Lecture 2: Preliminaries of Machine Learning

王志　南京大学

2023-03-04

Content

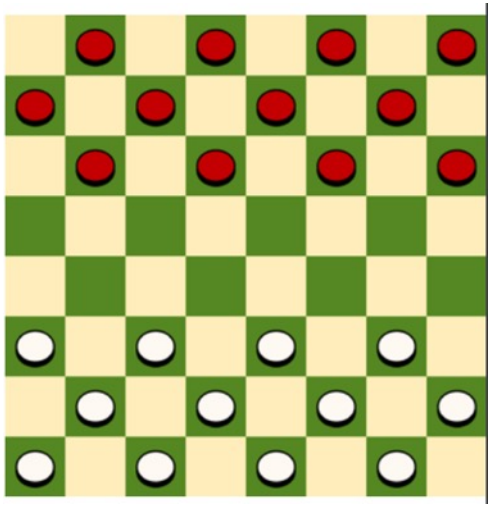
Definition of Machine Learning

Arthur Samuel (1959):

Machine Learning is the field of study that gives the computer the ability to learn without being explicitly programmed.

A. L. Samuel*

Some Studies in Machine Learning
Using the Game of Checkers. II—Recent Progress

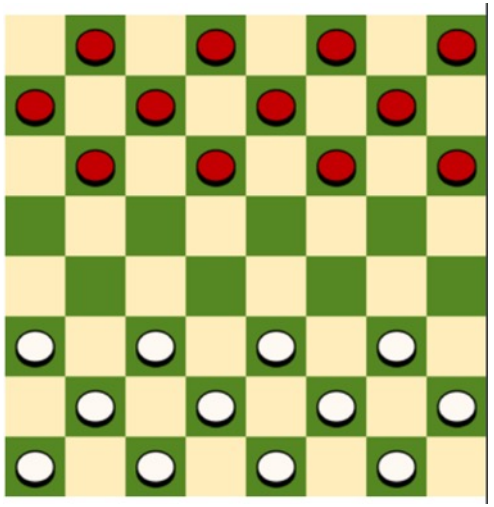


Definition of Machine Learning

Tom Mitchell (1998):

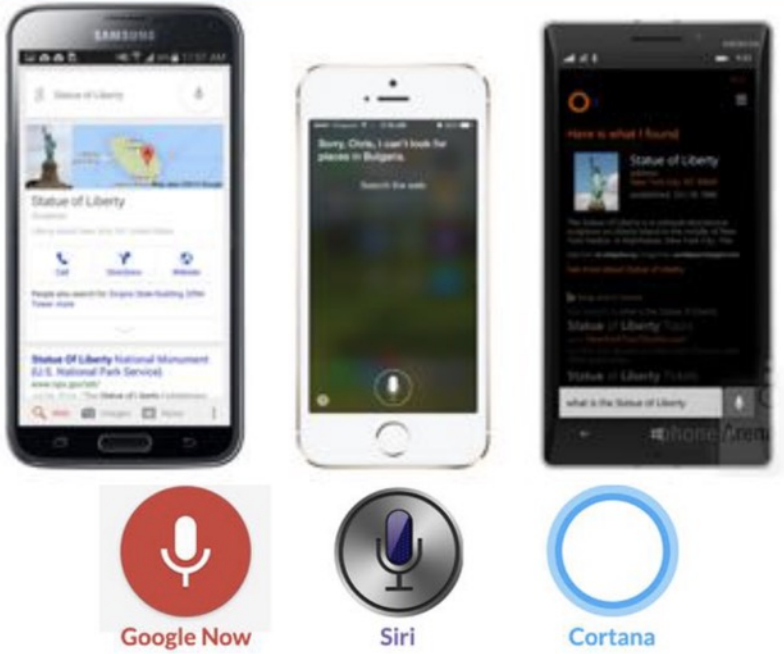
A computer program's performance P at task T improves with experience E.

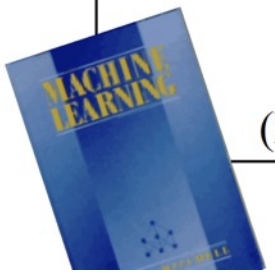
- Experience (data): games played by the program (with itself)
- Performance measure: winning rate



Example: Speech recognition

1. Learning to recognize spoken words

THEN	NOW
"…the SPHINX system (e.g. Lee 1989) learns speaker-specific strategies for recognizing the primitive sounds (phonemes) and words from the observed speech signal…neural network methods…hidden Markov models…" (Mitchell, 1997)	

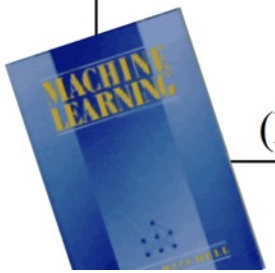


Source: https://www.stonetemple.com/great-knowledge-box-showdown/#VoiceStudyResults

Example: Autonomous driving

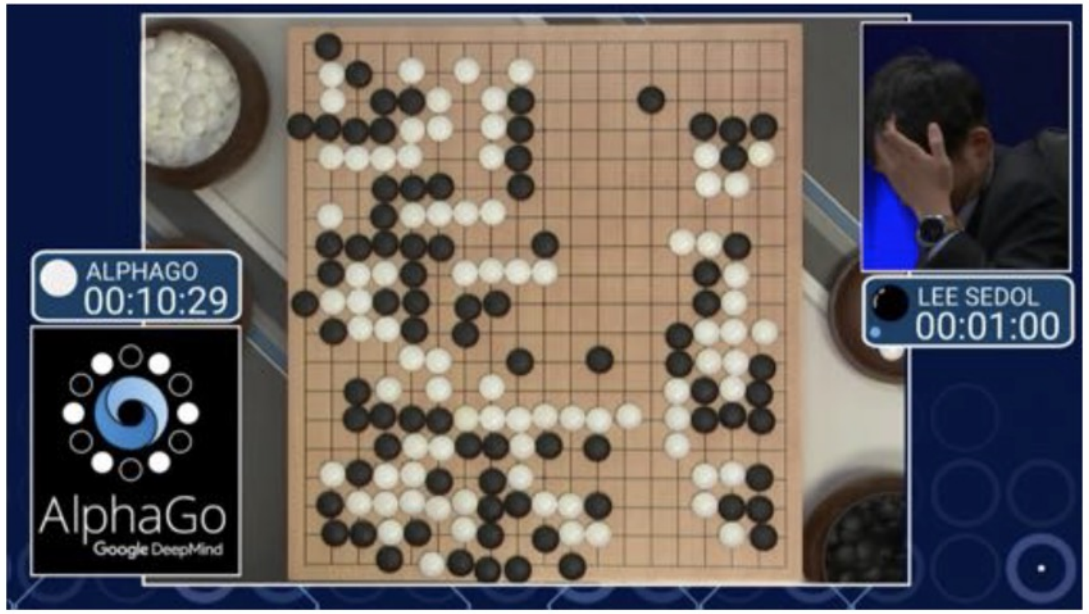
2. Learning to drive an autonomous vehicle

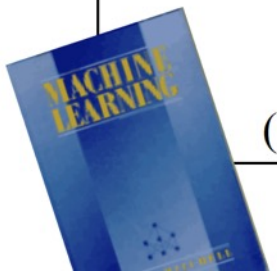
THEN	NOW
"…the ALVINN system (Pomerleau 1989) has used its learned strategies to drive unassisted at 70 miles per hour for 90 miles on public highways among other cars…" (Mitchell, 1997)	 https://www.geek.com/wp-content/uploads/2016/03/uber.jpg



Example: Games / Reasoning

3. Learning to beat the masters at board games

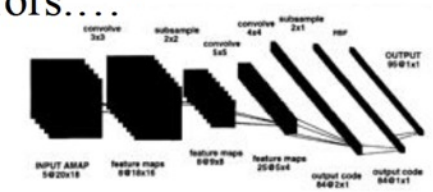
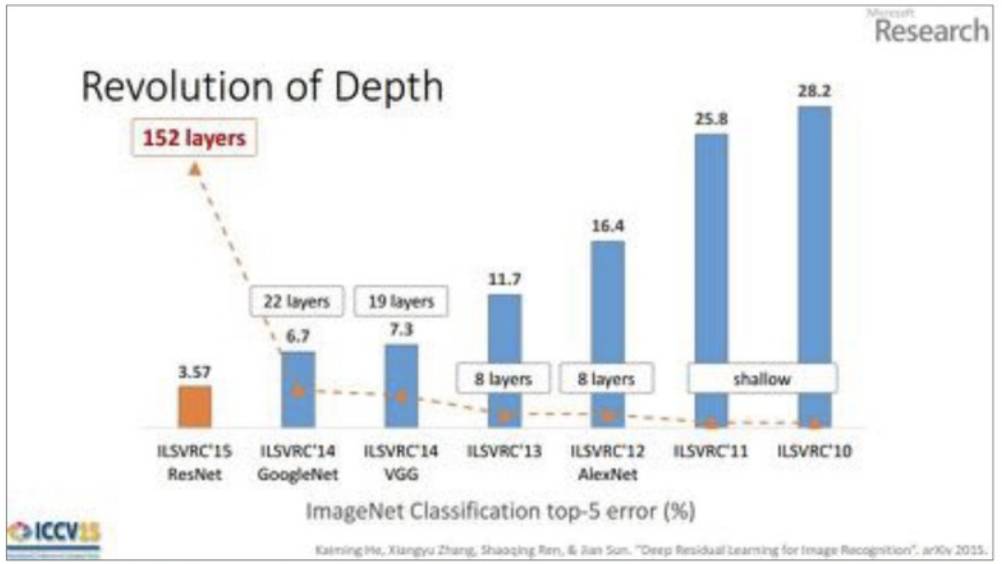
THEN	NOW
"…the world's top computer program for backgammon, TD-GAMMON (Tesauro, 1992, 1995), learned its strategy by playing over one million practice games against itself…"	



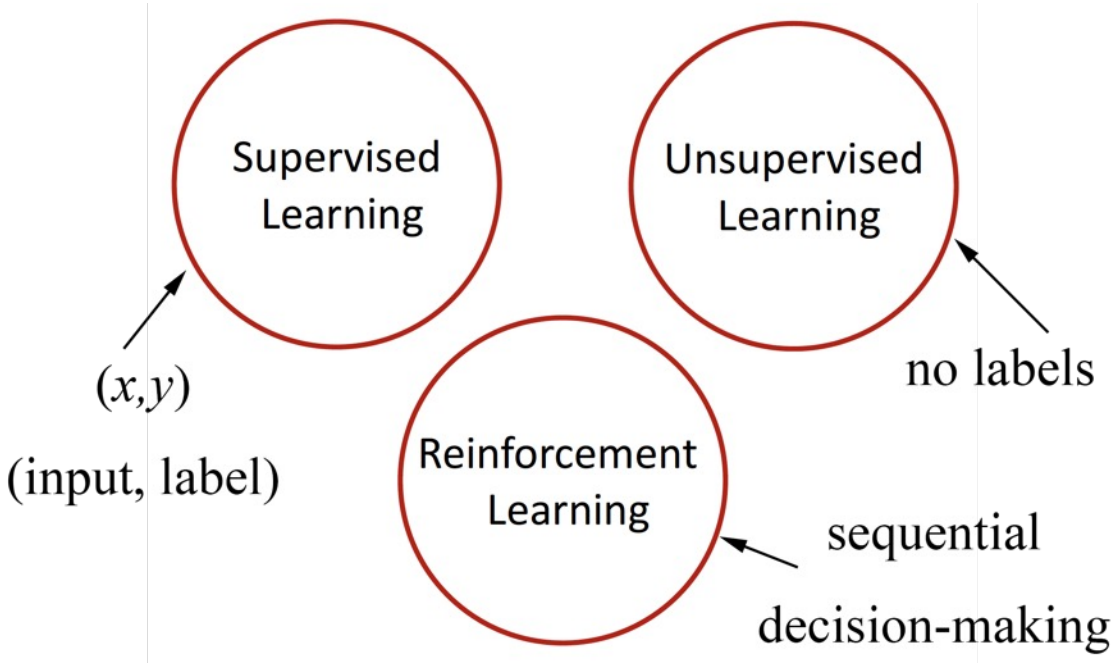
(Mitchell, 1997)

Example: Computer vision

4. Learning to recognize images

THEN	NOW
"…The recognizer is a convolution network that can be spatially replicated. From the network output, a hidden Markov model produces word scores. The entire system is globally trained to minimize word-level errors…."  (LeCun et al., 1995)	 

Taxonomy of machine learning



- a simplistic view based on tasks
- can also be viewed as tools/methods

- Yann LeCun's cake analogy at NIPS 2016:
  - If intelligence is a cake, the bulk of the cake (蛋糕胚) is unsupervised learning, the icing on the cake (奶油) is supervised learning, and the cherry (樱桃) on the cake is reinforcement learning.

Reinforcement Learning

- Supervised Learning
  - (input, label)
- Unsupervised Learning
  - (input)
- **Reinforcement Learning**
  - sequential decision-making

- Computer Vision
  - Input: image pixels
- Natural Language Processing
  - Input: sentences
- **Reinforcement Learning**
  - Input: states

The Dilemma of RL

RL = Artificial General Intelligence (AGI)?

Yet?

The Dilemma of RL

Transformers

Attention is **all you need**

A Vaswani, N Shazeer, N Parmar… - Advances in neural …, 2017 - proceedings.neurips.cc

… to attend to **all** positions in the decoder up to and including that position. **We need** to prevent … **We** implement this inside of scaled dot-product **attention** by masking out (setting to −∞) …

☆ Save 99 Cite Cited by 103261 Related articles All 62 versions »

Vision Transformers

An image is worth 16x16 words: Transformers for image recognition at scale

A Dosovitskiy, L Beyer, A Kolesnikov… - arXiv preprint arXiv …, 2020 - arxiv.org

While the Transformer architecture has become the de-facto standard for natural language processing tasks, its applications to computer vision remain limited. In vision, attention is either applied in conjunction with convolutional networks, or used to replace certain components of convolutional networks while keeping their overall structure in place. We show that this reliance on CNNs is not necessary and a pure transformer applied directly to sequences of image patches can perform very well on image classification tasks. When pre …

☆ Save 99 Cite Cited by 27144 Related articles All 12 versions »

Decision Transformers

Decision transformer: Reinforcement learning via sequence modeling

L Chen, K Lu, A Rajeswaran, K Lee… - Advances in neural …, 2021 - proceedings.neurips.cc

… of the **Transformer** architecture, and associated advances in language modeling such as GPT-x and BERT. In particular, we present **Decision Transformer**, … , **Decision Transformer** simply …

☆ Save 99 Cite Cited by 900 Related articles All 8 versions »

Artificial General Intelligence (AGI)

ChatGPT (Generative Pre-Training)

Next-token prediction

Transformer (Scaling Law)

Enter text:

The dog eats the apples.

