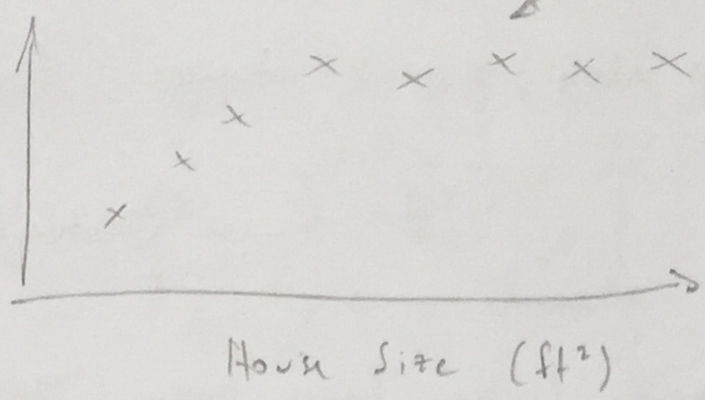
~~COURSERA~~ ML - WEEK 1

What Is Machine Learning?

• Supervised Learning:

eg.] Price in $1000's

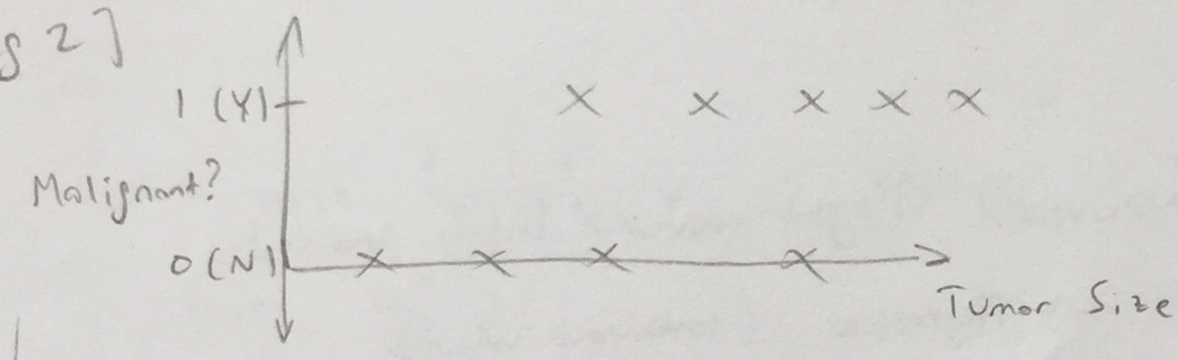


=> fit line, quadratic to data, etc.

↳ SUPERVISED → RIGHT ANSWERS GIVEN (fit algorithm to data set)

↳ Regression Problem: predict continuous valued output (e.g. price)

eg 2]

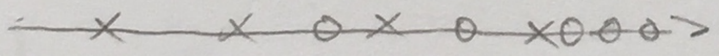


=> given tumor size, what is P(malignant)?

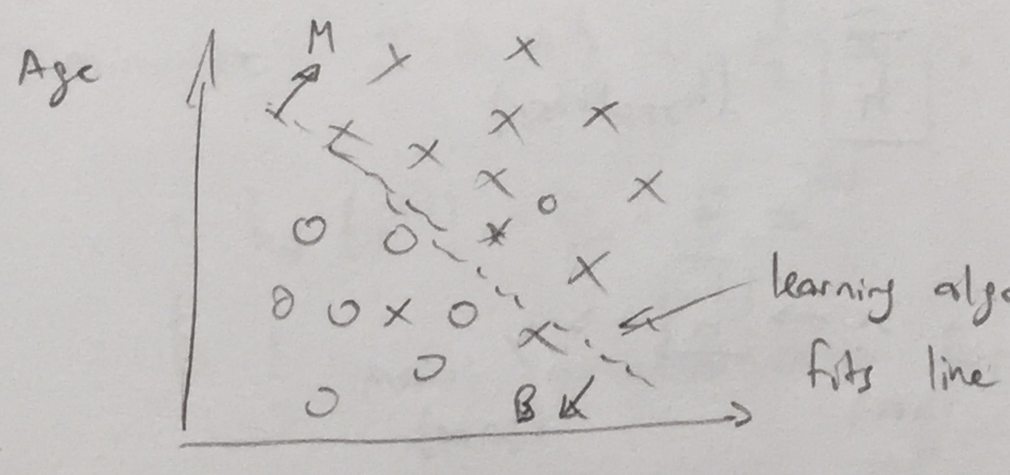
↳ CLASSIFICATION PROBLEM → predict discrete output

↳ can have 2, 3, ... discrete outcomes

REPLOT:



eg 3] Multiple Features / Attributes:



=> can have many other features (clump thickness, uniformity, etc.)

