CS5800: Algorithms

Week 12 – Machine Learning Algorithms

Dr. Ryan Rad

Fall 2023

Northeastern University

Today's Agenda

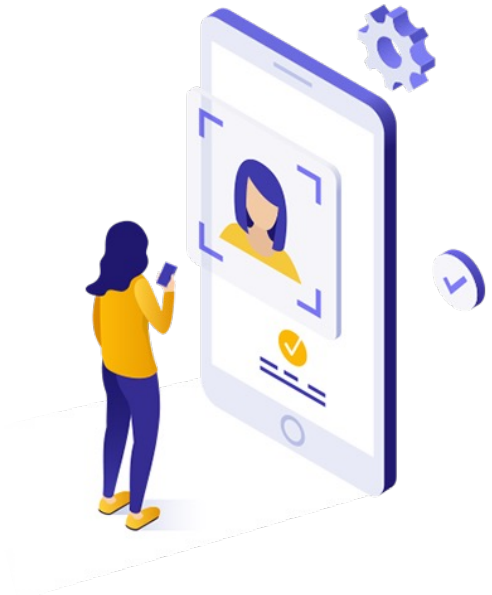
Mache Learning Algorithms

- Explicit Programming vs Machine Learning
- Supervised vs Unsupervised Learning
- Linear Regression
- Gradient Decent
- Clustering
- K-Means

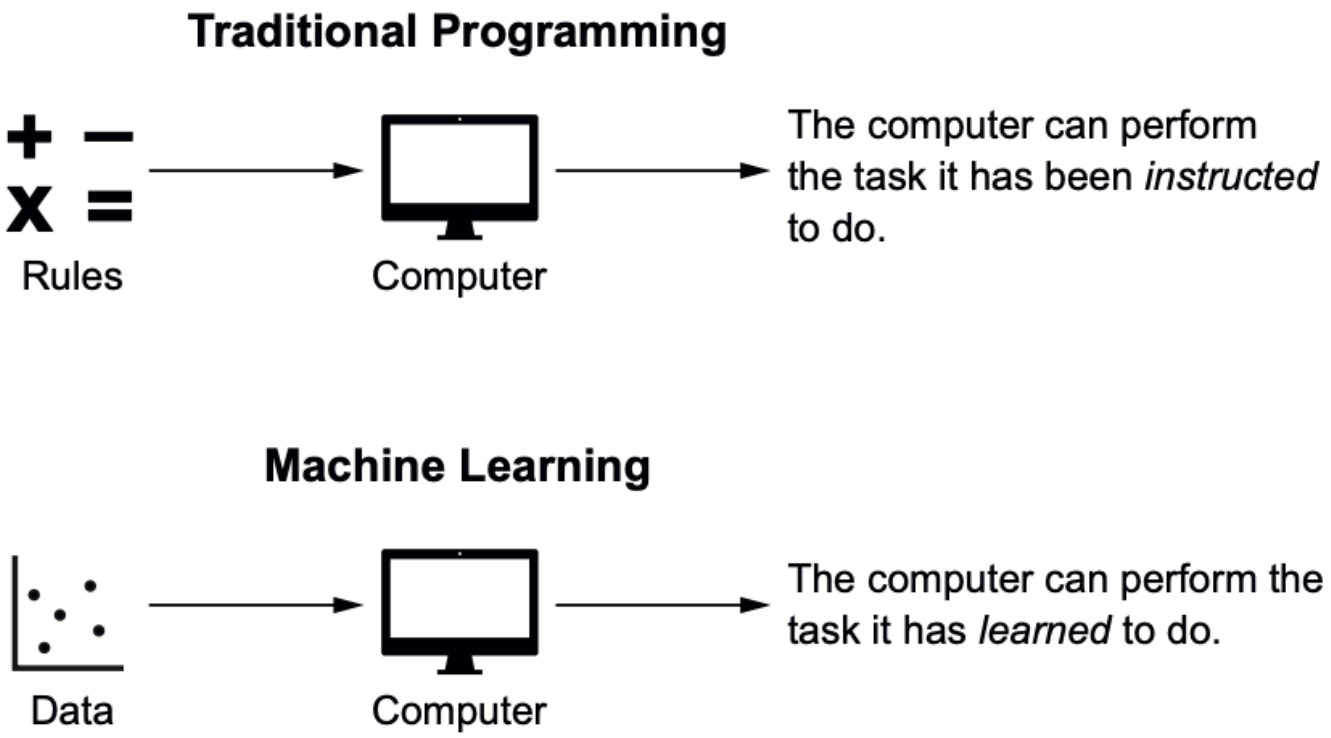
Traditional Programing VS Machine Learning

Access Control System with *ID Card Scanners*

Access Control System with *Face Recognition*



Traditional Programing VS Machine Learning?



The path to modern AI (Second Wave)

In 1980s, private companies such as IBM and Xerox started investing in a new AI spring. New hopes were fueled by a technology called ***expert systems***: computer programs that encode the knowledge of a human expert in a certain field in the form of precise, *if-then* rules.

