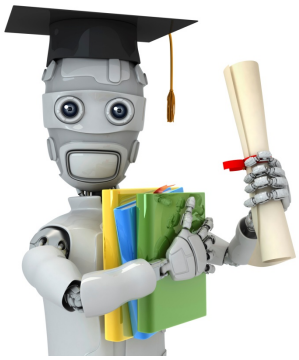
TELECOM Bretagne

Institut Mines-Télécom



Intersemestre Deep Learning
Machine Learning

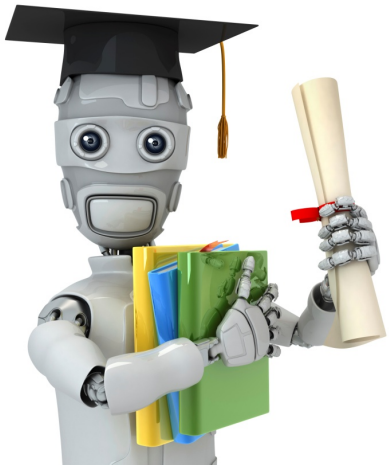
Mai Nguyen

Machine Learning

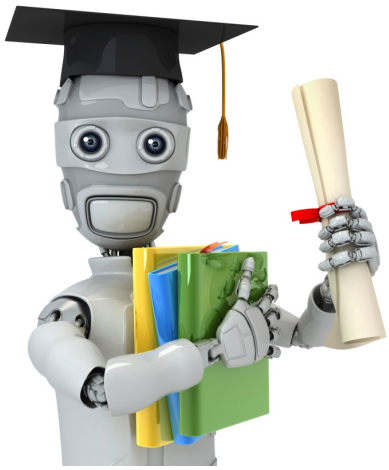
1. **What is machine learning?**
2. **Supervised learning**
3. **Neural networks**
4. **Practical memo for using machine learning**

References:

- ml : Coursera ml class (https://class.coursera.org/ml-005/lecture)
- Deep learning: https://www.coursera.org/specializations/deep-learning



1. What is machine learning?

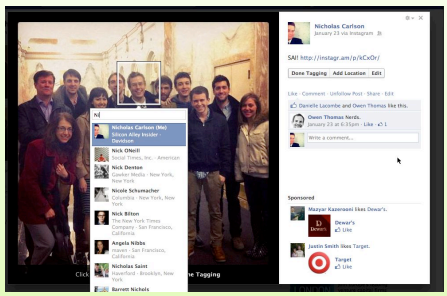


1.1. Definitions

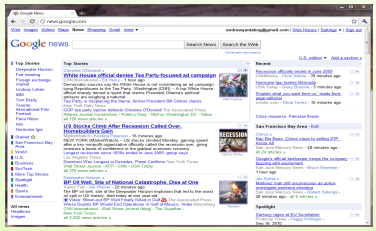
Machine Learning definition

Arthur Samuel (1959)

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



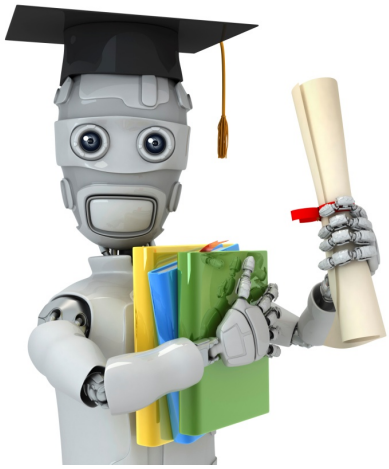
Supervised learning



Unsupervised learning

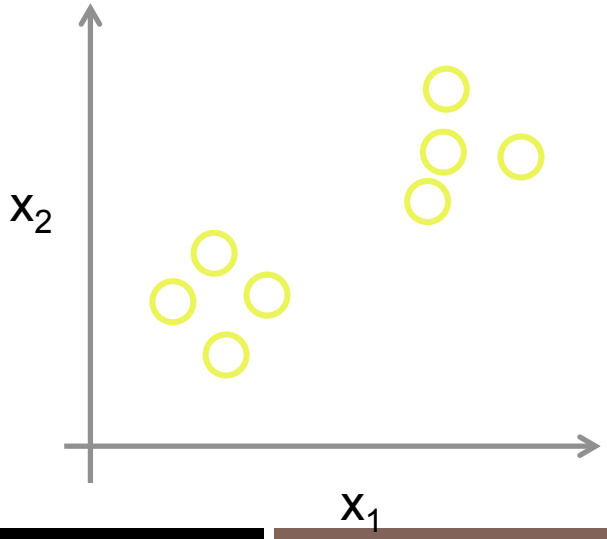
Recommender systems

Reinforcement learning



1.2. Unsupervised Learning

Unsupervised Learning



Science »

Irish Times

Planet Nine: a study in celestial shyness

Irish Times - 6 hours ago

Astronomers were shocked last January when a research team floated the possibility that there might be an as yet undiscovered large planet lurking far out at the distant edge of the solar system.

Planet Nine: Rogue world mystery baffles space boffins Daily Star

Planet Nine could be a 'rogue world' captured from another solar system Daily Mail

See realtime coverage

Highly Cited: Mysterious Planet Nine May Be a Captured 'Rogue' World Space.com

MIT brainiacs wrangle 2D graphene into super-strong 3D art homework

The Register - 5 hours ago

Video Graphene is said to be the wonder material of our age, but it's largely a 2D affair. Now scientists have made 3D structures out of the stuff that will be an engineer's wet dream.

Health »

Daily Mail

BBC doc shows NHS doctors forced to decide between saving cancer patient or pensioner

Daily Mail - 3 hours ago

A woman, known only as Janice, was rushed to the A&E department at St Mary's Hospital in Paddington with her life in the balance after suffering a ruptured blood vessel.

Science »

Irish Times

Planet Nine: a study in celestial shyness

Irish Times - 6 hours ago

Astronomers were shocked last January when a research team floated the possibility that there might be an as yet undiscovered large planet lurking far out at the distant edge of the solar system.

