ML Application

Data Mining (Web click data, biology, engineering)

Application can't program by hand

(NLP, CV, handwriting recognition)

Self-customizing programs

ML Learning definition

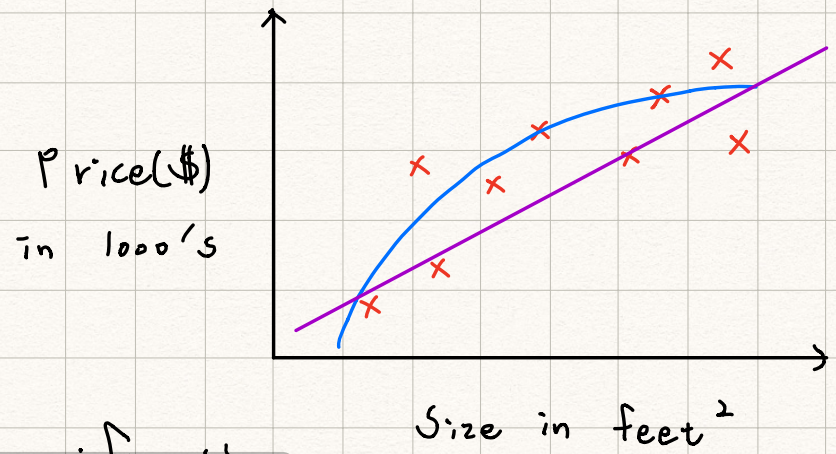
A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E.

Supervise Learning

Regression

Predict continuous valued output (price)

房地產預測

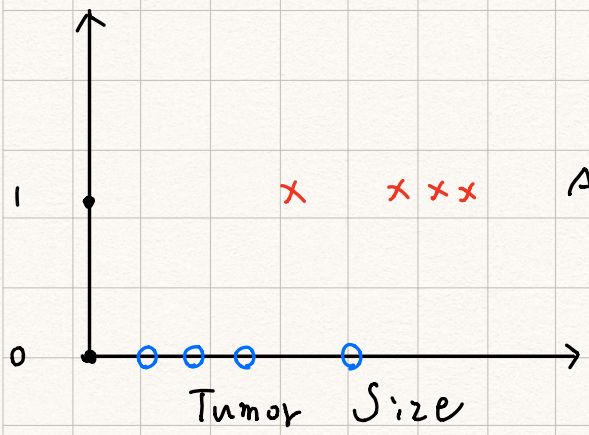


Classification

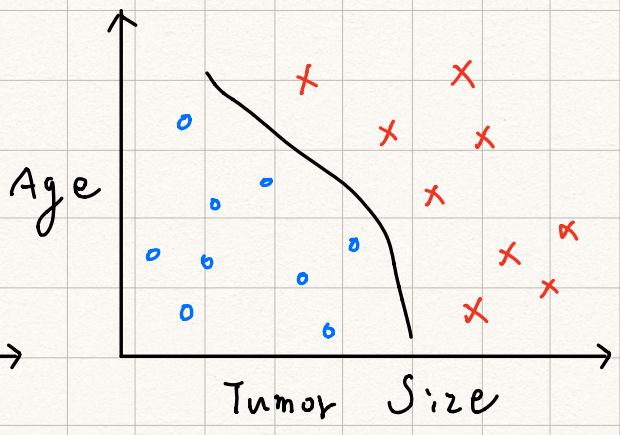
Discrete valued output (0 or 1)

可能不只 0.1 的分類，0. 1. 2. 3

健康 癌1 癌2 癌3



one feature



two features

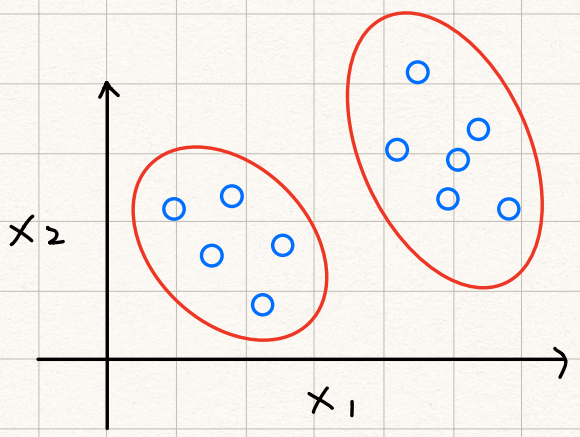
∞ 個 feautreas ?

