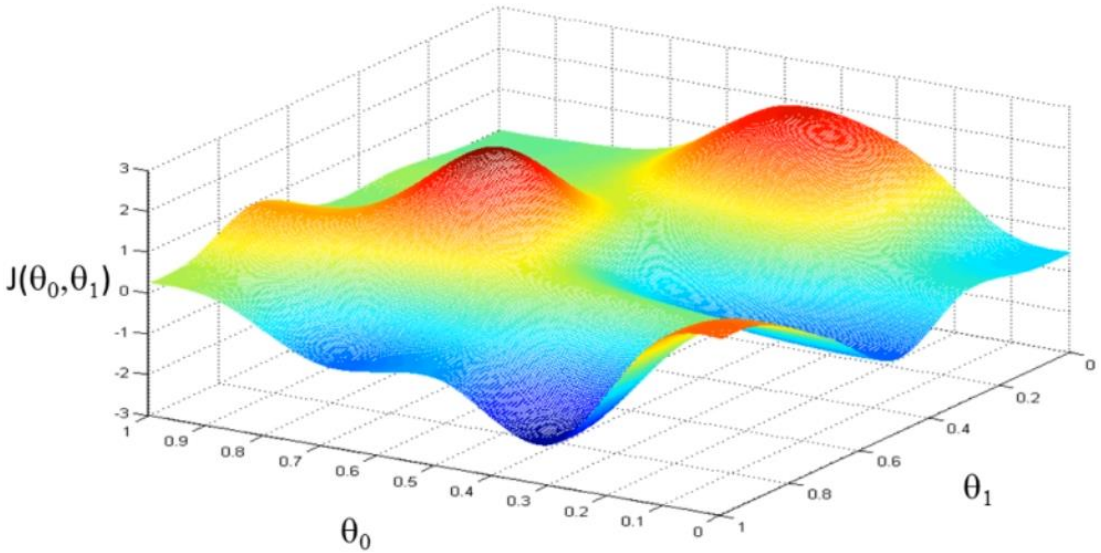
$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$: ***Gradient Descent***

**Hypothesis:** $h_\theta(x^i) = \theta_0 + \theta_1 x^i = \boldsymbol{\theta x^i}$

**Parameters:** $\theta_0, \theta_1$

**Cost Func:** $J(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^i) - y^i)^2$

**Goal:** $\underset{\theta_0,\theta_1}{minimize} J(\theta_0, \theta_1)$



# II. Supervised Learning

Explain

## C. Linear Regression:

### C3. Illustration-Gradient Descent

$J(\theta_j)$: ***Quadratic Function***

$\frac{d}{d\theta_j}J(\theta_j)$: ***Slope***

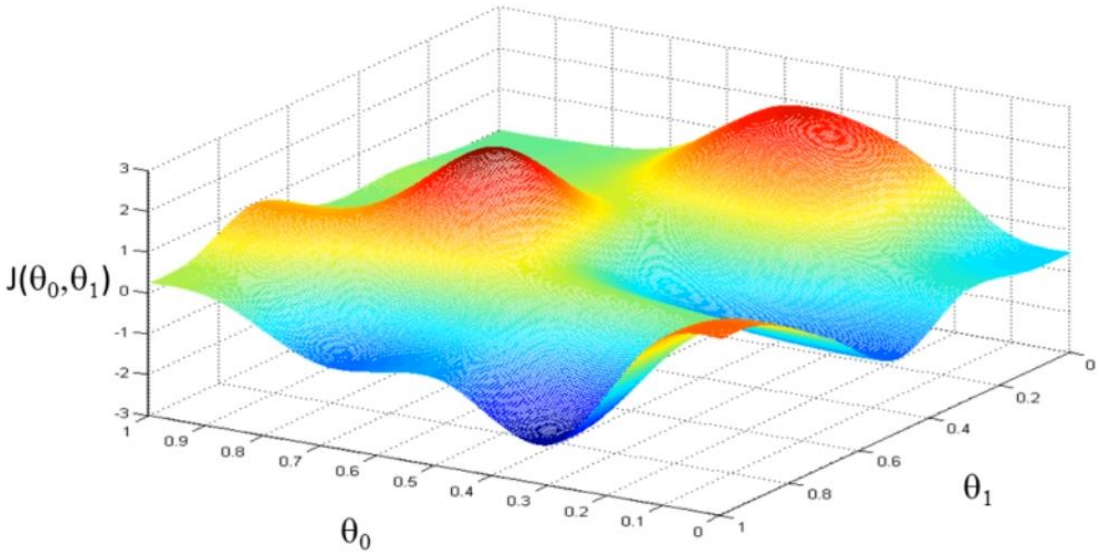
$\theta_j = \theta_j - \alpha\frac{\partial}{\partial\theta_j}J(\theta)$: ***Gradient Descent***

**Hypothesis:** $h_\theta(x^i) = \theta_0 + \theta_1 x^i = \boldsymbol{\theta x^i}$

**Parameters:** $\theta_0, \theta_1$

**Cost Func:** $J(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^i) - y^i)^2$

**Goal:** $\underset{\theta_0,\theta_1}{minimize} J(\theta_0, \theta_1)$




**Get to local minimum**

# II. Supervised Learning

## C. Linear Regression:

### C4. Extension-Multiple Variables

**Price:**

Size

# II. Supervised Learning

## C. Linear Regression:

### C4. Extension-Multiple Variables

**Price:**

Size, Community, Floor, Far, School, Hospital,

Subway, Traffic, Business, Entertainment,

Market, Policy, ……

# II. Supervised Learning

## C. Linear Regression:

### C4. Extension-Multiple Variables

**Price:**

$x_1$ Size, $x_2$ Community, $x_3$ Floor, $x_4$ Far, $x_5$ School, $x_6$ Hospital,

$x_7$ Subway, $x_8$ Traffic, $x_9$ Business, $x_{10}$ Entertainment,

$x_{11}$ Market, $x_{12}$ Policy, …… $x_n$

$x_0 = 1$

# II. Supervised Learning

## C. Linear Regression:

### C4. Extension-Multiple Variables

**Price:**

$x_1$ Size, $x_2$ Community, $x_3$ Floor, $x_4$ Far, $x_5$ School, $x_6$ Hospital,

$x_7$ Subway, $x_8$ Traffic, $x_9$ Business, $x_{10}$ Entertainment,

$x_{11}$ Market, $x_{12}$ Policy, ····· $x_n$ $\quad x_0 = 1$

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n = \boldsymbol{\theta}^T \boldsymbol{x}$$

# II. Supervised Learning

Explain

## C. Linear Regression:

### C4. Extension-Multiple Variables

**Hypothesis:**

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n = \boldsymbol{\theta}^T \boldsymbol{x}$$

$$h_\theta(x^i) = \theta_0 + \theta_1 x_1^i + \theta_2 x_2^i + \cdots + \theta_n x_n^i = \boldsymbol{\theta}^T \boldsymbol{x}^i$$

$n$: $dimension's\ total\ number$
$j$: $dimension\ id\ of\ a\ sample$

$m$: $sample's\ total\ number$
$i$: $sample's\ id$

**Gradient Descent:**

$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_j)$$

$$= \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^i) - y^i) \cdot x_j^i$$

# II. Supervised Learning

Explain: Why Elongated

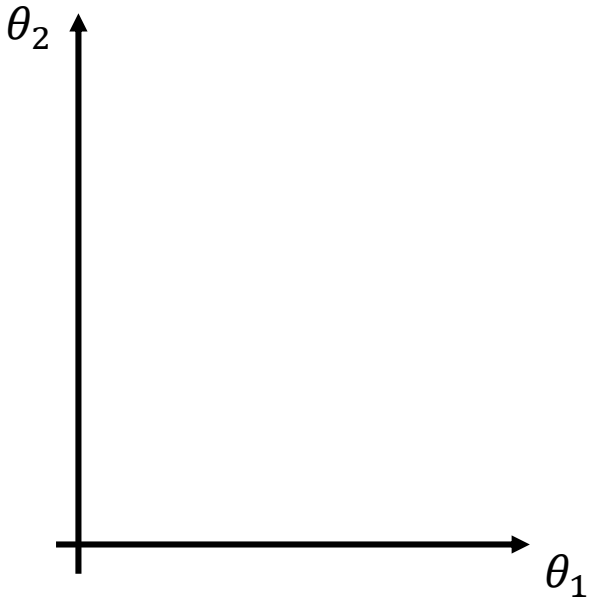
## C. Linear Regression:

### C5. Practice-Feature Scaling

**E.G.:**

$x_1 = (0, 500)$ — $size$

$x_2 = (0,10)$ — $\#\ of\ bedrooms$



# II. Supervised Learning

Explain: Why Zigzag

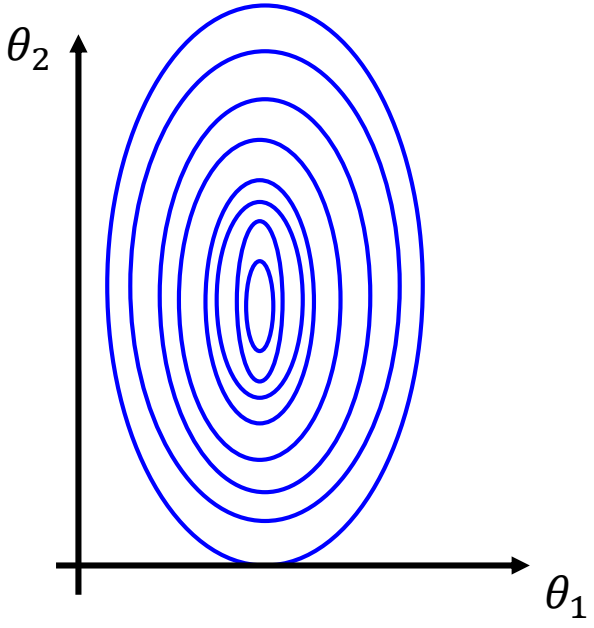
## C. Linear Regression:

### C5. Practice-Feature Scaling

**E.G.:**

$x_1 = (0, 500)$ — $size$

$x_2 = (0,10)$ — $\#\ of\ bedrooms$




**Why Elongated Ellipse?**

**Why Zigzag?**

**Why the gradient along with the "narrow" dimension won't be exploded?**

# II. Supervised Learning

## C. Linear Regression:

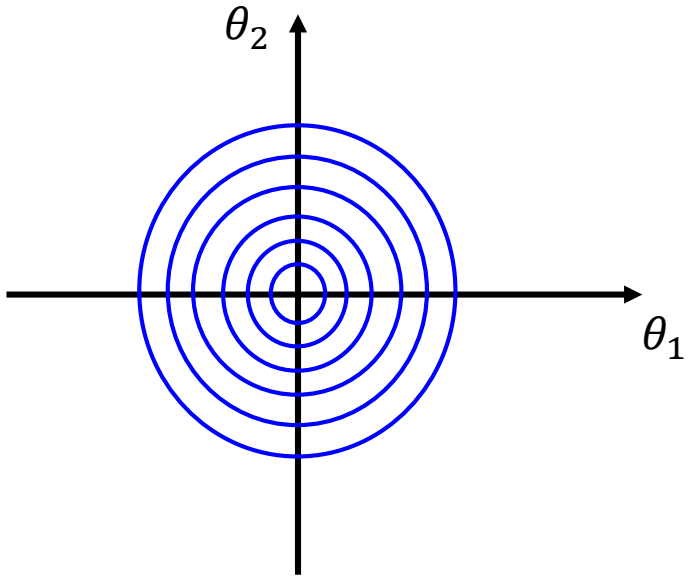
### C5. Practice-Feature Scaling

**E.G.:**

$x_1 = (0, 500)$ — $size$

$x_2 = (0,10)$ — $\#\ of\ bedrooms$

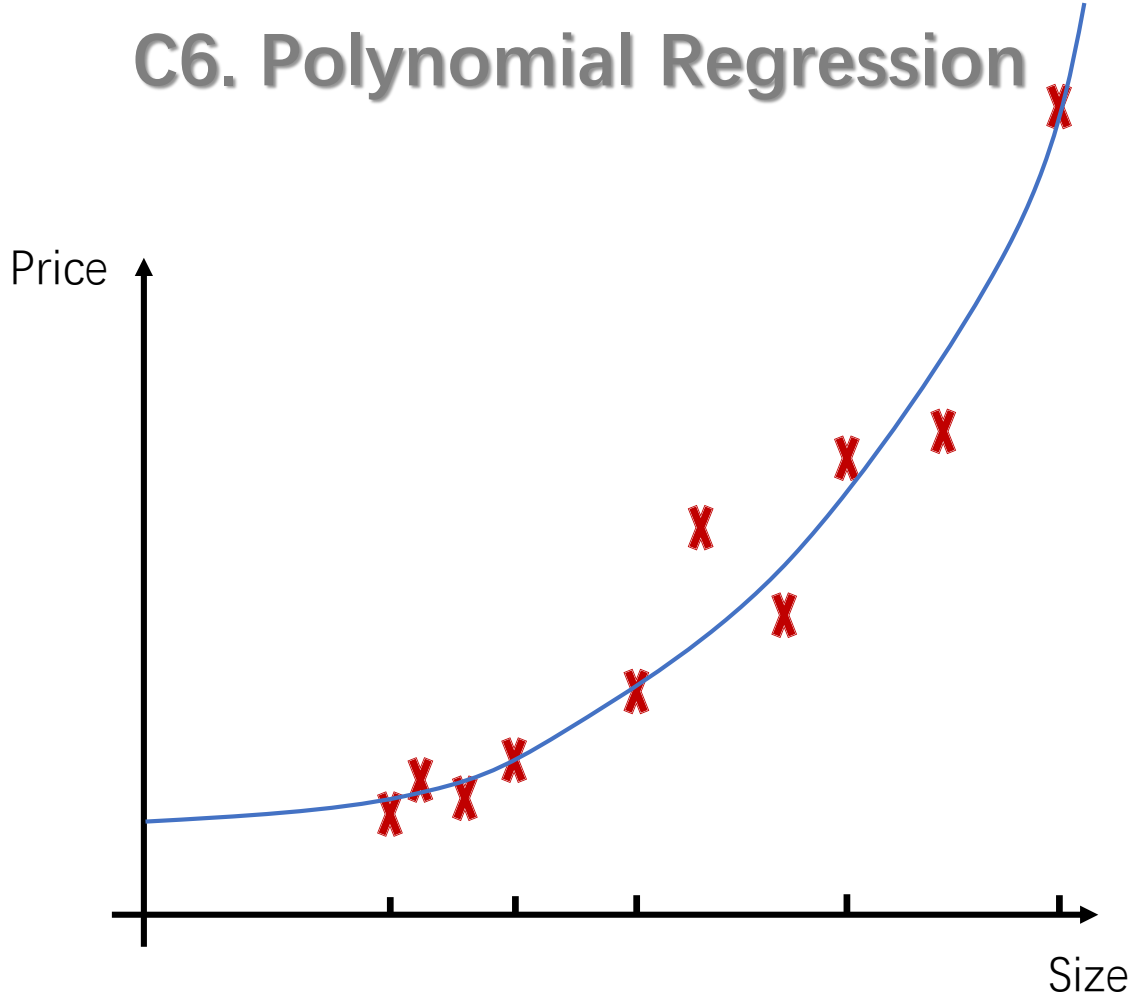
$$x_j' = \frac{x_j - \overline{x_j}}{\sigma_j} \quad s.t\ N \sim (0, 1)$$



# II. Supervised Learning

## C. Linear Regression:

### C6. Polynomial Regression




$$h_\theta(x) = \theta_0 + \theta_1 x + \theta_1 x^2$$

$$= \theta_0 + \theta_1(size) + \theta_1(size)^2$$

$$= \theta_0 + \theta_1 x_1 + \theta_1 x_2$$

# II. Supervised Learning

## C. Linear Regression:

### C7. Normal Equation

$$h_\theta(x) = \theta_0 x_0 + \theta_1 x_1 + \cdots + \theta_n x_n$$

$$J(\theta_{0\ldots n}) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

$$\begin{pmatrix} \theta_0 \\ \theta_1 \\ \ldots \\ \theta_n \end{pmatrix} \in \mathbb{R}^{n+1}$$

$$h_\theta(x) = \theta^T x$$

$$J(\theta) = \frac{1}{2m}(X\theta - y)^T(X\theta - y)$$

$$J(\theta) = ((X\theta)^T - y^T)(X\theta - y)$$

$$J(\theta) = (X\theta)^T X\theta - (X\theta)^T y - y^T(X\theta) + y^T y$$

$$J(\theta) = \theta^T X^T X\theta - 2(X\theta)^T y + y^T y$$

$$\frac{\partial J}{\partial \theta} = 2X^T X\theta - 2X^T y = 0$$

$$X^T X\theta = X^T y$$

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

# II. Supervised Learning

## D. Logistic Regression:

# II. Supervised Learning

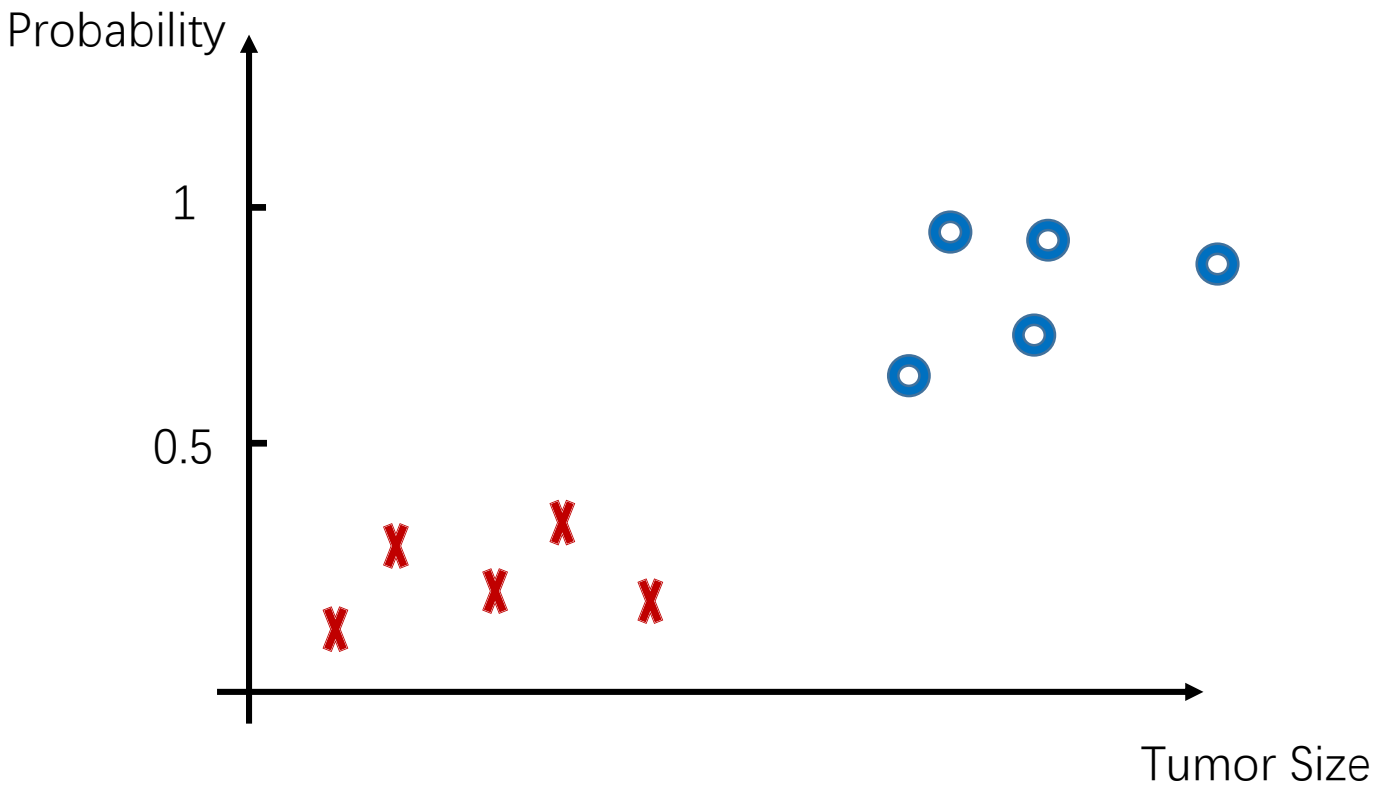
## D. Logistic Regression:

**Classification**

# II. Supervised Learning

## D. Logistic Regression:

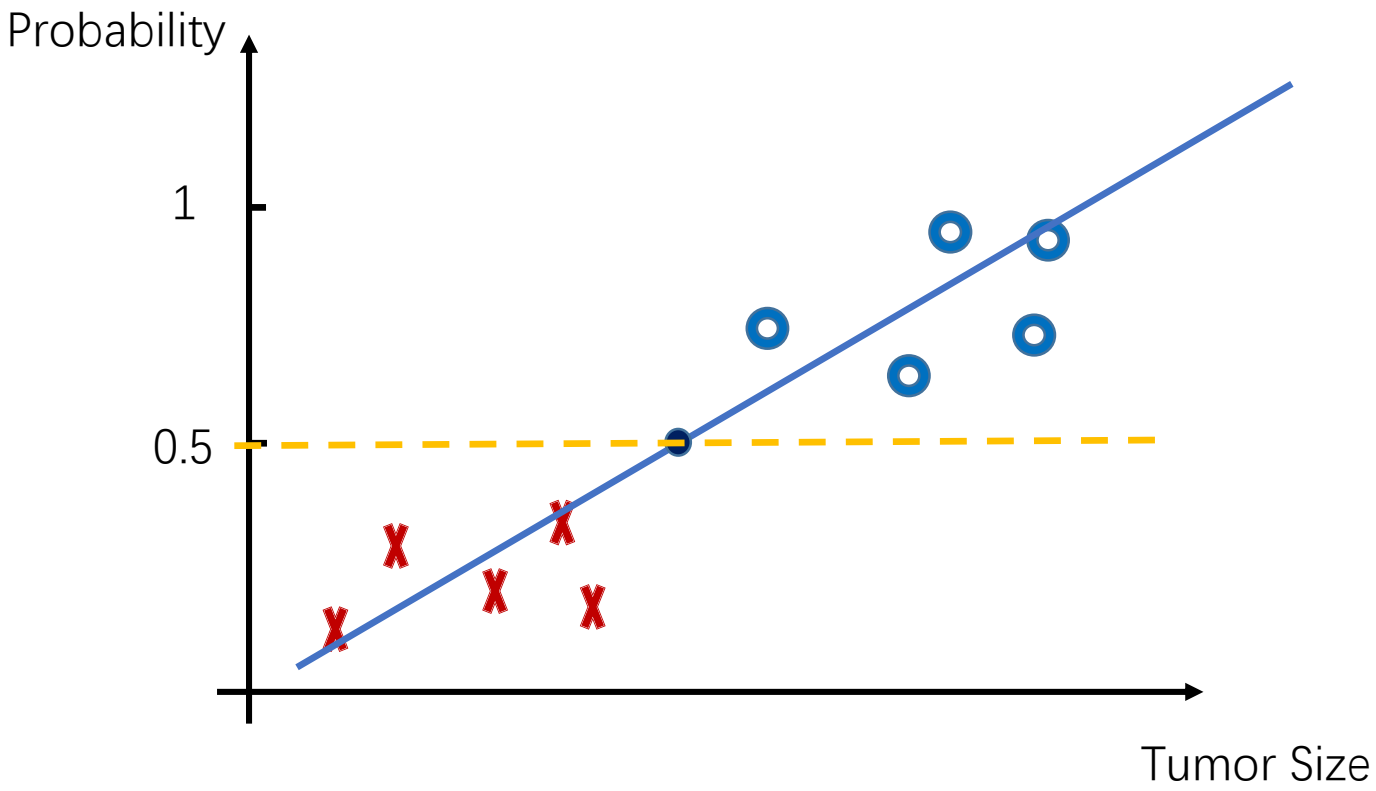
### D1. 2 Classes



# II. Supervised Learning

## D. Logistic Regression:

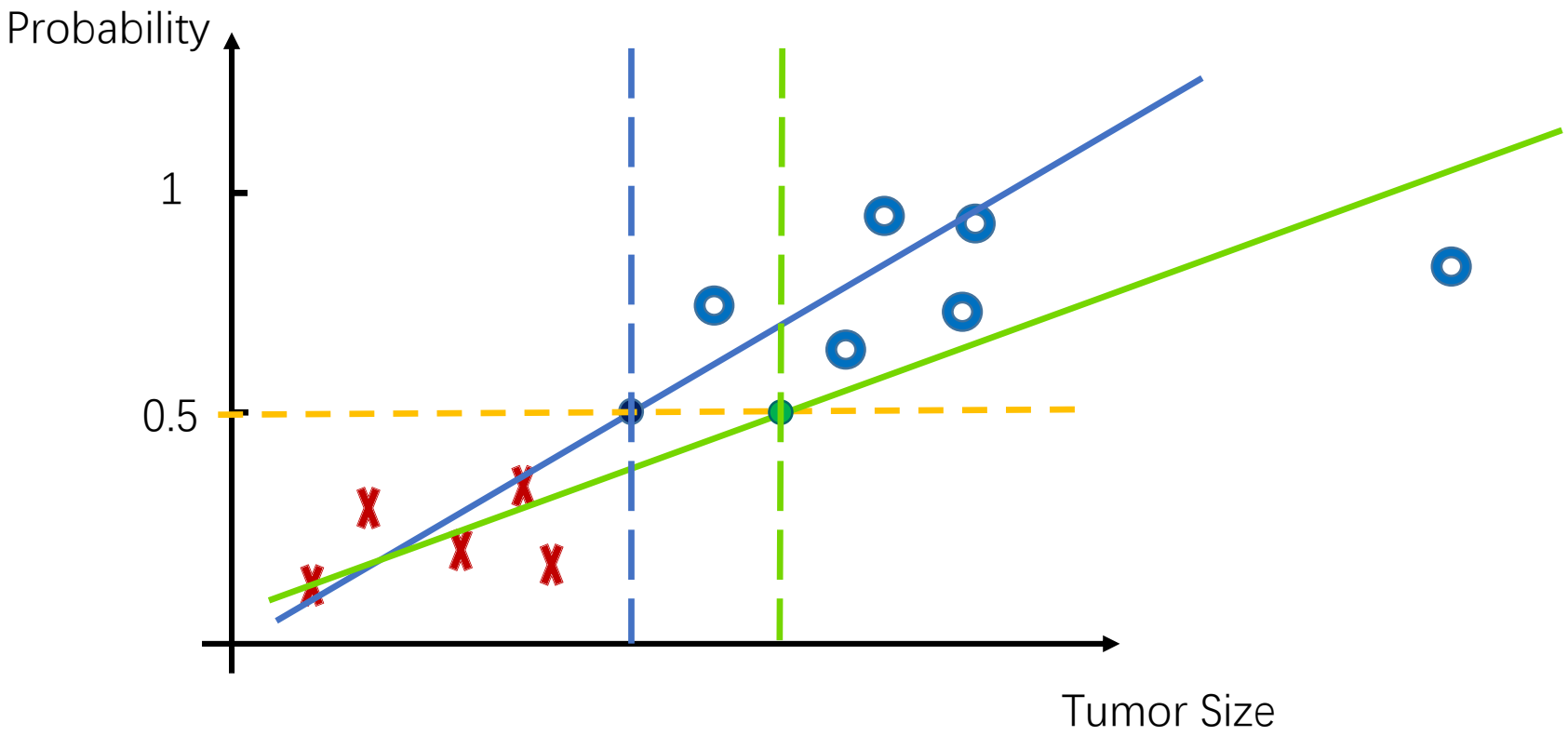
### D1. 2 Classes



# II. Supervised Learning

## D. Logistic Regression:

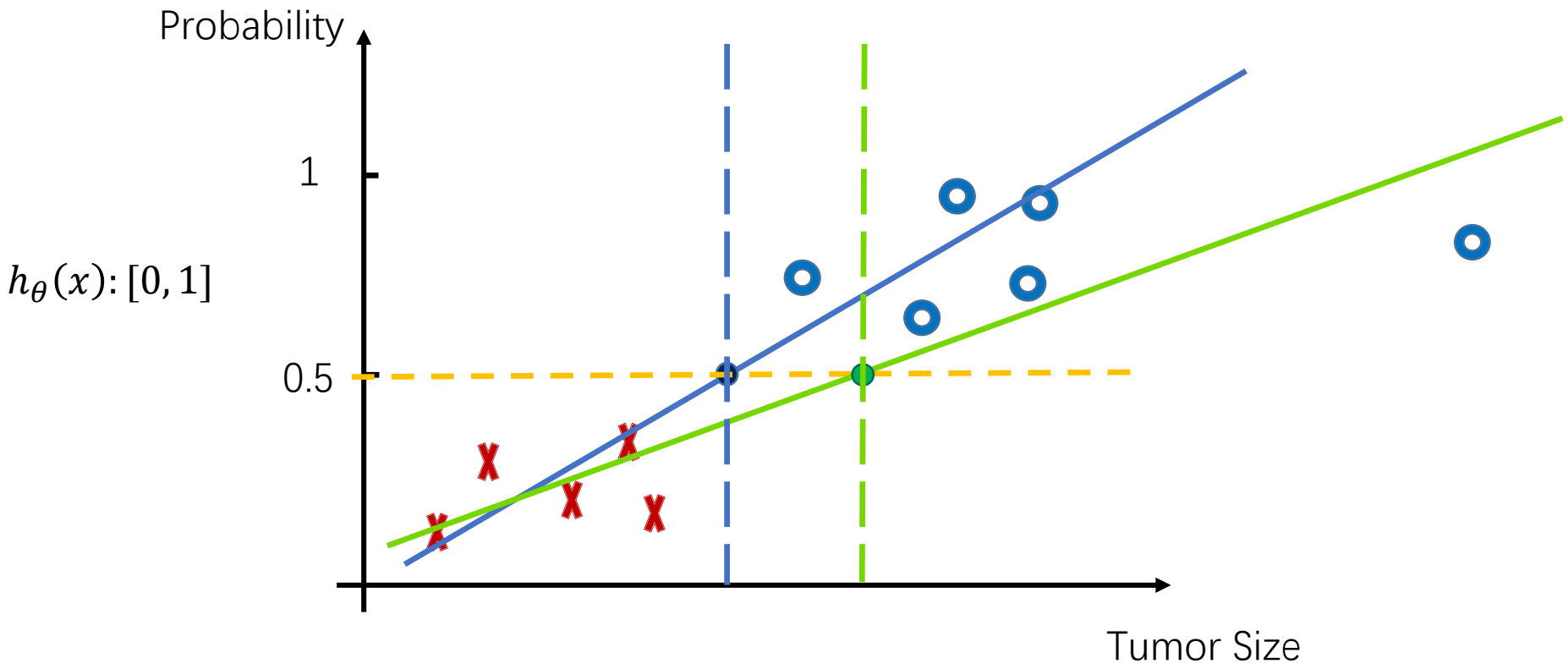
### D1. 2 Classes



# II. Supervised Learning

## D. Logistic Regression:

### D1. 2 Classes



# II. Supervised Learning

## D. Logistic Regression:

### D1. 2 Classes

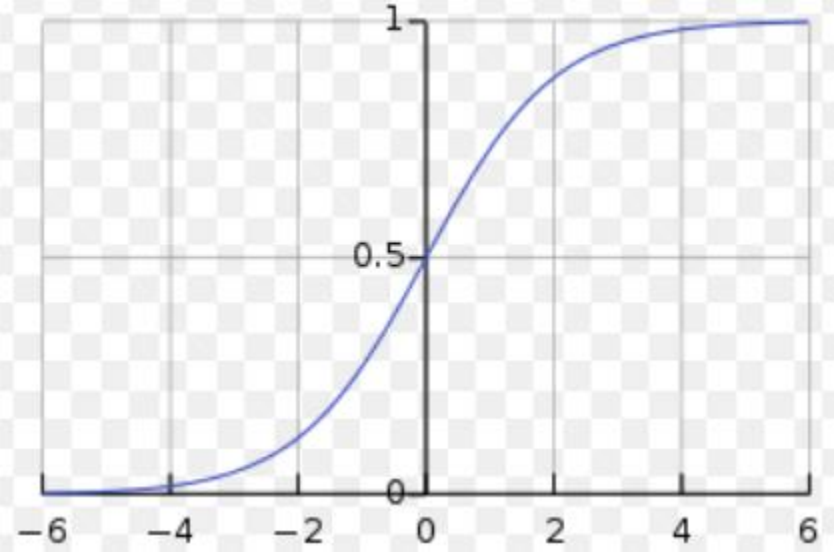
**Hypothesis:**

$$h_\theta(x) = g(\theta^T x)$$

$$= g(z)$$

$$= \frac{1}{1 + e^{-z}}$$

$$= \frac{1}{1 + e^{-\theta^T x}}$$




**Sigmoid Function**
**Logistic Function**

# II. Supervised Learning

## D. Logistic Regression:

### D1. 2 Classes

**Hypothesis:**

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

**Cost Function:**

$$J(\theta) = \begin{cases} -\log(h_\theta(x)) & if\ y = 1 \\ -\log\big(1 - h_\theta(x)\big) & if\ y = 0 \end{cases}$$

For single sample

# II. Supervised Learning

Explain: h(x)'s shape

## D. Logistic Regression:

### D1. 2 Classes

**Hypothesis:**

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

**Cost Function:**

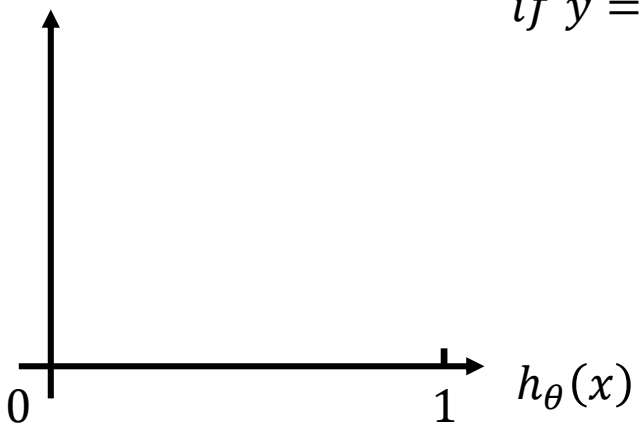
$$J(\theta) = \begin{cases} -\log(h_\theta(x)) & if\ y = 1 \\ -\log\big(1 - h_\theta(x)\big) & if\ y = 0 \end{cases}$$

$$J(\theta) = (h_\theta(x) - y)^2$$

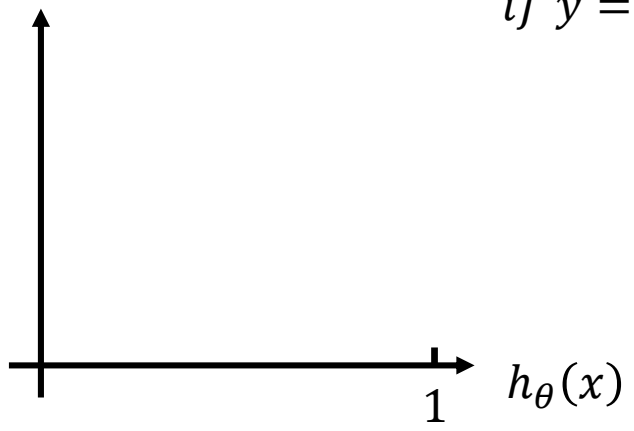
Linear Regression

For single sample

$if\ y = 1$




$if\ y = 0$



# II. Supervised Learning

Explain: why can write together

## D. Logistic Regression:

### D1. 2 Classes

**Hypothesis:**

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

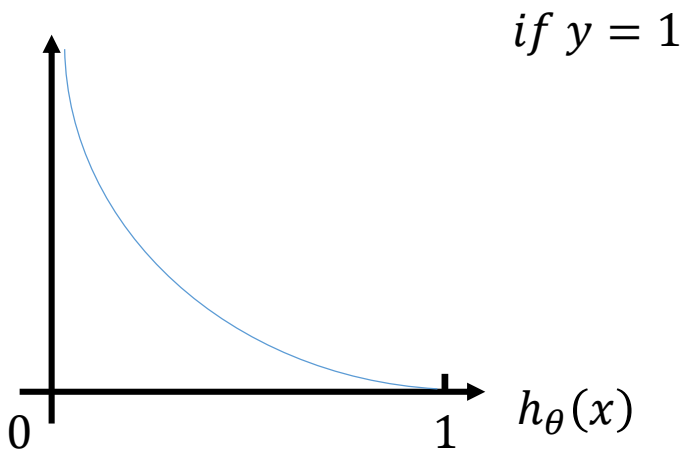
**Cost Function:**

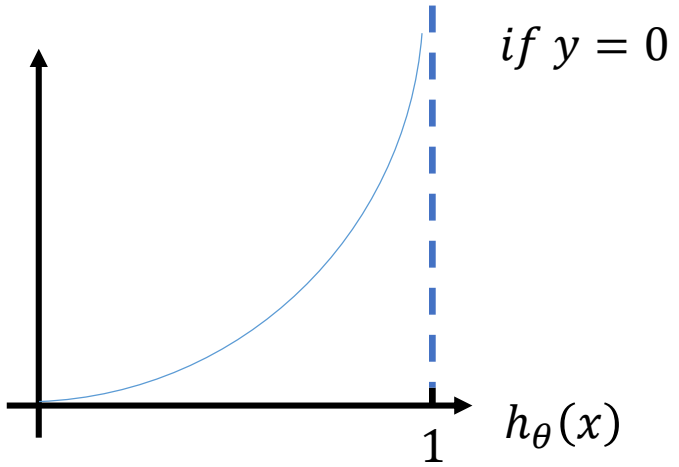
$$J(\theta) = -y log\big(h_\theta(x)\big) - (1 - y) log\big(1 - h_\theta(x)\big)$$

$$J(\theta) = (h_\theta(x) - y)^2$$

Linear Regression

For single sample



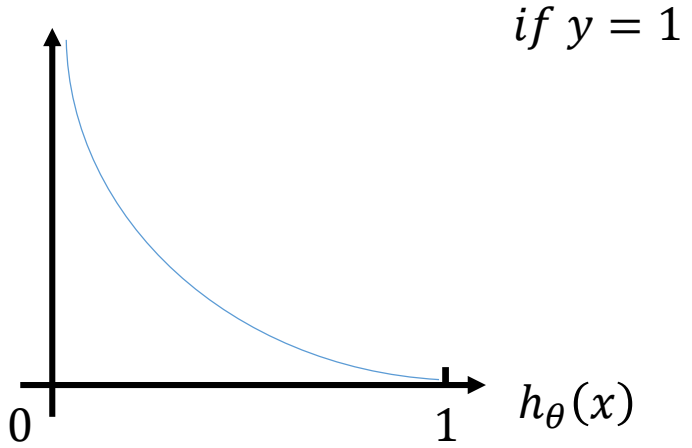
# II. Supervised Learning

## D. Logistic Regression:

### D1. 2 Classes

**Hypothesis:**

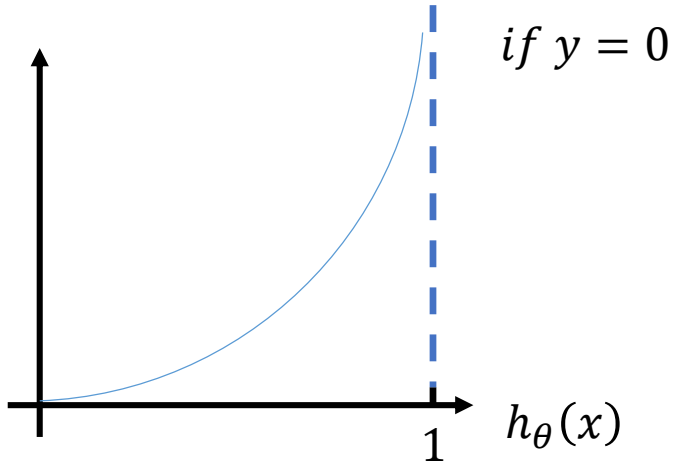
$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$




**Cost Function:**

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^i log\left(h_\theta(x^i)\right) + (1 - y^i) log\left(1 - h_\theta(x^i)\right)]$$

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^i) - y^i)^2$$ Linear Regression




For samples

Explain: derivative

# II. Supervised Learning

## D. Logistic Regression:

### D1. 2 Classes

**Hypothesis:**

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

**Gradient Descent:**

$$\frac{\partial}{\partial \theta_j} J(\theta_j) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^i) - y^i) \cdot x_j$$

**Cost Function:**

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} [y^i log\left(h_\theta(x^i)\right) + (1 - y^i) log\left(1 - h_\theta(x^i)\right)]$$

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^i) - y^i)^2$$ Linear Regression

For samples

Explain: derivative

# II. Supervised Learning

## D. Logistic Regression:

### D1. 2 Classes

**Hypothesis:**

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

**Cost Function:**

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^i log\left(h_\theta(x^i)\right) + (1 - y^i) log\left(1 - h_\theta(x^i)\right)]$$

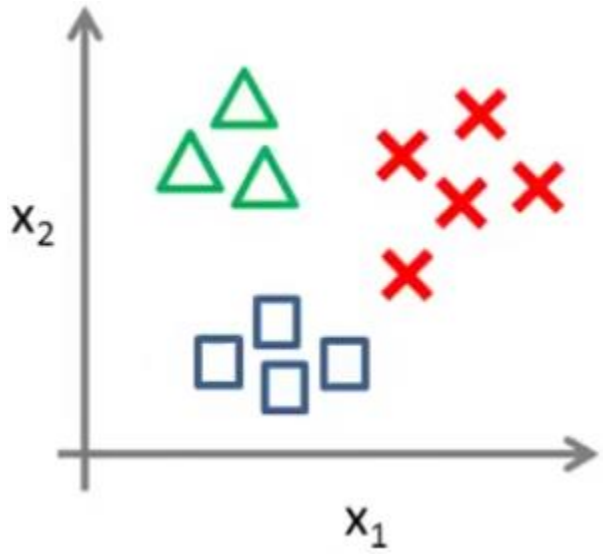
**Gradient Descent:**

$$\frac{\partial}{\partial \theta_j} J(\theta_j) = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^i) - y^i) \cdot x_j$$