Planet Nine: Rogue world mystery baffles space boffins Daily Star

Planet Nine could be a 'rogue world' captured from another solar system Daily Mail

See realtime

Highly Cited: Mysterious Planet Nine May Be a Captured 'Rogue' World Space.com

Planet Nine 'rogue world' mystery baffles space boffins

ASTRONOMERS believe Planet Nine could be a real rogue world captured by our solar system.

Share Tweet 2

By Harry Kemble / Published 12th January 2017

MYSTERY: Planet Nine is in the far away reaches of the solar system

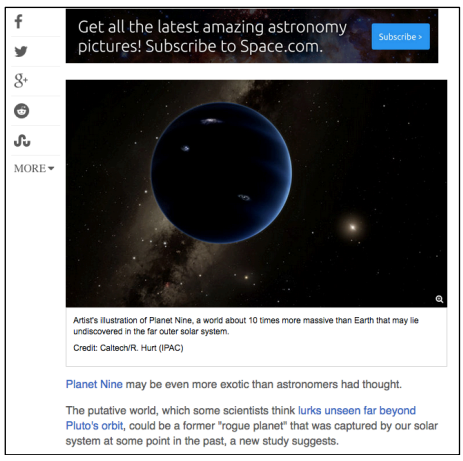
"Rogue, or free-floating, planets may be abundant in the Galaxy"

A ninth planet – 10 times the mass of Earth – is believed to exist beyond Pluto.

It is believed to be responsible for why the solar system lies on a strange tilt.

Professor Paul Mason

And the so-called Planet Nine is now sparking



Get all the latest amazing astronomy pictures! Subscribe to Space.com.

Subscribe ›

MORE

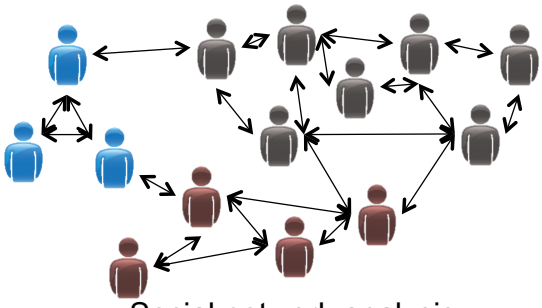
Artist's illustration of Planet Nine, a world about 10 times more massive than Earth that may lie undiscovered in the far outer solar system.

Credit: Caltech/R. Hurt (IPAC)

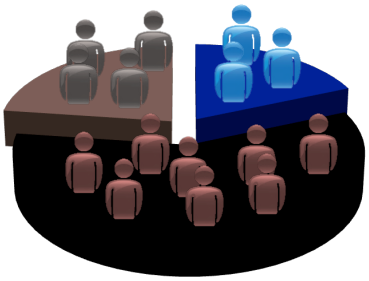
Planet Nine may be even more exotic than astronomers had thought.

The putative world, which some scientists think lurks unseen far beyond Pluto's orbit, could be a former "rogue planet" that was captured by our solar system at some point in the past, a new study suggests.

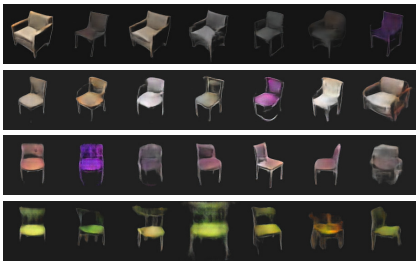
Organize computing clusters



Social network analysis



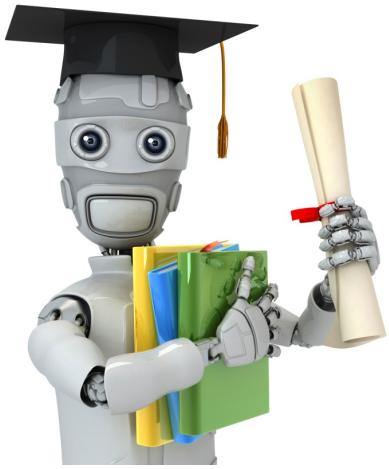
Market segmentation



Generative models

Unsupervised learning algorithms

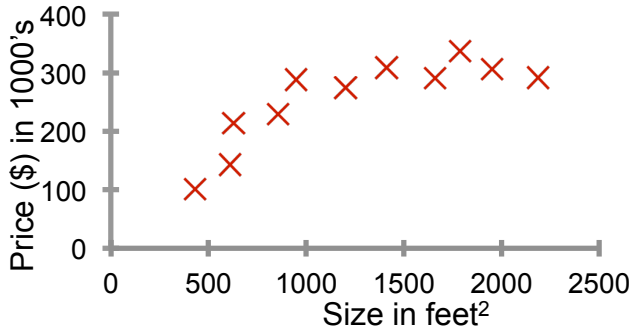
- **Clustering :**
  - Ex: K-means -> UV2 MAJ INF 413
- **Dimensionality reduction:**
  - Principal Component Analysis
  - Auto-encoder
- **Anomaly detection:**
  - Gaussian Models
- **Generative models:**
  - Deep Neural Networks



2. Supervised Learning

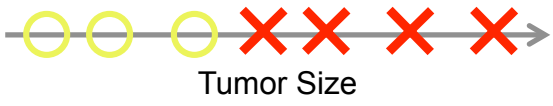
Supervised Learning : "right answers" given

Housing price prediction.