Some algorithms need ∞ features (math trick)

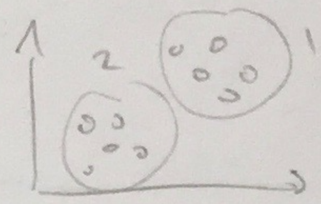
- Regression → CONTINUOUS VALUE OUTPUTS
- Classification → DISCRETE OUTPUTS

- Unsupervised Learning

↳ Supervised: given data set, get discrete outputs (classification) or continuous outputs (regression)

↳ Unsupervised: given data w/o labels → make inferences / find structure in data

↳ e.g. find clusters



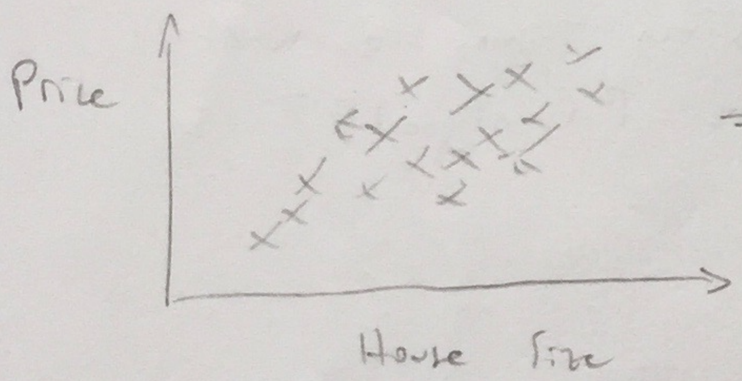
→ Clustering Algorithm

↳ eg. Google News, Genomics, Computer Clusters, Social Media, Market Segmentation, Astronomical

↳ e.g. cocktail problem → separate voices talking over each other

⇒ We are using Octave.

- Model Representation:



⇒ Supervised (right answer / labels provided)

↳ Regression (continuous output)

↳ Training Set: $x^{(1)}$ →

Size	Price
⋮	⋮

Set of training examples (x, y)

↓ # rows → m

x: input variable

y: output variable

$(x^{(i)}, y^{(i)})$

↳ ith training example

⇒ Training Set → Learning Algorithm → $\boxed{h}$ (hypothesis)

↳ h maps x to y ⇒ $x \rightarrow \boxed{h} \rightarrow y$

(size) (price)

↳ hypothesis written as $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 \ldots$

↳ shorthand $h(x)$

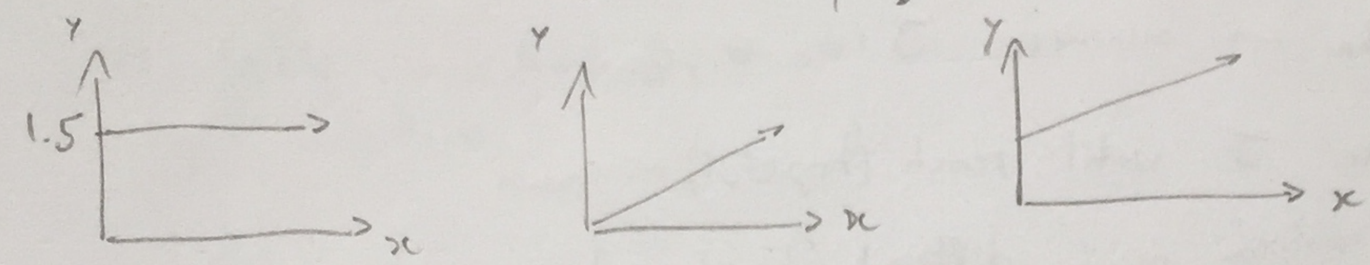
=> Linear Function: $h_\theta(x) = \theta_0 + \theta_1 x$ ← linear regression w/ one variable (x)

↳ univariate (one variable) lin. reg.