464 3290 25365 262 22514 13

Self-supervised learning

The Dilemma of RL

- Computer Vision
  - Input: **image pixels**
- Natural Language Processing
  - Input: **sentences**
- **Reinforcement Learning**
  - Input: **states, (states, actions)**

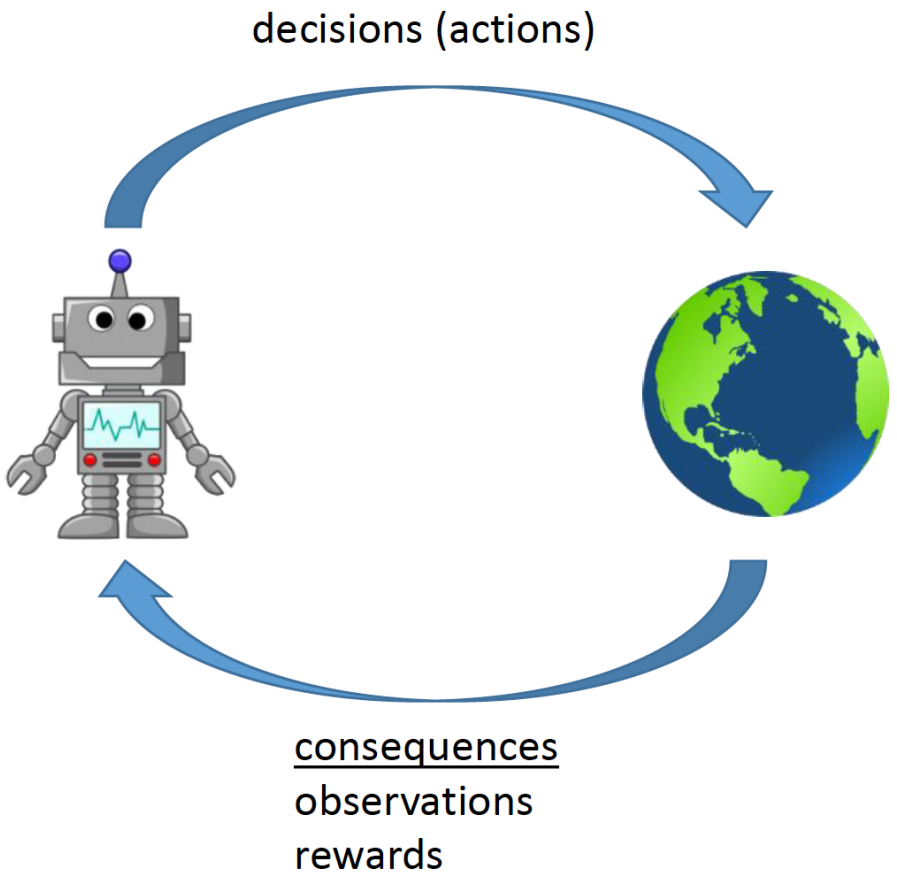
Semantics

not aligned

The Dilemma of RL

Data

From online interactions



RL in ChatGPT

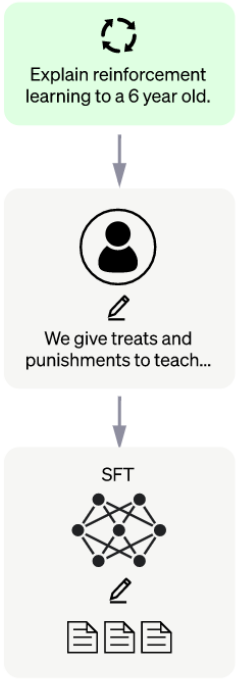
Step 1

Collect demonstration data and train a supervised policy.

A prompt is sampled from our prompt dataset.

A labeler demonstrates the desired output behavior.

This data is used to fine-tune GPT-3.5 with supervised learning.



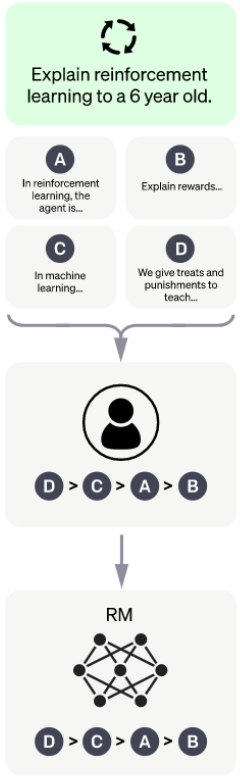
Step 2

Collect comparison data and train a reward model.

A prompt and several model outputs are sampled.

A labeler ranks the outputs from best to worst.

This data is used to train our reward model.



Step 3

Optimize a policy against the reward model using the PPO reinforcement learning algorithm.

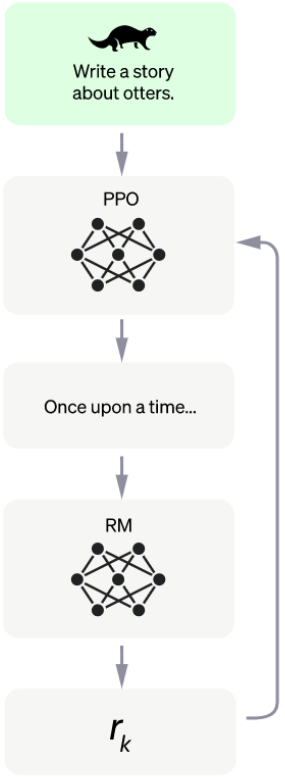
A new prompt is sampled from the dataset.

The PPO model is initialized from the supervised policy.

The policy generates an output.

The reward model calculates a reward for the output.

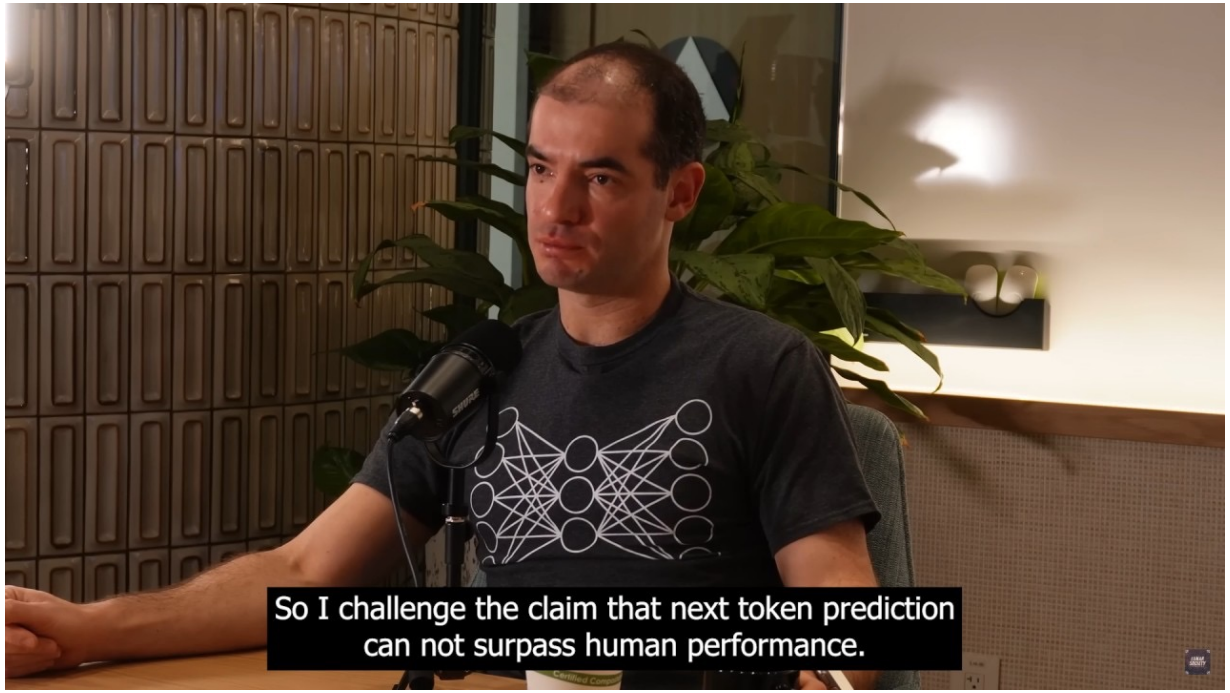
The reward is used to update the policy using PPO.



Confidence in RL

Supervised learning

Maybe imitating the intelligence within existing data?



Confidence in RL

Supervised learning

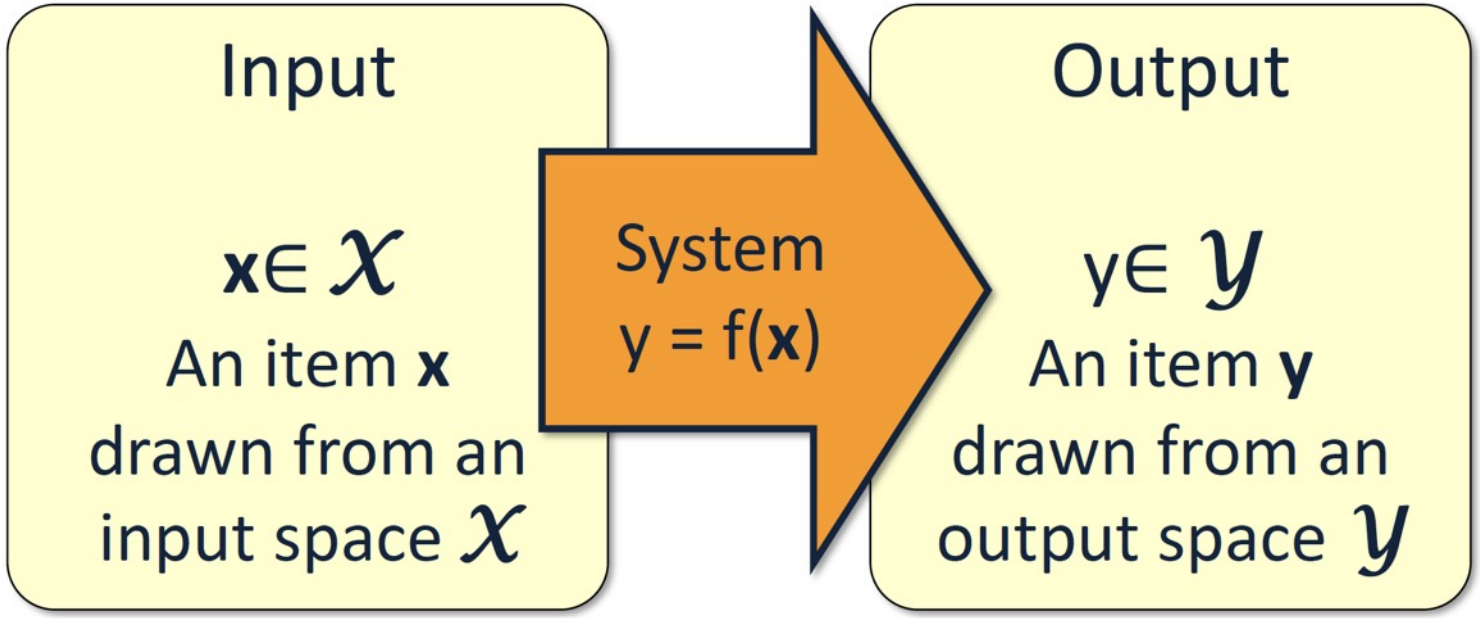
Maybe imitating the intelligence within existing data?

Reinforcement learning

Can surpass the intelligence within existing data definitely

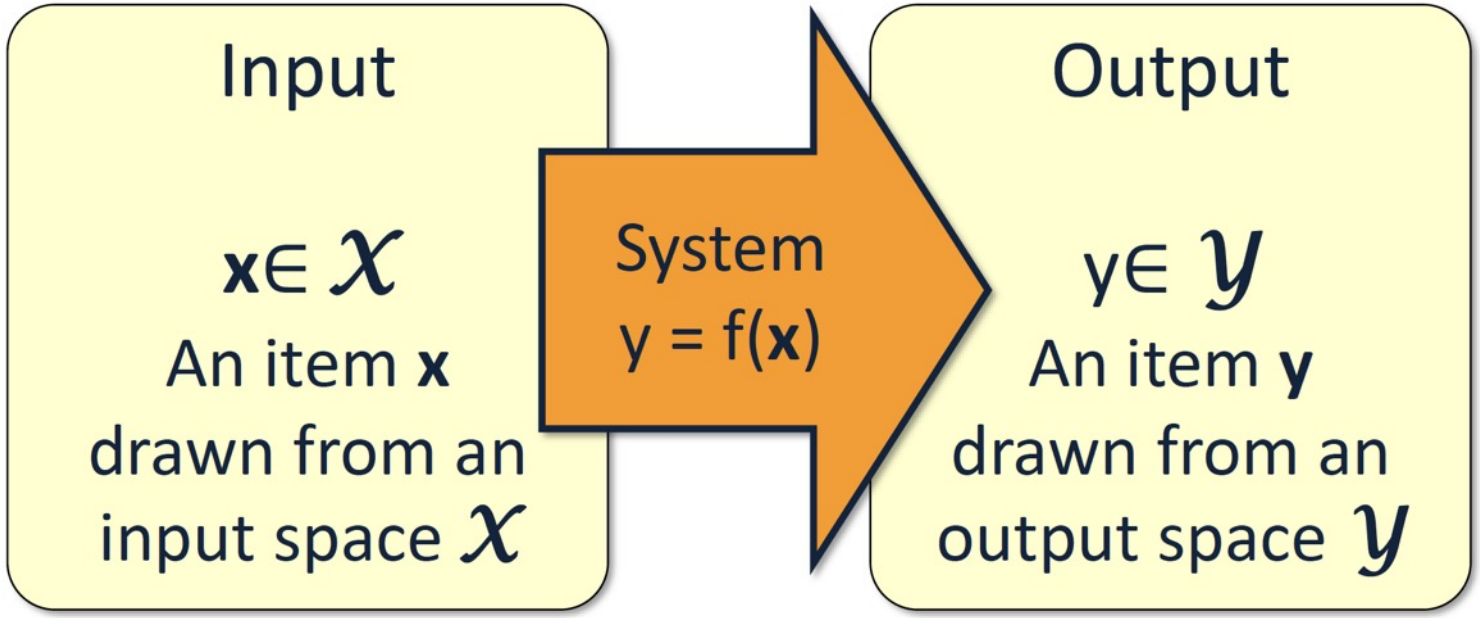
Content

Supervised Learning



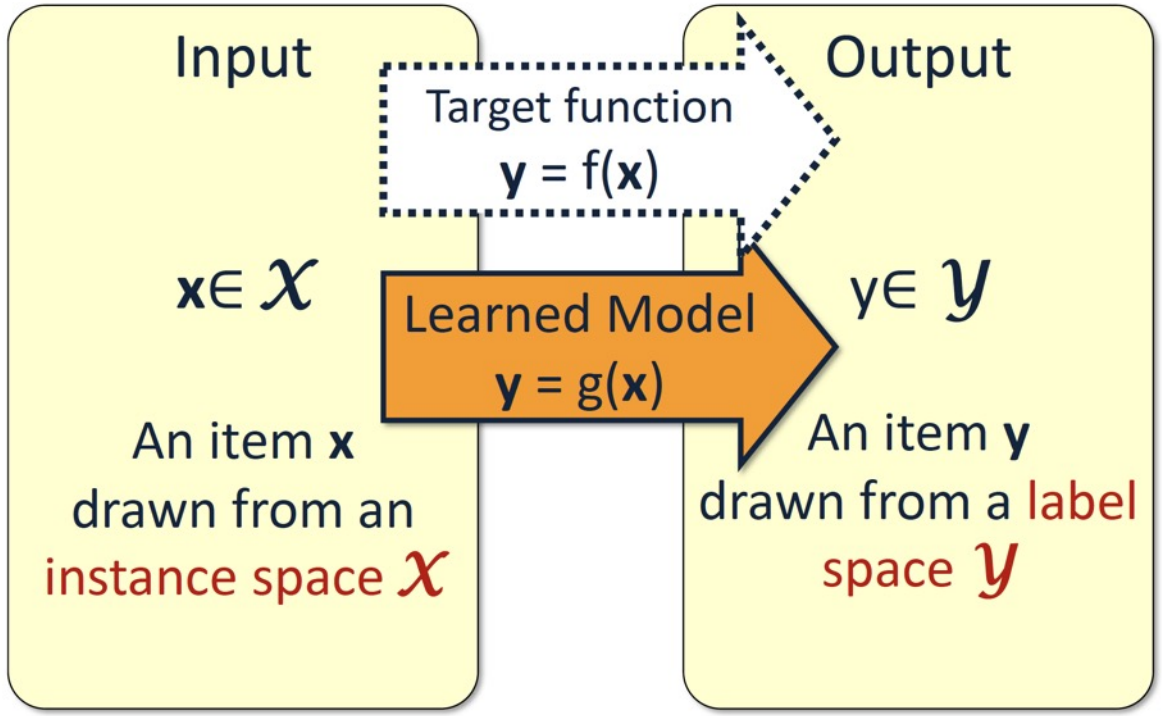
- We consider systems that apply a function $f(\cdot)$ to input items x and return an output $y = f(x)$
- In supervised learning, we deal with systems whose $f(\cdot)$ is learned from samples $\sum_i (x_i, y_i)$

Why use learning?