# II. Supervised Learning

## D. Logistic Regression:

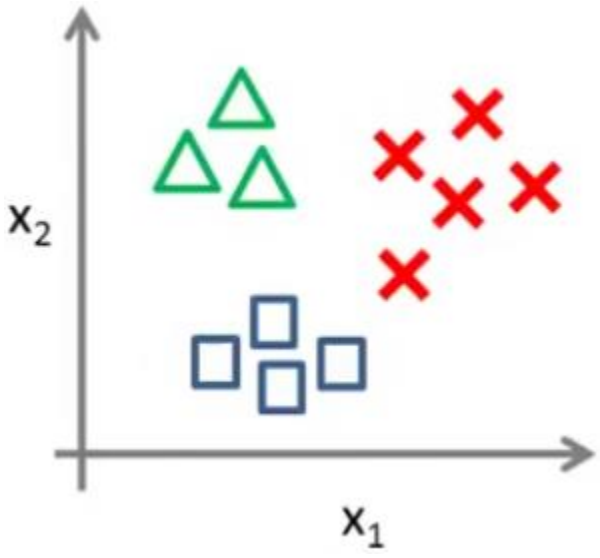
### D2. Multi-Classes

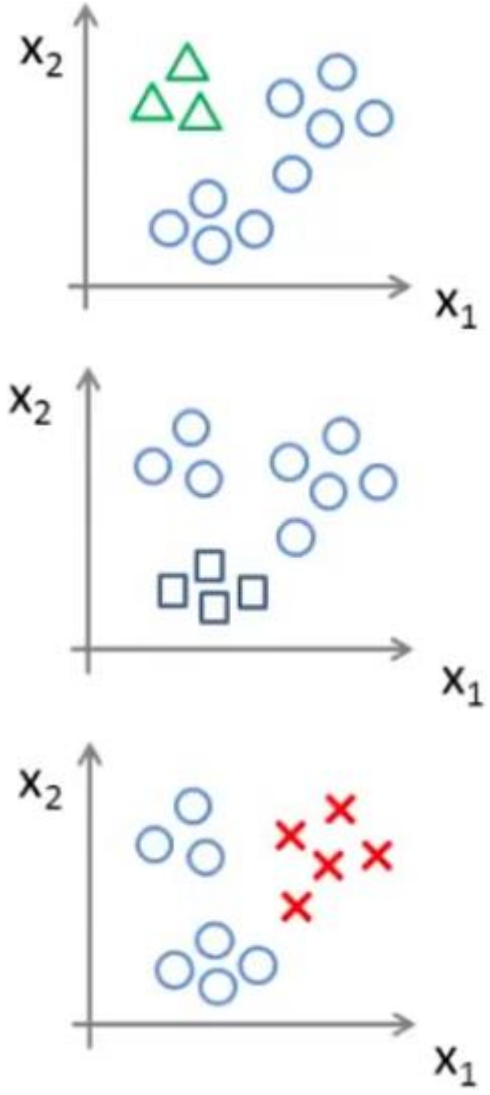


# II. Supervised Learning

## D. Logistic Regression:

### D2. Multi-Classes

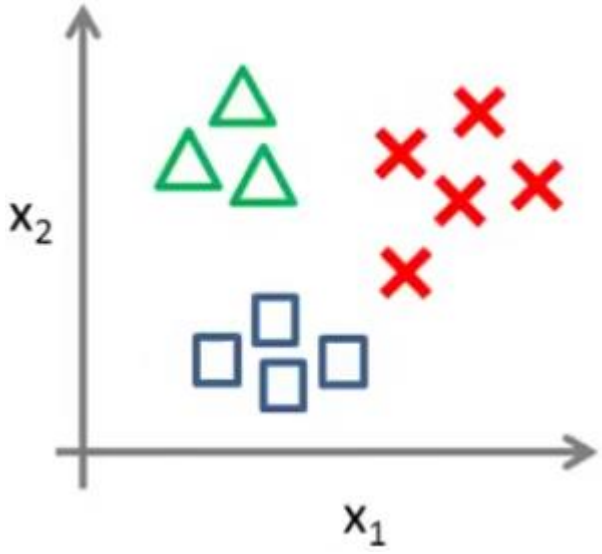


# II. Supervised Learning

## D. Logistic Regression:

### D2. Multi-Classes




Train a logistic regression classifier $h_\theta^{(i)}(x)$ for each class $i$ to predict the probability that $y = i$.

On a new input $x$, to make a prediction, pick the class $i$ that maximizes

$$\max_i h_\theta^{(i)}(x)$$

# II. Supervised Learning

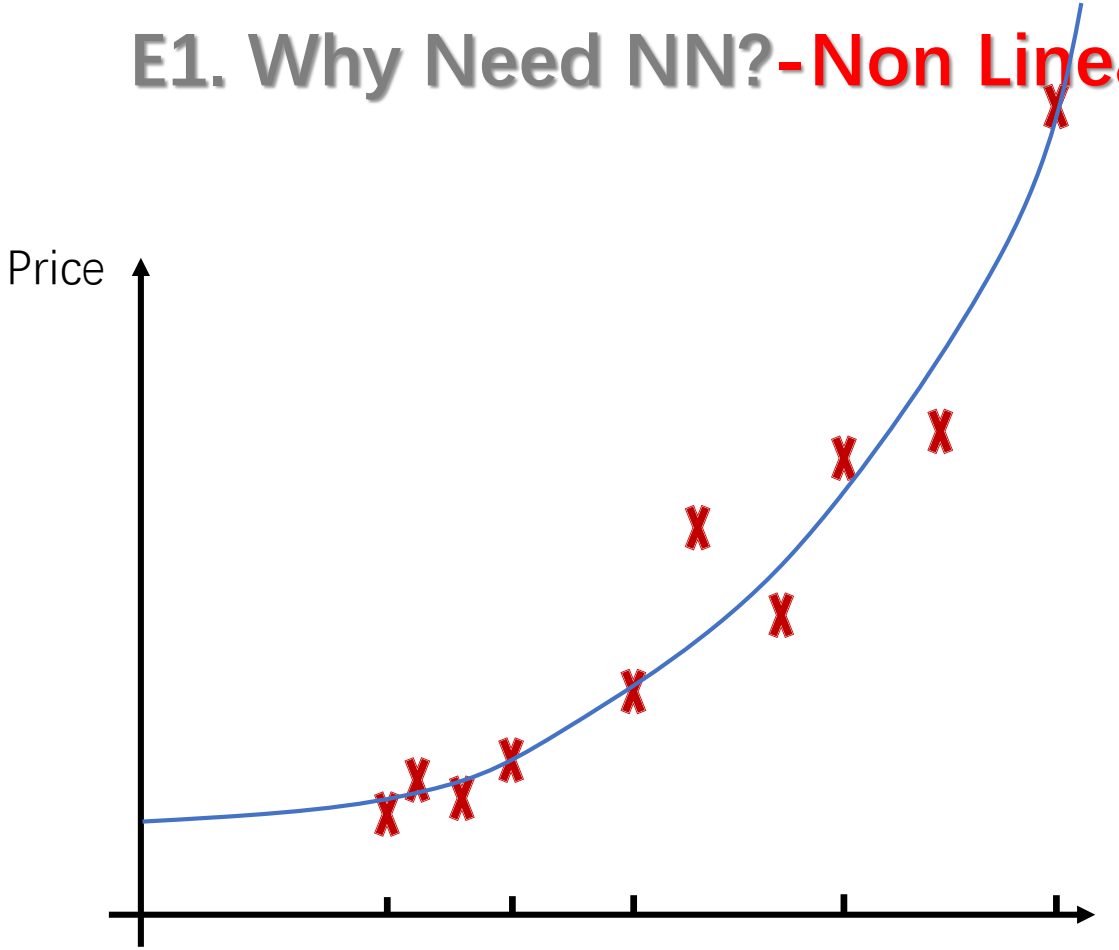
**Linear Regression:**

Implementation from scratch

# II. Supervised Learning

## E. Neural Network:

### E1. Why Need NN?-Non Linear




$$h_\theta(x) = \theta_0 + \theta_1 x + \theta_1 x^2$$

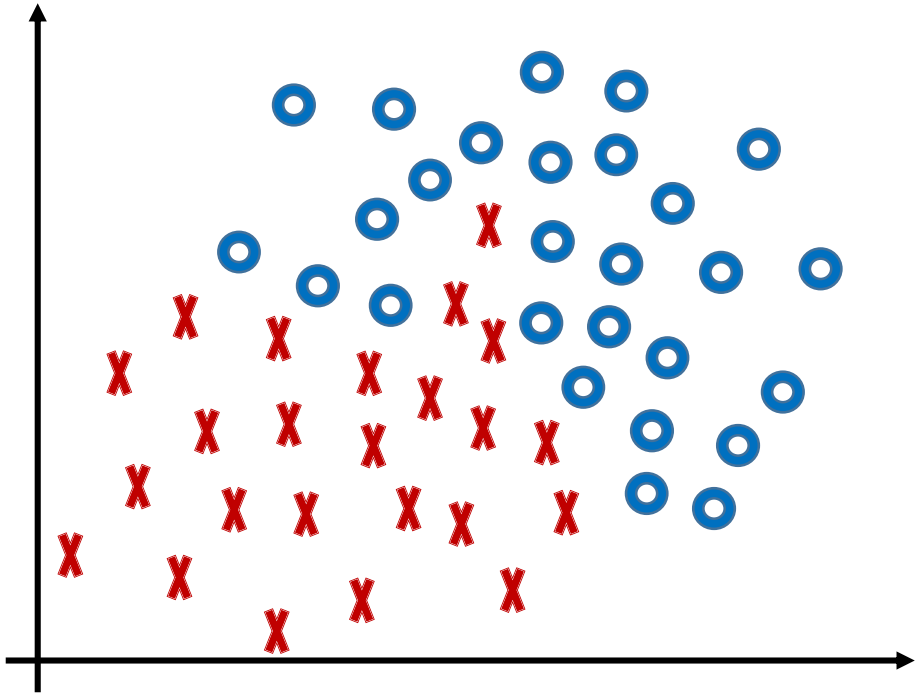
$$= \theta_0 + \theta_1(size) + \theta_1(size)^2$$

$$= \theta_0 + \theta_1 x_1 + \theta_1 x_2$$

# II. Supervised Learning

## E. Neural Network:

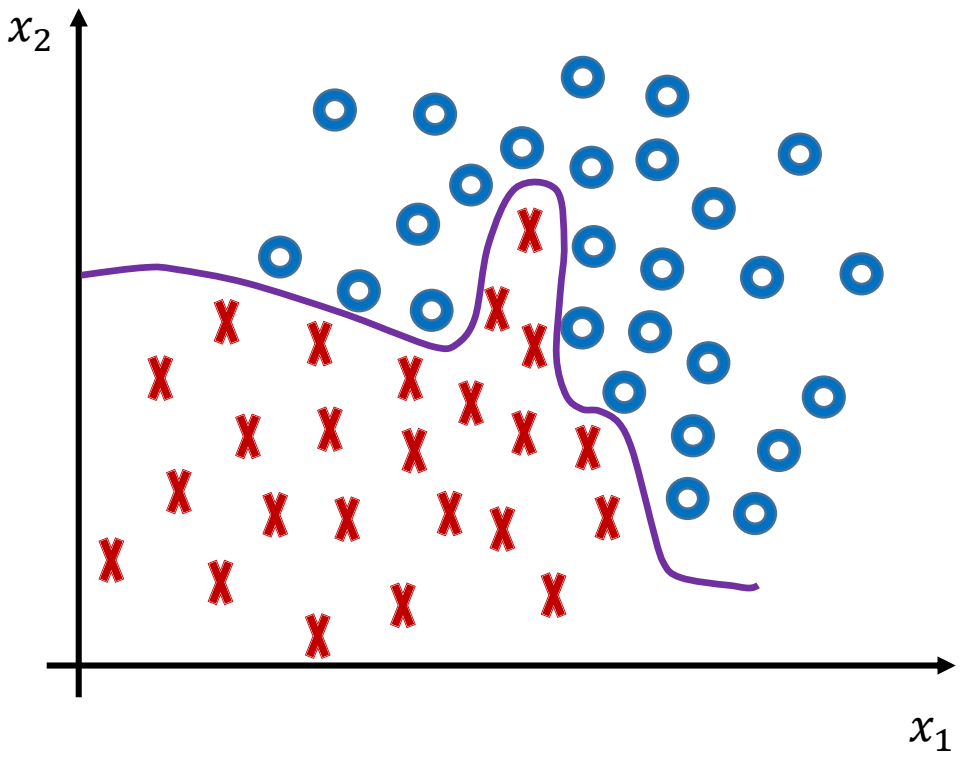
### E1. Why Need NN?-Non Linear



# II. Supervised Learning

## E. Neural Network:

### E1. Why Need NN?-Non Linear

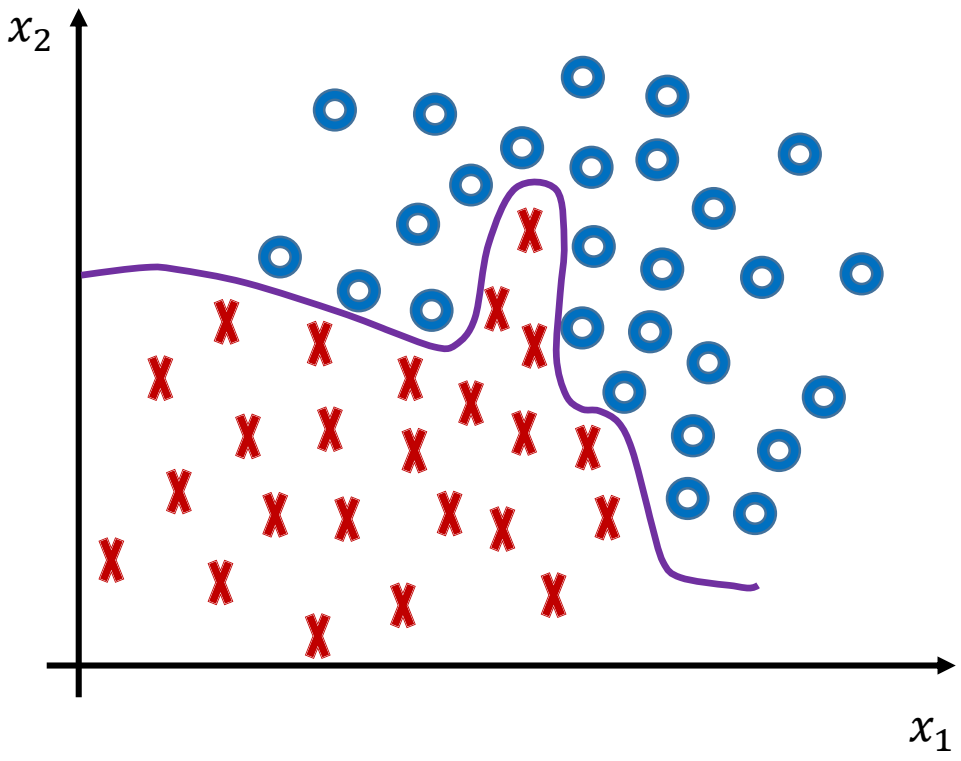



$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_1 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 x_2 + \theta_5 x_1^3 x_2 + \theta_6 x_1 x_2^2 + \cdots)$$

# II. Supervised Learning

## E. Neural Network:

### E1. Why Need NN?-Non Linear

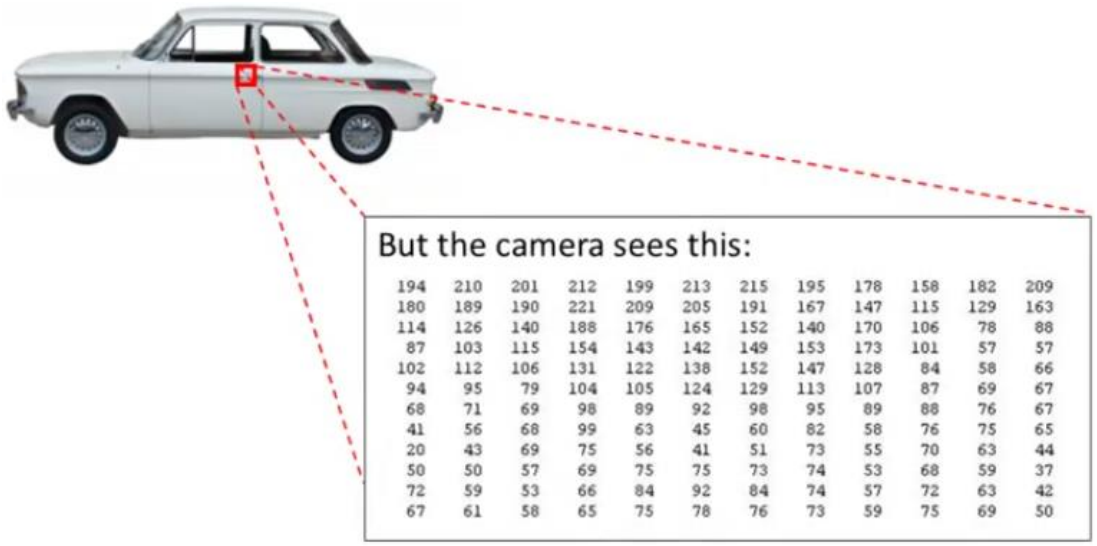



$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_1 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 x_2 + \theta_5 x_1^3 x_2 + \theta_6 x_1 x_2^2 + \cdots)$$