- Cost Function:

↳ We have training set $(x, y) \times m$ & hypothesis $h_\theta(x) = \theta_0 + x\theta_1$

↳ figure out θ_0 & θ_1!



← various θ_0, θ_1's

$\theta_0 = 1.5$ $\theta_1 = 0$

$\theta_0 = 0$ $\theta_1 = 0.5$

$\theta_0 = 1$ $\theta_1 = 0.5$

=> want θ_0 & θ_1 that fits data! -> $h_\theta(x)$ close to y for (x, y)'s (training examples)

↳ minimize $h_\theta(x) - y$ -> square it -> minimize $(h_\theta(x) - y)^2$

over θ_0, θ_1

↳ minimize $\frac{1}{2}$ average sum-of-squares error

$$\underset{\text{over } \theta_0, \theta_1}{\text{minimize}} \quad \frac{1}{2m} \sum_{i=1}^{m} \left[h_\theta(x^{(i)}) - y^{(i)} \right]^2 = J(\theta_0, \theta_1)$$

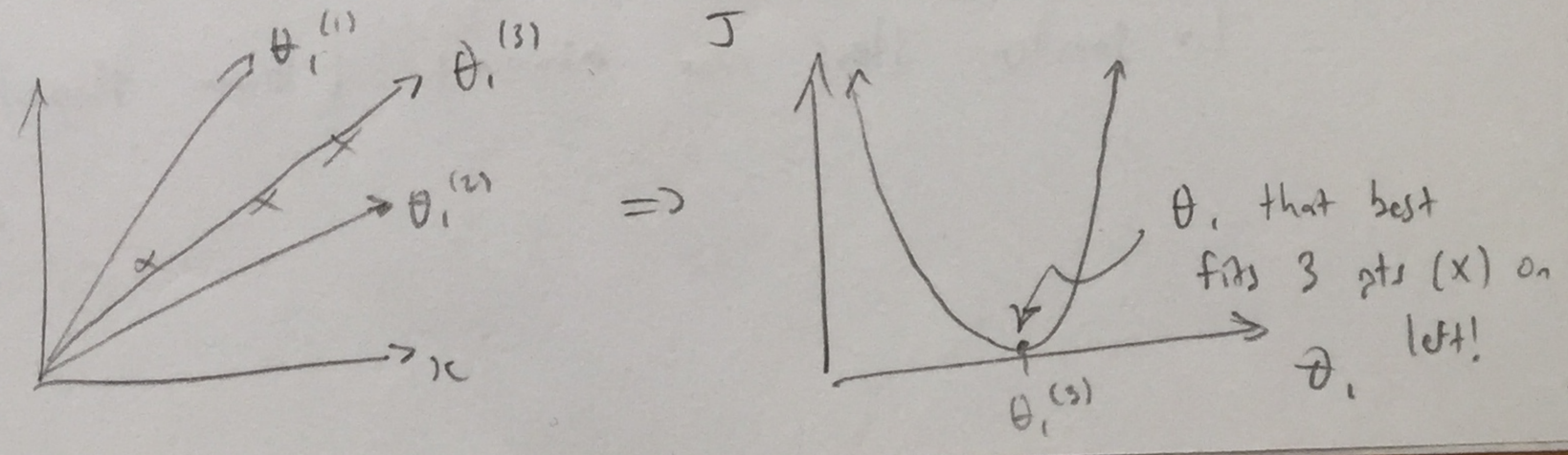
↳ minimize $J(\theta_0, \theta_1)$ "squared error function"

=> minimize $J(\theta_0, \theta_1) = \frac{1}{2} \bar{x}$, where $\bar{x}$ is mean of squares $h_\theta(x^{(i)}) - y^{(i)}$