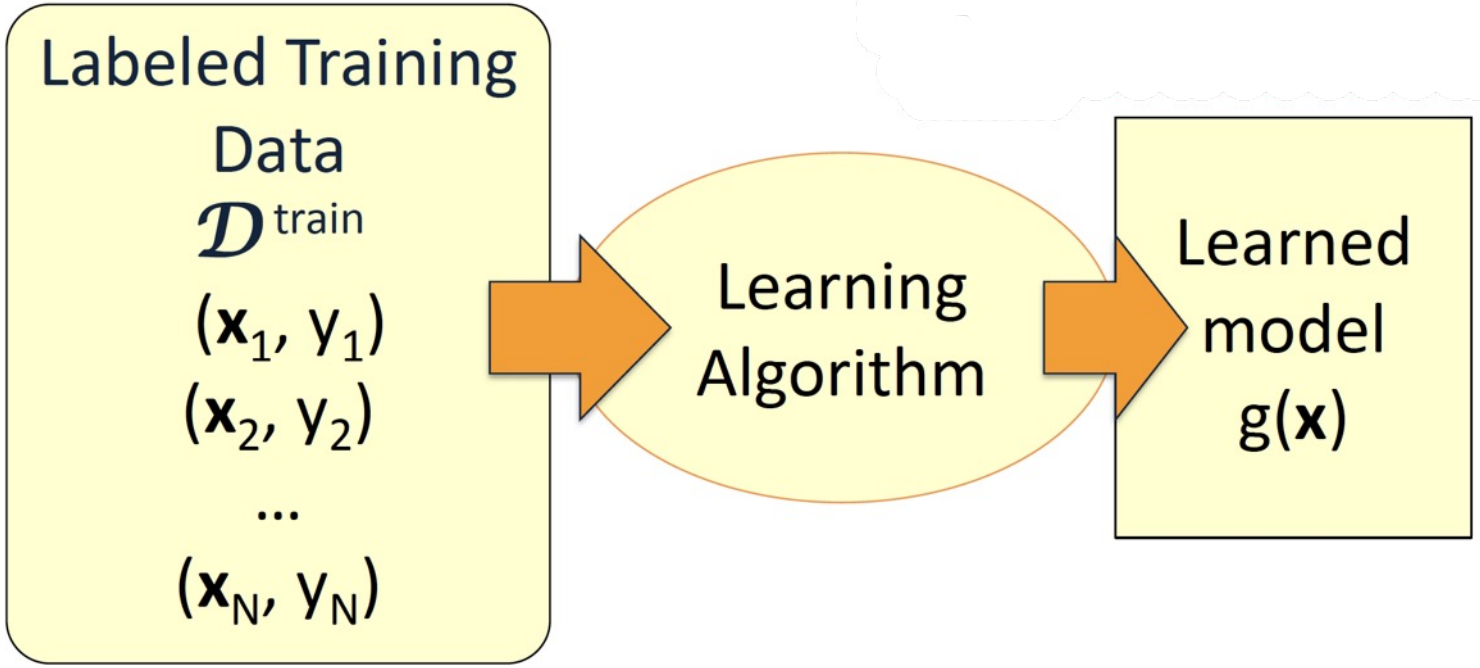
- We typically use machine learning when the function $f(\cdot)$ we want the system to apply is unknown to us, and we cannot "think" about it.
  - Even though the function could actually be simple.

Supervised Learning



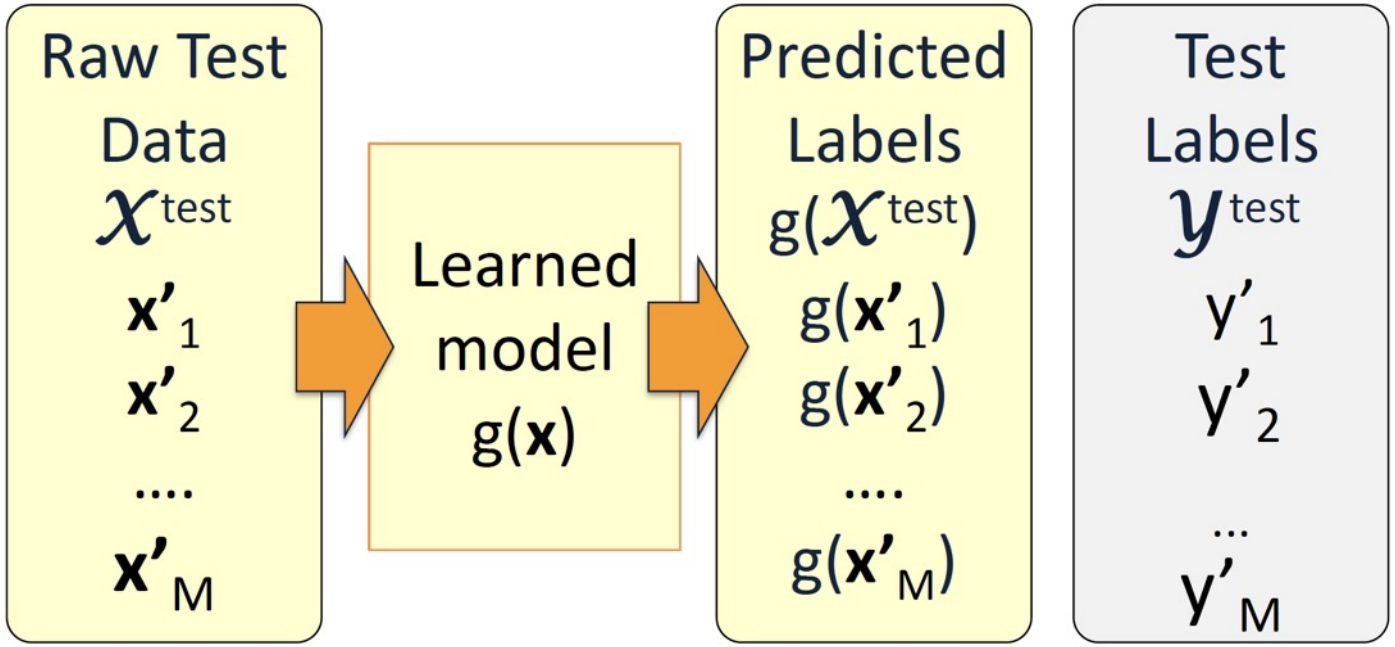
- We need to choose what kind of model we want to learn
  - Linear model, nonlinear model
  - Parametric model, nonparametric model
  - Decision trees, neural networks, Gaussian processes

Training



- Given the learning samples in D^{train}
- The learner returns a model $g(\cdot)$

Testing



- Apply the model to raw test data
- Evaluate by comparing predicted labels against the test labels

Goal of supervised learning

- Given: Samples $\sum_i (x_i, y_i)$ of some unknown function $f(\cdot): x \to y$
- Find: A good approximation of $f(\cdot)$

- x provides some representation of the input
  - $x \in \{0,1\}^d$ or $x \in \mathbb{R}^d$
- The target function (label, 标签)
  - $f(x) \in \{-1, +1\}$ Binary Classification (二分类)
  - $f(x) \in \{1,2, \dots, k\}$ Multi-class Classification (多分类)
  - $f(x) \in \mathbb{R}^k$ Regression (回归)

Linear Regression (线性回归)

- The output variable y as a linear combination of the input variables $(x_1, x_2, \ldots, x_d)$ plus a noise term ϵ

$$y = g_{\boldsymbol{\theta}}(\boldsymbol{x}) + \epsilon = \boldsymbol{\theta}^T \boldsymbol{x} + \epsilon \ = \theta_0 + \theta_1 x_1 + \ldots + \theta_d x_d + \epsilon$$

- $\boldsymbol{\theta} = [\theta_0, \ldots, \theta_d]^T$: model parameters, to be learned
- Use linear regression for at least two purposes:
  - describe relationships in the data by interpreting $\boldsymbol{\theta}$
  - predict future outputs for inputs that we have not yet seen

Example: House price prediction

x_1	x_2	y
Living area (feet^2)	#bedrooms	Price (1000$s)
2104	3	400
1600	3	330
2400	3	369
1416	2	232
3000	4	540
⋮	⋮	⋮

$$g_{\boldsymbol{\theta}}(\boldsymbol{x}) = \hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2$$

Cost function (损失函数 / 代价函数)

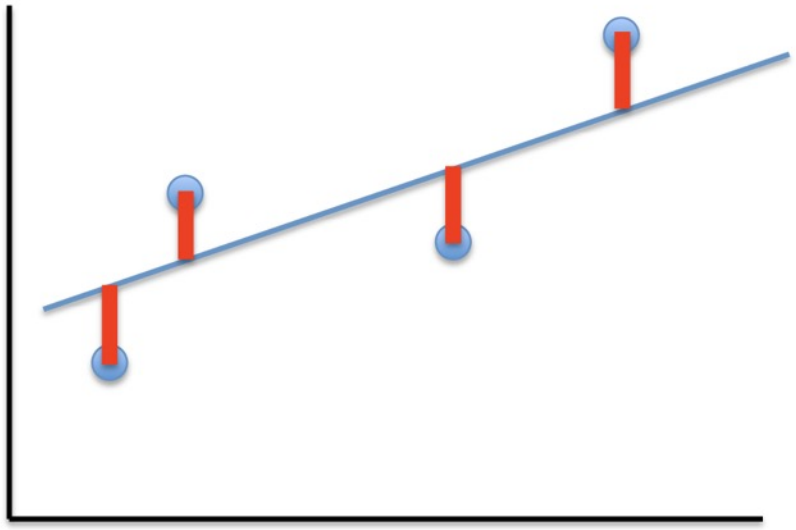
- Question:
  - Given a training set, how do we pick, or learn the parameters $\boldsymbol{\theta}$ **?**
- Goal:
  - Make $g_{\boldsymbol{\theta}}(\boldsymbol{x})$ close to y, at least for the training examples we have
- Cost function, e.g., Mean Squared Error (MSE)
  - It measures how close the predicted values is to the ground truth

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{n}(g_{\boldsymbol{\theta}}(\boldsymbol{x}_i) - y_i)^2$$

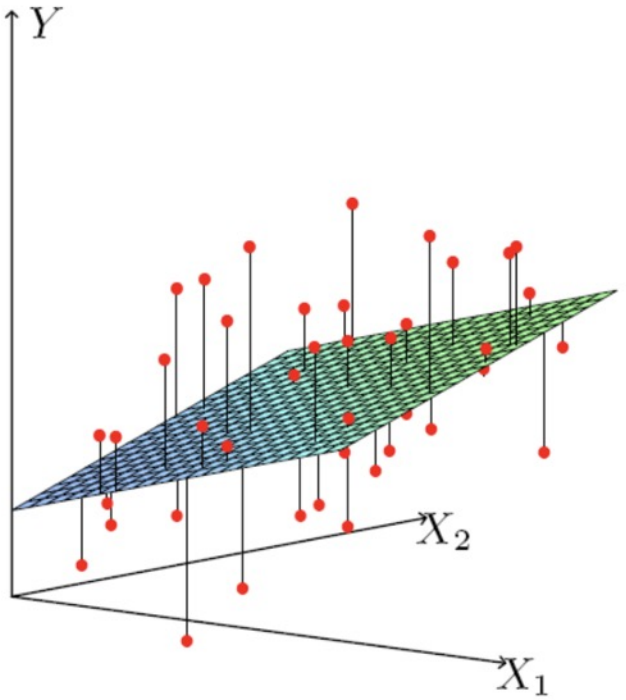
Ordinary least-squares regression model (最小二乘回归模型)

Cost function

$$\boldsymbol{\theta}^* = \arg\min_{\theta} J(\boldsymbol{\theta}) = \frac{1}{2}\sum_{i=1}^{n}(g_{\boldsymbol{\theta}}(\boldsymbol{x}_i) - y_i)^2$$