Suppose you want to build an AI system that can stand in for a gastroenterologist. This is how you do it with an expert system: you ask a doctor to describe with extreme precision how they make decisions about patients. You then ask a programmer to painstakingly transform the doctor's knowledge and diagnosis flow to if-then rules that can be understood and executed by a computer.

- *If the patient has a stomachache and the body temperature is high, then the patient has the flu.*
- *If the patient has a stomachache and has eaten expired food, then the patient has food poisoning.*

The path to modern AI (Second Wave)

Do you see any problems with this approach?

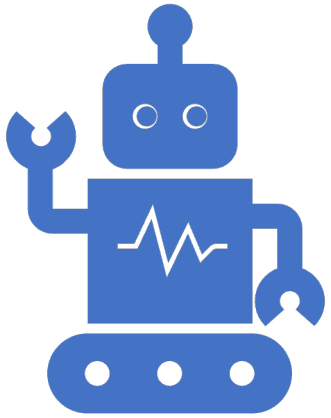
- ***Poor adaptability***—The only way for the software to improve is to go back to the drawing board with a computer scientist and the expert (in this case, the doctor).
- ***Extreme fragility***—The system will fail in situations that weren't part of the original design. What if a patient has a stomachache but normal body temperature, and hasn't eaten spoiled food?
- ***Tough to maintain***—The complexity of such a system is huge. When thousands of rules are put together, improving it or changing it is incredibly complicated, slow, and expensive. Have you ever worked with a huge Microsoft Excel sheet and struggled to find the root cause of a mistake? Imagine an Excel sheet 100 times bigger.

The path to modern AI (Third Wave): Early 2000s

Expert systems were a commercial failure. By the end of the 1980s, many of the companies that were developing them went out of business, marking the beginning of the *second AI winter*.

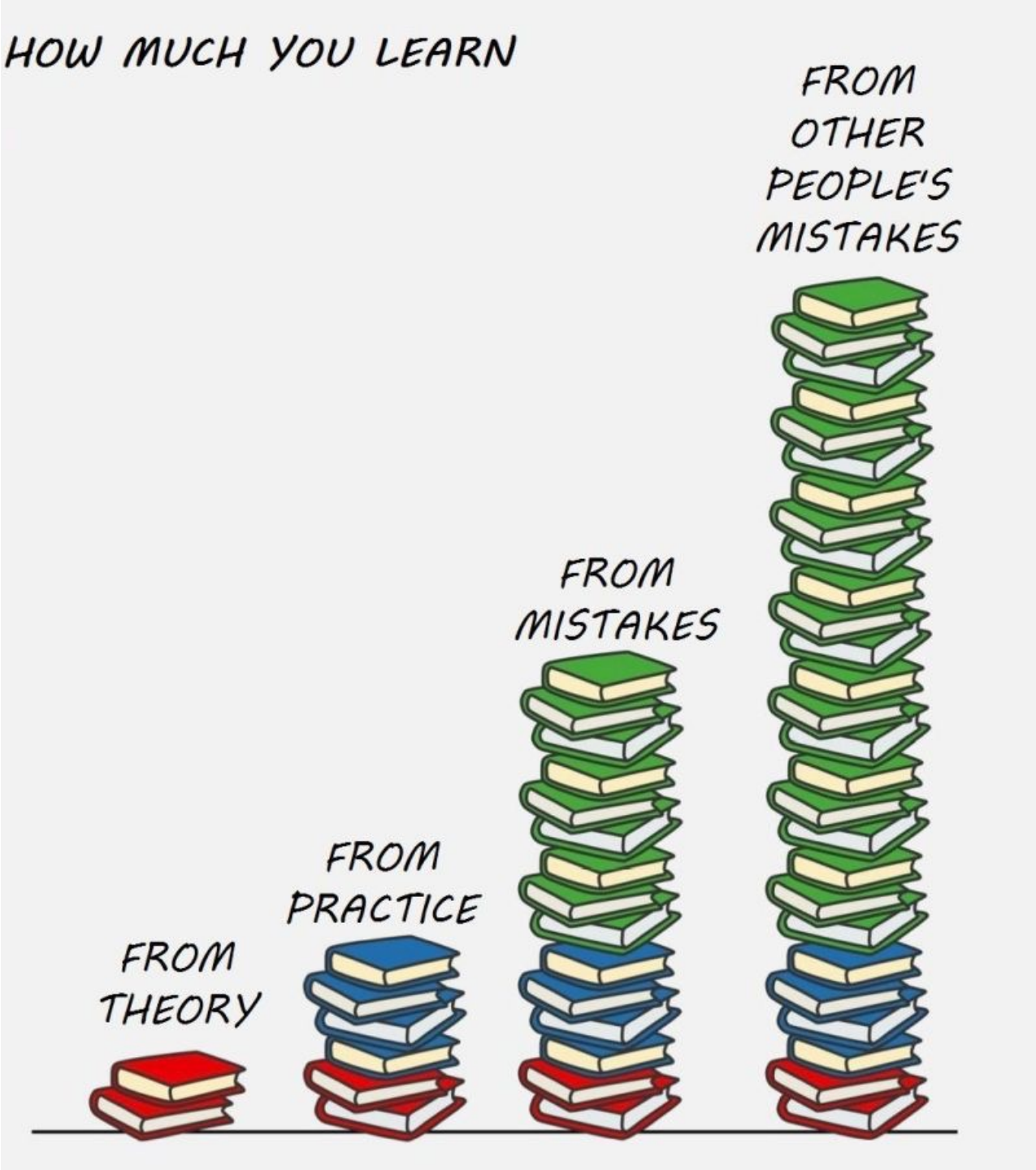
The first definition of *machine learning* dates back to 1959, from American AI pioneer Arthur Samuel:
"Machine learning is the field of study that gives computers the ability to learn without being explicitly programmed."