Regression: Predict **continuous** valued output

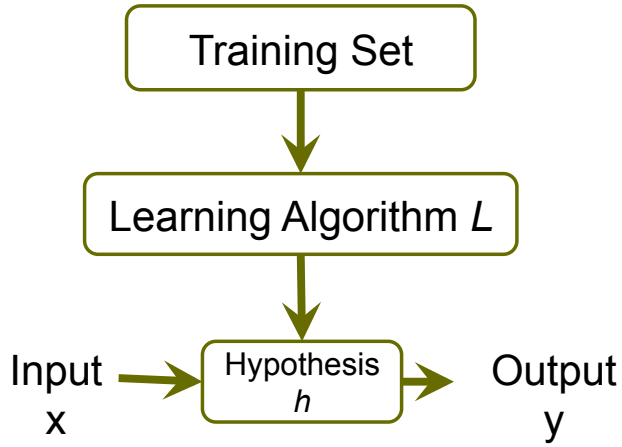
Breast cancer



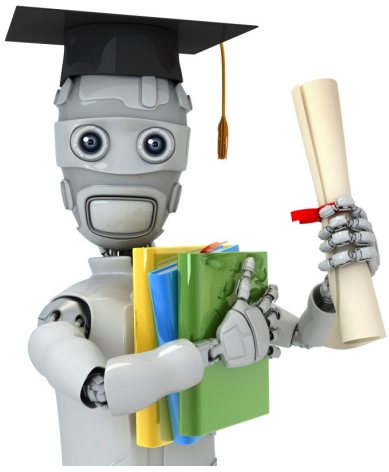
Classification

Discrete valued output

Supervised Learning



- If **output** has
  - Real values =>
    L regression algorithm
  - Discrete values =>
    L classification algorithm
- **Hypothesis/model/ mapping h** can be:
  - Linear function : linear regression (regression)
  - Logistic function : logistic regression (classification)
  - Neural networks

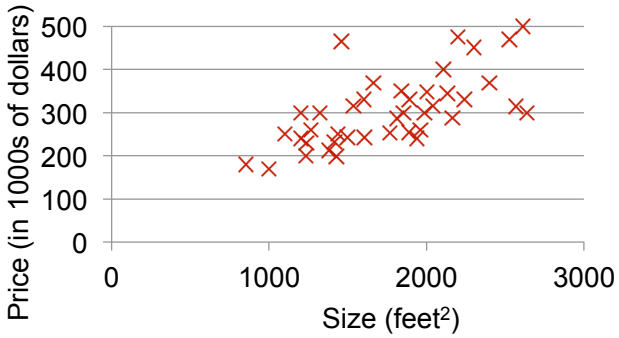


Machine Learning

2.1. Linear Regression

Notations

Training set of housing prices (Portland, OR)



Size in feet² (x)	Price in $ (y)
2104	460
1416	232
1534	315
852	178
…	…

Notation:

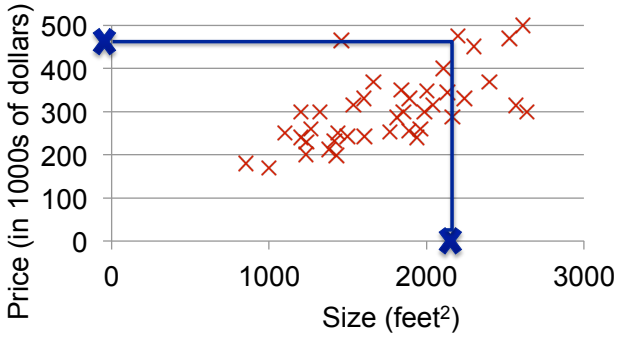
m = Number of training examples

$\mathbf{x}^{(i)}$ = "input" variable / features of the i^{th} example

$\mathbf{y}^{(i)}$ = "output" variable / "target" variable of the i^{th} example

Notations

Training set of housing prices (Portland, OR)



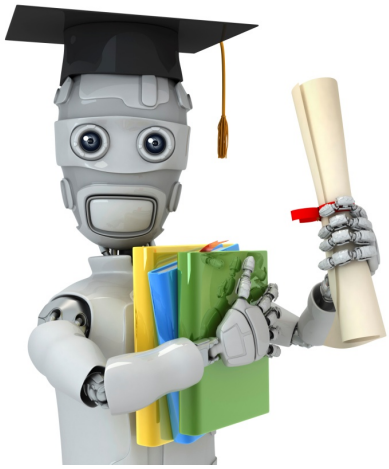
Size in feet² (x)	Price in $ (y)
2104	460
1416	232
1534	315
852	178
…	…

Notation:

m = Number of training examples

$\mathbf{x}^{(i)}$ = "input" variable / features of the i^{th} example

$\mathbf{y}^{(i)}$ = "output" variable / "target" variable of the i^{th} example

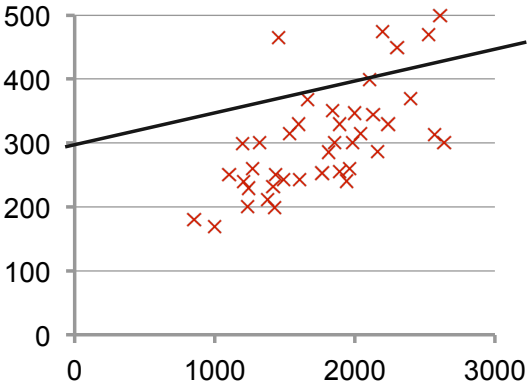


Machine Learning

2.1. Linear regression

2.1.1. Hypothesis representation

How to choose the parameters



$$h_\theta(x) = \theta^T x = \theta_0 + \theta_1 x_1$$

- Choose θ_0, θ_1 so that $h_\theta(x)$ is close to y for our training examples
- **Minimise the error** between the data and our model predictions :

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

- **J = "cost function"**

Multiple features (variables)

Size (feet²)	Number of bedrooms	Number of floors	Age of home (years)	Price ($1000)
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	1	36	178
…	…	…	…	…

Notation:

n = number of features

$x^{(i)}$ = input (features) of i^{th} training example.

$x_j^{(i)}$ = value of feature j in i^{th} training example.