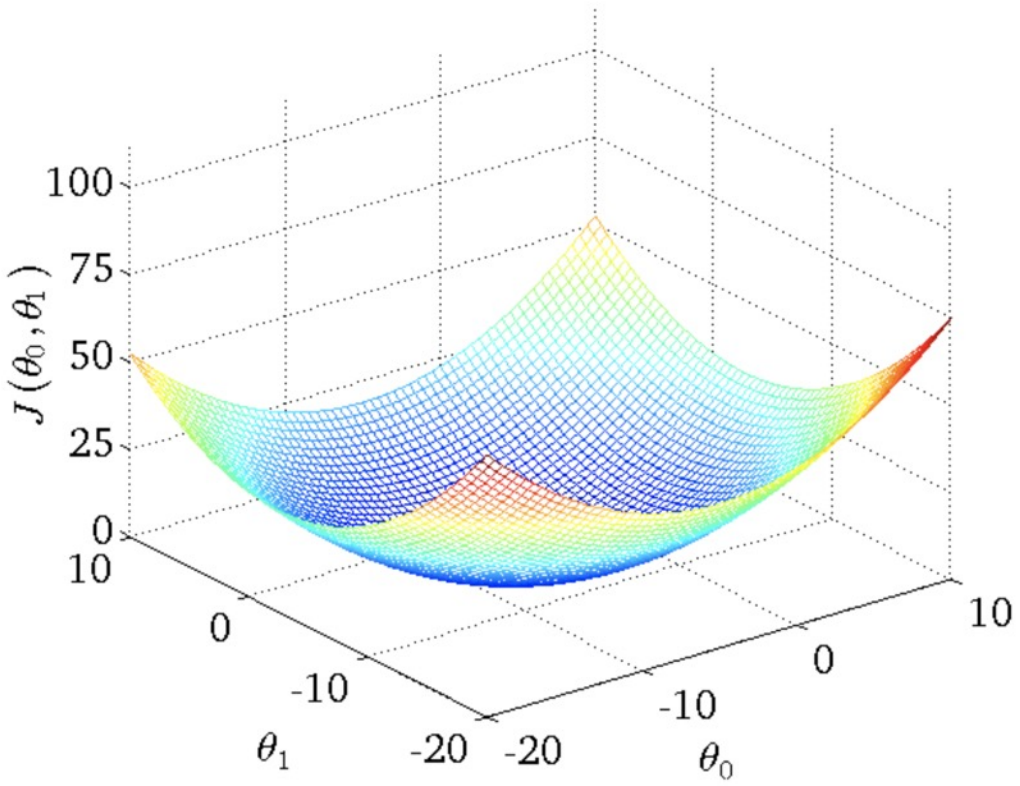
$$\hat{y} = \theta_0 + \theta_1 x$$



$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2$$

Cost function

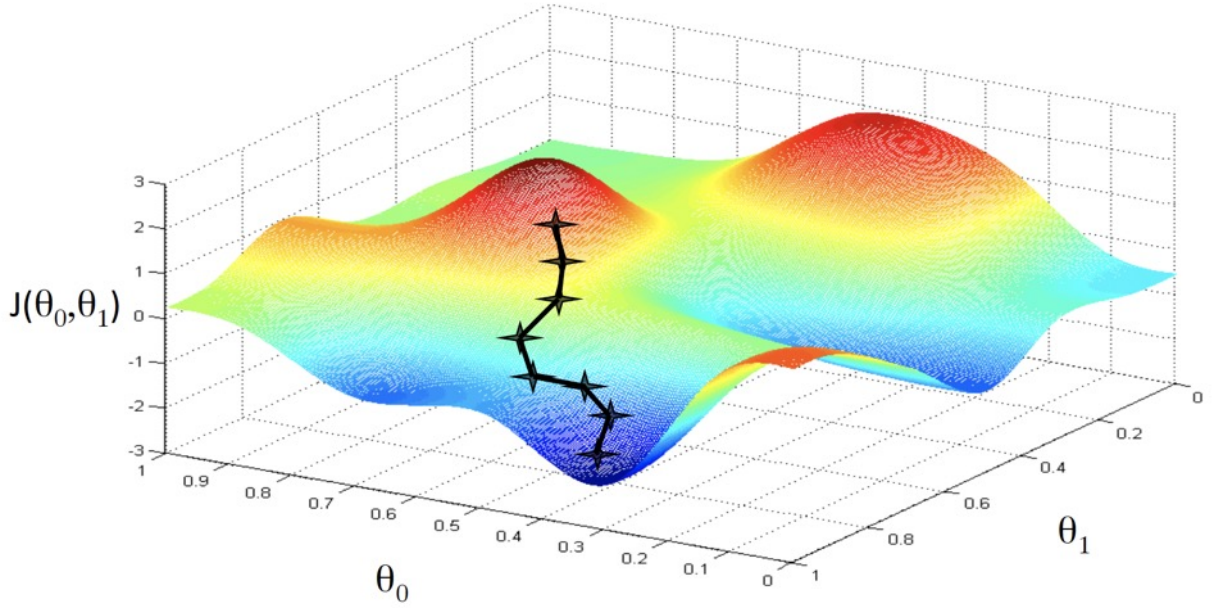
$$\boldsymbol{\theta}^* = \arg\min_{\theta} J(\boldsymbol{\theta}) = \frac{1}{2}\sum_{i=1}^{n}(g_{\boldsymbol{\theta}}(\boldsymbol{x}_i) - y_i)^2$$



Least-squares cost function is convex!

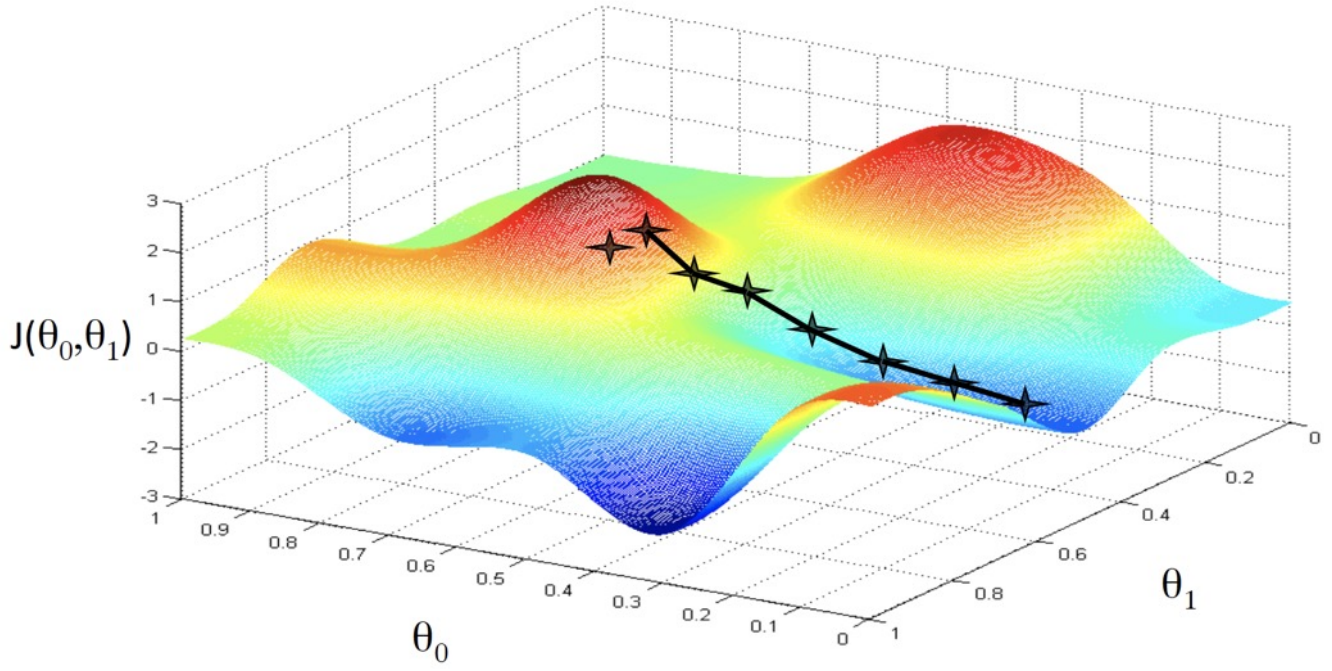
Gradient descent (梯度下降)

- A search procedure:
  - starts with some initial guess for $\boldsymbol{\theta}$
  - repeatedly changes $\boldsymbol{\theta}$ to make $J(\boldsymbol{\theta})$ smaller
  - until converging to a minimum



Gradient descent

- A search procedure:
  - starts with some initial guess for $\boldsymbol{\theta}$
  - repeatedly changes $\boldsymbol{\theta}$ to make $J(\boldsymbol{\theta})$ smaller
  - until converging to a minimum



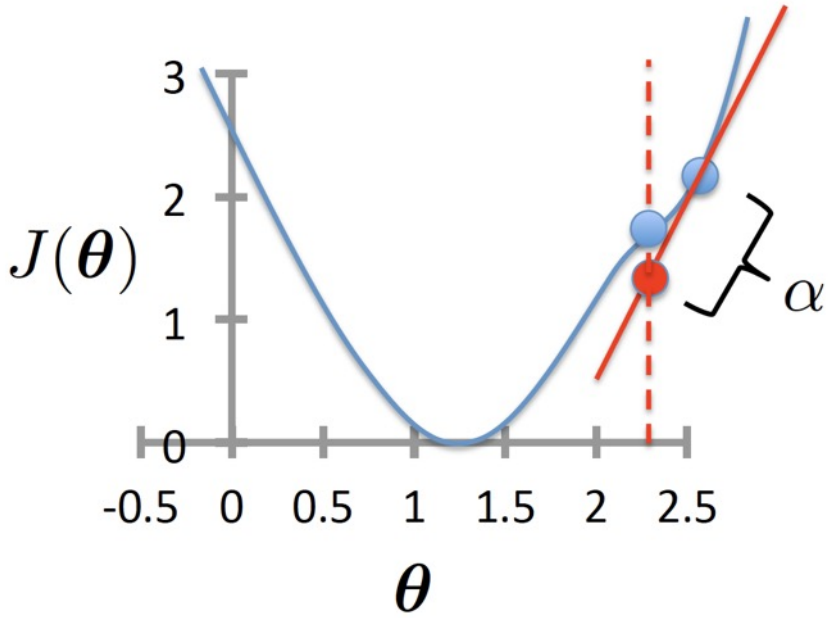
Gradient descent

- Initialize $\boldsymbol{\theta}$
- Repeat until convergence

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\boldsymbol{\theta})$$

simultaneous update
for j = 0 ... d

learning rate (small)
e.g., α = 0.05



19

Gradient descent for linear regression

$$\frac{\partial}{\partial \theta_j} J(\boldsymbol{\theta}) = \frac{\partial}{\partial \theta_j} \frac{1}{2} (g(x) - y)^2$$

$$= (g(x) - y) \cdot \frac{\partial}{\partial \theta_j} (g(x) - y)$$

$$= (g(x) - y) \cdot \frac{\partial}{\partial \theta_j} \left(\Sigma_{i=0}^{d} \theta_i x_i - y \right)$$

$$= (g(x) - y) \cdot x_j$$

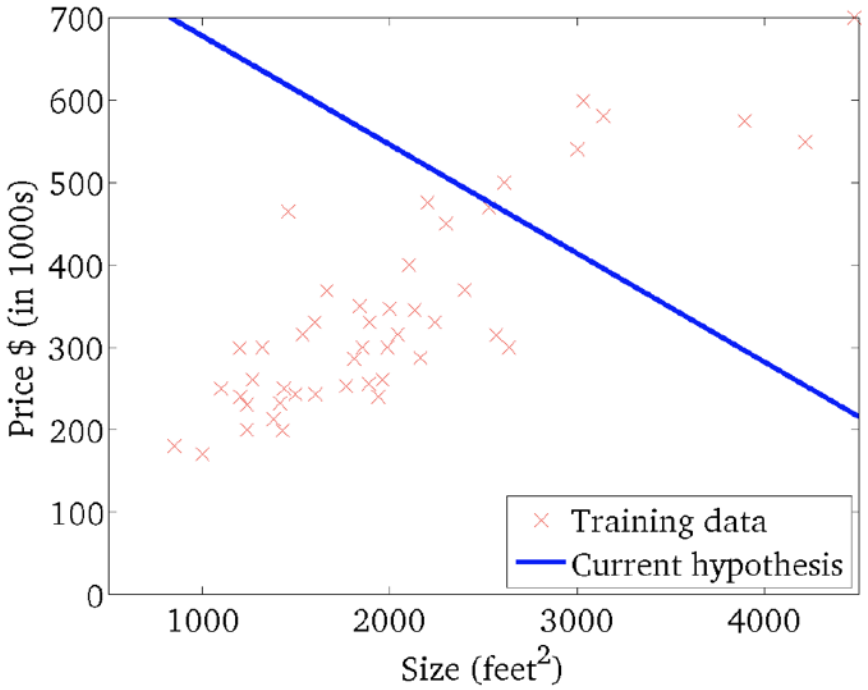
Least Mean Squares (LMS) update rule

- $\theta_j \leftarrow \theta_j - \alpha \cdot (g(x) - y) \cdot x_j$
- Assume convergence when $\|\boldsymbol{\theta}_{new} - \boldsymbol{\theta}_{old}\|_2 < \epsilon$

Visualization of gradient descent - 2

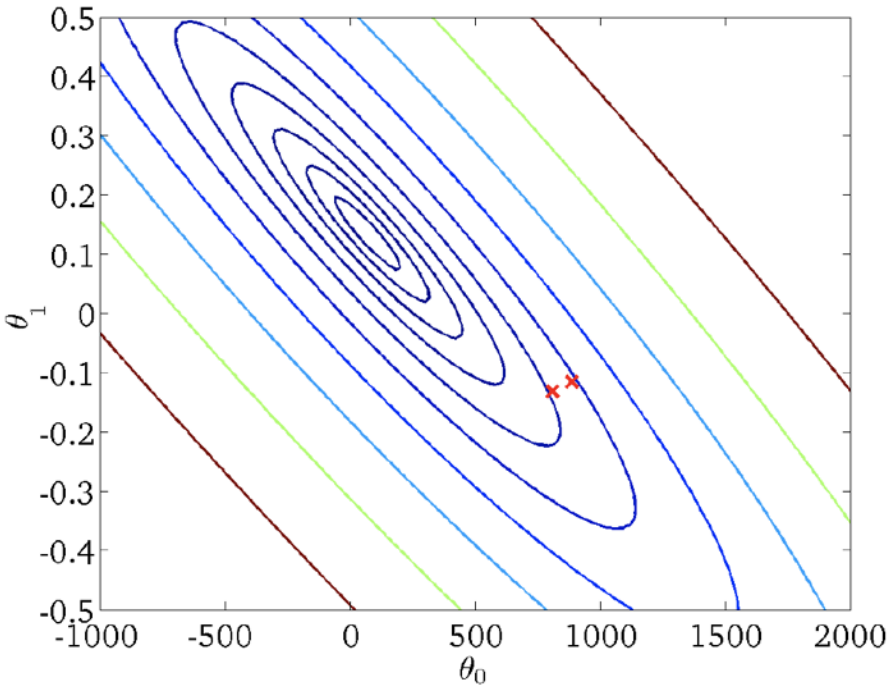
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

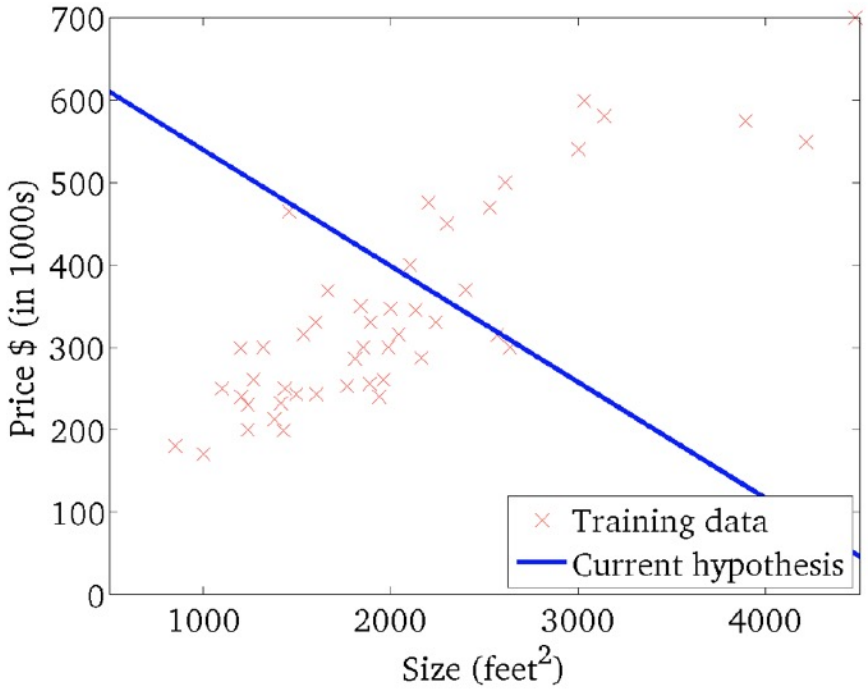
(function of the parameters θ_0, θ_1)



Visualization of gradient descent - 3

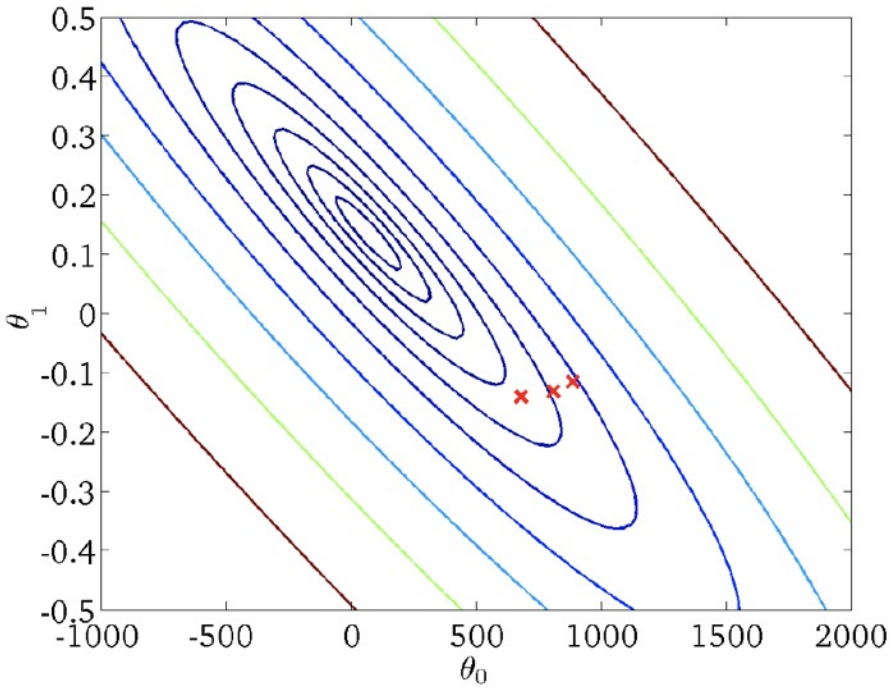
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