Introduction to Machine Learning



- **Machine Learning** is a set of techniques that teach computers to do what comes naturally to humans and animals: learn from experience.
- **Machine Learning** algorithms use computational methods to “learn” information directly from data without relying on predefined rules.
- The algorithms adaptively improve their performance as the number of samples available for learning increases

Traditional Programing VS Machine Learning?



Supervised vs Unsupervised

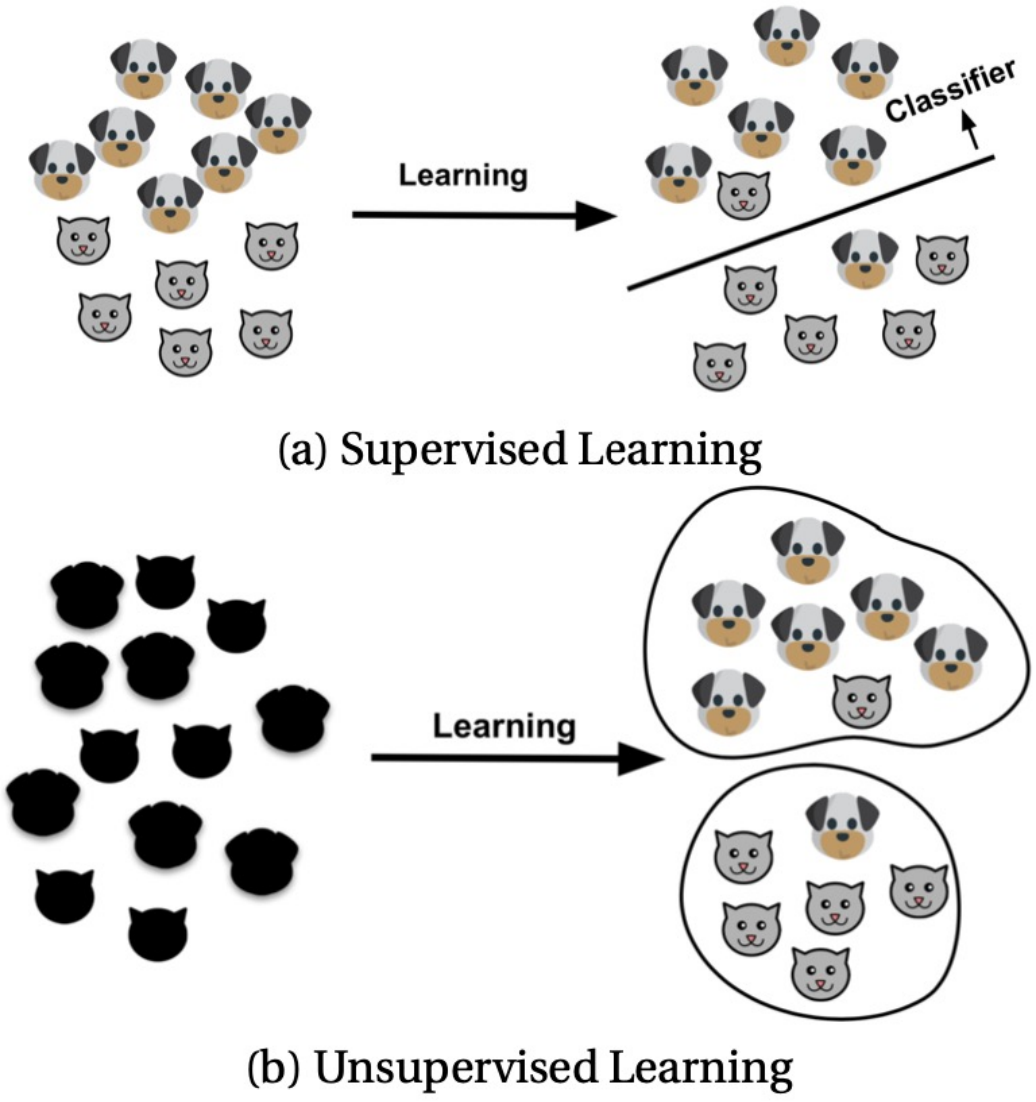
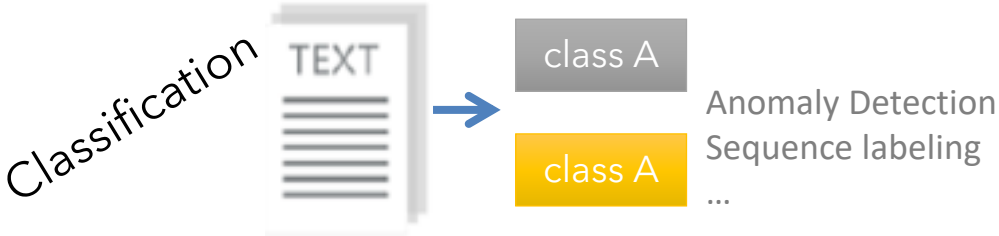