Hypothesis function

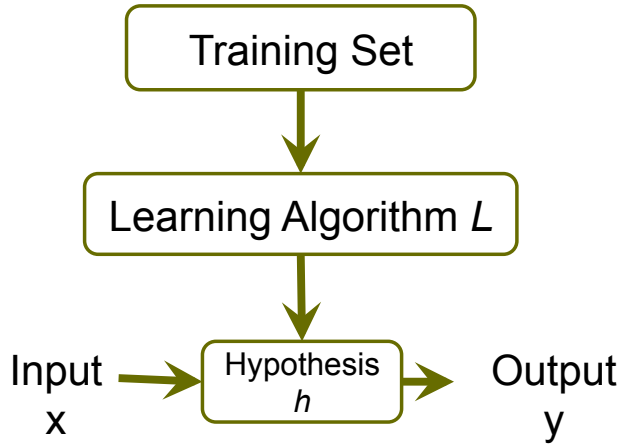
Hypothesis:

1 variable: $h_\theta(x) = \theta_0 + \theta_1 x$

multivariate: $h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$

$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix} \quad h_\theta(x) = \theta^T x$$

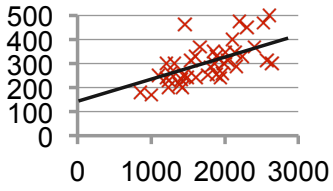
How do we represent h (hypothesis)?

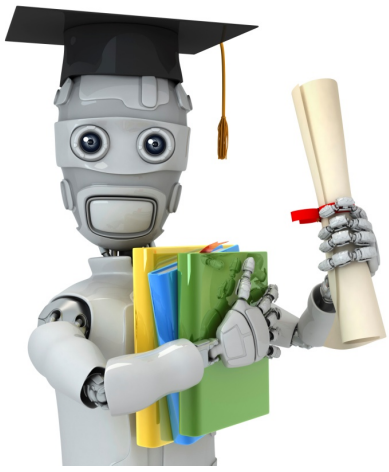


- **Linear hypothesis**

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

- **Linear regression** with one variable
  - θ_i's: Parameters
  - How to choose θ_i's ?





Machine Learning

2.1. Linear regression

2.1.2. Cost function

Hypothesis: $h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$

Parameters: θ_0, θ_1

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

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

Gradient descent for multiple variables

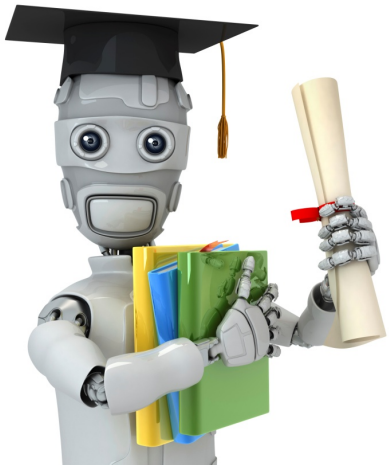
Hypothesis: $h_\theta(x) = \theta^T x = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$

Parameters: $\theta_0, \theta_1, \ldots, \theta_n$

Cost function:

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

Goal: $\underset{\theta_0, \theta_1}{\text{minimize}}\ J$

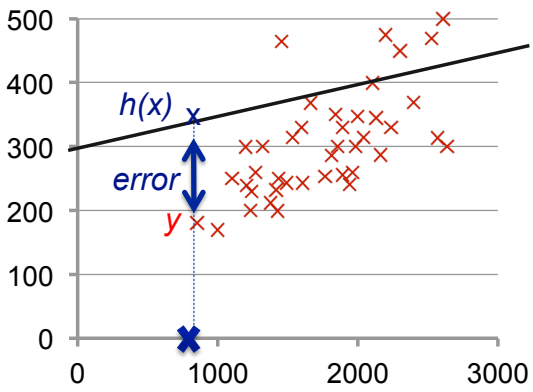


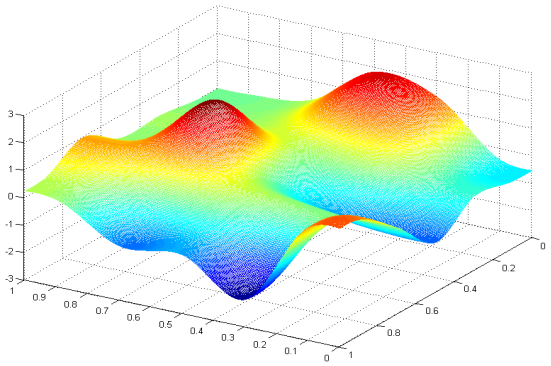
Machine Learning

2.1. Linear regression with one variable

2.1.3. Gradient Descent

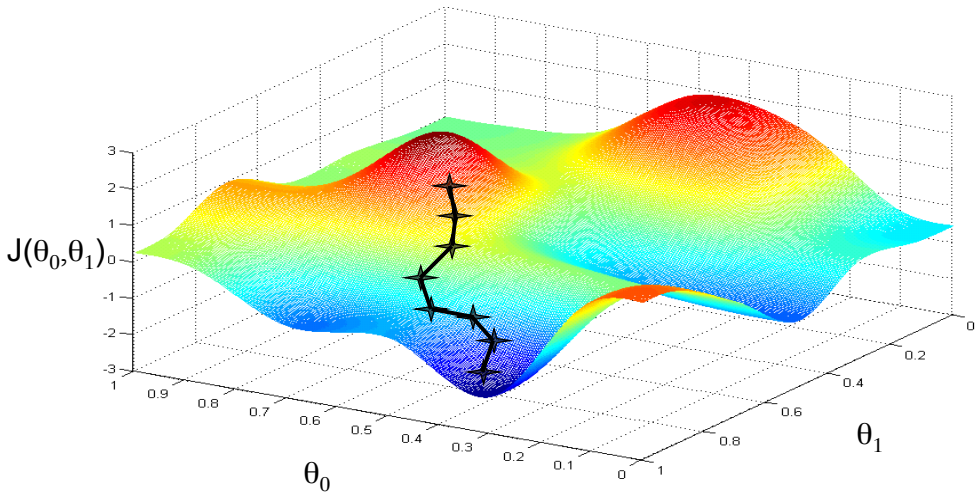
Cost function





$$h_\theta(x) = \theta^T x = \theta_0 + \theta_1 x_1$$

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



Gradient descent: intuition

- **Have some function** $J(\theta_0, \theta_1)$
- **Want** $\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$
- **Outline:**
  - Start with some θ_0, θ_1
  - Keep changing θ_0, θ_1 to reduce $J(\theta_0, \theta_1)$ until we hopefully end up at a minimum