# II. Supervised Learning

## E. Neural Network:

### E1. Why Need NN?-Non Linear



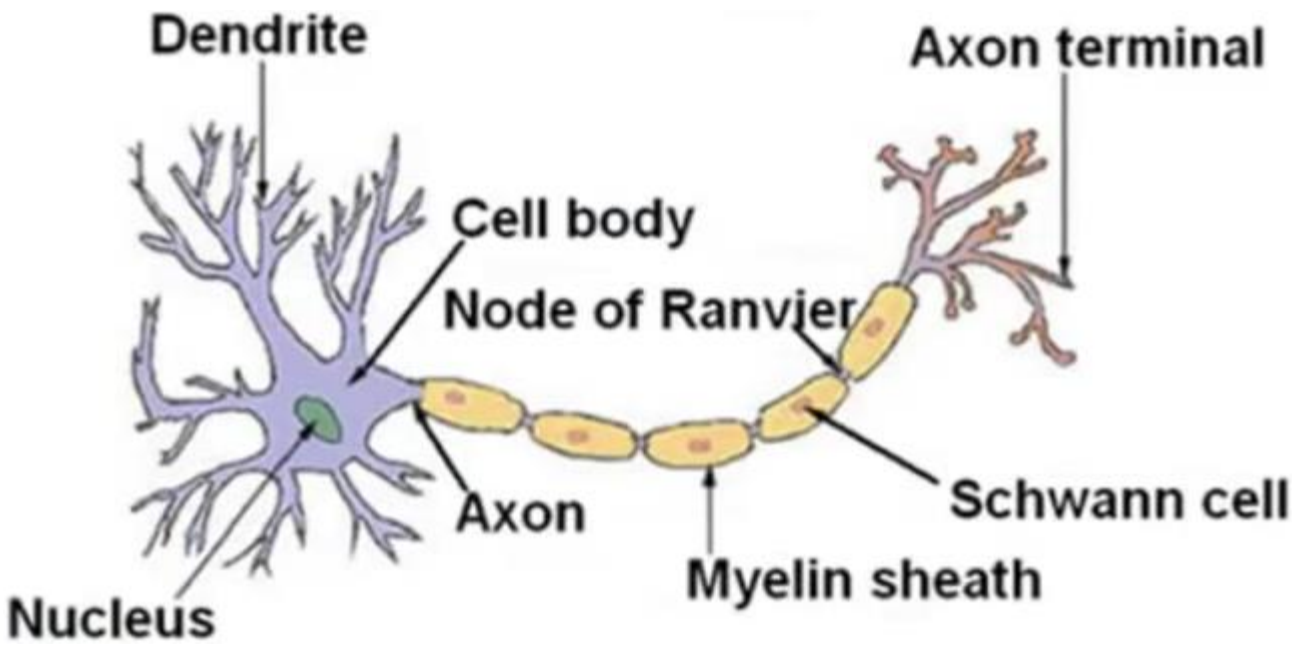

***Not possible by just combining features***

***Need advanced techs***

# II. Supervised Learning

## E. Neural Network:

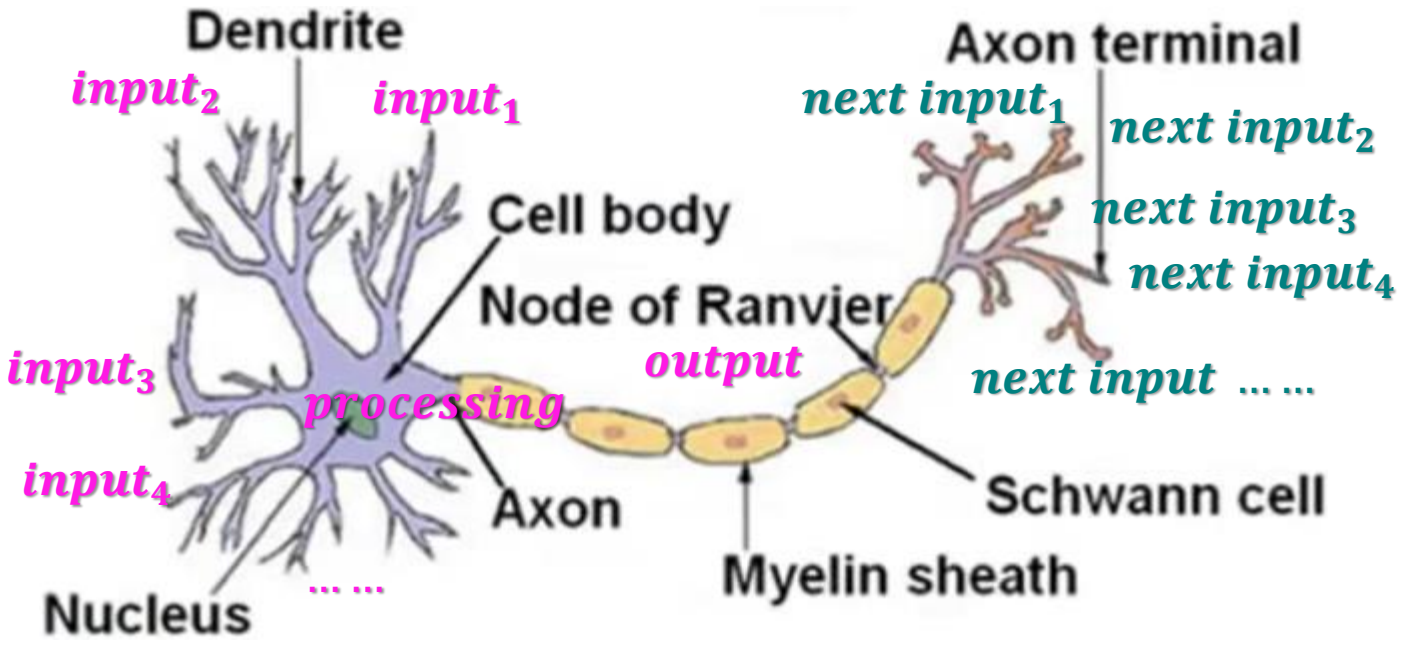
### E1. Why Need NN?-Mimic Brain



# II. Supervised Learning

## E. Neural Network:

### E1. Why Need NN?-Mimic Brain



# II. Supervised Learning

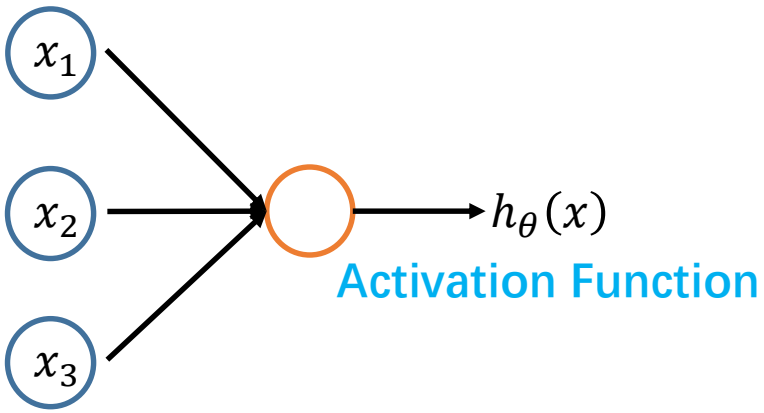
Explain: bias

## E. Neural Network:

### E2. Logistic Unit

$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \qquad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix}$$

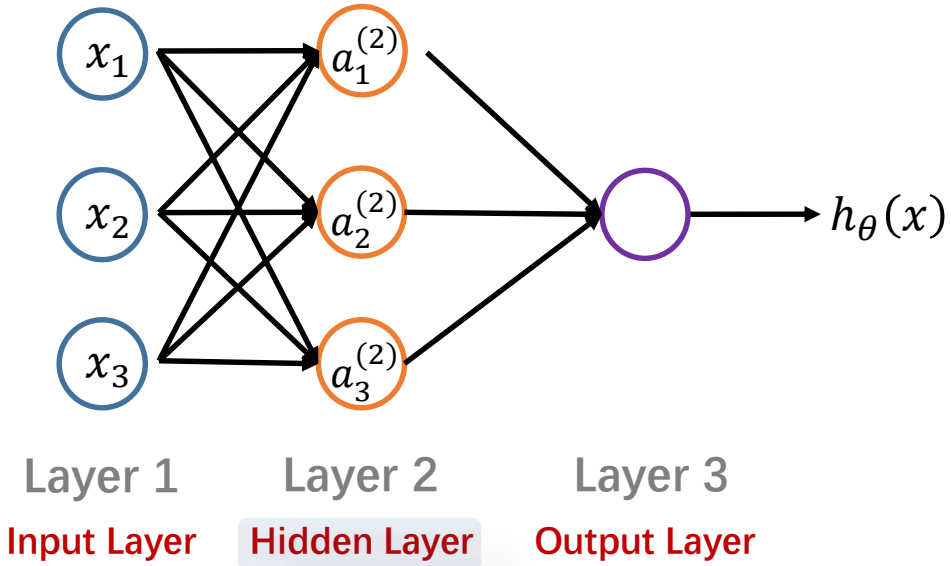
feature weights



# II. Supervised Learning

## E. Neural Network:

### E3. Neural Network



# II. Supervised Learning

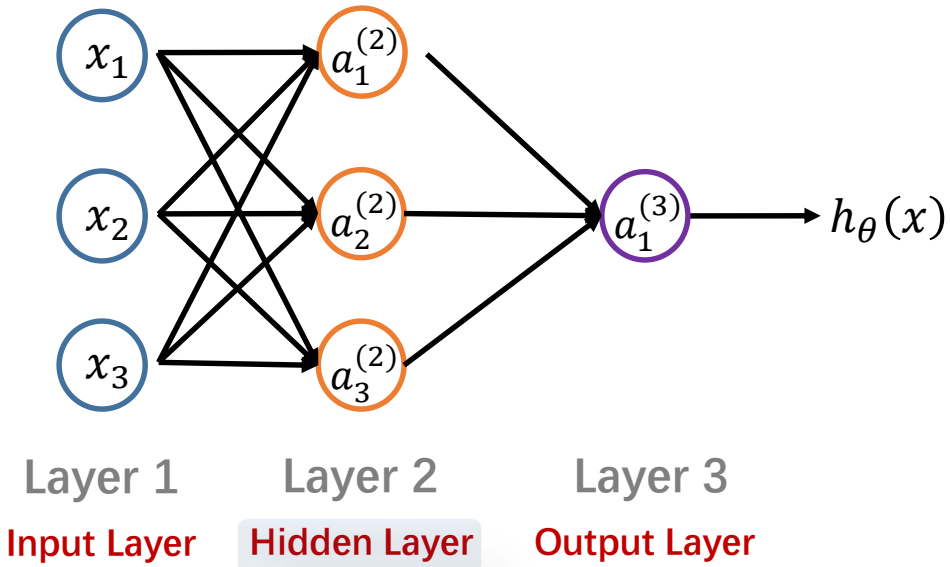
Explain: formulas

## E. Neural Network:

### E3. Neural Network

$a_i^{(j)}$ = "activation" of unit ***i*** in layer ***j***

$\theta^{(j)}$ = ***matrix*** of weights controlling function mapping from layer $j$ to layer $j+1$




Layer 2:

$$a_1^{(2)} = g(\theta_{10}^{(1)}x_0 + \theta_{11}^{(1)}x_1 + \theta_{12}^{(1)}x_2 + \theta_{13}^{(1)}x_3)$$

$$a_2^{(2)} = g(\theta_{20}^{(1)}x_0 + \theta_{21}^{(1)}x_1 + \theta_{22}^{(1)}x_2 + \theta_{23}^{(1)}x_3)$$

$$a_3^{(2)} = g(\theta_{30}^{(1)}x_0 + \theta_{31}^{(1)}x_1 + \theta_{32}^{(1)}x_2 + \theta_{33}^{(1)}x_3)$$

Layer 3:

$$h_\theta(x) = a_1^{(3)} = g(\theta_{10}^{(2)}a_0^{(2)} + \theta_{11}^{(2)}a_1^{(2)} + \theta_{12}^{(2)}a_2^{(2)} + \theta_{13}^{(2)}a_3^{(2)})$$

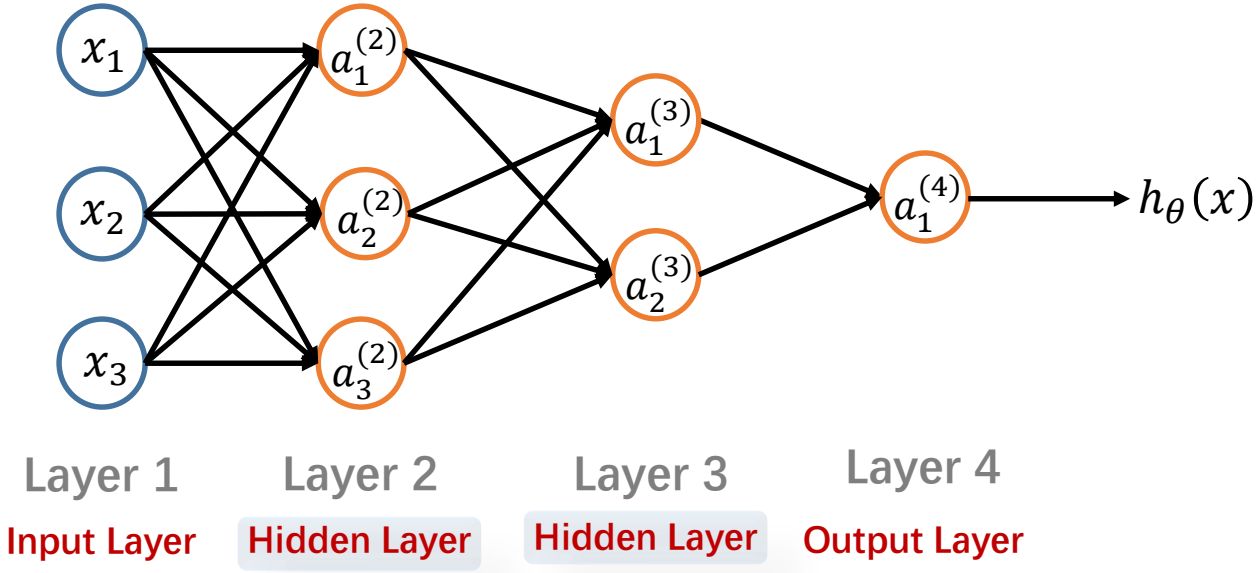
# II. Supervised Learning

## E. Neural Network:

### E3. Neural Network

$a_i^{(j)}$ = "activation" of unit ***i*** in layer ***j***

$\theta^{(j)}$ = ***matrix*** of weights controlling function mapping from layer $j$ to layer $j+1$



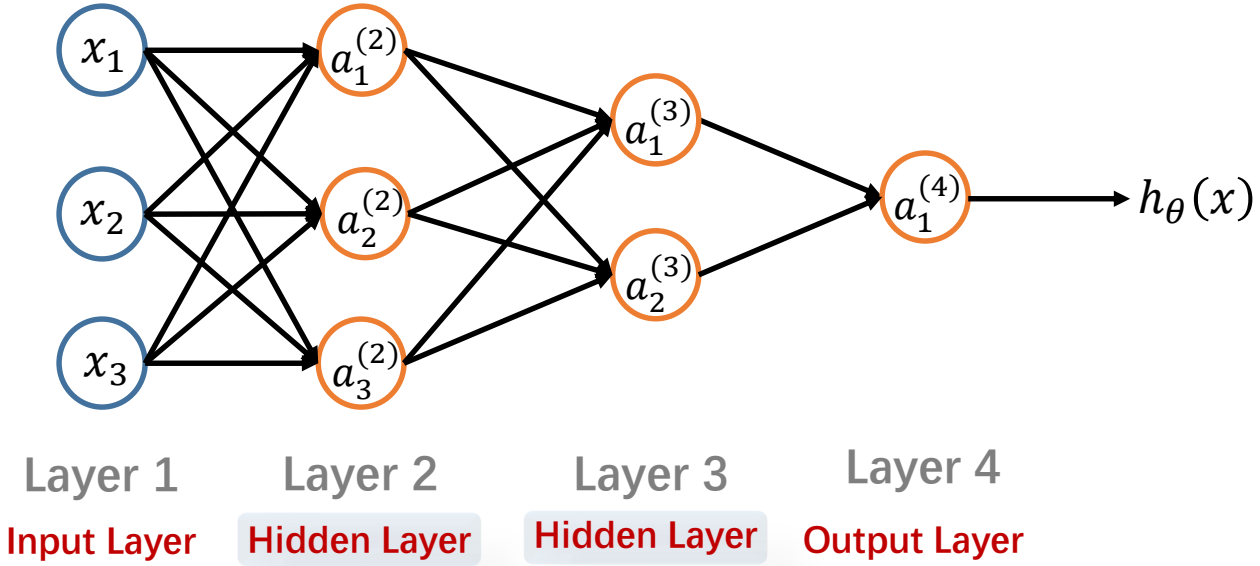
# II. Supervised Learning

## E. Neural Network:

### E3. Neural Network

$a_i^{(j)}$ = "activation" of unit ***i*** in layer ***j***

$\theta^{(j)}$ = ***matrix*** of weights controlling function mapping from layer $j$ to layer $j+1$



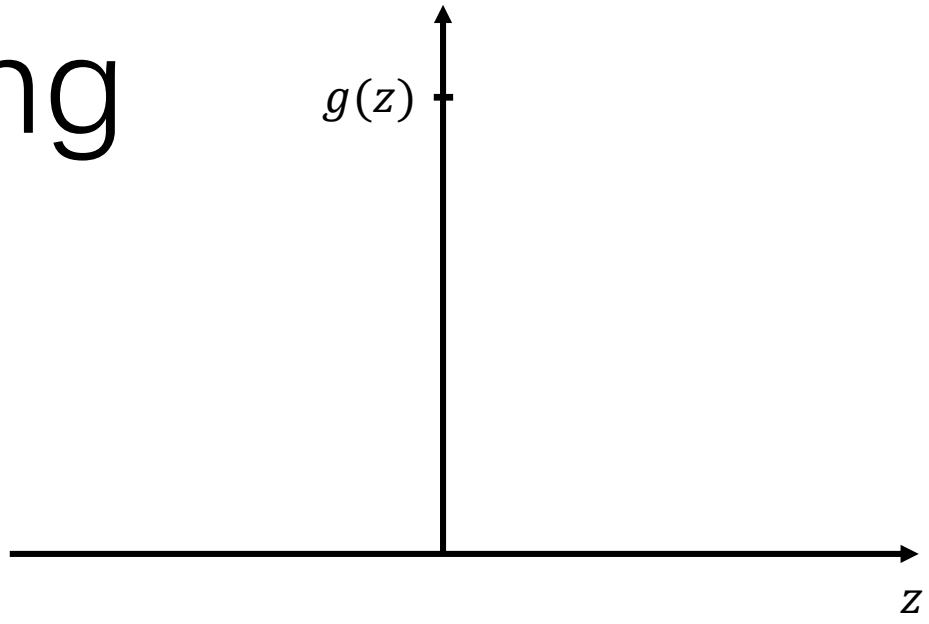
Explain: AND

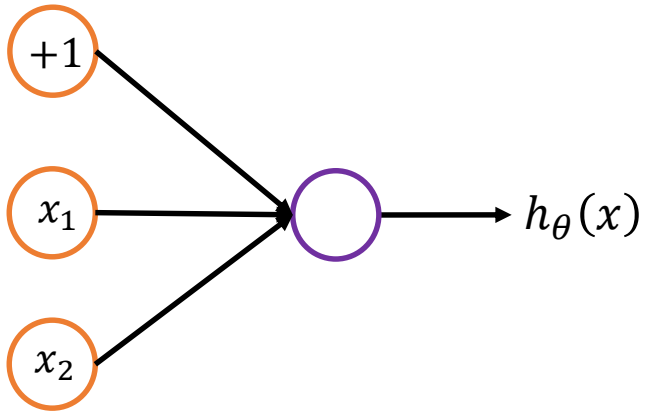
# II. Supervised Learning

## E. Neural Network:

### E3. Neural Network-E.G: AND

$y = x_1 \,\&\, x_2, \quad x_1, x_2 \in \{0, 1\}$



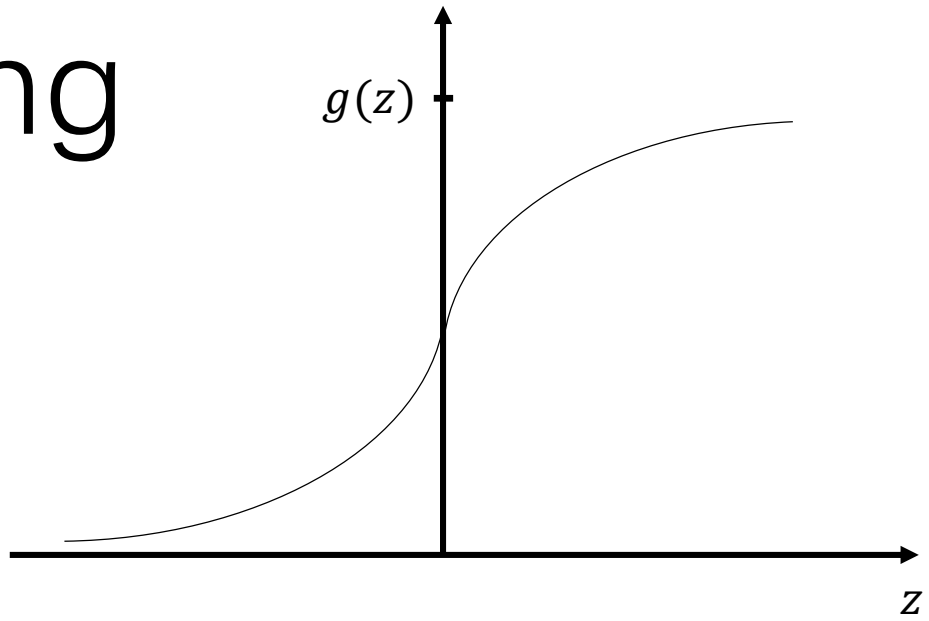




| $x_1$ | $x_2$ | $h_\theta(x)$ |
|---|---|---|
| 0 | 0 | |
| 0 | 1 | |
| 1 | 0 | |
| 1 | 1 | |

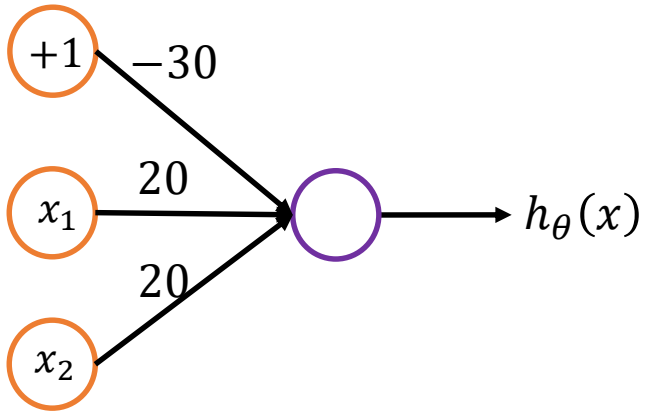
# II. Supervised Learning

## E. Neural Network:

### E3. Neural Network-E.G: AND




$y = x_1 \,\&\, x_2, \quad x_1, x_2 \in \{0, 1\}$

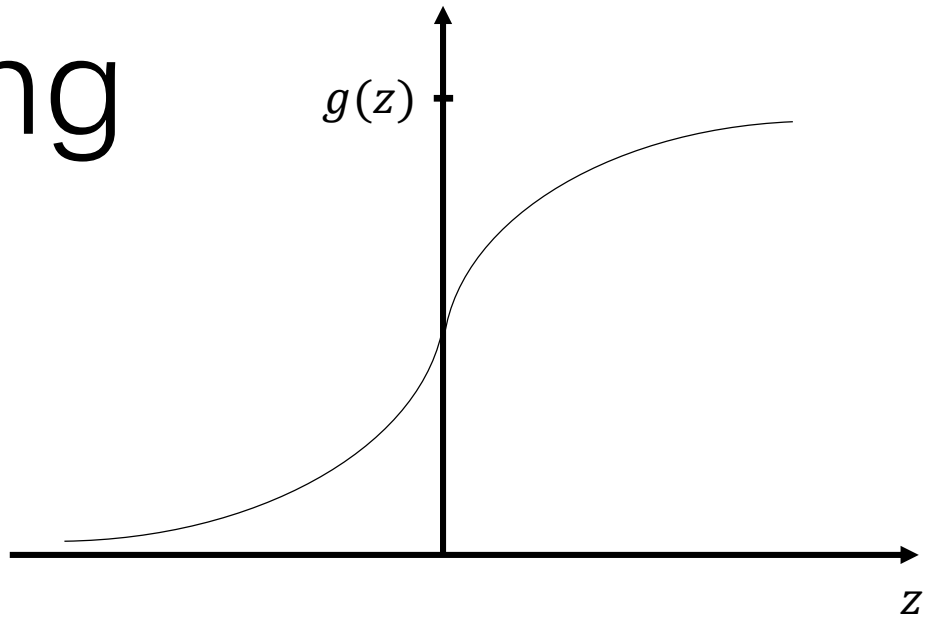



| $x_1$ | $x_2$ | $h_\theta(x)$ |
|---|---|---|
| 0 | 0 | |
| 0 | 1 | |
| 1 | 0 | |
| 1 | 1 | |