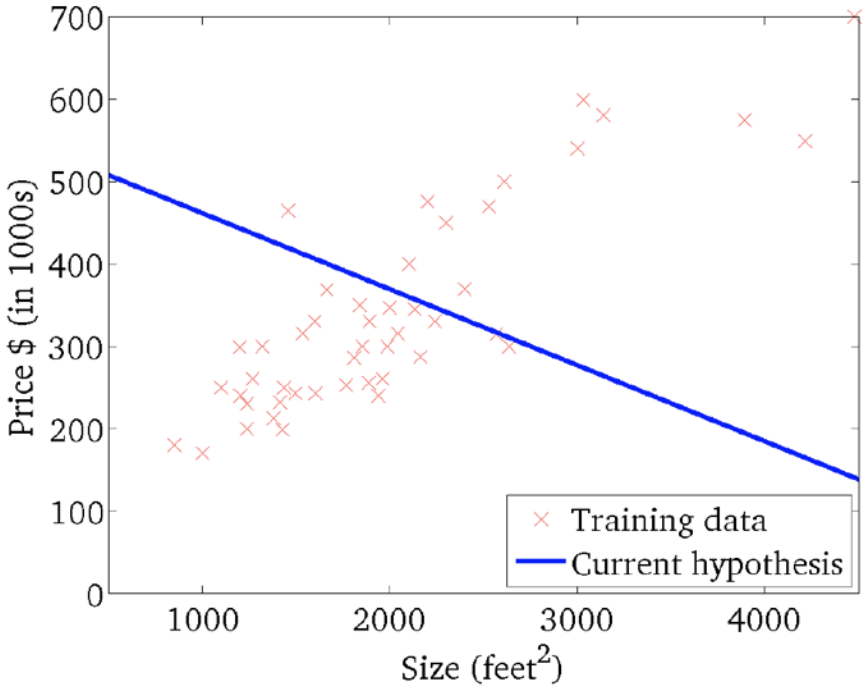
(function of the parameters θ_0, θ_1)



Visualization of gradient descent - 4

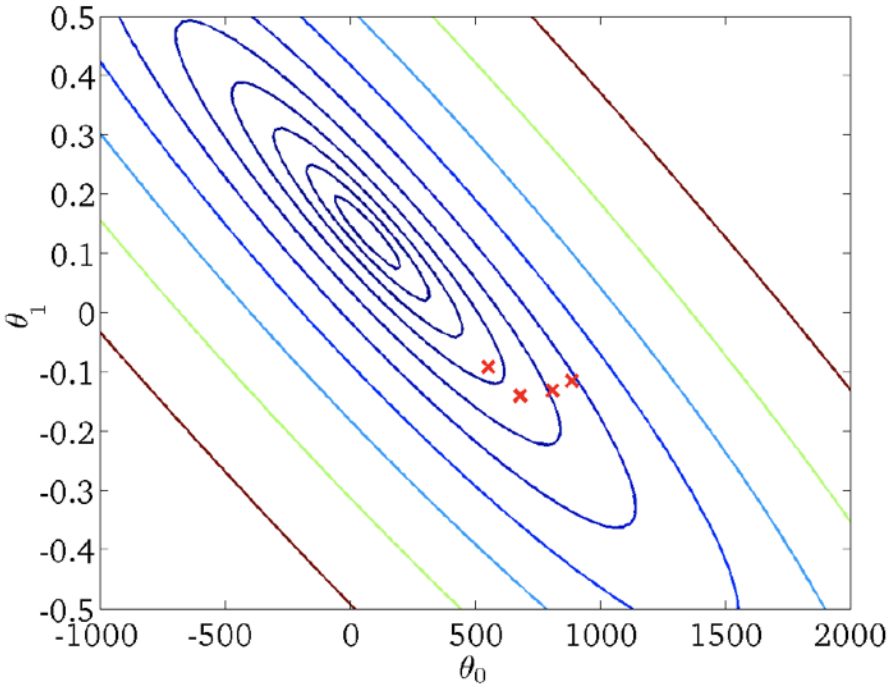
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

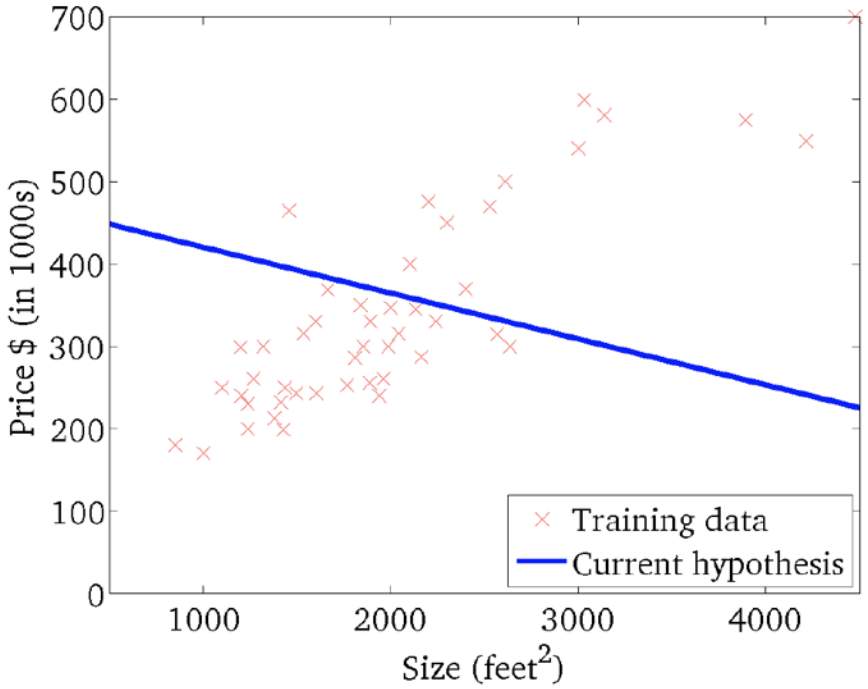
(function of the parameters θ_0, θ_1)



Visualization of gradient descent - 5

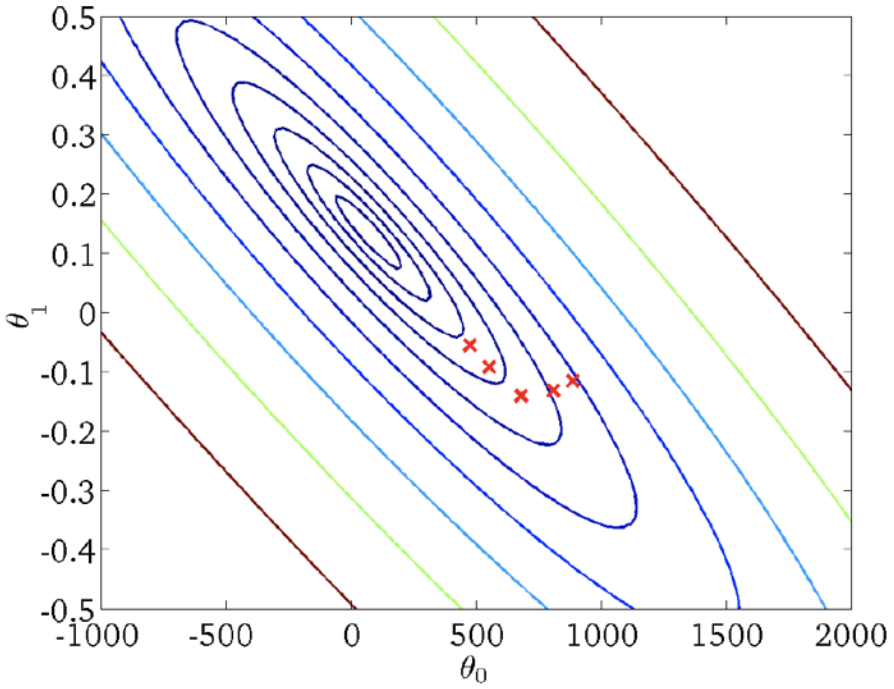
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

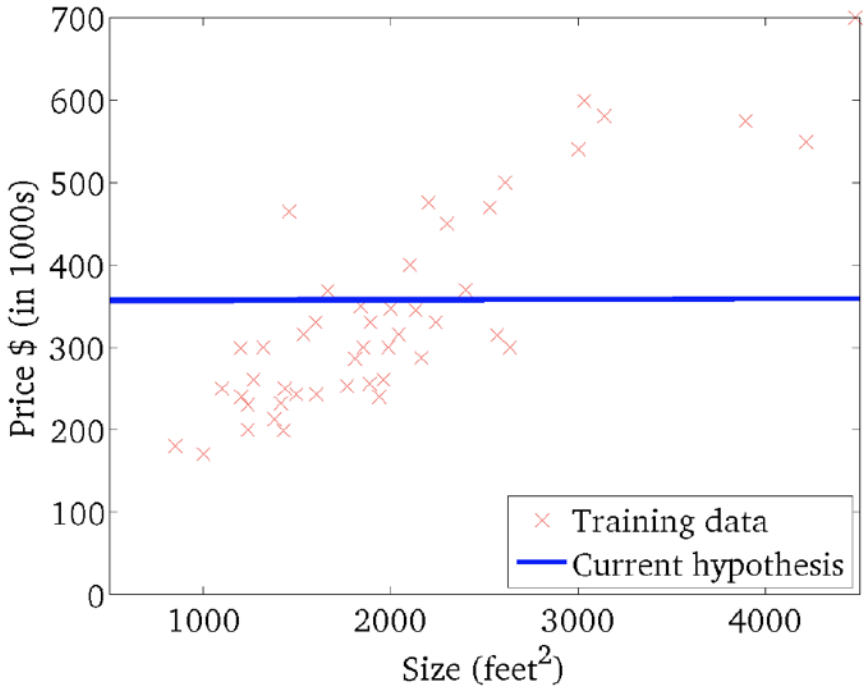
(function of the parameters θ_0, θ_1)



Visualization of gradient descent - 6

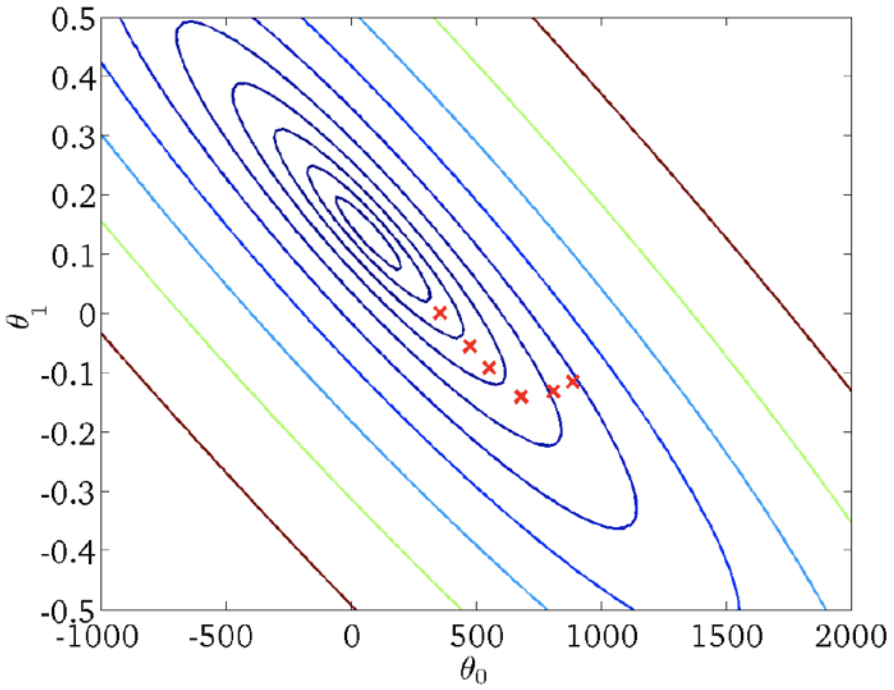
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

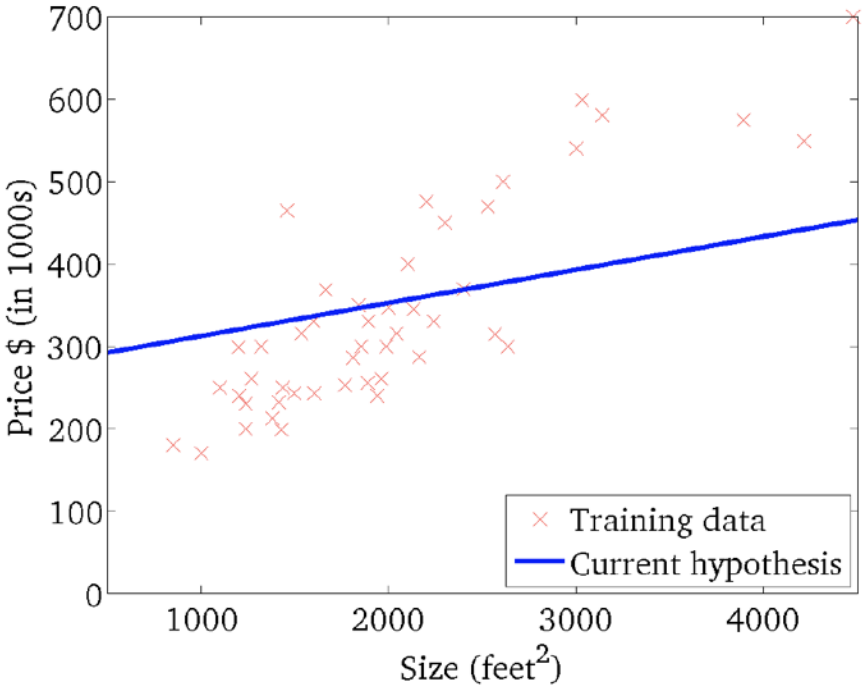
(function of the parameters θ_0, θ_1)



Visualization of gradient descent - 7

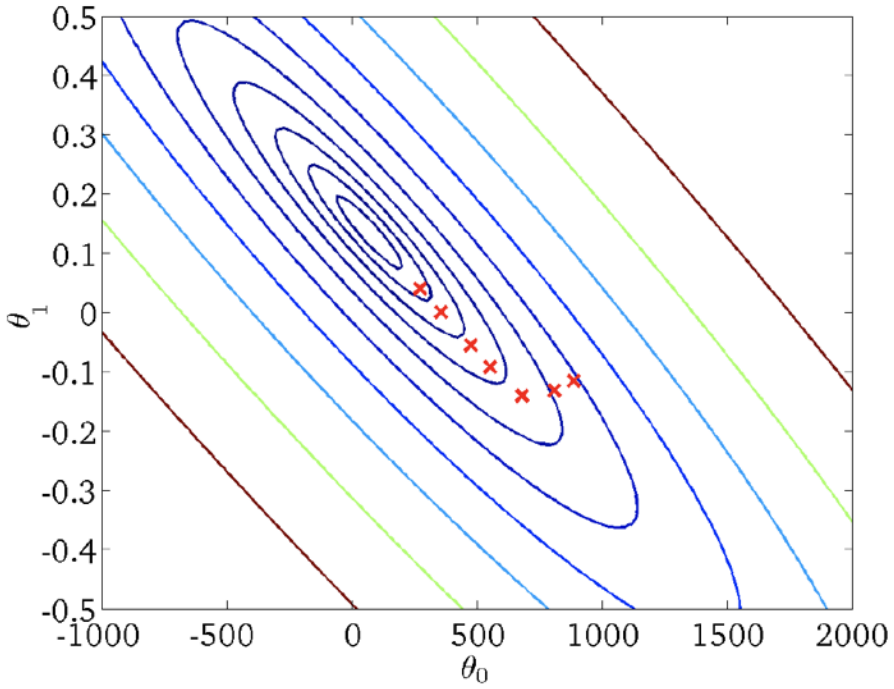
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