支持向量機

Unsupervise Learning

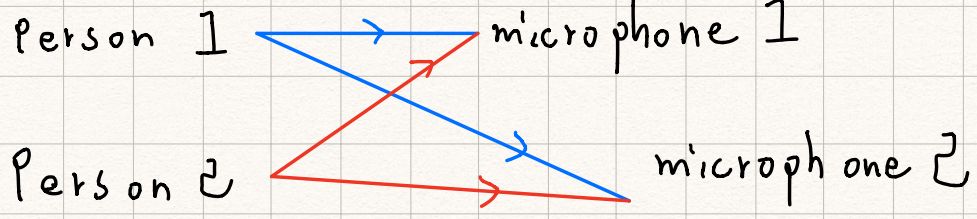
資料沒有標籤 or 屬性
沒有 Test data

Clustering Algorithm



用來組織大量計算群

Cocktail Party Algorithm



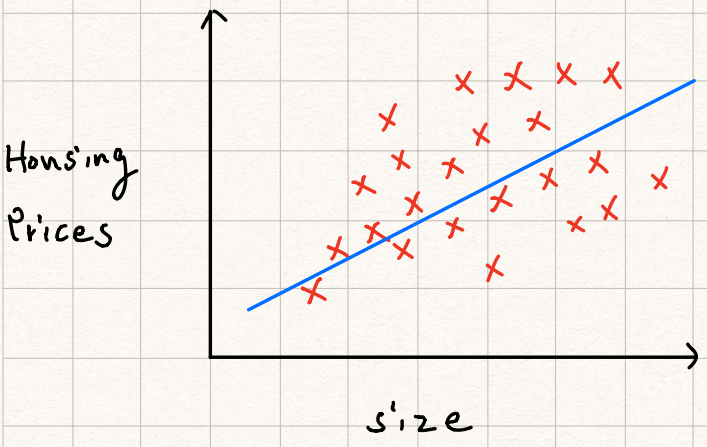
利用算法剔除其中一種聲源

Octave 確認算法可行後，

再至 C++、Java 環境重新建置

Model Representation

Linear Regression



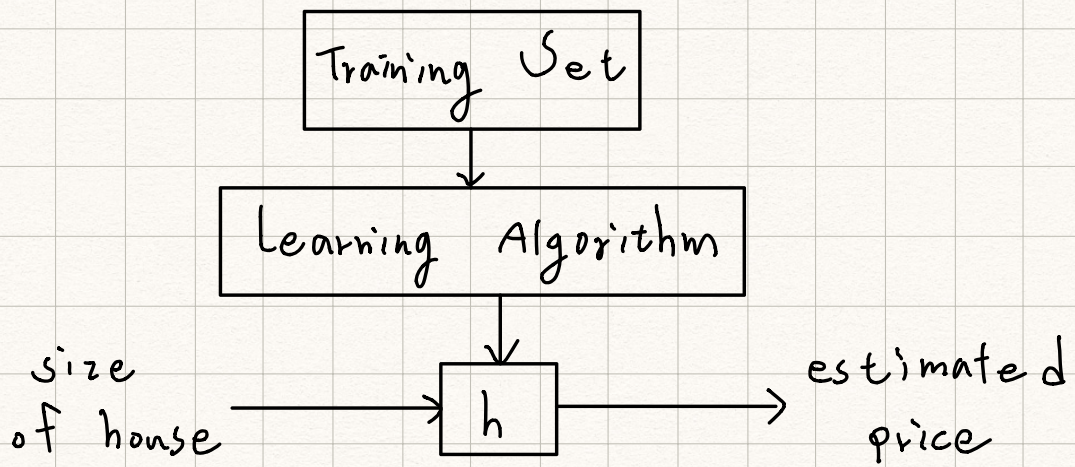
Supervised Learning ↓

Given the "right answer" for each example in the data.

有test data

本課的符號說明：

m: Number of training data
x's: "input" variable / features
y's: "output" variable / "target" variable



Cost function

h (hypothesis): $h_\theta(x) = \theta_0 + \theta_1 x$

θ_0, θ_1 被稱為 模型參數

minimize θ_0, θ_1 $\rightsquigarrow$ $\frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$