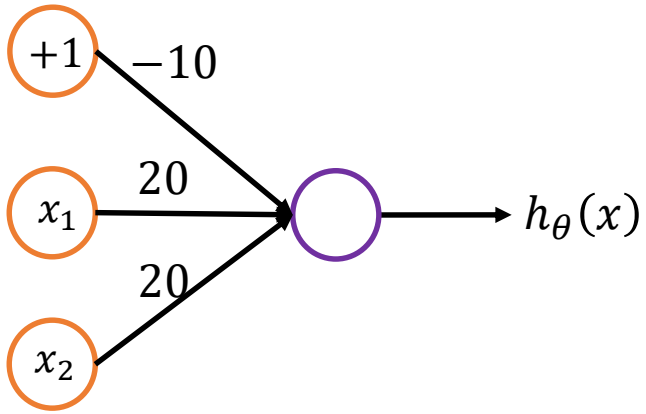
# II. Supervised Learning

## E. Neural Network:

### E3. Neural Network-E.G: OR




$y = x_1 \mid x_2, \quad x_1, x_2 \in \{0, 1\}$

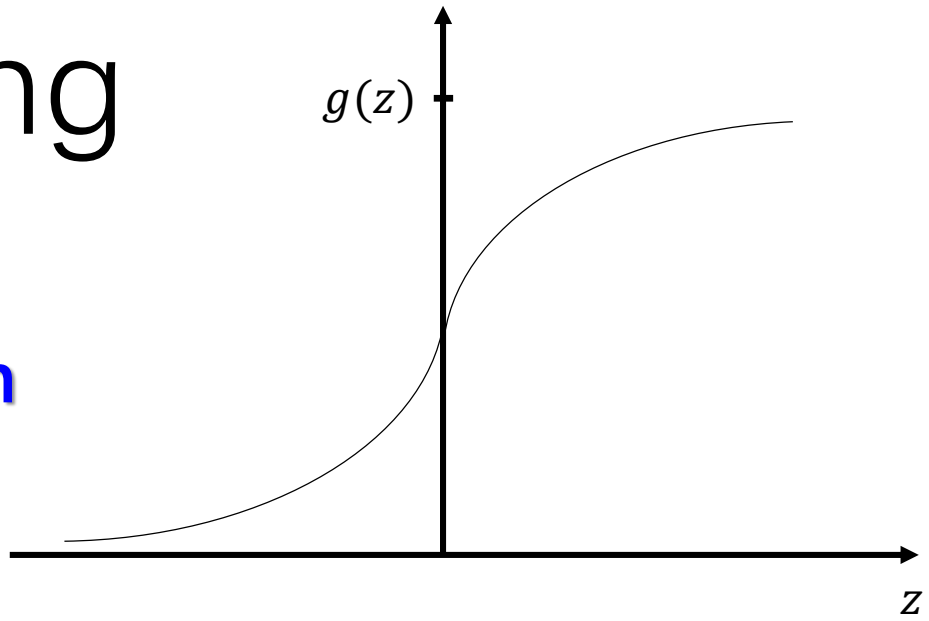



| $x_1$ | $x_2$ | $h_\theta(x)$ |
|---|---|---|
| 0 | 0 | |
| 0 | 1 | |
| 1 | 0 | |
| 1 | 1 | |

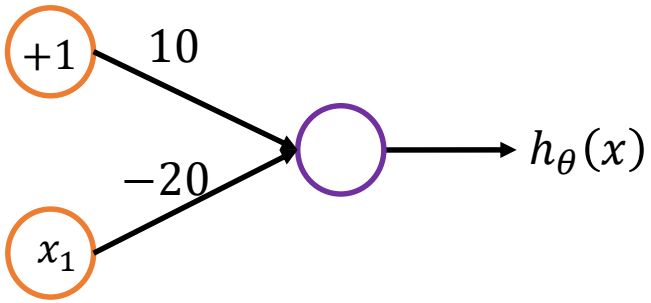
# II. Supervised Learning

## E. Neural Network:

### E3. Neural Network-E.G: Negation




$y = !\,x_1, \quad x_1 \in \{0, 1\}$




| $x_1$ | $h_\theta(x)$ |
|---|---|
| 0 | |
| 1 | |

Explain: xnor

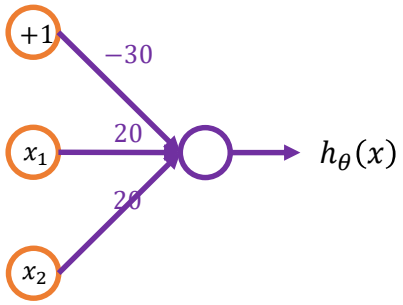
# II. Supervised Learning

## E. Neural Network:

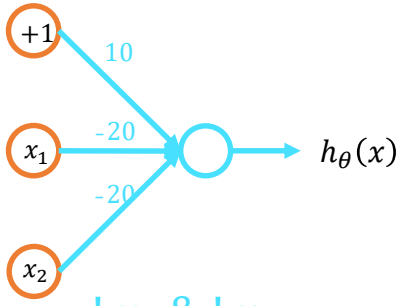
### E3. Neural Network-E.G: Combination

$y = x_1 \textbf{ XNOR } x_2, \quad x_1, x_2 \in \{0, 1\}$

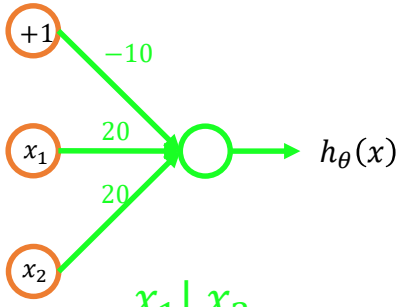
| $x_1$ | $x_2$ | $a_1^{(2)}$ | $a_2^{(2)}$ | $h_\theta(x)$ |
|---|---|---|---|---|
| 0 | 0 | | | |
| 0 | 1 | | | |
| 1 | 0 | | | |
| 1 | 1 | | | |




$x_1 \,\&\, x_2$




$!\,x_1 \,\&\, !\,x_2$




$x_1 \,|\, x_2$

Explain: multi-label

# II. Supervised Learning

## E. Neural Network:

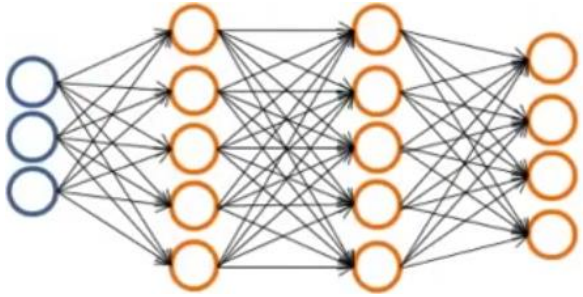
### E3. Neural Network-E.G: Classification

Pedestrian

Car

Motorcycle

Truck



$h_\Theta(x) \in \mathbb{R}^4$

Want $h_\Theta(x) \approx \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$, $h_\Theta(x) \approx \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$, $h_\Theta(x) \approx \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$, etc.

when pedestrian when car when motorcycle

# II. Supervised Learning

## E. Neural Network:

### E4. Gradient Descent

**Cost Function:**

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m,}\sum_{k=1}^{K} y_k^i log(h_\theta(x^i))_k + \left(1 - y_k^i\right) log(1 - \left(h_\theta(x^i))_k\right)\right]$$

$i: sample\ ID$

$m: \#\ of\ samples$

$k: output\ ID$

$K: \#\ of\ outputs$

# II. Supervised Learning

## F. Back Propagation:

Explain: scratch
Intuitive Chain rule

# II. Supervised Learning

## F. Back Propagation:

**Chain Rule**

Explain: Theoretical Math Chain rule

# II. Supervised Learning

## F. Back Propagation:

### F1. Chain Rule

Explain: real e.g1
Math Chain rule

# II. Supervised Learning

## F. Back Propagation:

### F1. Chain Rule-Real Example

Explain: real e.g2
Math Chain rule

# II. Supervised Learning

## F. Back Propagation:

### F1. Chain Rule-Real Example

Explain: real e.g3
Math Chain rule

# II. Supervised Learning

## F. Back Propagation:

### F1. Chain Rule-Real Example

Explain:
Chain rule problems

# II. Supervised Learning

## F. Back Propagation:

### F1. Chain Rule-Problems

# II. Supervised Learning

## F. Back Propagation:

### F2. Implementation: NN

# II. Supervised Learning

## G. Regularization:

### G1. What Is Regularization

**Linear Regression:** $$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^i) - y^i)^2$$

**Logistic Regression:** $$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^i log\left(h_\theta(x^i)\right) + (1 - y^i)log\left(1 - h_\theta(x^i)\right)]$$

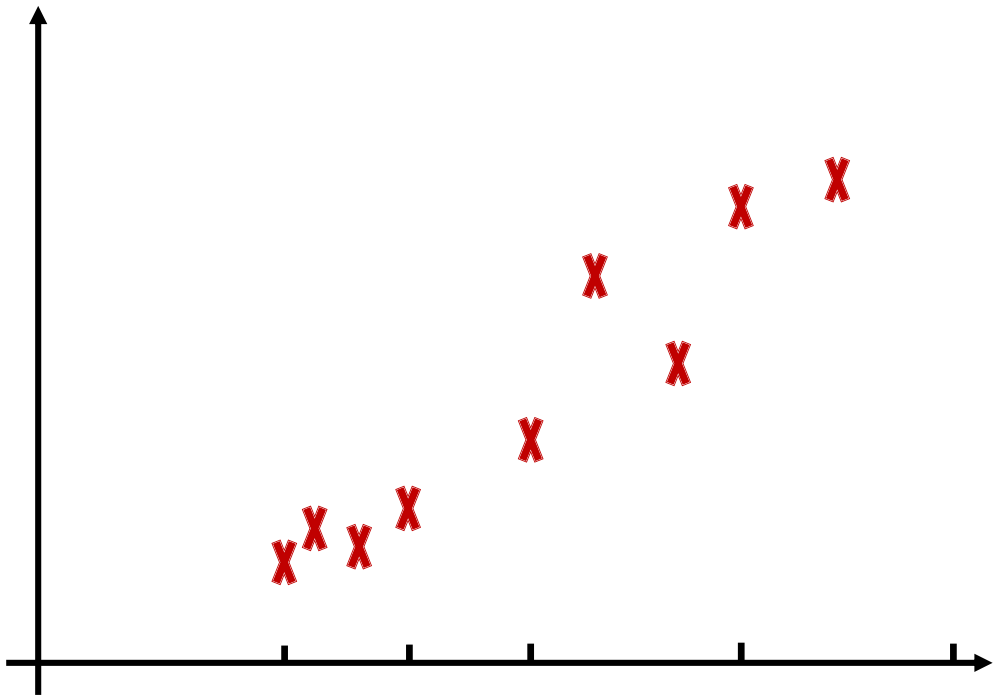
**Neural Network:** $$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m,}\sum_{k=1}^{K} y_k^i log(h_\theta(x^i))_k + \left(1 - y_k^i\right) log(1 - \left(h_\theta(x^i))_k\right)\right]$$

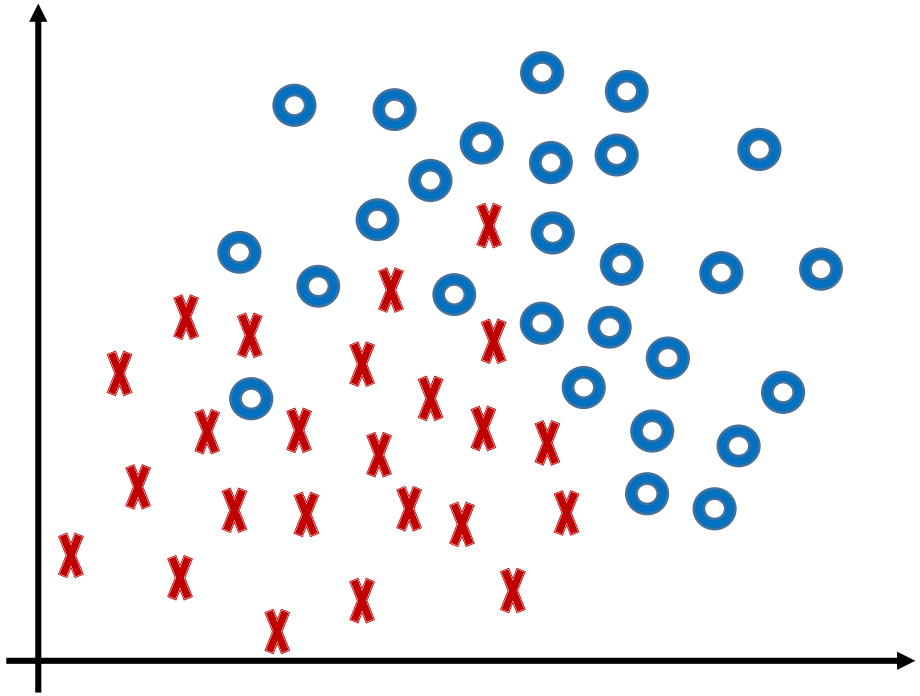
Explain: Fit to what
Extension
Overfit & prevension

# II. Supervised Learning

## G. Regularization:

### G1. What Is Regularization





# II. Supervised Learning

## G. Regularization:

### G1. What Is Regularization

**Linear Regression:**

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^i) - y^i)^2 + \boxed{\lambda\sum_{j=1}^{n}\theta_j^2}\right]$$

**Logistic Regression:**

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^i log\left(h_\theta(x^i)\right) + (1-y^i)log\left(1-h_\theta(x^i)\right)] + \boxed{\frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2}$$

**Neural Network:**

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m,}\sum_{k=1}^{K} y_k^i \log(h_\theta(x^i))_k + \left(1-y_k^i\right) log(1-\left(h_\theta(x^i)\right)_k)\right]$$

$$+\boxed{\frac{\lambda}{2m}\sum_{l=1}^{L-1,}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}(\theta_{ji}^l)^2}$$

Explain: theta, lambda effects

# II. Supervised Learning

## G. Regularization:

### G1. What Is Regularization

**Linear Regression:**

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^i) - y^i)^2 + \lambda\sum_{j=1}^{n}\theta_j^2\right]$$

**Gradient Descent:**

$$\begin{aligned}&\textit{while not converge}\ \{\\&\quad \theta_j = \theta_j - \alpha\left[\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^i) - y^i)\cdot x^i + \frac{\lambda}{m}\theta_j\right]\\&\}\end{aligned}$$

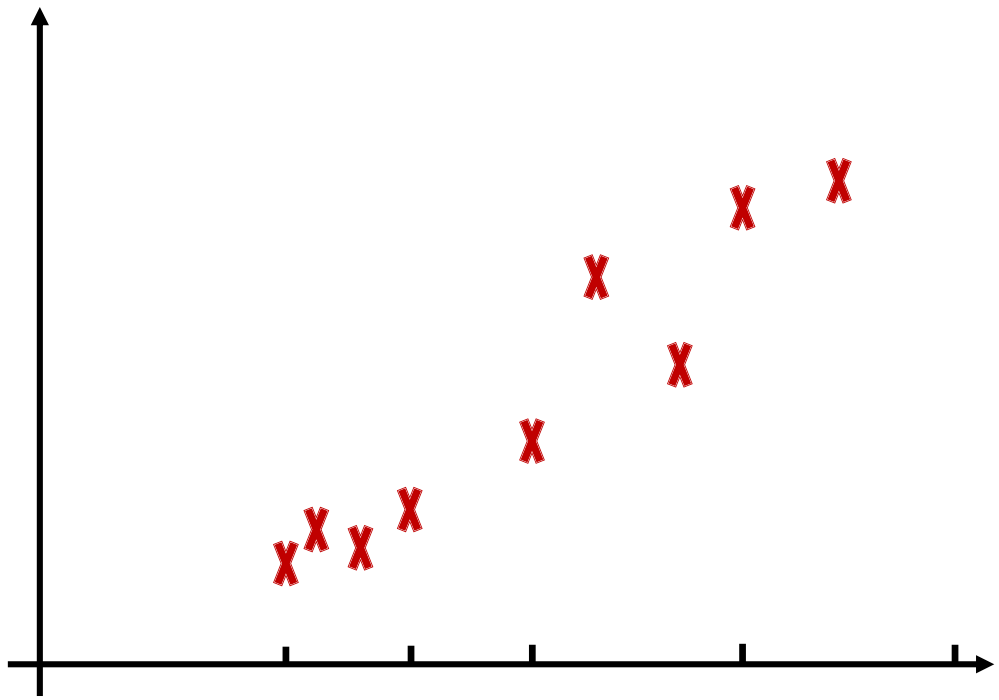
$$\theta_j = \theta_j\left(1 - \boxed{\alpha\frac{\lambda}{m}}\right) - \alpha\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^i) - y^i)\cdot x^i$$

Explain:
Restrain can be better

# II. Supervised Learning

## G. Regularization:

### G1. What Is Regularization

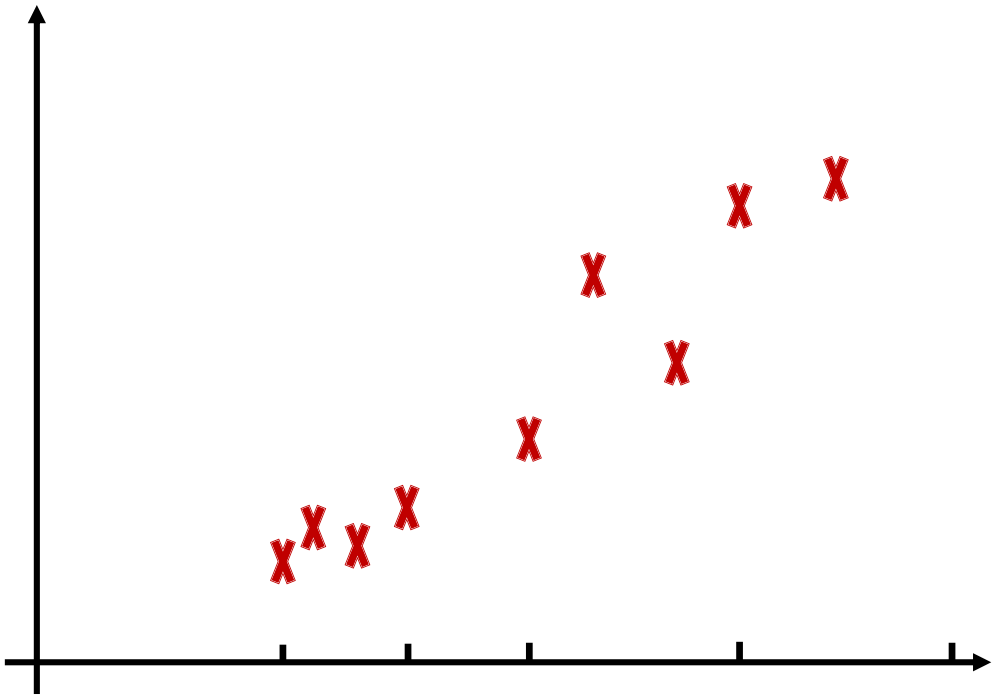


Explain:
Restrain can be better

# II. Supervised Learning

## G. Regularization:

### G1. What Is Regularization



为了拟合不同点，overfit的函数往往需要在相对较小的区间内迅速变化，因而需要较大的梯度。为了保证较大的梯度，只能令权重较大。

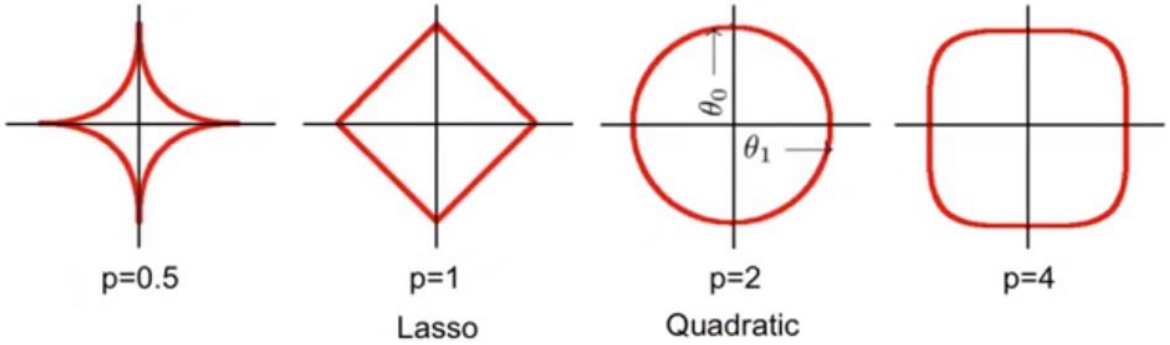
因而，Regularization保证了较小的权重，所以能够起到防止overfit的作用。