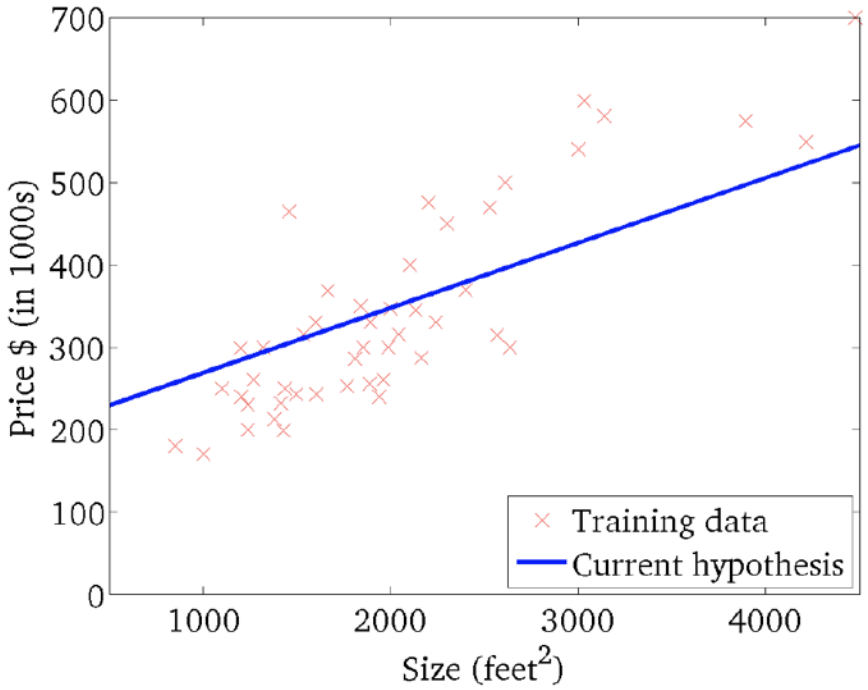
(function of the parameters θ_0, θ_1)



Visualization of gradient descent - 8

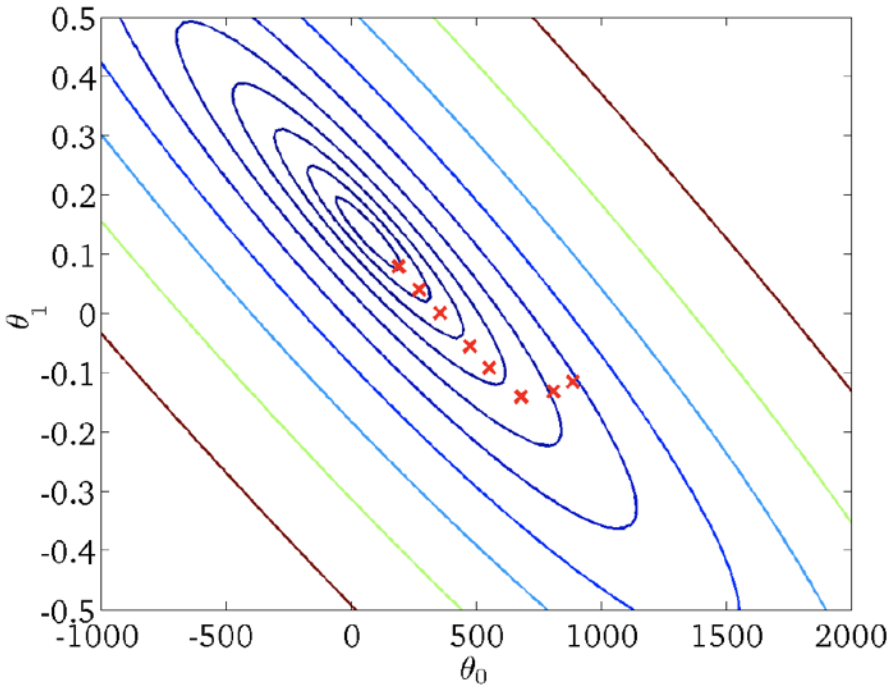
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

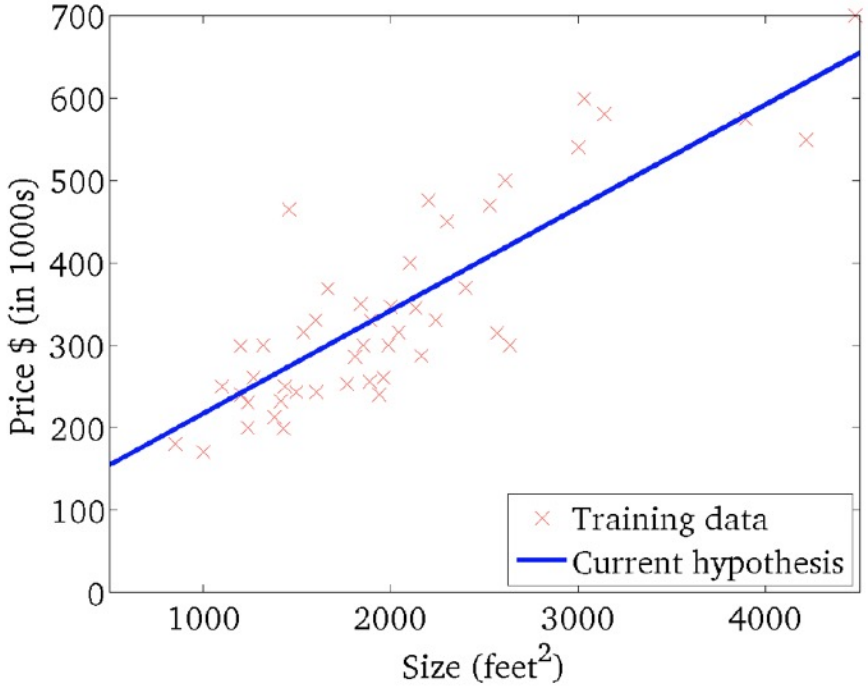
(function of the parameters θ_0, θ_1)



Visualization of gradient descent - 9

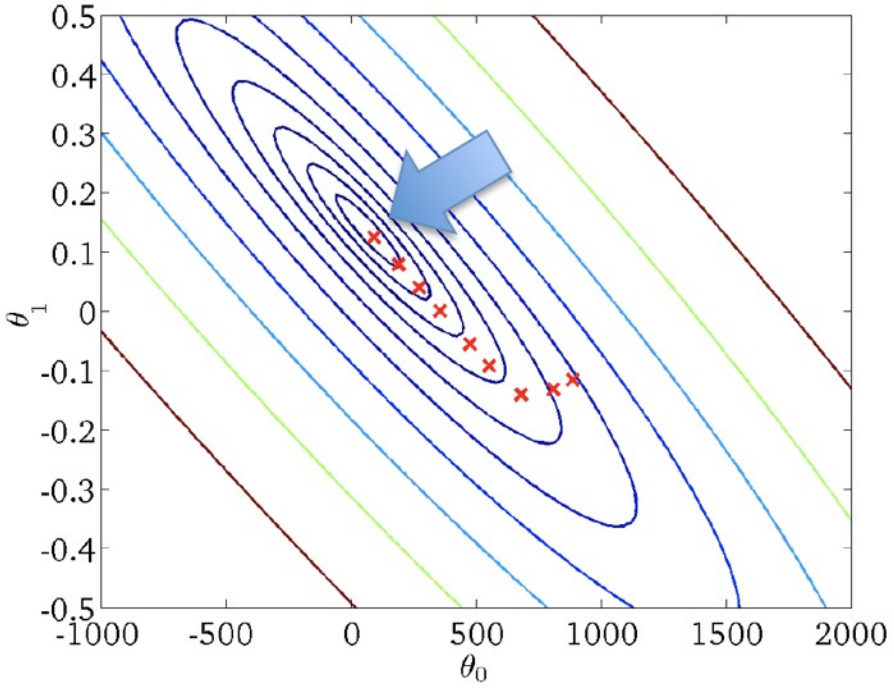
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



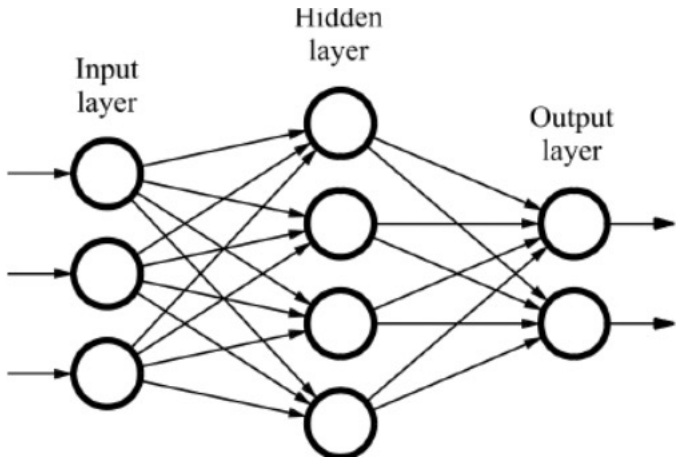
$J(\theta_0, \theta_1)$

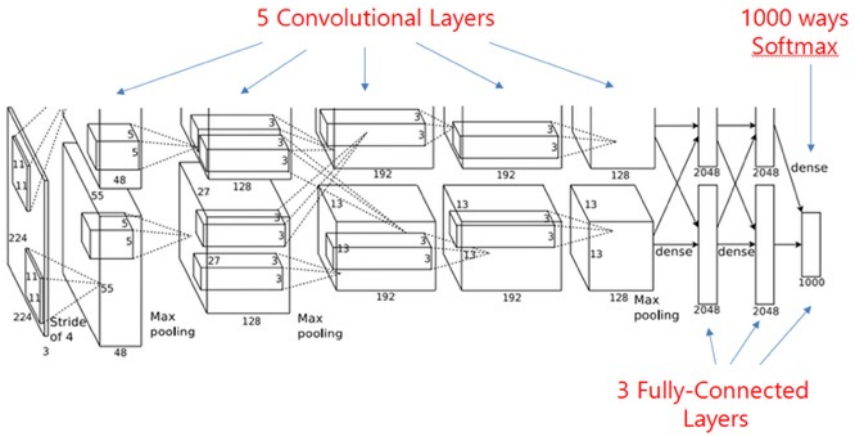
(function of the parameters θ_0, θ_1)

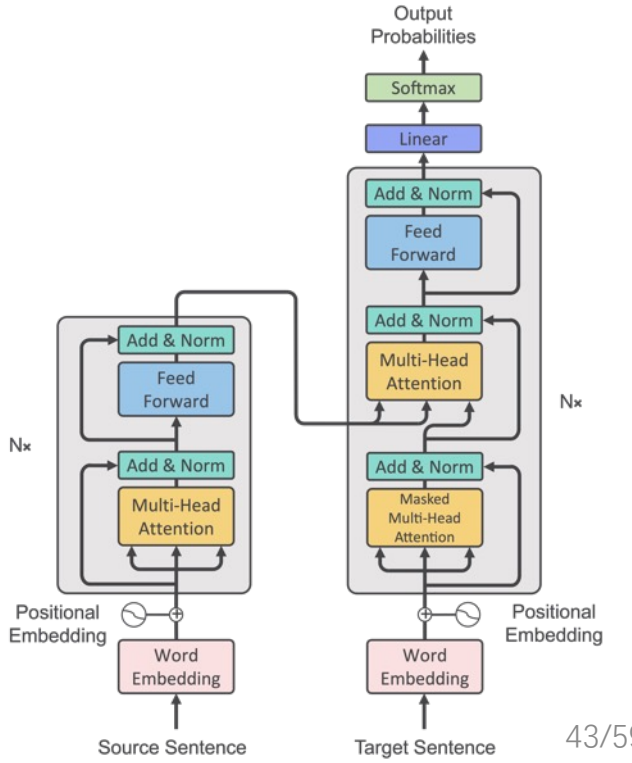


Neural networks? Learn more after class...

- Input-output pairs $\Sigma_i(x_i, y_i)$
- Function approximator $y \approx g(x)$, using a neural network?
- Cost function, e.g., mean-squared error, cross-entropy loss (交叉熵损失)
- Optimization – gradient descent

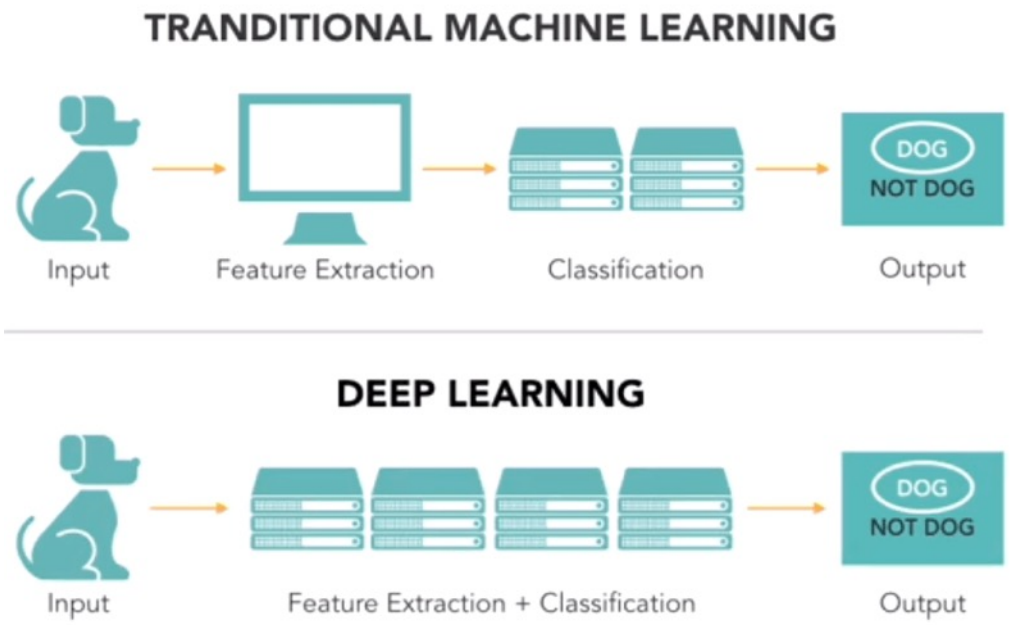






Deep learning? Learn more after class...