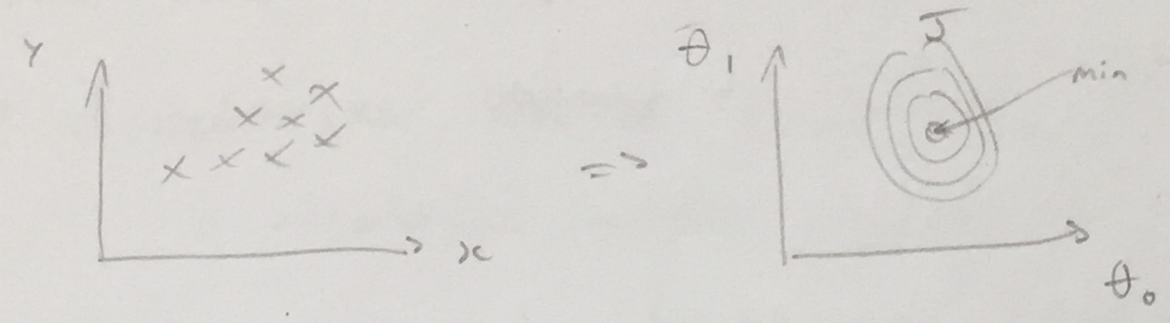
↳ $h_\theta(x^{(i)}) = \theta_0 + \theta_1 x^{(i)}$

↳ $\frac{1}{2}$ makes derivative easier for gradient descent later

=> 2 plots:



- If θ_0 not fixed, $J(\theta_0, \theta_1)$ plotted on contour / 3D plot!



- Gradient Descent:

↳ a minimization algorithm → minimize $J(\theta_0, \theta_1, \theta_2 \dots)$

↳ change θ_i to reduce J until reach (hopeful) minimum

↳ initial choice matters! → can reach different local optimums depending on initial choice

Algorithm:

$$\text{temp}0 := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$$

$$\text{temp}1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$$

α = step size ("learning rate")

$$\theta_0 := \text{temp}0$$

$$\theta_1 := \text{temp}1$$

⇒ Note order of operations!

"Simultaneous Update"

↳ $:=$ means assignment (a := b means val. a = val. b)

ex] $\theta_0 = 1$, $\theta_1 = 2$ $\quad \theta_j := \theta_j + \sqrt{\theta_0 \theta_1}$ → find next θ_0 & θ_1

$$\Rightarrow \text{temp}0 := 1 + \sqrt{1 \cdot 2} = 1 + \sqrt{2}$$

$$\text{temp}1 := 2 + \sqrt{1 \cdot 2} = 2 + \sqrt{2}$$

$$\boxed{\begin{aligned} \theta_0 &= 1 + \sqrt{2} \\ \theta_1 &= 2 + \sqrt{2} \end{aligned}}$$

⇒ α too small → SLOW, but α too large → possibly overshoot!

⇒ $\alpha \cdot \frac{\partial}{\partial \theta_i} J(\theta_i \dots)$ dictates θ motion → both step size & derivative matters

↳ smaller steps near minimums (lower slope)

Regression w/ One Variable:

recall → linear regression model: $h_\theta(x) = \theta_0 + \theta_1 x$ ← hypothesis (line here)

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

→ gradient descent: $\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$ for $j = 0, 1$ → repeat until convergence

⇒ SEEK $\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$

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

⇒ we need partial deriv term: $\frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) = \frac{\partial}{\partial \theta_j} \left[\frac{1}{2m} \sum_{i=1}^{m} \left(h(x^{(i)}) - y^{(i)}\right)^2 \right]$

↳ for $j = 0$: part. deriv $= \frac{1}{m} \sum_{i=1}^{m} \left(h(x^{(i)}) - y^{(i)}\right) \cdot (1 - 0)$

for $j = 1$: part deriv $= \frac{1}{m} \sum_{i=1}^{m} \left(h(x^{(i)}) - y^{(i)}\right) \cdot (x^{(i)} - 0)$

↳ gradient descent becomes:

$\text{temp}0 := \theta_0 - \alpha \left[\frac{1}{m} \sum_{i=1}^{m} \left(h(x^{(i)}) - y^{(i)}\right) \right]$

$\text{temp}1 := \theta_1 - \alpha \left[\frac{1}{m} \sum_{i=1}^{m} \left(h(x^{(i)}) - y^{(i)}\right) x^{(i)} \right]$

$\theta_0 := \text{temp}0$

$\theta_1 := \text{temp}1$ ← simultaneous update

⇒ for linear regression always bowl-shaped (convex function)

↳ only 1 global optimum → great for gradient descent!

⇒ "batch" gradient descent → each step uses all training examples $\left(\sum_{i=1}^{m}\right)$