Gradient descent for linear regression

Hypothesis: $h_\theta(x) = \theta^T x = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$

Parameters: $\theta_0, \theta_1, \ldots, \theta_n$

Cost function:

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

Gradient descent:

Repeat $\{$

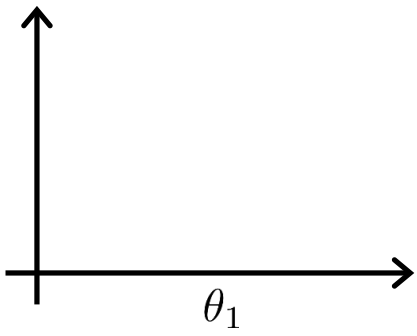
$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \ldots, \theta_n)$$

$\}$ (simultaneously update for every $j = 0, \ldots, n$)

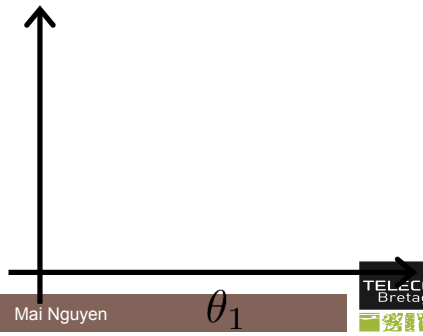
Gradient descent algorithm

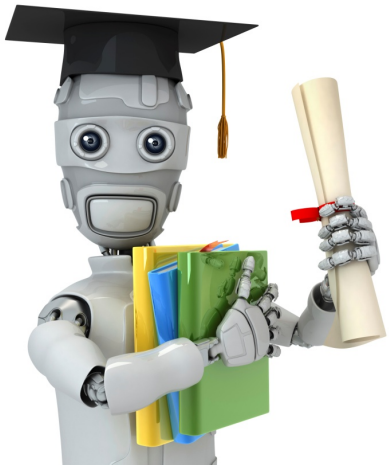
$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

If α is too small, gradient descent can be slow.

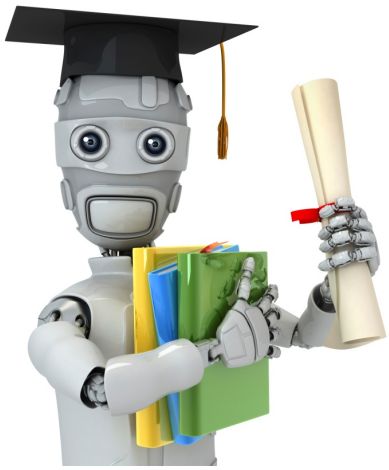


If α is too large, gradient descent can overshoot the minimum. It may fail to converge, or even diverge.



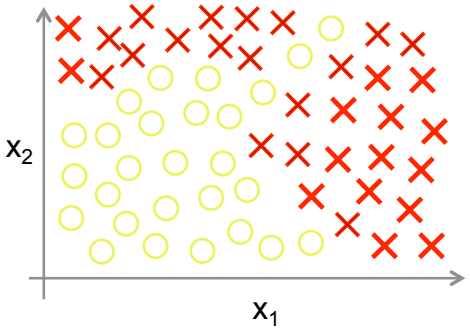


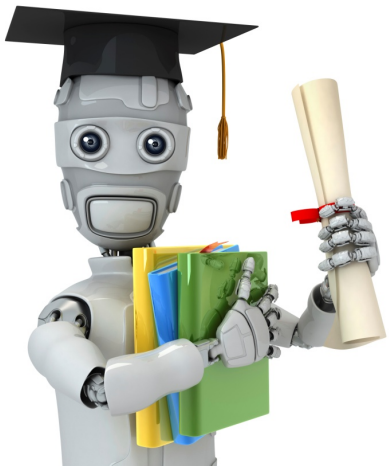
2.2. Neural Networks



2.2.1. Non-linear hypotheses

Non-linear Classification





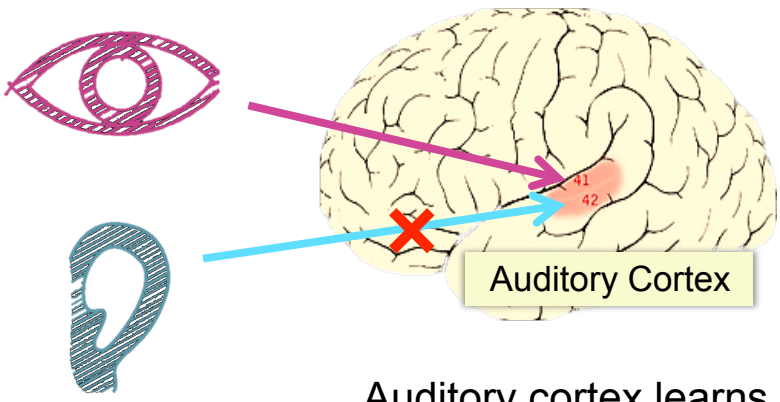
Machine Learning

2.2.2. Neurons and the brain

Neural Networks

Origins: Algorithms that try to mimic the brain.
Was very widely used in 80s and early 90s; popularity diminished in late 90s.
Recent resurgence: State-of-the-art technique for many applications