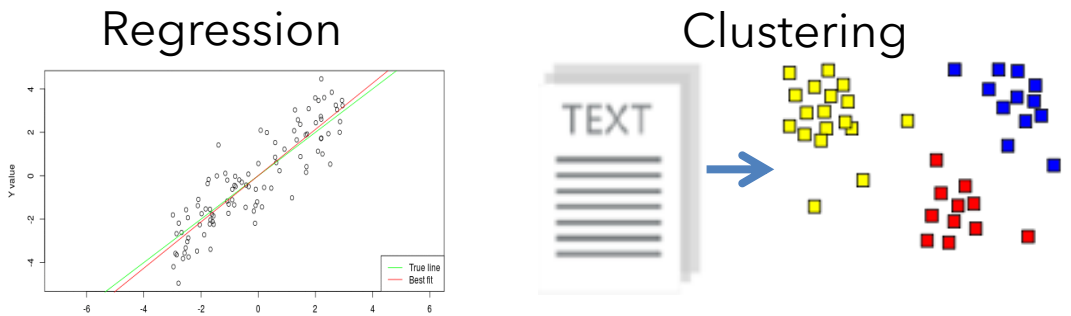
Figure from: Goyal, Anil. (2018). Learning a Multiview Weighted Majority Vote Classifier: Using PAC-Bayesian Theory and Boosting.

Types of Machine Learning

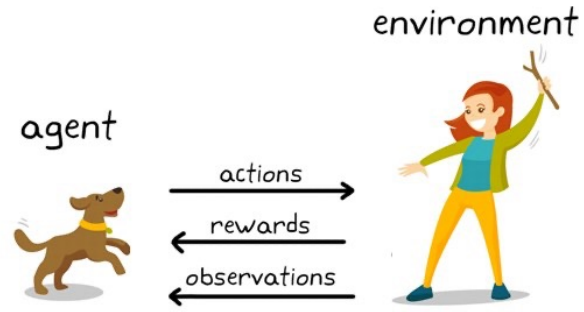
Supervised: Learning with a **labeled training** set
Example: email *classification* with already labeled emails



Unsupervised: Discover **patterns** in **unlabeled** data
Example: *cluster* similar documents based on text



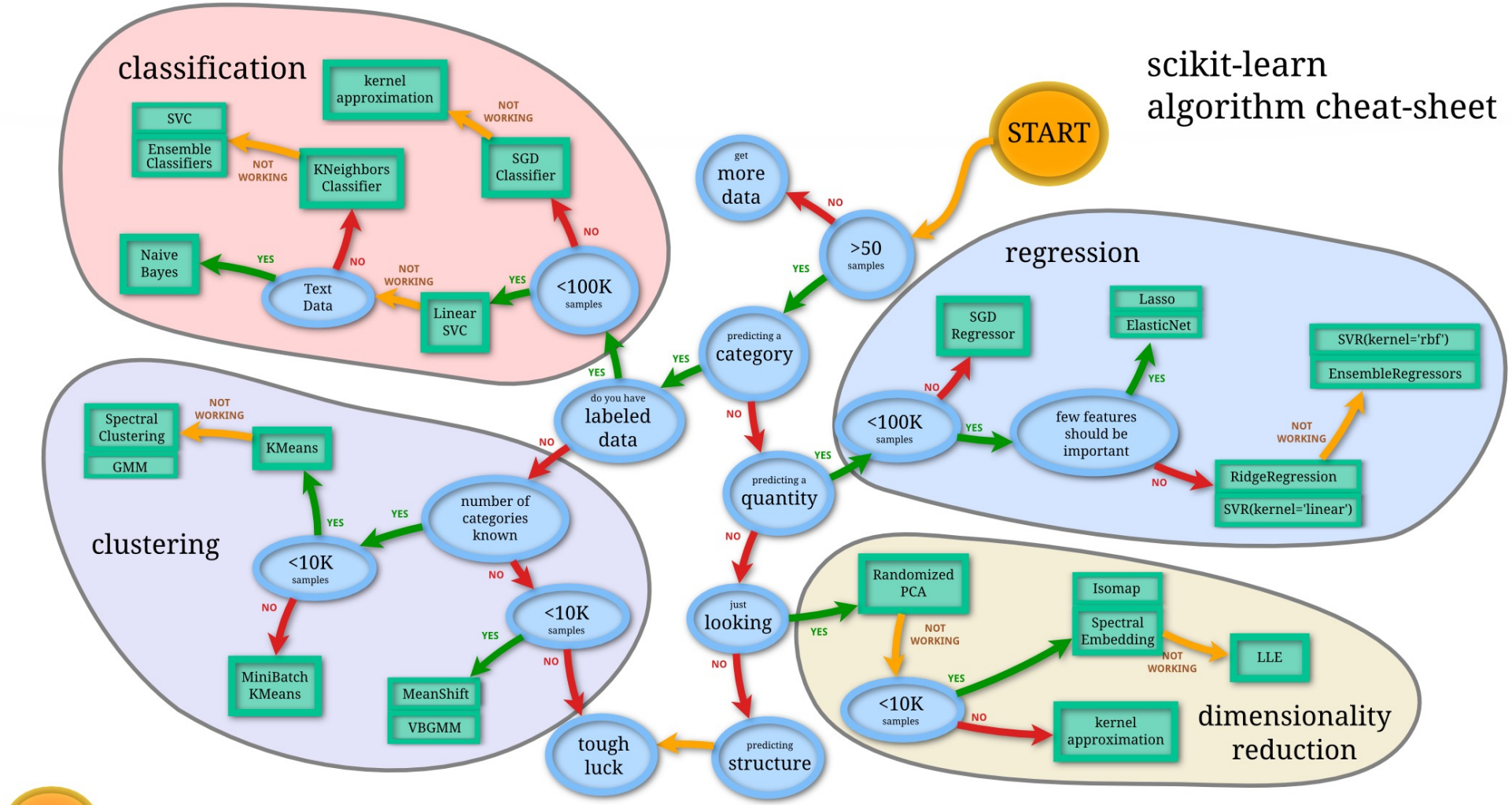
Reinforcement learning: learn to **act** based on **feedback/reward**
Example: learn to play Go, reward: *win or lose*