- a broader family of machine learning methods based on artificial neural networks with representation learning
  - deep neural networks, deep belief networks, recurrent neural networks, convolutional neural networks, etc
  - learning can be supervised, semi-supervised or unsupervised
  - in an end-to-end manner

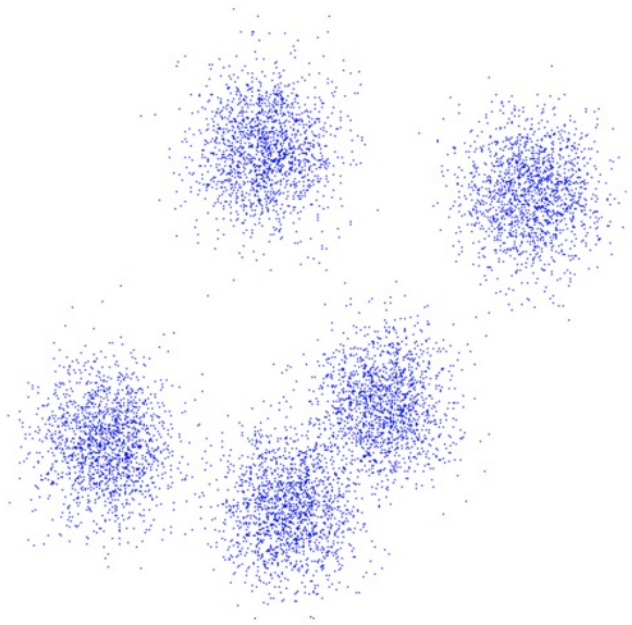


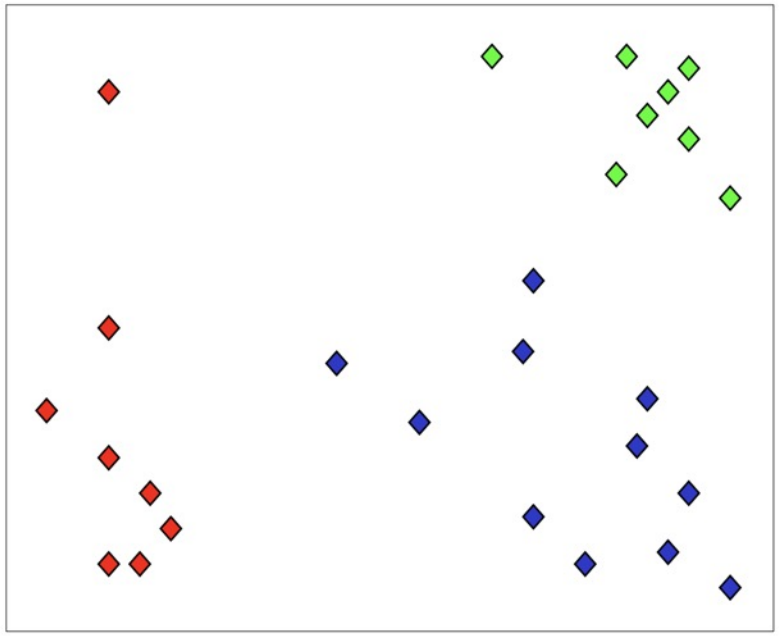
Content

1. Overview
2. Supervised Learning, Linear Regression
3. Unsupervised Learning, K-Means Clustering

Unsupervised learning

- A type of machine learning looks for previously undetected patterns in a data set with no pre-existing labels and with a minimum of human supervision
- Main methods: Clustering and dimensionality reduction





Clustering

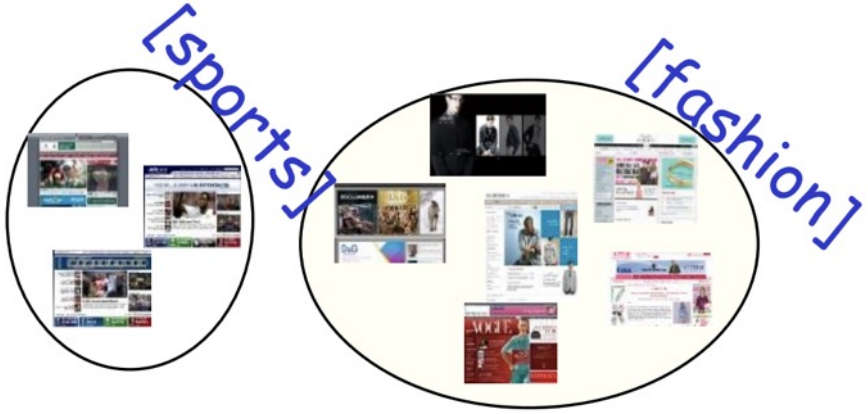
- **Goal**: Automatically partition unlabeled data into groups of similar data points.
- **Question**: When and why would we want to do this?
  - Automatically organizing data
  - Understanding hidden structure in data
  - Preprocessing for further analysis, representing high-dimensional data in a low-dimensional space (e.g., for visualization purposes)
- Typical algorithms: K-means, Gaussian mixture models (GMM, 高斯混合模型)…

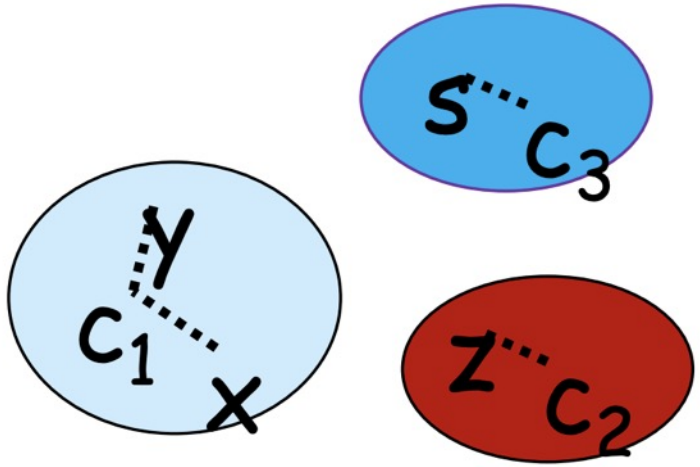
Applications (Clustering comes up everywhere...)

- Cluster news articles or web pages or search results by topic
- Cluster protein sequences by function or genes according to expression profile
- Cluster users of social networks by interest (community detection)
- Cluster customers according to purchase history (用户画像)
- Cluster galaxies or nearby stars (e.g. Sloan Digital Sky Survey)
- And many many more applications...

K-means clustering

- **Input**:
  - A set S of n points
  - A dissimilarity measure specifying the distance $d(x_i, x_j)$ between data pairs (x_i, x_j)
- **Goal**: Output a partition of the data
- K-means: Find center points $(c_1, c_2, \ldots, c_k)$ to minimize $\sum_{i=1}^{n} \min_{j \in \{1,\ldots,k\}} d(x_i, c_j)$

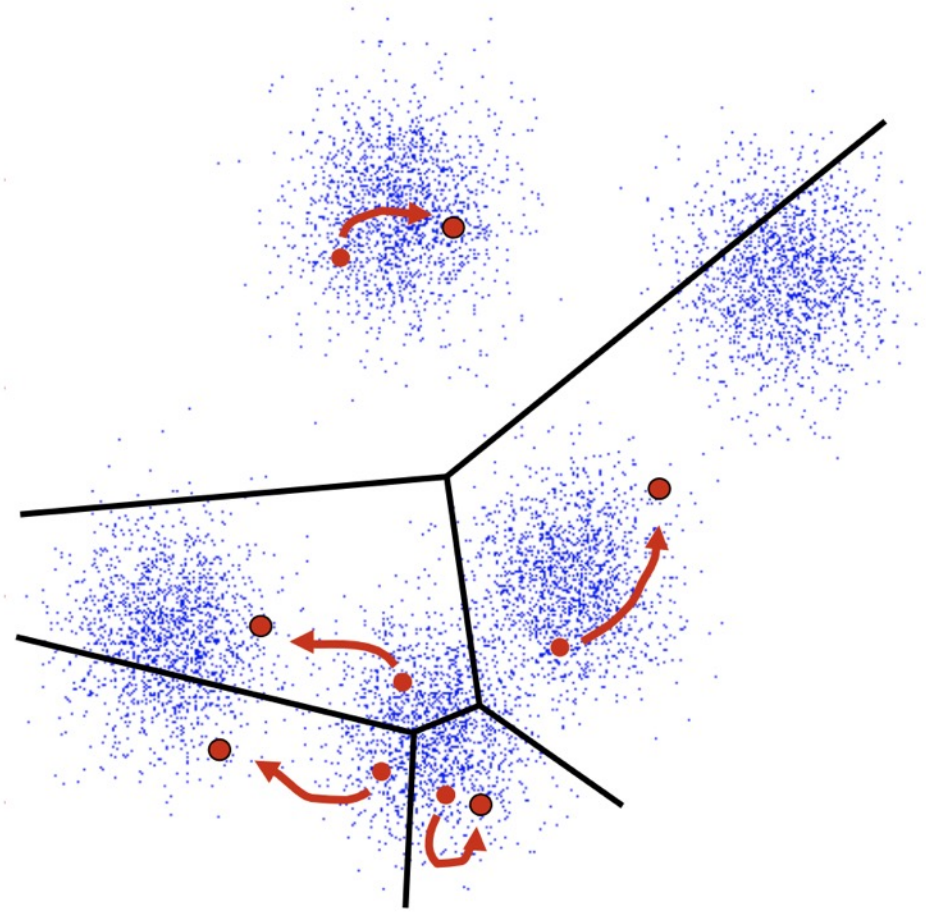




Euclidean K-means clustering

- **Input**:
  - a set of n datapoints, $(x_1, x_2, \dots, x_n)$ in $\mathbb{R}^d$
  - target number of clusters k
- **Output**:
  - k representatives, $(c_1, c_2, \dots, c_k)$ in $\mathbb{R}^d$
- Objective:
  - Choose $(c_1, c_2, \dots, c_k)$ to minimize

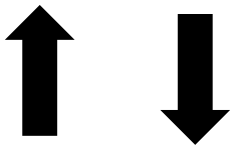
$$\min \sum_{i=1}^{n} \min_{j \in \{1,\dots,k\}} \|x_i - c_j\|^2$$



The iterative method

- **Input**:
  - a set of n datapoints, $(x_1, x_2, \dots, x_n)$ in $\mathbb{R}^d$
  - target number of clusters k
- **Initialize (in any way)**:
  - Centers $(c_1, c_2, \dots, c_k) \in \mathbb{R}^d$
  - Clusters $(C_1, C_2, \dots, C_k) \in \mathbb{R}^d$
- Repeat until no further change in the cost
  - For each j: $C_j \leftarrow \{\text{whose closest center is } c_j\}$
  - For each j: $c_j \leftarrow \text{mean of } C_j$

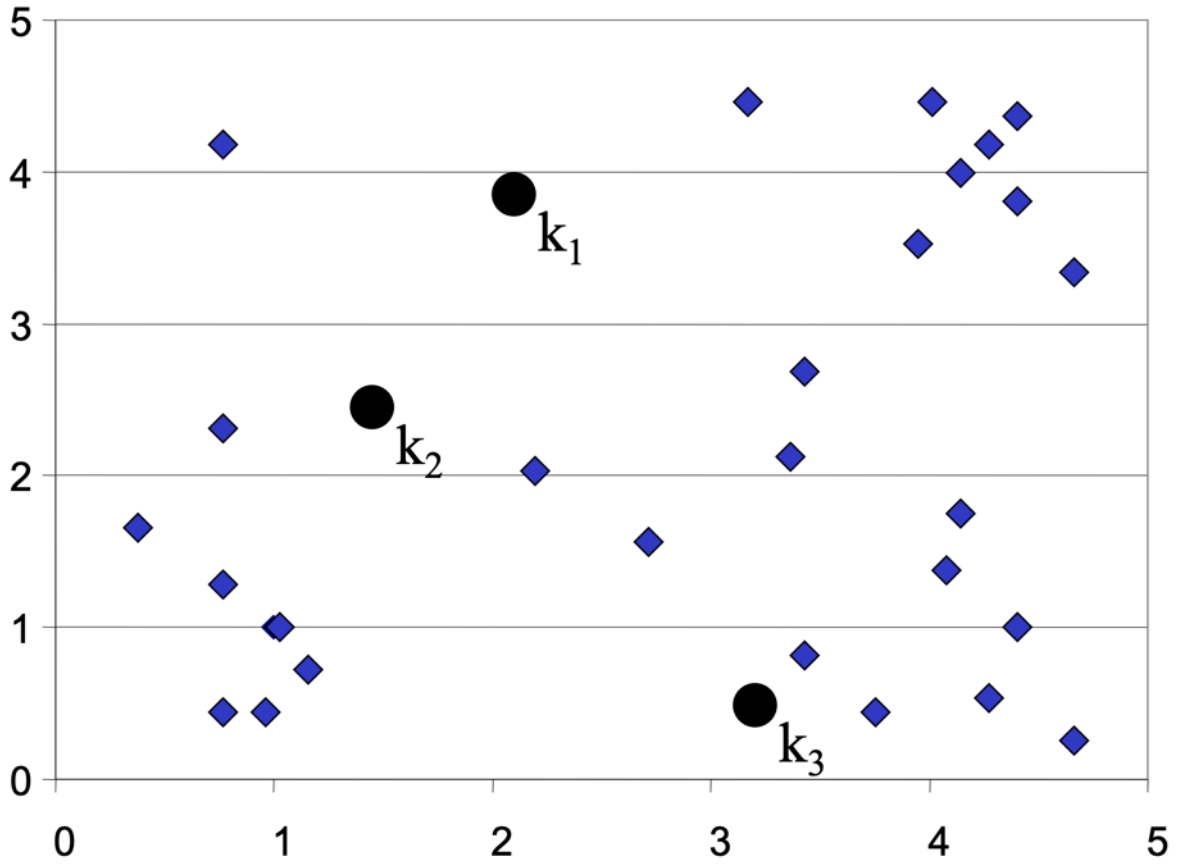
Holding $(c_1, c_2, \dots, c_k)$ fixed, picking optimal $(C_1, C_2, \dots, C_k)$



Holding $(C_1, C_2, \dots, C_k)$ fixed, picking optimal $(c_1, c_2, \dots, c_k)$

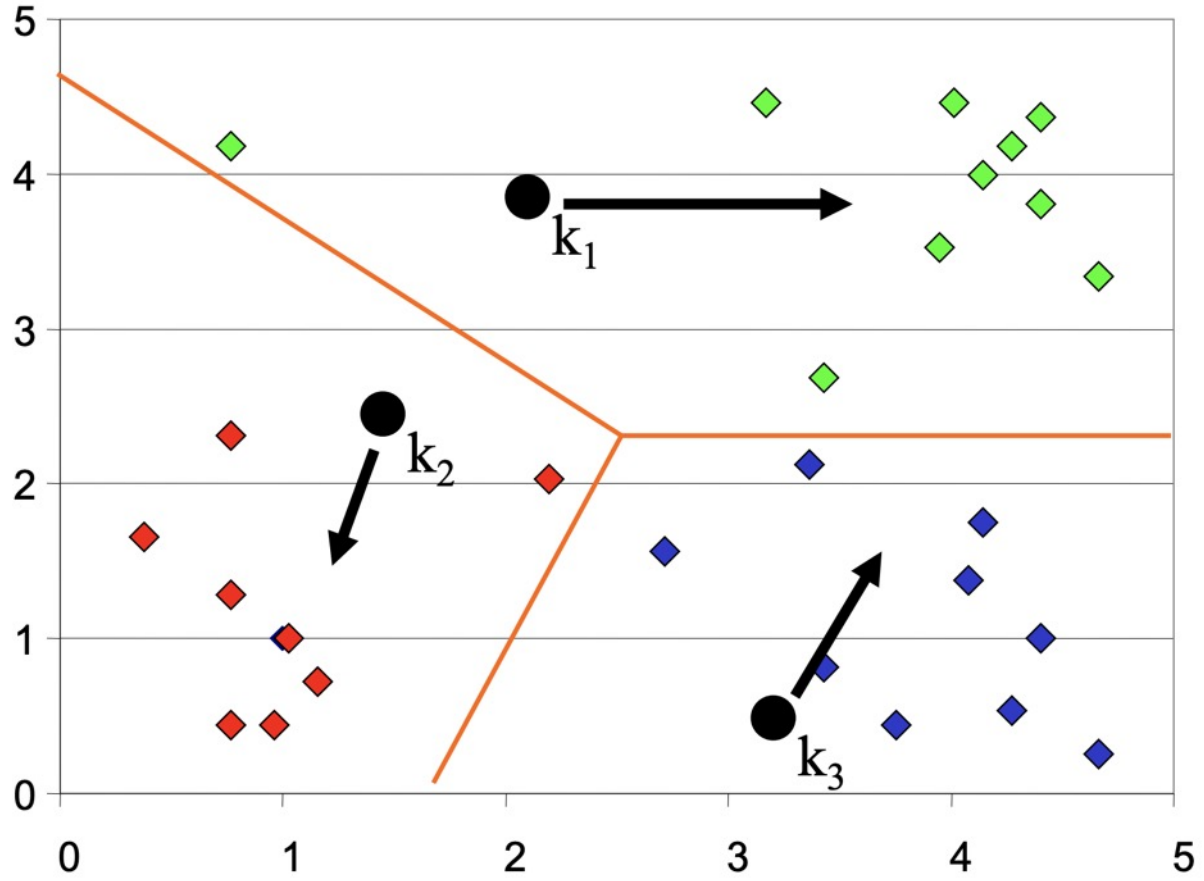
The iterative method

- Decide k, and initialize k centers (randomly)