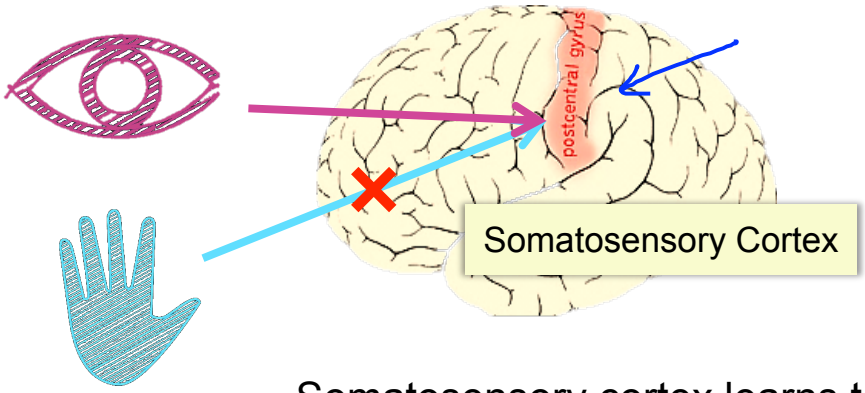
The "one learning algorithm" hypothesis



Auditory cortex learns to see

[Roe et al., 1992]

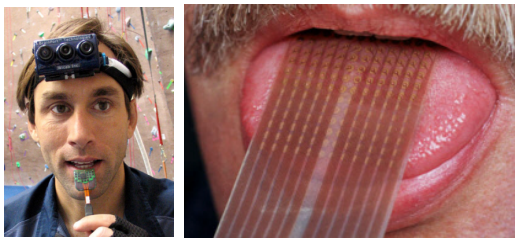
The "one learning algorithm" hypothesis



Somatosensory cortex learns to see

[Metin & Frost, 1989]

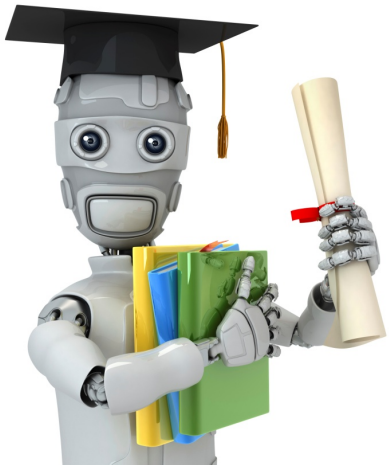
Sensor representations in the brain



Seeing with your tongue

Human echolocation (sonar)

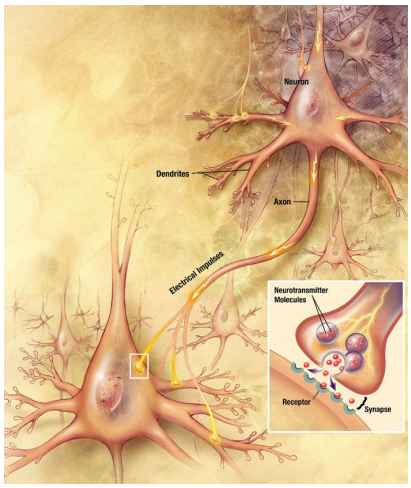
[BrainPort; Welsh & Blasch, 1997; Nagel et al., 2005; Constantine-Paton & Law, 2009]



Machine Learning

2.2.3. Model Representation

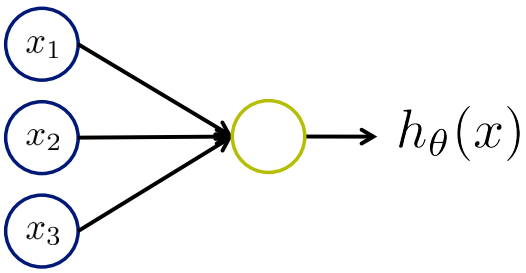
Neurons in the brain



[Credit: US National Institutes of Health, National Institute on Aging]

Neuron model: Logistic unit

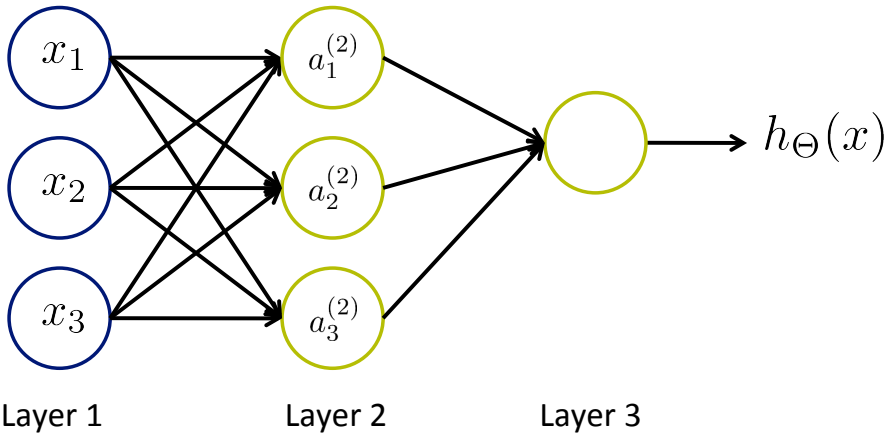
Graphical model -> Pyrat, UV2 MAJ INF 435



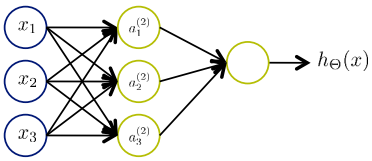
$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix}$$

Activation function g: linear, sigmoid, tanh

Neural Network



Neural Network

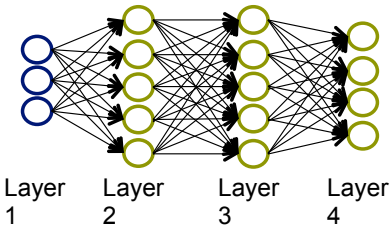


$a_i^{(j)} =$ "activation" of unit i in layer j

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

If network has s_j units in layer j, s_{j+1} units in layer $j+1$, then $\Theta^{(j)}$ will be of dimension .