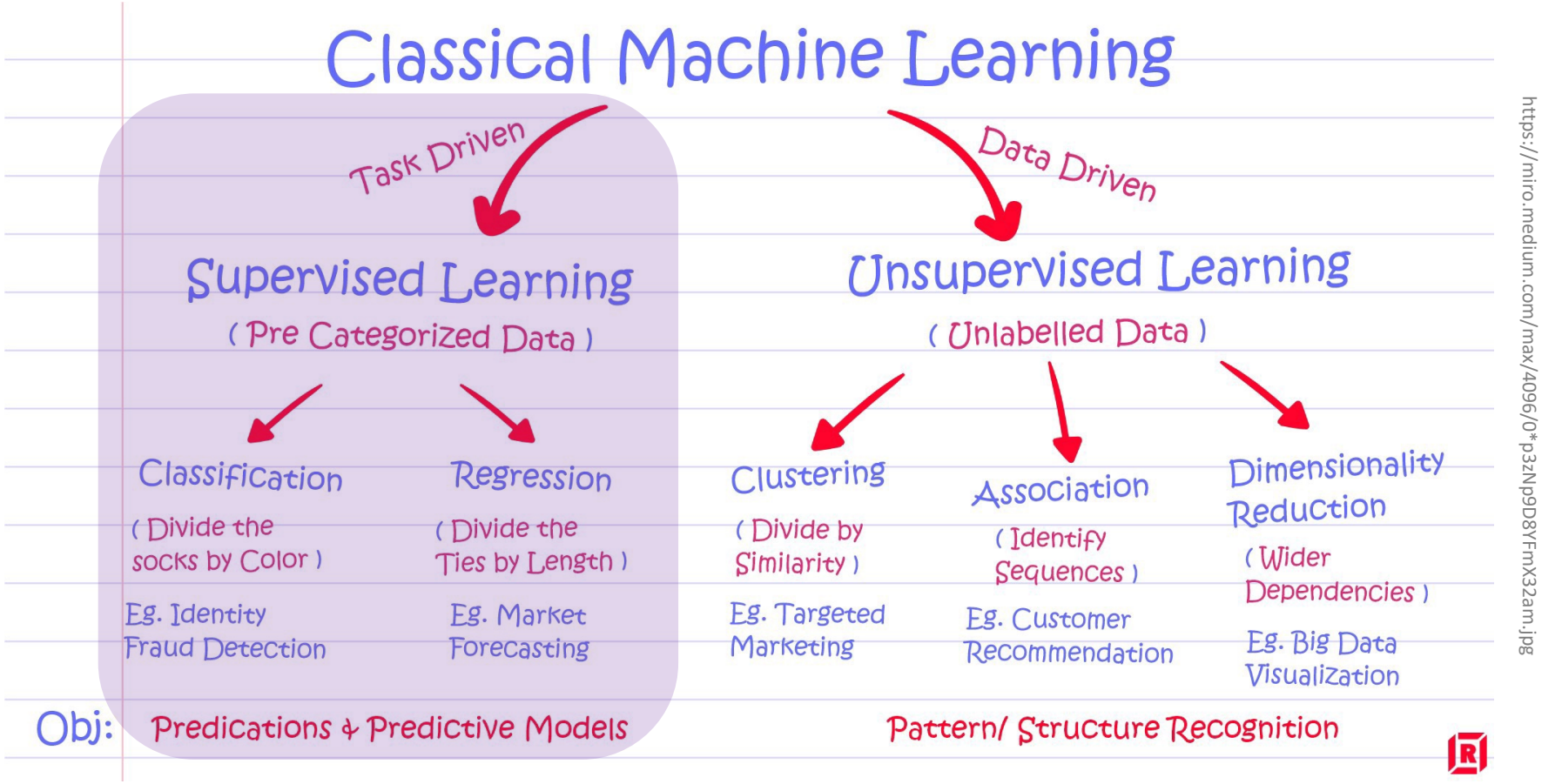
https://www.kdnuggets.com/images/mathworks-reinforcement-learning-fig1-543.jpg