# II. Supervised Learning

## G. Regularization:

### G2. Type Of Regularization

**Lp Norm:** $$\left(\sum_i |\theta_i|^p\right)^{\frac{1}{p}}$$




**L2 Regularization:** $\|\theta\|^2$ Ridge Regression

**L1 Regularization:** $\|\theta\|$ Lasso Regression

# II. Supervised Learning

## G. Regularization:

### G2. Type Of Regularization

**L1 Regularization:**

**Linear Regression:**

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^i) - y^i)^2 + \lambda\sum_{j=1}^{n}\|\theta_j\|_1\right]$$

**Logistic Regression:**

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^i log\left(h_\theta(x^i)\right) + (1 - y^i)log\left(1 - h_\theta(x^i)\right)] + \frac{\lambda}{2m}\sum_{j=1}^{n}\|\theta_j\|_1$$

**Neural Network:**

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m,}\sum_{k=1}^{K} y_k^i \log(h_\theta(x^i))_k + \left(1 - y_k^i\right) log(1 - \left(h_\theta(x^i))_k\right)\right]$$

$$+\frac{\lambda}{2m}\sum_{l=1}^{L-1,}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}\|\theta_j\|_1$$

Explain:
L1 Pattern

# II. Supervised Learning

## G. Regularization:

### G2. Type Of Regularization

**L1 Regularization:**

**Linear Regression:**

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^i) - y^i)^2 + \lambda\sum_{j=1}^{n}\|\theta_j\|_1\right]$$

**Gradient Descent:**

$$\begin{aligned}&\textit{while not converge}\ \{\\ &\qquad \theta_j = \theta_j - \alpha\left[\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^i) - y^i)\cdot x^i + \boxed{\frac{\lambda}{m}sgn(\theta_j)}\right]\\ &\}\end{aligned}$$

$$\theta_j = \theta_j - \boxed{\alpha\frac{\lambda}{m}sgn(\theta_j)} - \alpha\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^i) - y^i)\cdot x^i$$

# II. Supervised Learning

## G. Regularization:

### G2. Type Of Regularization

The Figure 3.11 from *Elements of Statistical Learning* by Hastie, Tibshirani, and Friedman is very illustrative:

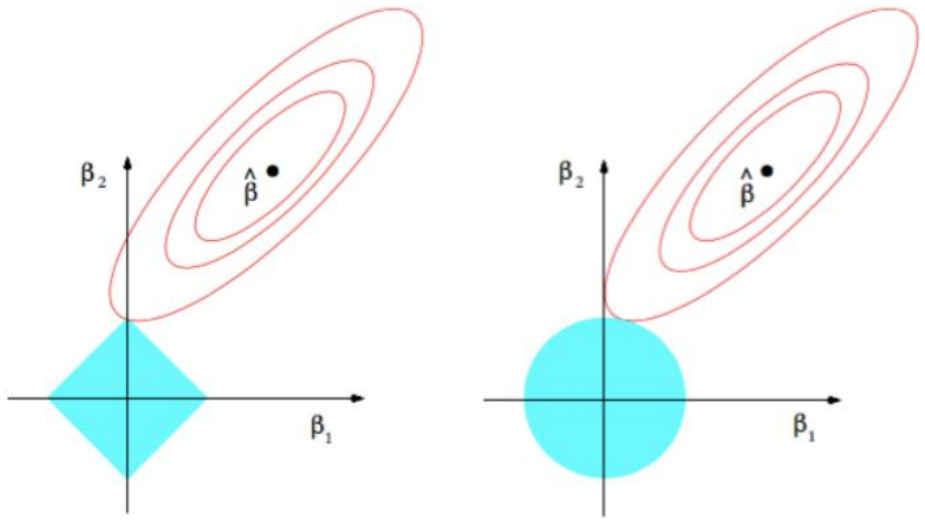



**FIGURE 3.11.** *Estimation picture for the lasso (left) and ridge regression (right). Shown are contours of the error and constraint functions. The solid blue areas are the constraint regions* $|\beta_1| + |\beta_2| \leq t$ *and* $\beta_1^2 + \beta_2^2 \leq t^2$*, respectively, while the red ellipses are the contours of the least squares error function.*

# II. Supervised Learning

## G. Regularization:

### G2. Type Of Regularization

**L1 Sparsity Theoretical Explanation:** Link 1

# II. Supervised Learning

## G. Regularization:

### G2. Type Of Regularization

| L2 regularization | L1 regularization |
|---|---|
| Computational efficient due to having analytical solutions | Computational inefficient on non-sparse cases |
| Non-sparse outputs | Sparse outputs |
| No feature selection | Built-in feature selection |

Explain:
SVM 1 fundamental

# II. Supervised Learning

## H. SVM:

### H1. Basic SVM: Support Vector Machine

Explain:
SVM 2 fundamental

# II. Supervised Learning

## H. SVM:

### H1. Basic SVM

Explain:
SVM 3 fundamental

# II. Supervised Learning

## H. SVM:

### H1. Basic SVM

Explain:
SVM soft margin

# II. Supervised Learning

## H. SVM:

### H2. Soft margin

Explain:
SVM soft margin

# II. Supervised Learning

## H. SVM:

### H3. Kernel

# II. Supervised Learning

## H. SVM:

### H3. Kernel

| Type | Expression: $\kappa(x_i, x_j)$ | Parameters |
|---|---|---|
| linear | | |
| polynomial | | |
| Gaussian (RBF) | | |
| Laplacian | | |
| Sigmoid | | |

# II. Supervised Learning

## H. SVM:

### H4. Applying SVM

**n: # of features, m: # of samples**

- **n~ & n > m: logistic regression || SVM w/o kernel**
- **n_, m-: SVM w kernel**
- **n_, m~: add more features, then logistic || SVM w/o kernel**
- **Usually, SVM is not so good for large # of samples.**
- **NN can handle. Always.**

- **See L3**

# Summary: Supervised Learning

# II. Supervised Learning

## Summary:

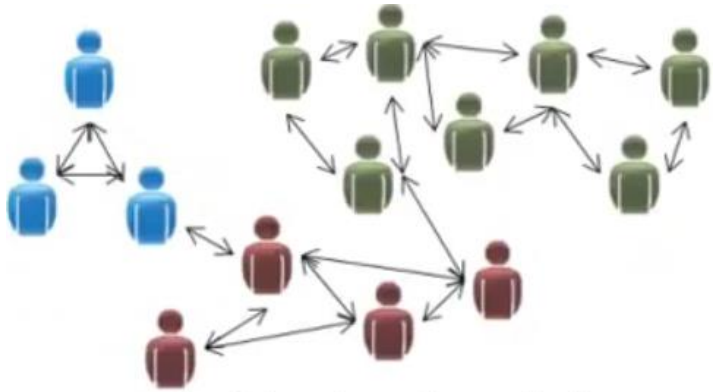
- **Linear Regression**
- **Logistic Regression / Classification**
- **L2 Loss**
- **Sigmoid Function**
- **Cross Entropy / Multi-Label**
- **Neural Network**
- **L1 / L2 Regularization**
- **Overfit / Zigzag**
- **BP / Cost Function / Hypothesis**
- **Gradient Vanishing / Explosion**
- **SVM / Kernel / Derivation**
- **Normal Equation**