Multi-output neural network



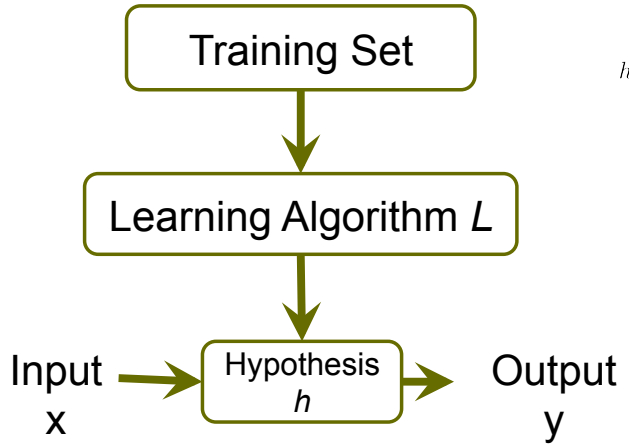
$L =$ total no. of layers in network

$s_l =$ no. of units (not counting bias unit) in layer l

$\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\}$

$y \in \mathbb{R}^K$

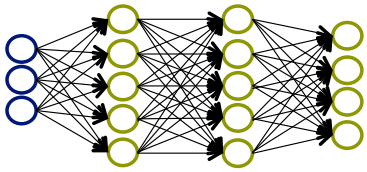
Neural network

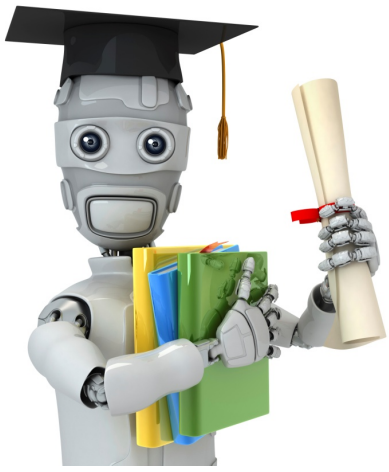


- **Hypothesis**

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

- **Linear regression** with one variable
  - $\Theta^{(j)}$: Parameters
  - How to choose $\Theta^{(j)}$ s ?





Machine Learning

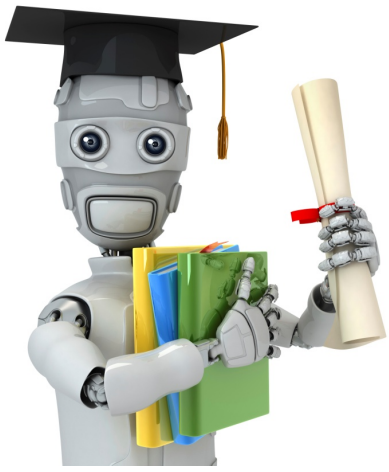
2.2.4. Cost function

Cost function

Regression:

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

$$h_\Theta(x) \in \mathbb{R}^K \quad (h_\Theta(x))_i = i^{th} \text{ output}$$



Machine Learning

2.2.5. Learning algorithm

Backpropagation

Gradient computation

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

$$\min_{\Theta} J(\Theta)$$

Need code to compute:

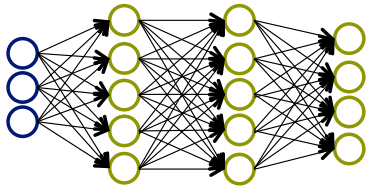
- $J(\Theta)$
- $\frac{\partial}{\partial \Theta_{ij}^{(l)}} J(\Theta)$

Gradient computation

Given one training example (x, y):

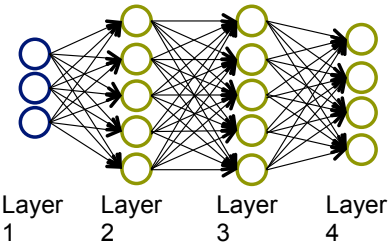
Forward propagation:

$$a^{(1)} = x$$
$$z^{(2)} = \Theta^{(1)} a^{(1)}$$
$$a^{(2)} = g(z^{(2)}) \quad (\text{add } a_0^{(2)})$$
$$z^{(3)} = \Theta^{(2)} a^{(2)}$$
$$a^{(3)} = g(z^{(3)}) \quad (\text{add } a_0^{(3)})$$
$$z^{(4)} = \Theta^{(3)} a^{(3)}$$
$$a^{(4)} = h_\Theta(x) = g(z^{(4)})$$



Gradient computation: Backpropagation algorithm

Intuition: $\delta_j^{(l)} =$ "error" of node j in layer l.



For each output unit (layer L = 4)

$$\delta_j^{(4)} = a_j^{(4)} - y_j$$

$$\delta^{(3)} = (\Theta^{(3)})^T \delta^{(4)} .* g'(z^{(3)})$$

$$\delta^{(2)} = (\Theta^{(2)})^T \delta^{(3)} .* g'(z^{(2)})$$

Backpropagation algorithm

Training set $\{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\}$

Set $\triangle_{ij}^{(l)} = 0$ (for all l, i, j).

For $i = 1$ to m

Set $a^{(1)} = x^{(i)}$

Perform forward propagation to compute $a^{(l)}$ for $l = 2, 3, \ldots, L$

Using $y^{(i)}$, compute $\delta^{(L)} = a^{(L)} - y^{(i)}$

Compute $\delta^{(L-1)}, \delta^{(L-2)}, \ldots, \delta^{(2)}$

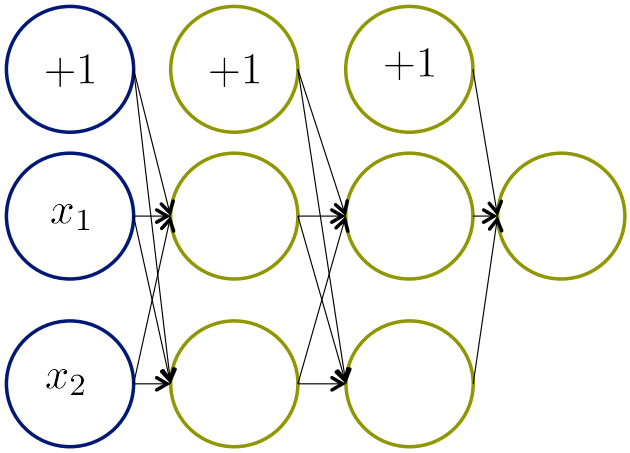
$\triangle_{ij}^{(l)} := \triangle_{ij}^{(l)} + a_j^{(l)} \delta_i^{(l+1)}$