http://mbjoseph.github.io/2013/11/27/measure.html

Scikit-learn algorithm cheat-sheet



Classical Machine Learning Methods



https://miro.medium.com/max/4096/0*p3zNp9D8YFmX32am.jpg

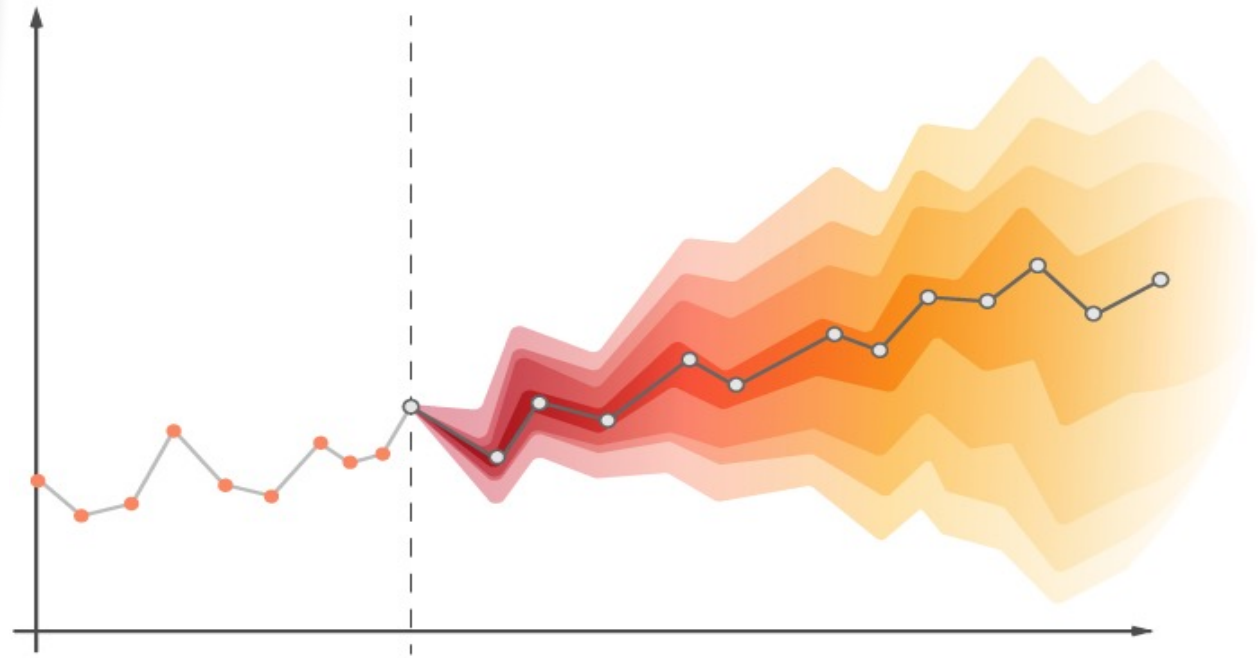
Week 12 Quiz – Q2

For which of the following tasks might K-means/GMM clustering be a suitable algorithm. Select ALL that apply.

a) Given a set of news articles from many different news websites, find out what are the main topics covered.
b) Given a dataset with thousands of historical emails with known spam status, you want to specifically determine if they are Spam or Non-Spam emails.
c) Given a database of information about your users, automatically group them into different market segments.
d) Given sales data from a large number of products in a supermarket, estimate future sales for each of these products.

Housing Price Prediction

Using Linear Regression



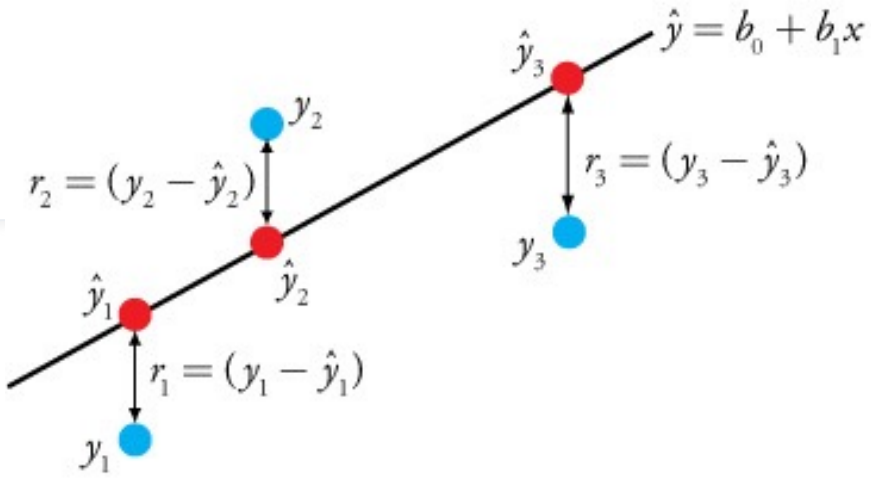
Linear Regression

Big Idea

General line formula:

Y = mX + b

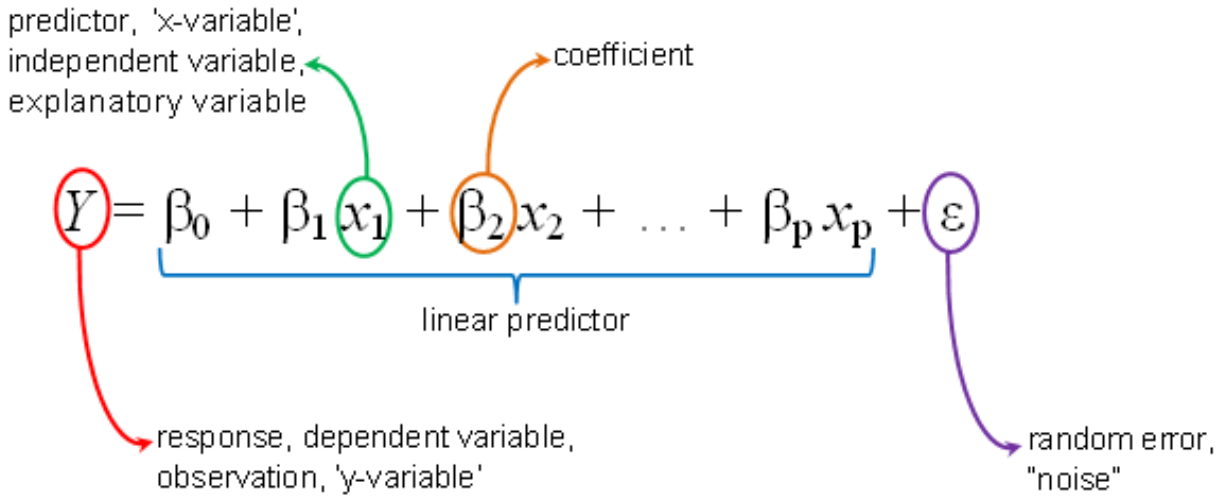
Y = X (+ 0)