# III. Unsupervised Learning

# III. Unsupervised Learning

## I. K-Means:

### I1. Clustering



Social network analysis

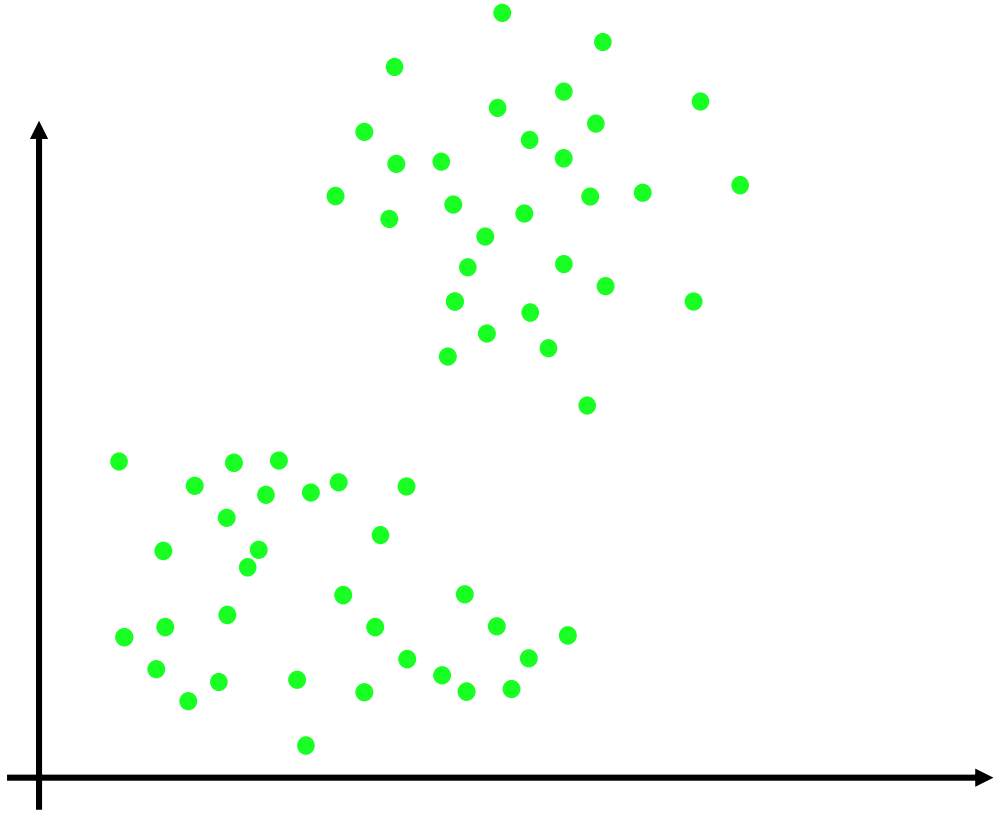
Organize computing clusters

Astronomical data analysis

# III. Unsupervised Learning

## I. K-Means:

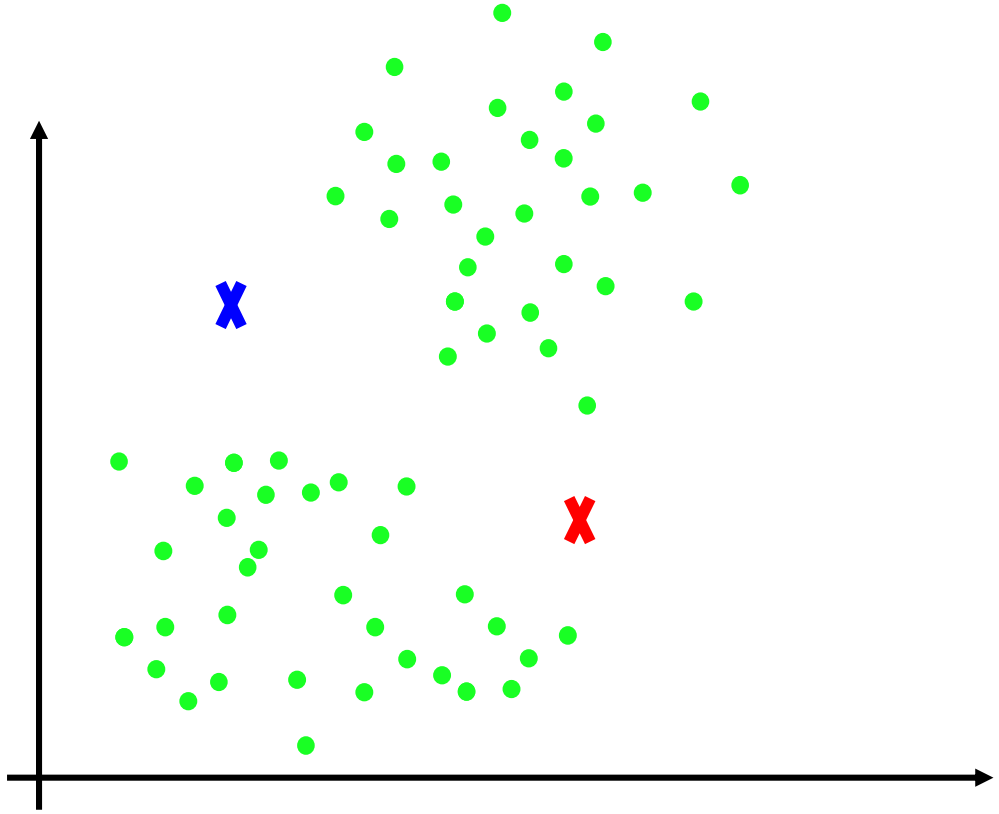
### I2. K-Means



# III. Unsupervised Learning

## I. K-Means:

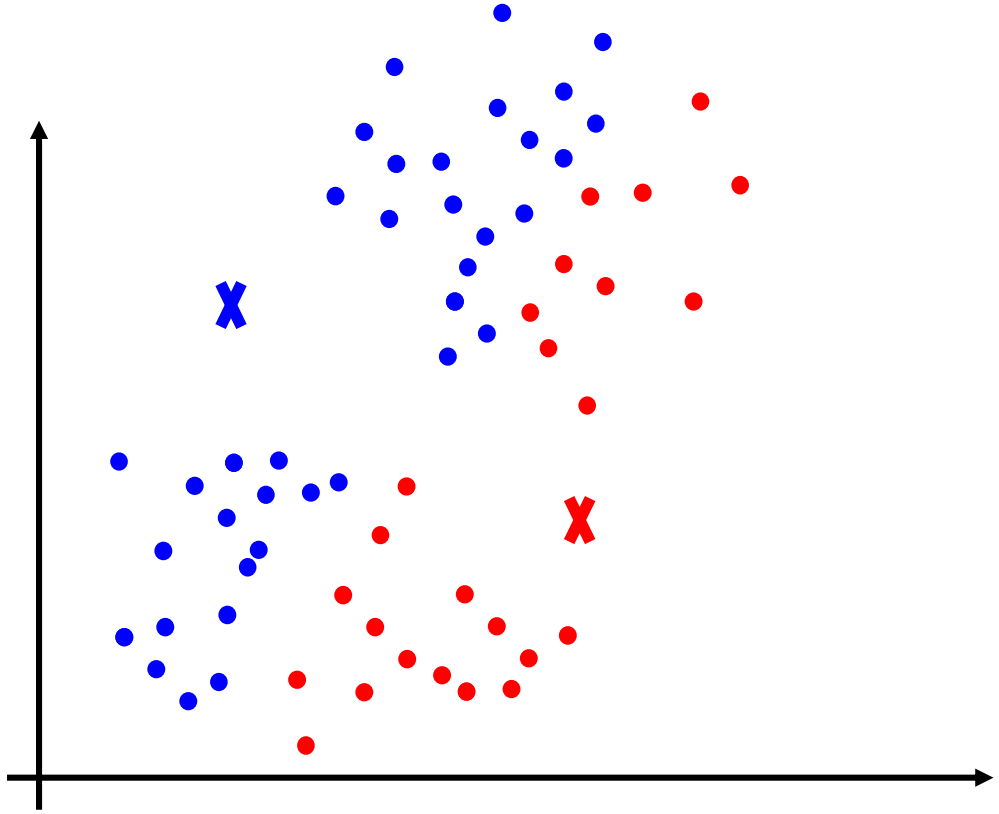
### I2. K-Means



# III. Unsupervised Learning

## I. K-Means:

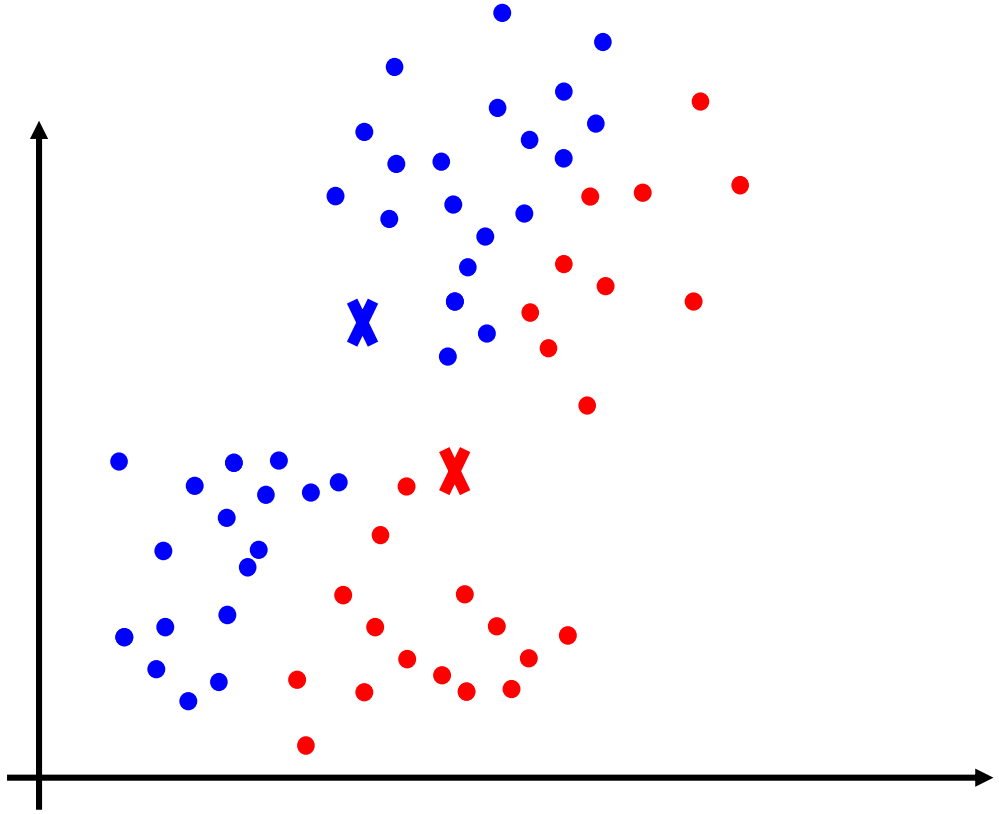
### I2. K-Means



# III. Unsupervised Learning

## I. K-Means:

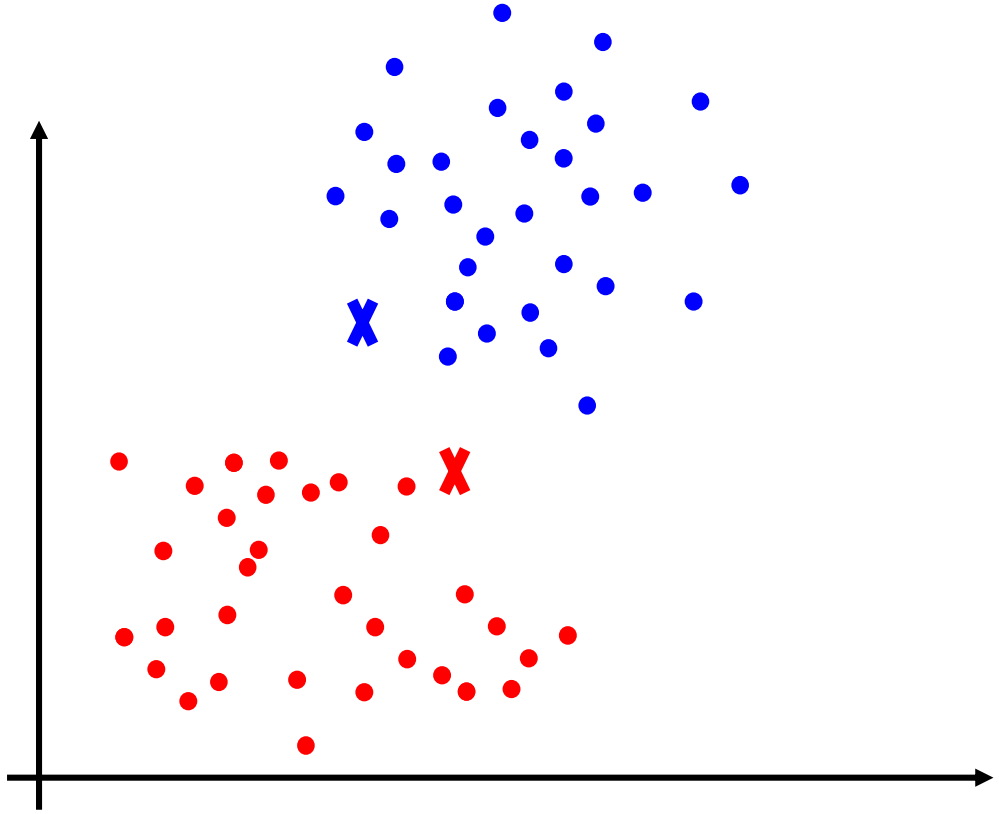
### I2. K-Means



# III. Unsupervised Learning

## I. K-Means:

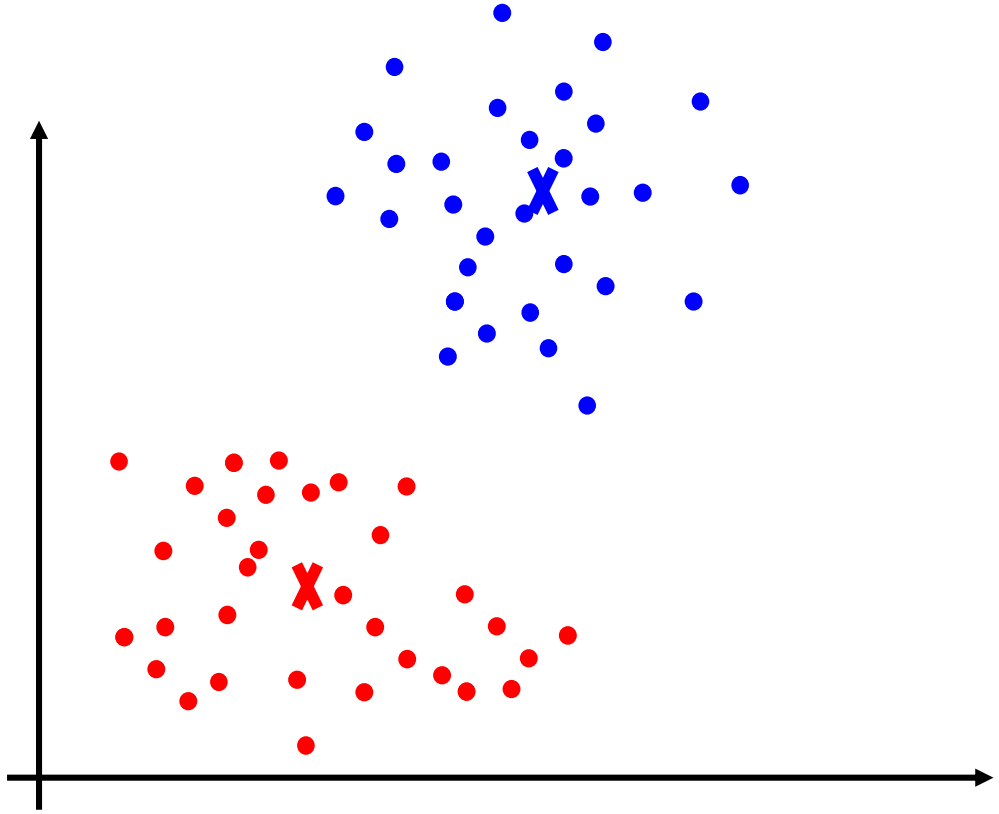
### I2. K-Means



# III. Unsupervised Learning

## I. K-Means:

### I2. K-Means



# III. Unsupervised Learning

## I. K-Means:

### I2. K-Means

**Input:** K: # of clusters
X: samples

**Start:** Randomly initialize K centroids $c_1, c_2, \cdots, c_K \epsilon \mathbb{R}^n$

**While:** $c_i$ not converging {
【step 1: update assignment for each sample $x_i$】
for $i$ = 1 to $m$:
$\underset{k}{argmin}(\|x_i - c_k\|)$
【step 2: update centroids】
for $k$ = 1 to $K$:
$c_k$ = mean of points assigned to cluster $k$
}

# III. Unsupervised Learning

## I. K-Means:

### I2. K-Means

**Implement: K-Means**

Explain:
K-Means++

# III. Unsupervised Learning

## I. K-Means:

### I3. K-Means-improved

Assignment!

**Problems:**

1. **Initialized seeds/centroids.** **K-Means++ [2007, Arthur & Vassilvitskii]**

1a. Take one center $c_1$, chosen uniformly at random from $\mathcal{X}$.
1b. Take a new center $c_i$, choosing $x \in \mathcal{X}$ with probability $\frac{D(x)^2}{\sum_{x \in \mathcal{X}} D(x)^2}$.
1c. Repeat Step 1b. until we have taken $k$ centers altogether.

**E.g.:**

Explain:
K-Means++

# III. Unsupervised Learning

## I. K-Means:

### I3. K-Means-improved

**Problems:**

1. **Initialized seeds/centroids.** **-K-Means++**

**E.g.:**

Explain:
K-Means++

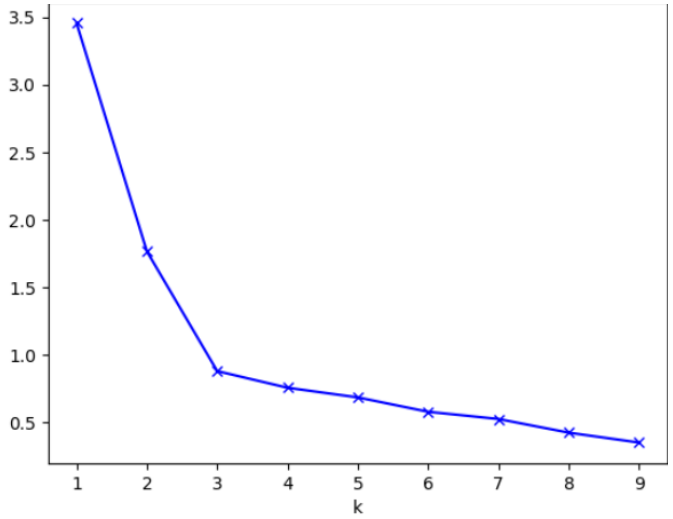
# III. Unsupervised Learning

## I. K-Means:

### I3. K-Means-improved

**Problems:**

1. **Initialized seeds/centroids.**
2. **K chosen. elbow method / more / ISODATA**



# III. Unsupervised Learning

## I. K-Means:

### I3. K-Means-improved

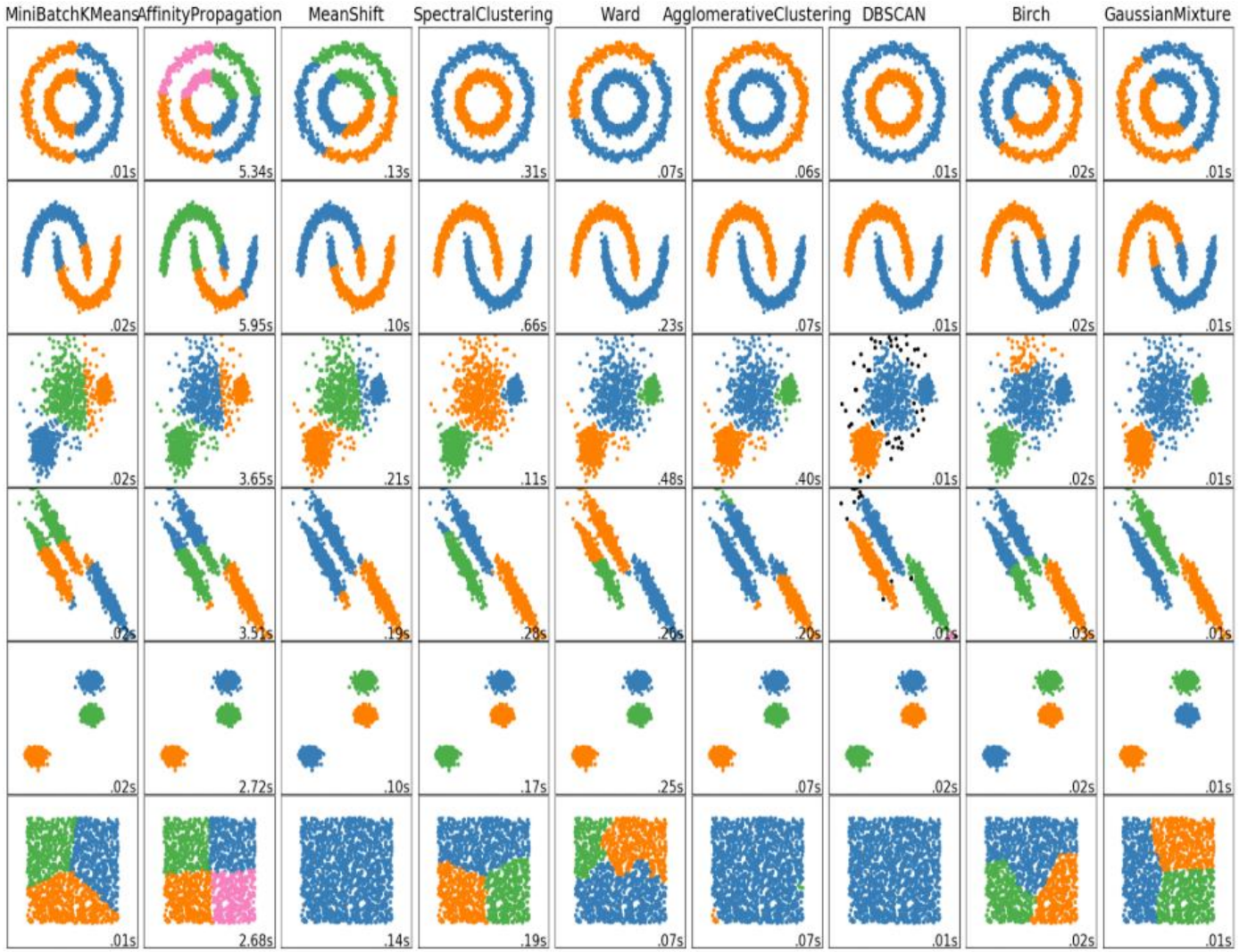
**Problems:**

1. Initialized seeds/centroids.
2. K chosen.
3. Not good for structural distribution
4. Slow when # of samples is big

# III. Unsupervised Learning

## I. K-Means:

### I4. Modern Methods




A comparison of the clustering algorithms in scikit-learn

# III. Unsupervised Learning

## I. K-Means:

### I5. kNN

**Decide a sample according to its k nearest neighbors**
**[Lazy-learning]**

Explain:
kNN

# III. Unsupervised Learning

## I. K-Means:

### I6. Comparison

| kNN | K-Means |
|---|---|
| 1. Classification / Regression<br>2. Supervised (Labeled) | 1. Clustering<br>2. Unsupervised(Unlabeled) |
| Memory-based learning (Needless to learn) | Have to learn |
| k: k neighbors | k: k clusters |
| | |
| Need to find Nearest Samples (NN), use KD-Tree sometimes. | |

# IV. Concepts & Problems

Explain:
Split

# IV. Concepts & Problems

## J. Training / Validation / Test Set

**Target:** To find a really good model

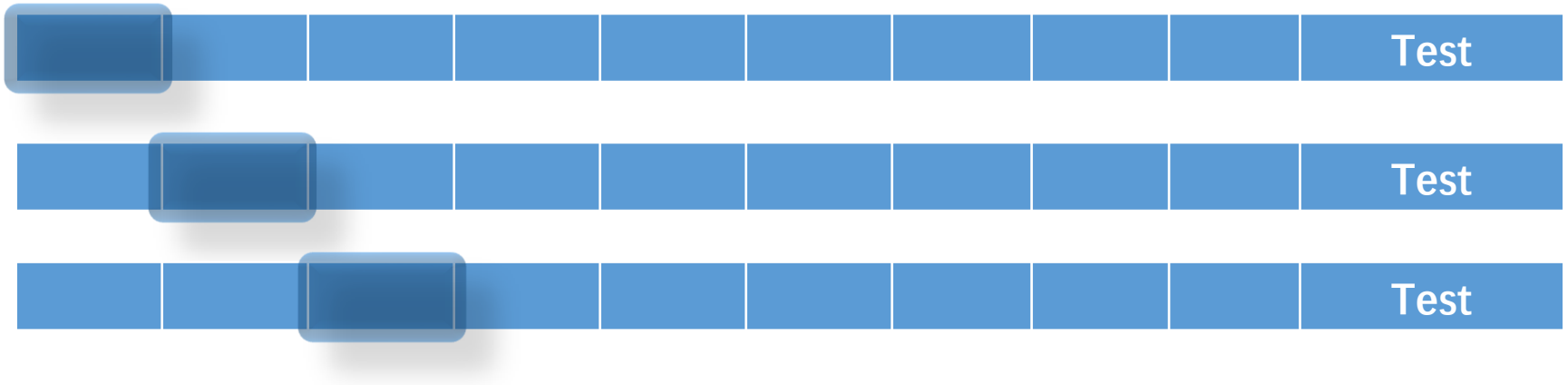
| Training | Validation | Test |
|---|---|---|

# IV. Concepts & Problems

## J. Training / Validation / Test Set

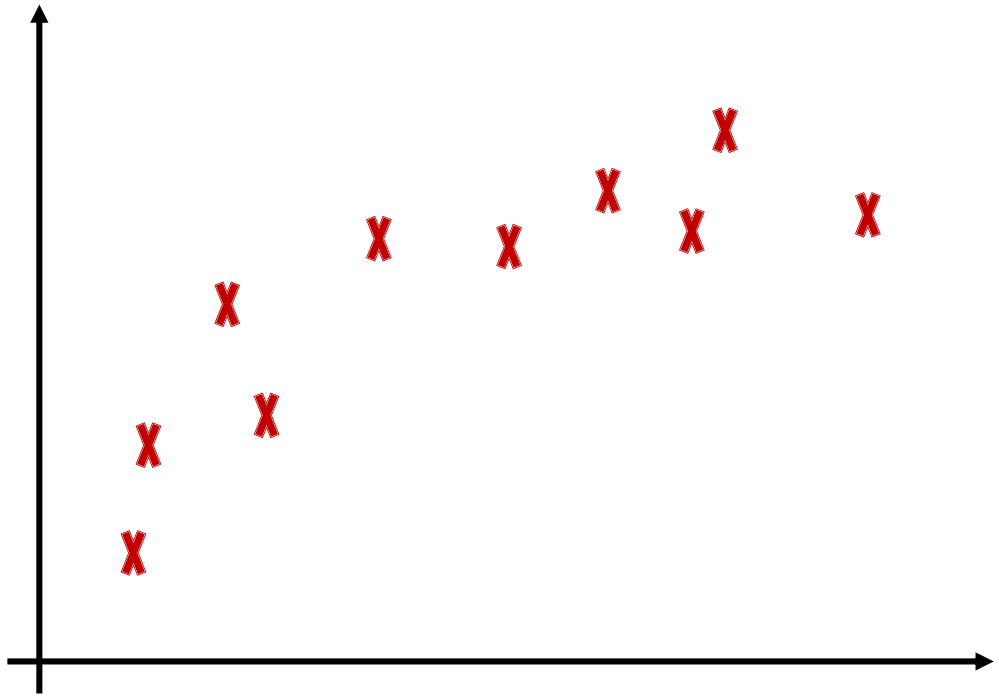
Explain:
Better model |
Hyperparameter
2 ways
ratio

**Target:** To find a really good model



# IV. Concepts & Problems

## K. Underfit / Overfit



# IV. Concepts & Problems

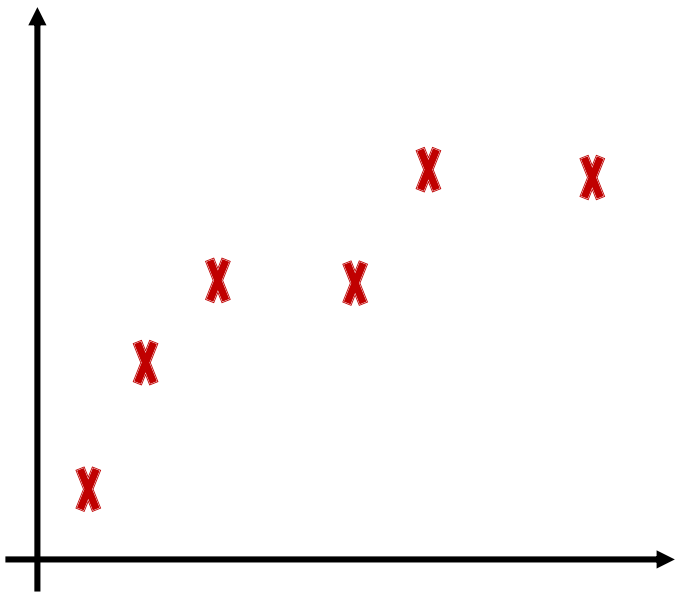
## K. Underfit / Overfit

| | Underfit | Overfit | |
|---|---|---|---|
| Phenomenon | 1. Not good even when training | 1. Very good when training, bad when validating | |
| Reasons | 1. Model's too simple<br>2. Data's too complex<br>3. Gradient/Weight's too small<br>4. Parameters' too less<br>5. Too much regularization | 1. Model's too complex<br>2. Data's too simple<br>3. Gradient/ Weight's too big<br>4. Parameters' too many<br>5. Less regularization | |
| Solutions | 1. More complex structure<br>2. Less data<br>3. Simple data<br>4. Less regularization | 1. Simpler structure<br>2. More data<br>3. Complex data<br>4. More regularization<br>5. Dropout<br>6. Batch Norm<br>7. Perturb Label<br>8. Noise | 9. Early stop<br>10. …… |