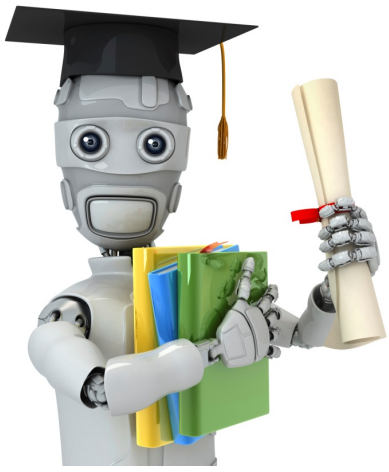
$D_{ij}^{(l)} := \frac{1}{m} \triangle_{ij}^{(l)}$

$\frac{\partial}{\partial \Theta_{ij}^{(l)}} J(\Theta) = D_{ij}^{(l)}$

Backpropagation intuition

Forward Propagation





Machine Learning

2.2. (Logistic regression) Classification

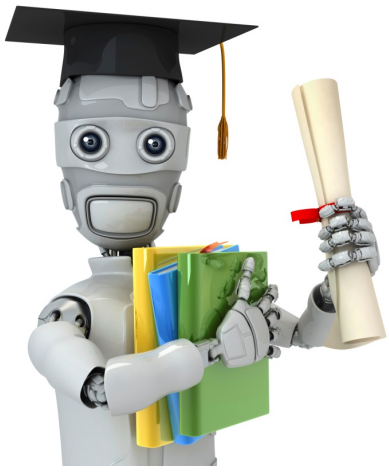
Debugging a learning algorithm:

Suppose you have implemented linear regression to predict housing prices.

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

However, when you test your hypothesis on a new set of houses, you find that it makes unacceptably large errors in its predictions. What should you try next?

- Get more training examples
- Try smaller sets of features
- Try getting additional features



Machine Learning

4.3. Evaluating a hypothesis

Evaluating your hypothesis

Dataset:

Size	Price
2104	400
1600	330
2400	369
1416	232
3000	540
1985	300
1534	315
1427	199
1380	212
1494	243

$(x^{(1)}, y^{(1)})$
$(x^{(2)}, y^{(2)})$
$\vdots$
$(x^{(m)}, y^{(m)})$

$(x_{test}^{(1)}, y_{test}^{(1)})$
$(x_{test}^{(2)}, y_{test}^{(2)})$
$\vdots$
$(x_{test}^{(m_{test})}, y_{test}^{(m_{test})})$

Training/testing procedure

- Learn parameter θ from training data (minimizing training error $J(\theta)$)

- Compute test set error:

Model selection

1. $h_\theta(x) = \theta_0 + \theta_1 x$
2. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2$
3. $h_\theta(x) = \theta_0 + \theta_1 x + \cdots + \theta_3 x^3$

$\vdots$

Choose $\theta_0 + \ldots \theta_5 x^5$

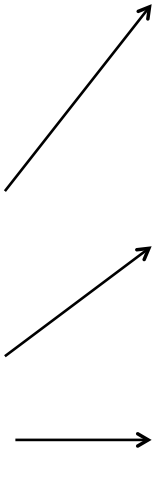
How well does the model generalize? Report test set error $J_{test}(\theta^{(5)})$.

Problem: $J_{test}(\theta^{(5)})$ is likely to be an optimistic estimate of generalization error. I.e. our extra parameter (d = degree of polynomial) is fit to test set.

Evaluating your hypothesis

Dataset:

Size	Price
2104	400
1600	330
2400	369
1416	232
3000	540
1985	300
1534	315
1427	199
1380	212
1494	243



$(x^{(1)}, y^{(1)})$
$(x^{(2)}, y^{(2)})$
$\vdots$
$(x^{(m)}, y^{(m)})$

$(x_{cv}^{(1)}, y_{cv}^{(1)})$
$(x_{cv}^{(2)}, y_{cv}^{(2)})$
$\vdots$
$(x_{cv}^{(m_{cv})}, y_{cv}^{(m_{cv})})$

$(x_{test}^{(1)}, y_{test}^{(1)})$
$(x_{test}^{(2)}, y_{test}^{(2)})$
$\vdots$
$(x_{test}^{(m_{test})}, y_{test}^{(m_{test})})$

Train/validation/test error

Training error:

$$J_{train}(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

Cross Validation error:

$$J_{cv}(\theta) = \frac{1}{2m_{cv}} \sum_{i=1}^{m_{cv}} (h_\theta(x_{cv}^{(i)}) - y_{cv}^{(i)})^2$$

Test error:

$$J_{test}(\theta) = \frac{1}{2m_{test}} \sum_{i=1}^{m_{test}} (h_\theta(x_{test}^{(i)}) - y_{test}^{(i)})^2$$

Model selection

1. $h_\theta(x) = \theta_0 + \theta_1 x$ → $\min_\theta J(\theta) \rightarrow \theta^{(1)} \rightarrow J_{cv}(\theta^{(1)})$
2. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2$ → $\theta^{(2)} \rightarrow J_{cv}(\theta^{(2)})$
3. $h_\theta(x) = \theta_0 + \theta_1 x + \cdots + \theta_3 x^3$ → $\theta^{(3)}$ … $J_{cv}(\theta^{(4)})$

⋮

10. $h_\theta(x) = \theta_0 + \theta_1 x + \cdots + \theta_{10} x^{10}$ → $\theta^{(10)} \rightarrow J_{cv}(\theta^{(10)})$

$d = 4$

Pick $\theta_0 + \theta_1 x_1 + \cdots + \theta_4 x^4$

Estimate generalization error for test set $J_{test}(\theta^{(4)})$