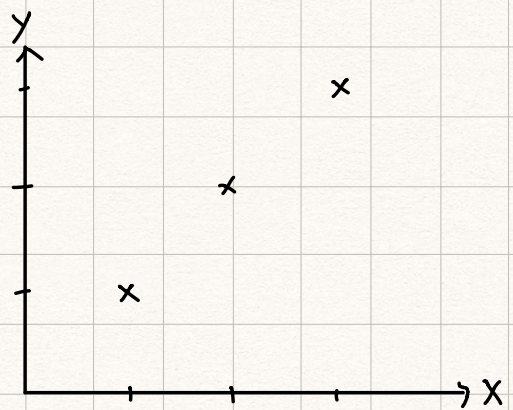
$h_\theta(x^{(i)}) = \theta_0 + \theta_1 x^{(i)}$

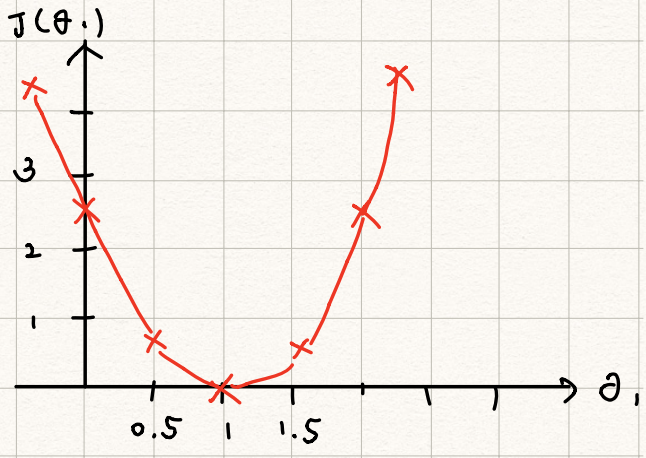
squared error function = $J(\theta_0, \theta_1)$ cost function

對於大部分回歸問題皆適用

Figure Comparison I

hyphethesis : $h_\theta(x)$ $\theta_0 = 0$, cost function $\therefore J(\theta_1)$



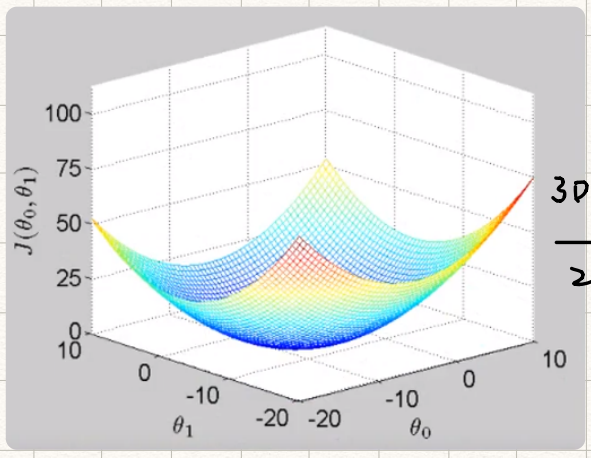


當 θ_1 改變, cost function 值也不一樣

最小化 $J(\theta_1)$，找到擬合最佳點

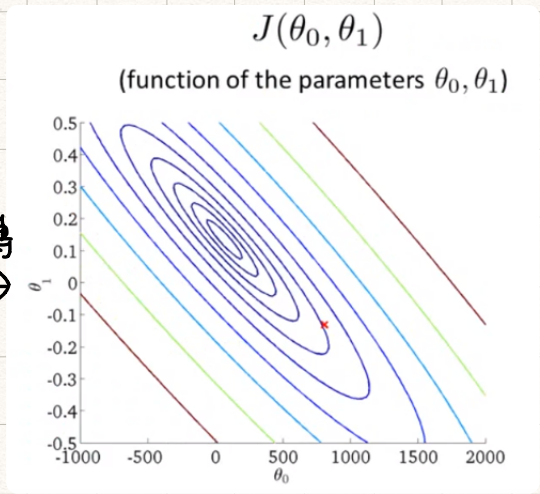
Figure Comparison II

$\theta_0, \theta_1 \neq 0$，
則根據訓練樣本，會長得類似此圖



3D轉 → 2D

輪廓圖
(contour plot)