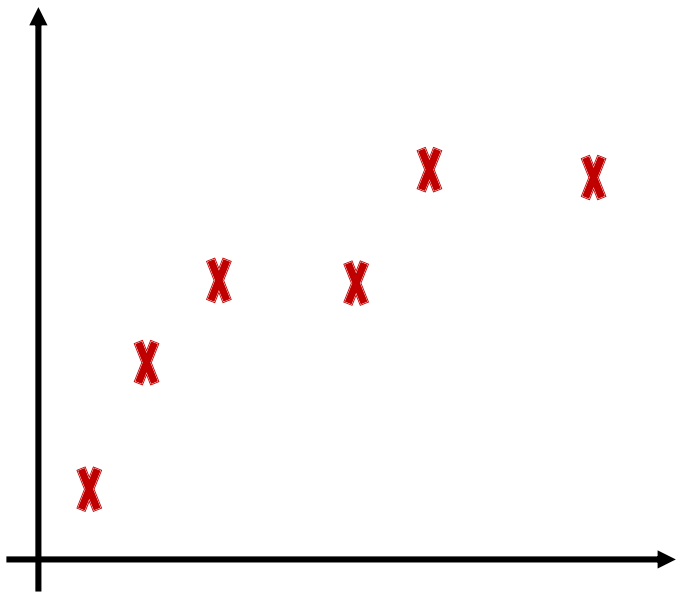
# IV. Concepts & Problems

## L. Bias / Variance

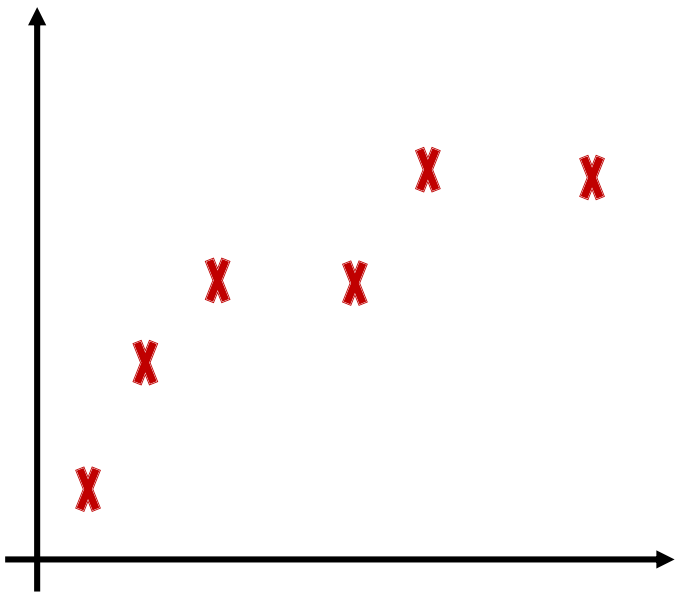
**High Bias: Underfit**



**Good**



**High Variance: Overfit**



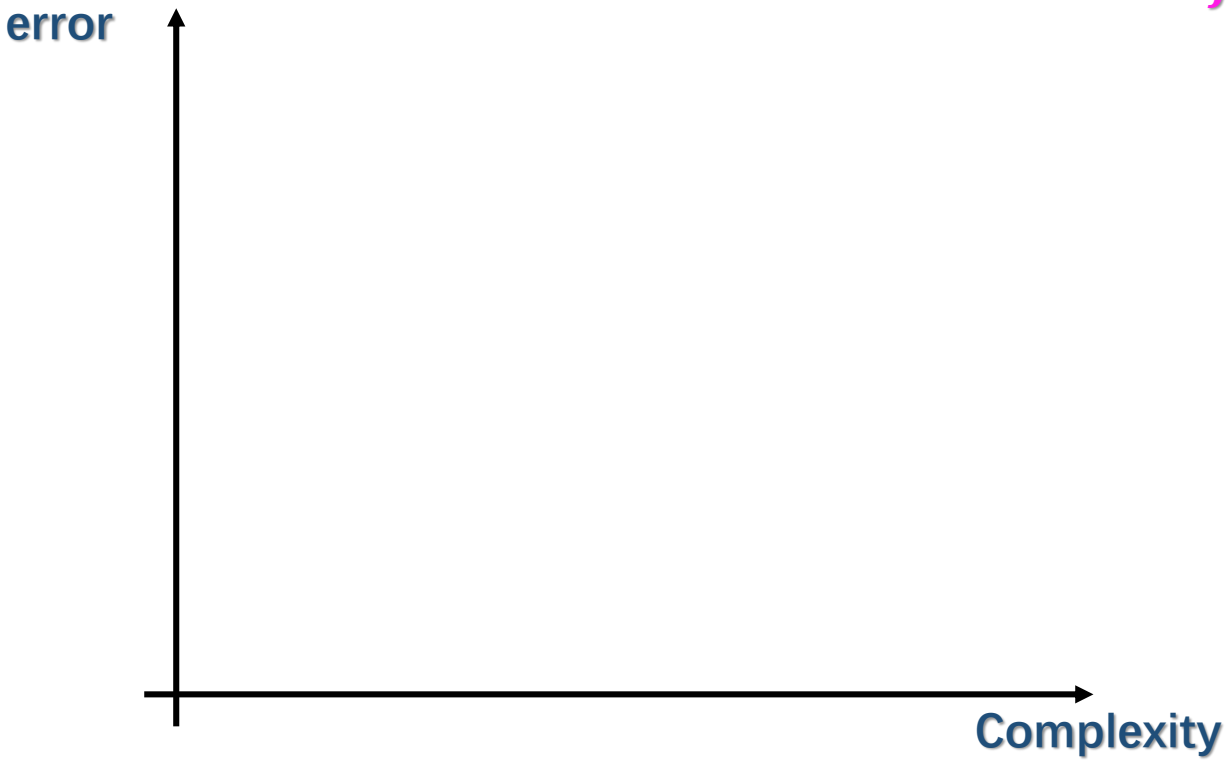
# IV. Concepts & Problems

## L. Bias / Variance

### L1. Complexity

$J_{train}(\theta)$

$J_{test}(\theta)$



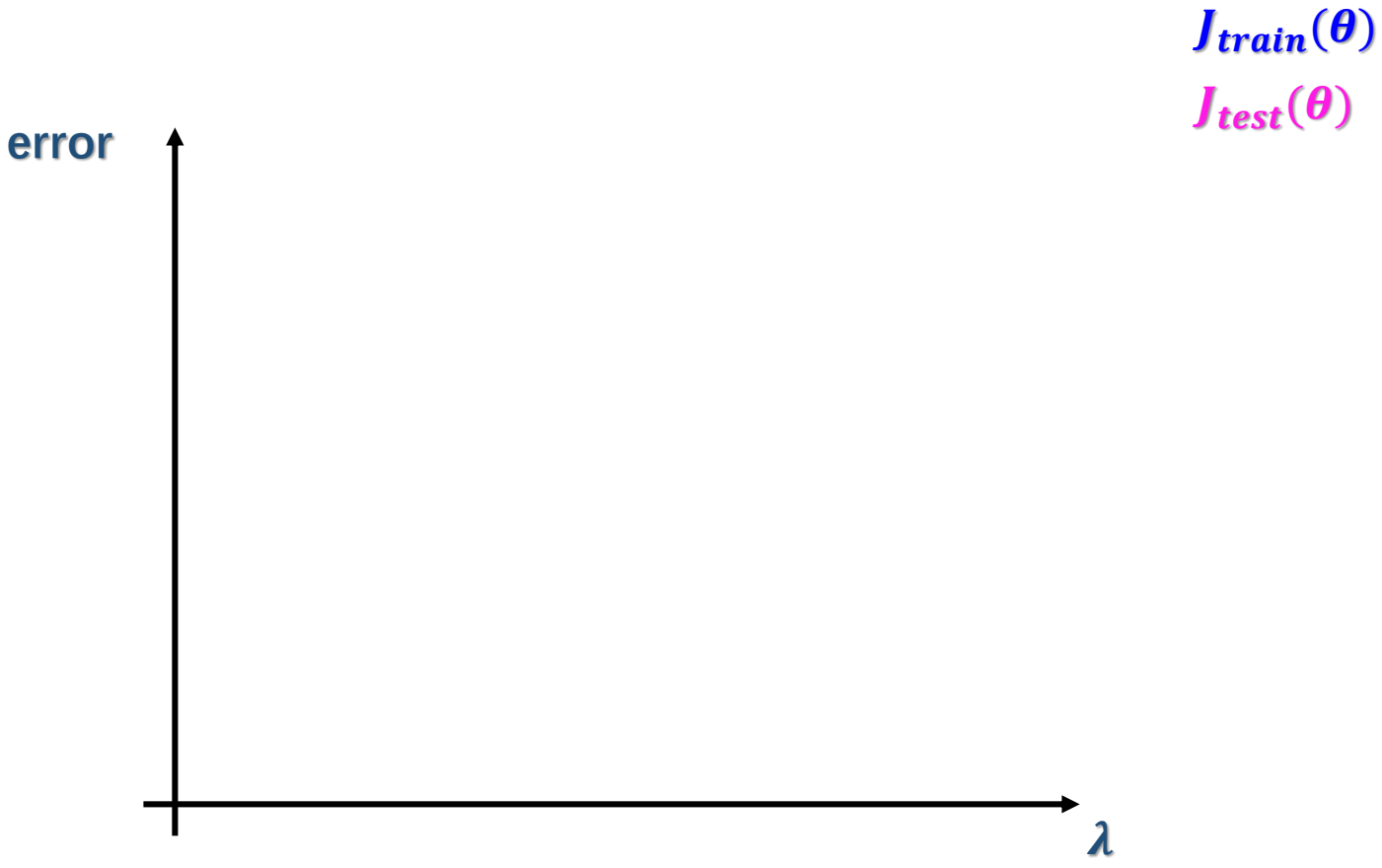
Explain: regularization
Bias Variance

# IV. Concepts & Problems

## L. Bias / Variance

### L2. Regularization

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^i) - y^i)^2 + \lambda\sum_{j=1}^{n}\theta_j^2\right]$$



# IV. Concepts & Problems

## L. Bias / Variance

### L3. Applying SVM

$$\begin{cases} \min_{\theta}\left(\frac{1}{2}\|\theta\|^2 + C\sum_{i}^{m}\xi_i\right) \\ s.t. \begin{cases} y_i\big(\theta\phi(x_i)\big) \geq 1 - \xi_i \\ \xi_i \geq 0 \end{cases} \end{cases}$$

Explain: why
No sample in bias

# IV. Concepts & Problems

## L. Bias / Variance

### L4. In Sum

| | |
|---|---|
| To reduce high variance | $+\lambda$ |
| | -complexity |
| | +samples |
| To reduce high bias | +complexity |
| | $-\lambda$ |

# IV. Concepts & Problems

## M. Gradient Vanishing / Explosion

# V. Other Classical ML Tools

# V. Other Classical ML Tools

## N. Decision Tree:

### Some concepts:

- $\boldsymbol{Entropy}$: 随机变量的不确定，称“(信息)**熵**”，也“即系统混乱程度”

$$H(Y) = -\sum_{i=1}^{n} p_i log p_i$$

- $\boldsymbol{Conditional\ Entropy}$: 随机变量X条件下Y的不确定性

$$H(Y|X) = -\sum_{i=1}^{n} p_i H(Y|X = x_i), p_i = P(X = x_i)$$

- $\boldsymbol{Information\ Gain}$: 信息增益：信息F对系统带来信息G

$$G(Y|X) = H(Y) - H(Y|X)$$

# V. Other Classical ML Tools

## N. Decision Tree: ID3 / C4.5 / CART

**History:**

Quinlan 1979

Quinlan 1993

Breiman 1984

**I: Iterative**

**D: Dichotomiser**

**3: 3rd generation of inductive learning**

# V. Other Classical ML Tools

## N. Decision Tree: ID3

### E.G.: 女婿受丈母娘欢迎度

| ID | Appearance | Income | Age | Profession | 是否受欢迎 |
|---|---|---|---|---|---|
| 1 | Good | Low | Older | Steady | N |
| 2 | Good | Low | Older | Unstable | N |
| 3 | Great | Low | Older | Steady | Y |
| 4 | Ah | Good | Older | Steady | Y |
| 5 | Ah | Great | Younger | Steady | Y |
| 6 | Ah | Great | Younger | Unstable | N |
| 7 | Great | Great | Younger | Unstable | Y |
| 8 | Good | Good | Older | Steady | N |
| 9 | Good | Great | Younger | Steady | Y |
| 10 | Ah | Good | Younger | Steady | Y |
| 11 | Good | Good | Younger | Unstable | Y |
| 12 | Great | Good | Older | Unstable | Y |
| 13 | Great | Low | Younger | Steady | Y |
| 14 | Ah | Good | Older | Unstable | N |

**Target:**

是否受欢迎: {Y:9, N:5}

**Attribute:**

Appearance: { Ah: 5=3Y+2N,
Good: 5=2Y+3N,
Great: 4=4Y}

Income: { Low: 4=2Y+2N,
Good: 6=4Y+2N,
Great: 4=3Y+1N}

Age: { Younger: 7=3Y+4N,
Older: 7=4Y+2N}

Profession: { Unstable: 6=3Y+3N,
Steady: 8=6Y+2N}

# V. Other Classical ML Tools

## N. Decision Tree: ID3

### E.G.: 女婿受丈母娘欢迎度

**Target:**

是否受欢迎: {Y:9, N:5}

**Attribute:**

Appearance: { Ah: 5=3Y+2N,
Good: 5=2Y+3N,
Great: 4=4Y}

Income: { Low: 4=2Y+2N,
Good: 6=4Y+2N,
Great: 4=3Y+1N}

Age: { Younger: 7=3Y+4N,
Older: 7=4Y+2N}

Profession: { Unstable: 6=3Y+3N,
Steady: 8=6Y+2N}

- ***Entropy***:

$$H(Y) = -\sum_{i=1}^{n} p_i log p_i$$

- ***Conditional Entropy***:

$$H(Y|X) = -\sum_{i=1}^{n} p_i H(Y|X = x_i), p_i = P(X = x_i)$$

**a. Calculate the entropy of the system**

$$H(D) = -\sum_{k=1}^{K} p_k log p_k = -\frac{9}{14}\log(\frac{9}{14}) - \frac{5}{14}\log(\frac{5}{14}) = 0.94$$

**b. Calculate entropies for each feature**

$$Appearance: H(F_{Great}) = -\frac{4}{4}\log\left(\frac{4}{4}\right) = 0$$

$$H(F_{Good}) = -\frac{2}{5}\log\left(\frac{2}{5}\right) - \frac{3}{5}\log\left(\frac{3}{5}\right) = 0.971$$

$$H(F_{Ah}) = -\frac{3}{5}\log\left(\frac{3}{5}\right) - \frac{2}{5}\log\left(\frac{2}{5}\right) = 0.971$$

$$\boldsymbol{H(D|F_{App})} = \frac{4}{14}H(F_{Great}) + \frac{5}{14}H(F_{Good}) + \frac{5}{14}H(F_{Ah})$$
$$= \mathbf{0.693}$$

# V. Other Classical ML Tools

## N. Decision Tree: ID3

### E.G.: 女婿受丈母娘欢迎度

**Target:**

是否受欢迎: {Y:9, N:5}

**Attribute:**

Appearance: { Ah: 5=3Y+2N,
Good: 5=2Y+3N,
Great: 4=4Y}

Income: { Low: 4=2Y+2N,
Good: 6=4Y+2N,
Great: 4=3Y+1N}

Age: { Younger: 7=3Y+4N,
Older: 7=4Y+2N}

Profession: { Unstable: 6=3Y+3N,
Steady: 8=6Y+2N}

- ***Entropy***:

$$H(Y) = -\sum_{i=1}^{n} p_i log p_i$$

- ***Conditional Entropy***:

$$H(Y|X) = -\sum_{i=1}^{n} p_i H(Y|X = x_i), p_i = P(X = x_i)$$

**a. Calculate the entropy of the system**

$$H(D) = -\sum_{k=1}^{K} p_k log p_k = -\frac{9}{14}\log(\frac{9}{14}) - \frac{5}{14}\log(\frac{5}{14}) = 0.94$$

**b. Calculate entropies for each feature**

$$H(D|F_{App}) = 0.693 \qquad H(D|F_{Age}) = 0.789$$

$$H(D|F_{Inc}) = 0.911 \qquad H(D|F_{Job}) = 0.892$$

**c. Get Info Gain & d. Split feature with max gain**

$$G(D|F_{App}) = H(D) - H(D|F_{App}) = 0.94 - 0.693 = 0.246$$

$$G(D|F_{Inc}) = H(D) - H(D|F_{Inc}) = 0.94 - 0.911 = 0.029$$

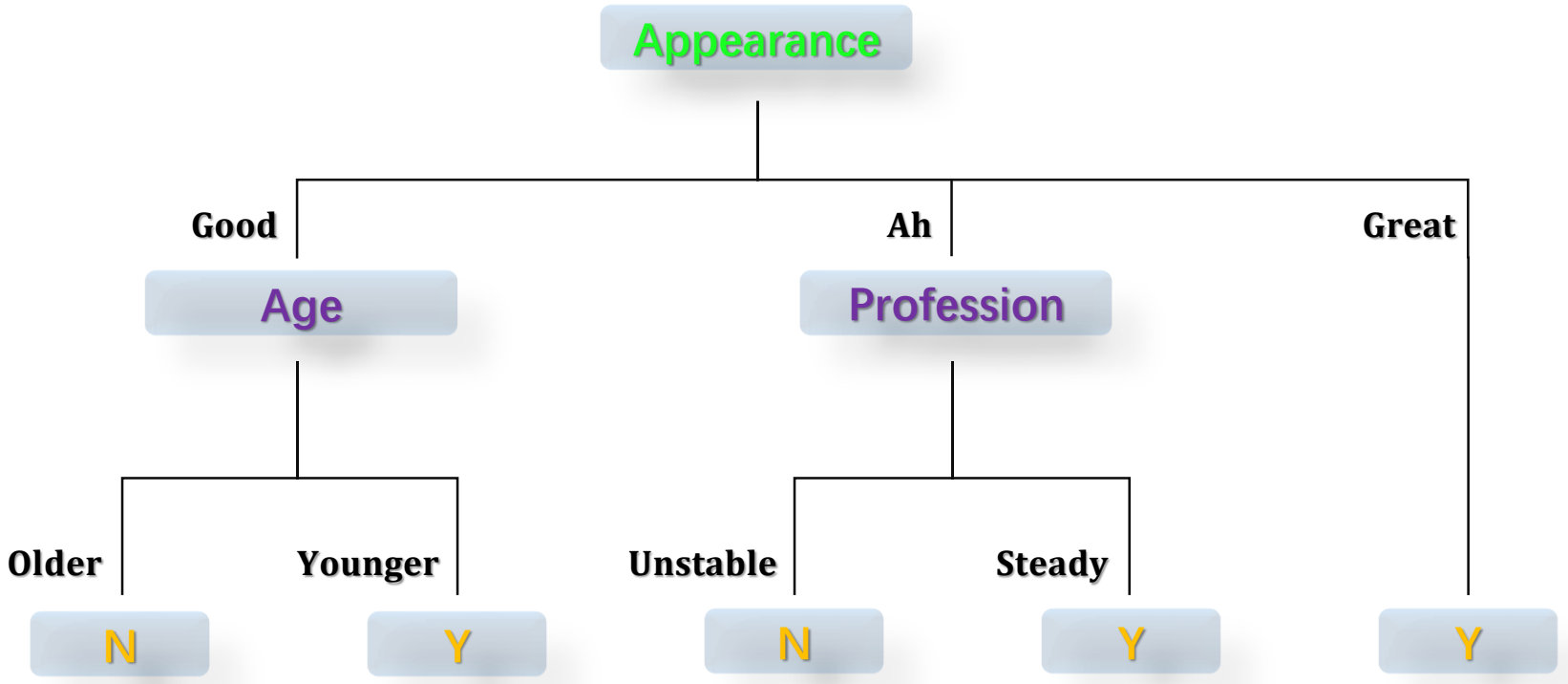
$$G(D|F_{Age}) = H(D) - H(D|F_{Age}) = 0.94 - 0.798 = 0.151$$

$$G(D|F_{Job}) = H(D) - H(D|F_{Job}) = 0.94 - 0.892 = 0.048$$

# V. Other Classical ML Tools

## N. Decision Tree: ID3

### E.G.: 女婿受丈母娘欢迎度



# V. Other Classical ML Tools

## N. Decision Tree: ID3

### Pros & Cons:

Pros: 1. Easy to understand
2. Classification + Regression

Cons: 1. Discrete
2. Prone to overfit
3. NP-Complete [Greedy algorithm: local optimum]
4. Usually easy to choose features having more attributes

Solutions: 1. Prune
2. # in leaf node
3. C4.5

# V. Other Classical ML Tools

## N. Decision Tree: ID3 / C4.5 / CART

### More than ID3:

**Except Gain:**

- $\boldsymbol{SplitInformation(D|F) = -\sum_{i=1}^{n} \frac{|D_i|}{|D|} log \frac{|D_i|}{|D|}}$

- $\boldsymbol{GainRatio(D|F) = \frac{G(D|F)}{SplitInformation(D|F)}}$

- $\boldsymbol{Choose\ argmax}$

# V. Other Classical ML Tools

## N. Decision Tree: ID3 / C4.5 / CART

### It's a binary tree:

a. **Calculate the Gini Index**

$$Gini(D) = \sum_{k=1}^{K} p_k(1 - p_k) = 1 - \sum_{k=1}^{K} p_k^2$$

b. **Calculate conditional Gini Index for each feature**

$$Gini(D|F) = \frac{D_1}{D} Gini(D_1) + \frac{D_2}{D} Gini(D_2)$$

c. **Split feature with min Gini Index**

# V. Other Classical ML Tools

## N. Decision Tree: ID3 / C4.5 / CART

### E.G.: 女婿受丈母娘欢迎度

| ID | Appearance | Income | Age | Profession | 是否受欢迎 |
|---|---|---|---|---|---|
| 1 | Good | Low | Older | Steady | N |
| 2 | Good | Low | Older | Unstable | N |
| 3 | Great | Low | Older | Steady | Y |
| 4 | Ah | Good | Older | Steady | Y |
| 5 | Ah | Great | Younger | Steady | Y |
| 6 | Ah | Great | Younger | Unstable | N |
| 7 | Great | Great | Younger | Unstable | Y |
| 8 | Good | Good | Older | Steady | N |
| 9 | Good | Great | Younger | Steady | Y |
| 10 | Ah | Good | Younger | Steady | Y |
| 11 | Good | Good | Younger | Unstable | Y |
| 12 | Great | Good | Older | Unstable | Y |
| 13 | Great | Low | Younger | Steady | Y |
| 14 | Ah | Good | Older | Unstable | N |

**Target:**

是否受欢迎: {Y:9, N:5}

**Attribute:**

Appearance: { Ah: 5=3Y+2N,
Good: 5=2Y+3N,
Great: 4=4Y}

Income: { Low: 4=2Y+2N,
Good: 6=4Y+2N,
Great: 4=3Y+1N}

Age: { Younger: 7=3Y+4N,
Older: 7=4Y+2N}

Profession: { Unstable: 6=3Y+3N,
Steady: 8=6Y+2N}

# V. Other Classical ML Tools

## N. Decision Tree: ID3 / C4.5 / CART

### E.G.: 女婿受丈母娘欢迎度

**Target:**

是否受欢迎: {Y:9, N:5}

**Attribute:**

Appearance: { Ah: 5=3Y+2N,
Good: 5=2Y+3N,
Great: 4=4Y}

Income: { Low: 4=2Y+2N,
Good: 6=4Y+2N,
Great: 4=3Y+1N}

Age: { Younger: 7=3Y+4N,
Older: 7=4Y+2N}

Profession: { Unstable: 6=3Y+3N,
Steady: 8=6Y+2N}

**a. Calculate Gini Index for Profession**

$$Gini(F_{Unstable}) = 1 - (\frac{3}{6})^2 - (\frac{3}{6})^2$$

$$Gini(F_{Steady}) = 1 - (\frac{2}{8})^2 - (\frac{6}{8})^2$$

$$Gini(D|F_{Job}) = \frac{6}{14} Gini(F_{Unstable}) + \frac{8}{14} Gini(F_{Steady}) = A$$

**b. Calculate Gini Index for Appearance [> 2 *branches*]**

1. $Ah|Good, Great :=> B_1$
2. $Good|Ah, Great :=> B_2$
3. $Great|Ah, Good :=> B_3$

$B = \min(B_1, B_2, B_3)$

**c. Split Feature according to min Gini Index**

Split feature ID = $argmin\{A, B, C, \ldots\})$

# V. Other Classical ML Tools

## O. Reading Parts: AdaBoost / Haar Feature

- **AdaBoost:** adaptive boost [1995, Freund & Schapire]
  [source paper / explanation]

- **Haar Feature:** [2001 Viola & Jones]
  [source paper / explanation]