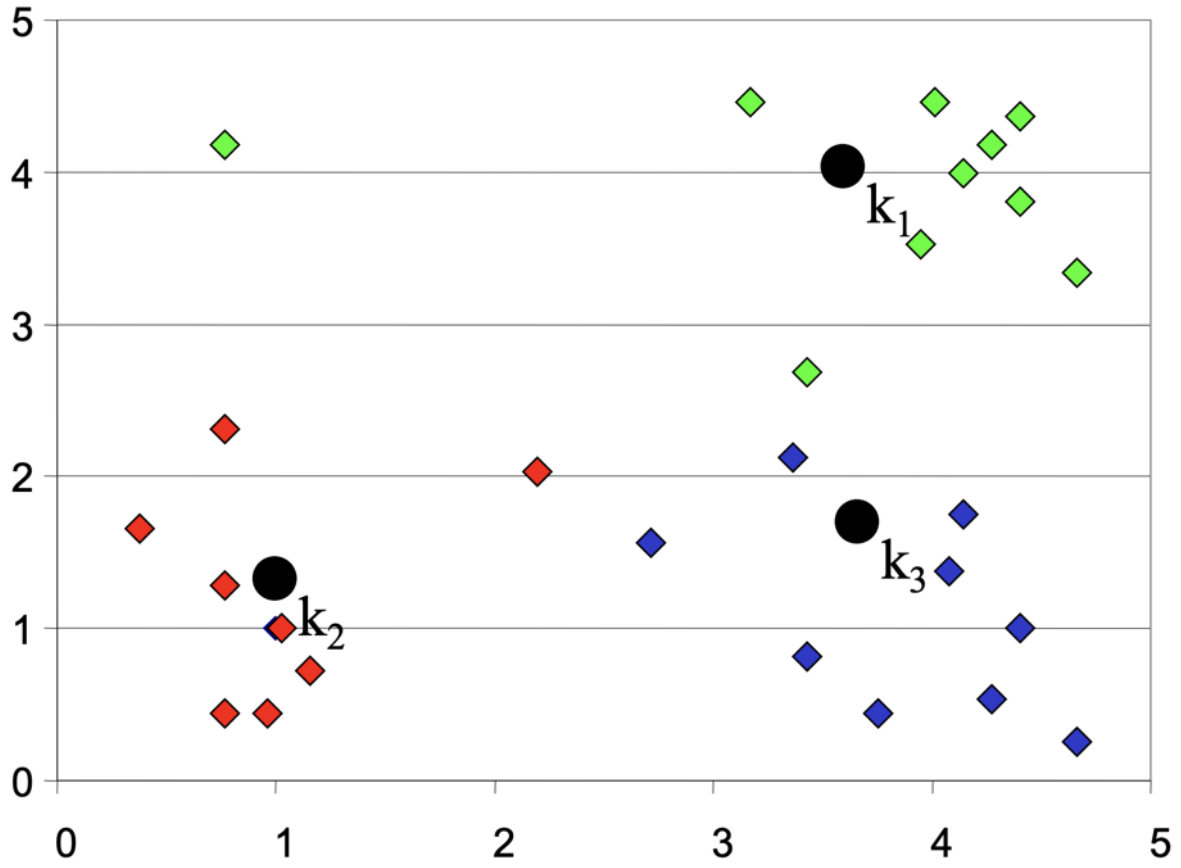
K-means clustering: Iteration 1

- Assign all objects to the nearest center.
- Move a center to the mean of its members.



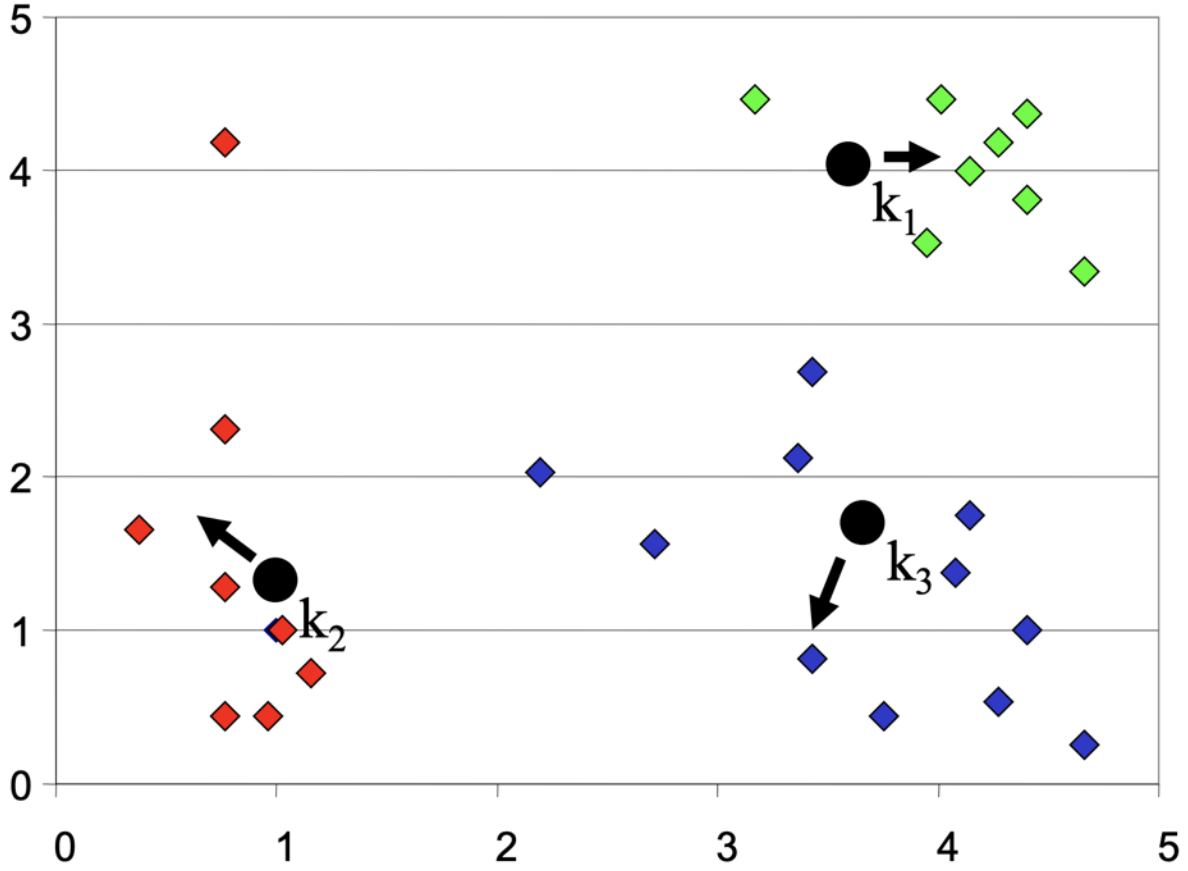
K-means clustering: Iteration 2

- After moving centers, re-assign the objects…



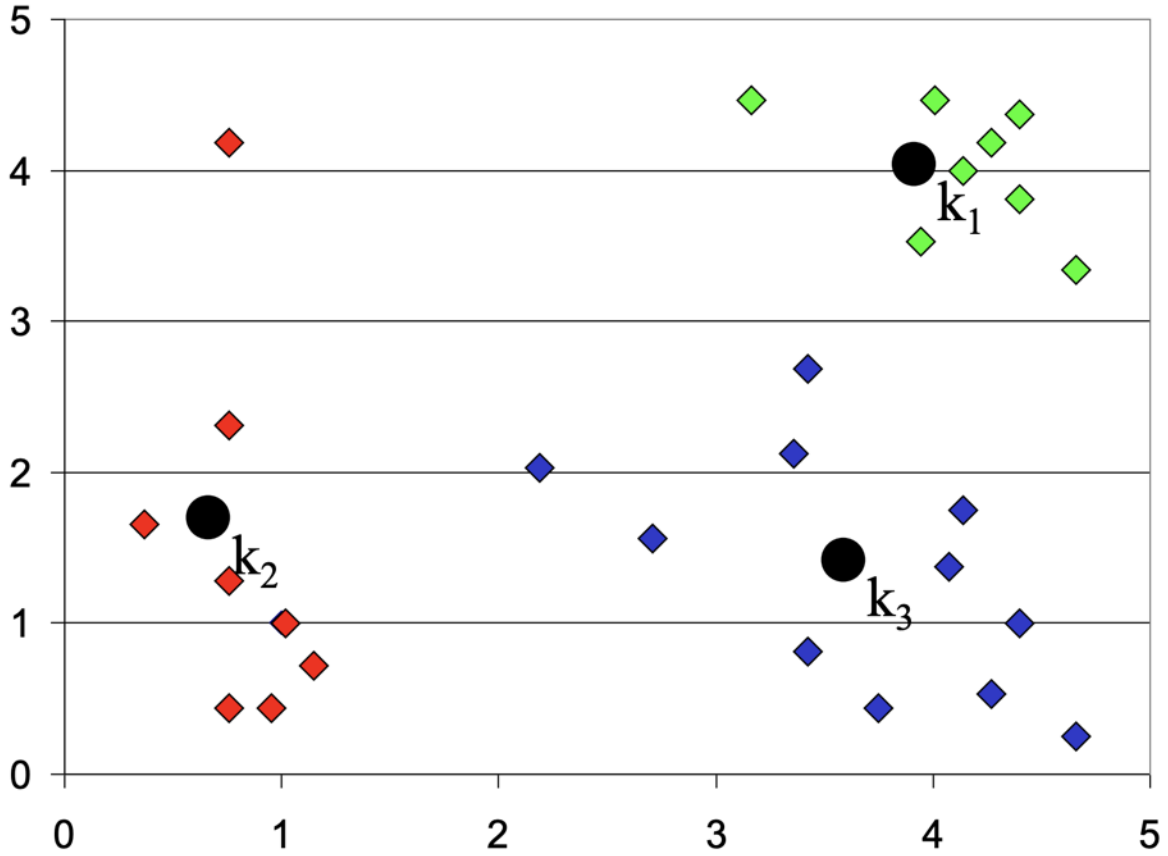
K-means clustering: Iteration 2

- After moving centers, re-assign the objects to nearest centers.
- Move a center to the mean of its new members.

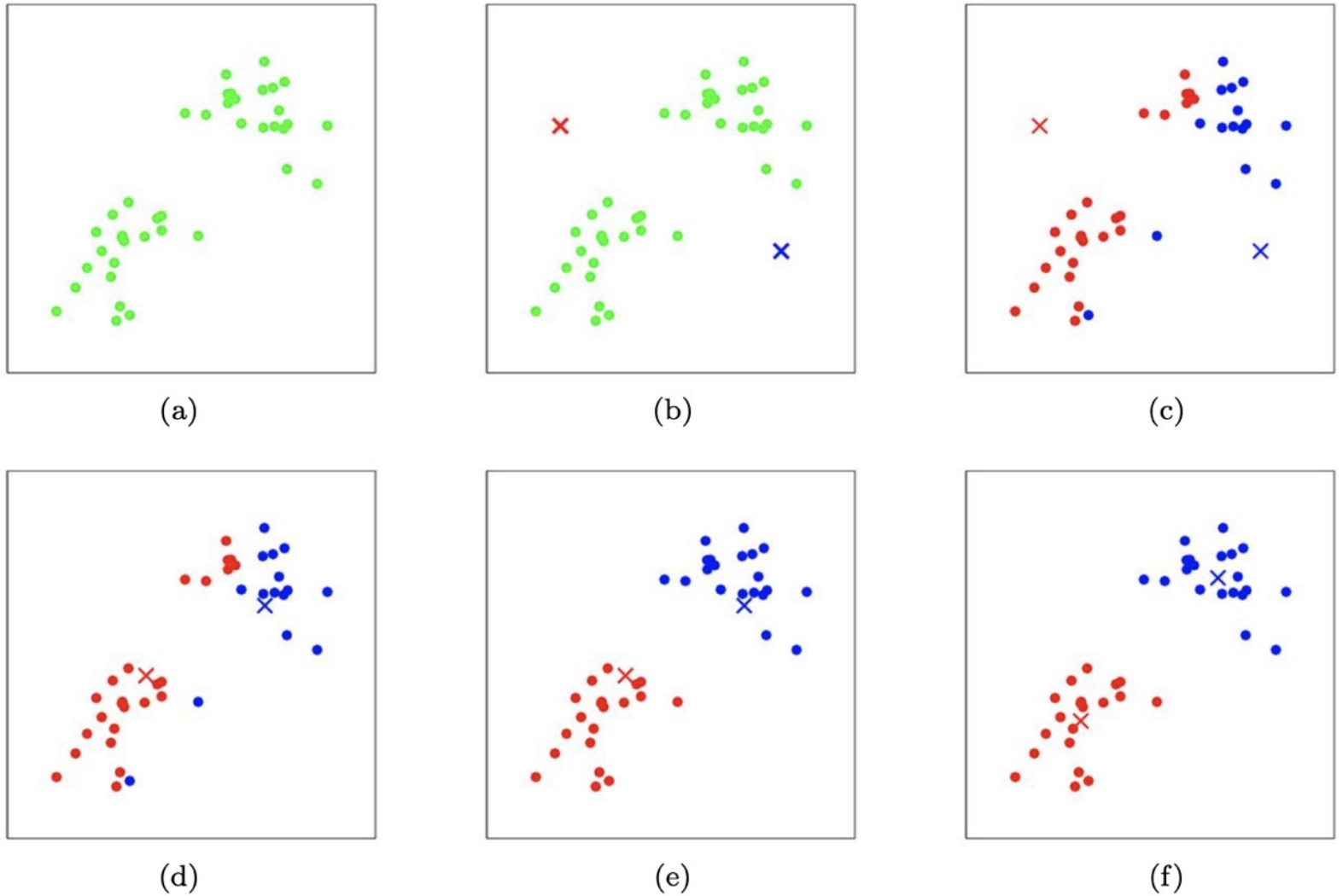


K-means clustering: Finished

- Re-assign and move centers, until no objects changed membership.



Example for K-means



Learning objectives of this lecture

- You should be able to…
  - Know elements of supervised learning and unsupervised learning
  - Be familiar with basic algorithms, e.g .,linear regression and K-means
  - Learn neural networks after class

THE END