每圈的 cost function 值是一樣的

Gradient Descent

$J(\theta_0, \theta_1)$, want $\min_{\theta_0, \theta_1} (\theta_0, \theta_1)$

$J(\theta_0, \theta_1, \theta_2, \theta_3 \ldots\ldots \theta_n)$ 的題也可用此法解

山坡 → 山底

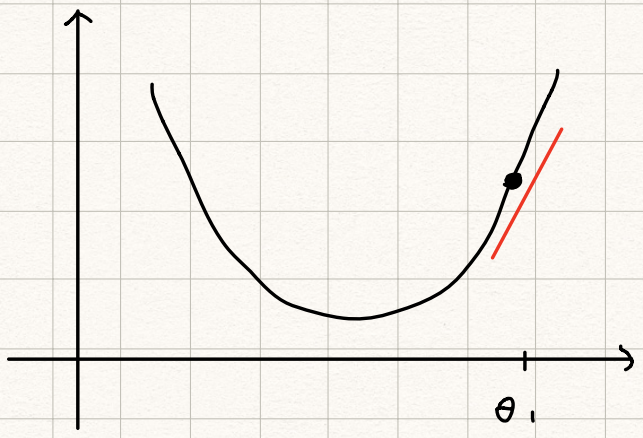
repeat until convergence

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \quad (\text{for } j=0 \text{ and } j=1)$$

assignment　learning rate　simultaneously update

人們談的 gradient descent

Intuition



$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1)$$

≥ 0

$$\theta_1 := \theta_1 - \alpha \, (\text{positive num})$$

α 太大，導致收斂 or 發散

Gradient descent can converge to a local minimum, even with the learning rate α fixed.

gradient descent can automatically take smaller steps.

因此 α 值不必特意再調整

越近 Local minimum，斜率越小

每次更新值隨之越小

Gradient Descent For Linear Regression

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \quad (\text{for } j=0 \text{ and } j=1)$$

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

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

$$\frac{\partial}{\partial \theta_j} = \frac{\partial}{\partial \theta_j} \cdot \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2$$

$$= \frac{\partial}{\partial \theta_j} \cdot \frac{1}{2m} \sum_{i=1}^{m} \left(\theta_0 + \theta_1 x^{(i)} - y^{(i)} \right)^2$$

θ_0 $j=0$: $\frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)$

θ_1 $j=1$: $\frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right) \cdot x^{(i)}$

$$\begin{cases} \theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right) \\ \theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right) \cdot x^{(i)} \end{cases}$$

※

Linear Regression 的 loss function 圖形總為 Convex Function (凸函數)，無局部極小值

Linear Algebra Review

Matrix: Rectangular arry of numbers

$$A = \begin{bmatrix} a & b \\ c & d \\ e & f \end{bmatrix}_{x \times y}$$

$x = 3,\ y = 2$

$A_{11} = a,\ A_{22} = d.$

Dimension of matrix:
number of rows x number of columns

Vector: An $n \times 1$ matrix

1-indexed vs 0-indexed

$$y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} \qquad y = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix}$$

程式語言通常是以 1-indexed 為主

Matrix - vector multiplication

pratical problem : $h_\theta(x) = -40 + 0.25x$

House sizes : 2104 . 1416 . 1534 . 852

$$\begin{bmatrix} 1 & 2104 \\ 1 & 1416 \\ 1 & 1534 \\ 1 & 852 \end{bmatrix} \times \begin{bmatrix} -40 \\ 0.25 \end{bmatrix} = \begin{bmatrix} -40\times1 + 0.25\times 2104 \\ -40\times1 + 0.25\times 1416 \\ \vdots \end{bmatrix}$$

可以以矩陣運算特性計算題目，如此一來便不須使用 loop

Matlab 對 Array Calculation 有優化！

承上題，

$h_\theta(x) = -40 + 0.25x$

$h_\theta(x) = 200 + 0.1x$

$h_\theta(x) = -150 + 0.4x$

$$\underset{4\times2}{\begin{bmatrix} 1 & 2104 \\ 1 & 1416 \\ 1 & 1534 \\ 1 & 852 \end{bmatrix}} \times \underset{2\times3}{\begin{bmatrix} -40 & 200 & -150 \\ 0.25 & 0.1 & 0.4 \end{bmatrix}} = \underset{4\times3}{\begin{bmatrix} \cdots\cdots \end{bmatrix}}$$

$A \times B \neq B \times A$

I: identity matric $\begin{bmatrix} 1 & 0 & 0 \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ 0 & \cdots & \ddots & 0 \\ 0 & 0 & 0 \cdots & 1 \end{bmatrix}$

$A \times I = I \times A$

Inverse and transpose

$A(A^{-1}) = A^{-1}A = I$ A^{-1}: inverse

Matrices that don't have an inverse are "singular" or "degenerate"

$A = \begin{pmatrix} 1 & 2 & 0 \\ 3 & 5 & 9 \end{pmatrix}$ $B = A^T = \begin{bmatrix} 1 & 3 \\ 2 & 5 \\ 0 & 9 \end{bmatrix}$

$B_{ij} = A_{ji}$