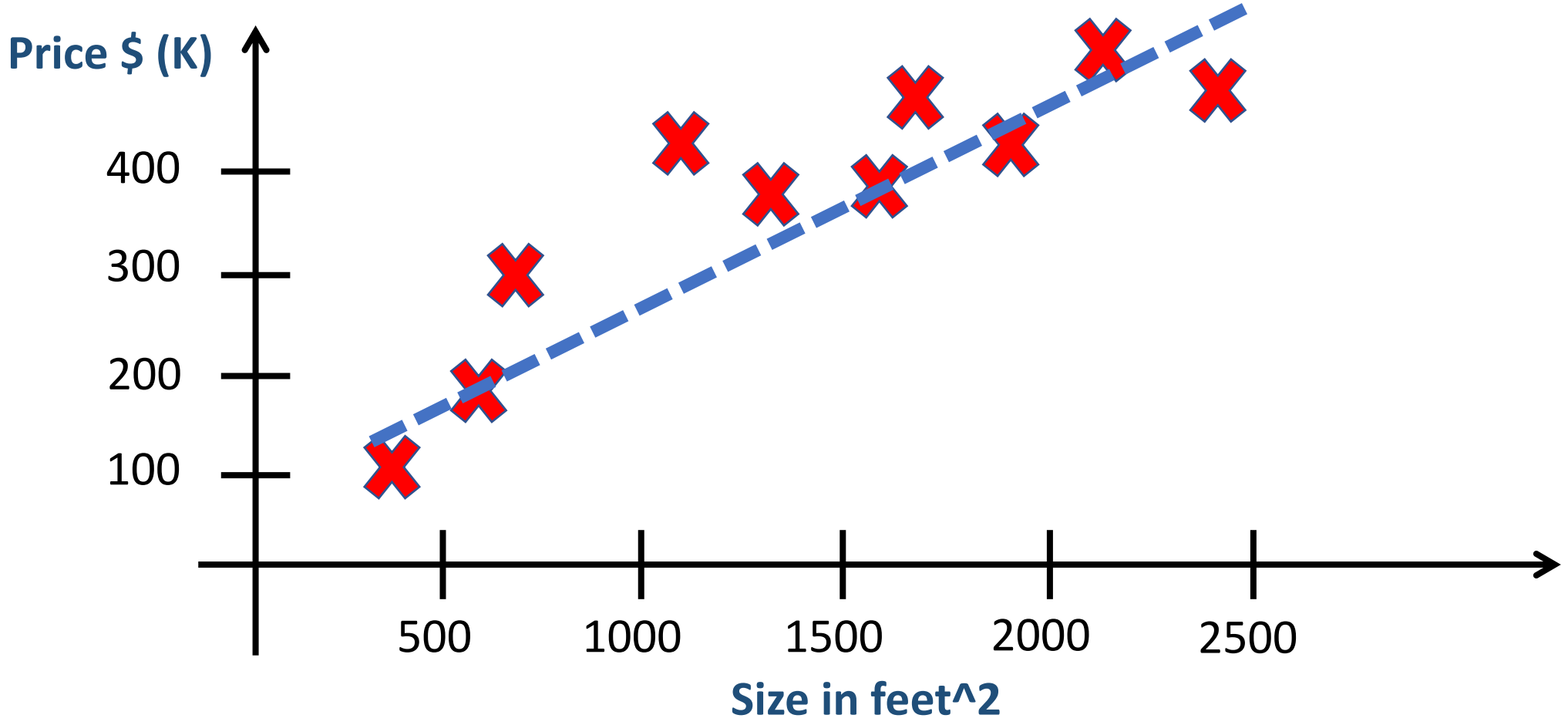
- **Linear Regression Theory**

The term “linearity” in algebra refers to a linear relationship between two or more variables (straight line).

Linear regression performs the task to predict a dependent variable value (y) based on a given independent variable (x). So, this regression technique finds out a linear relationship between x (input) and y(output).



Let's take a look at the data



Training Set

- **Notation:**
  - m = Number of training examples
  - n = Number of features
  - x = Input variable / features
  - y = Output variable / target variable
  - (x, y) = One training example
  - $(x^{(i)}, y^{(i)})$ = i^{th} training example
  - $(x_j^{(i)}$ = Feature j for the i^{th} training example

Size in feet^2 (x)	Price ($) in k (y)
1504	760
1216	632
1034	515
852	378
…	…

$m = 47$

Examples:

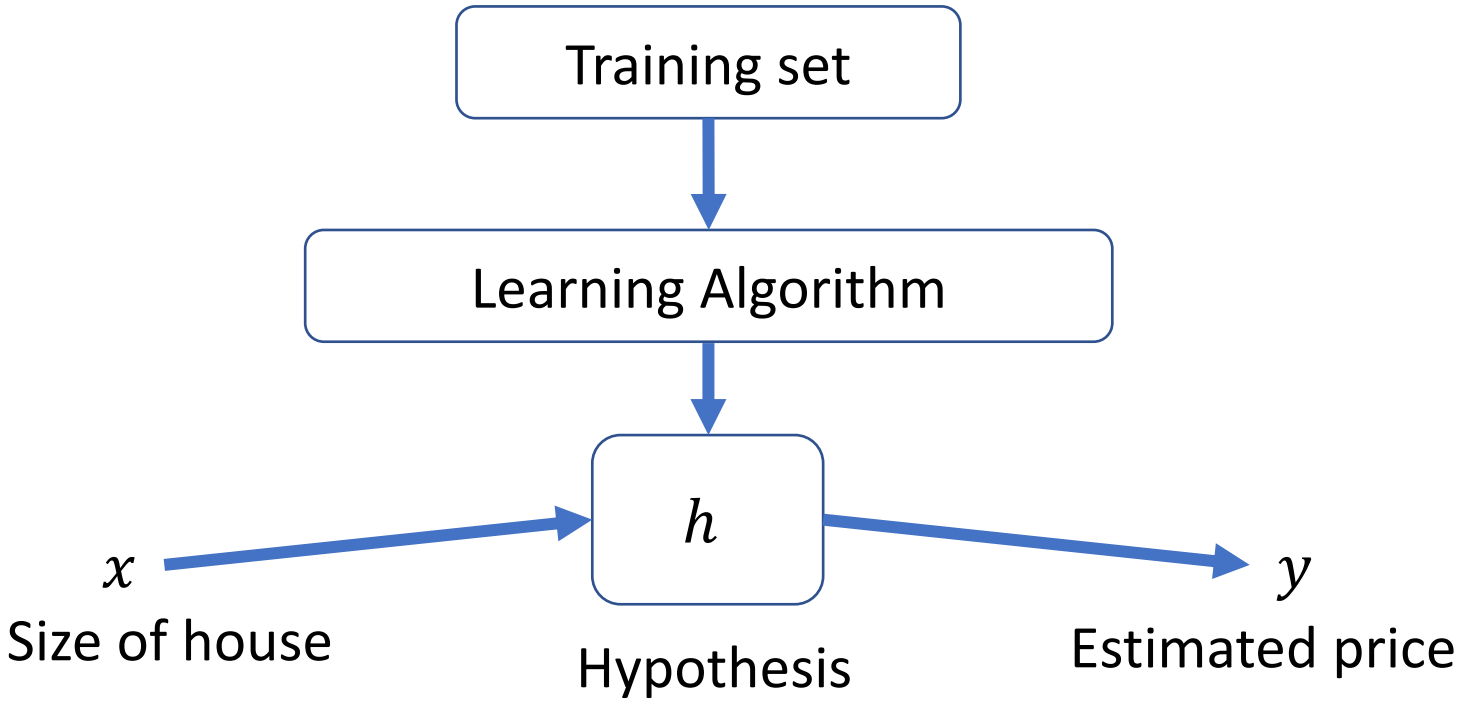
$x^{(1)} = 1504$ $y^{(1)} = 760$

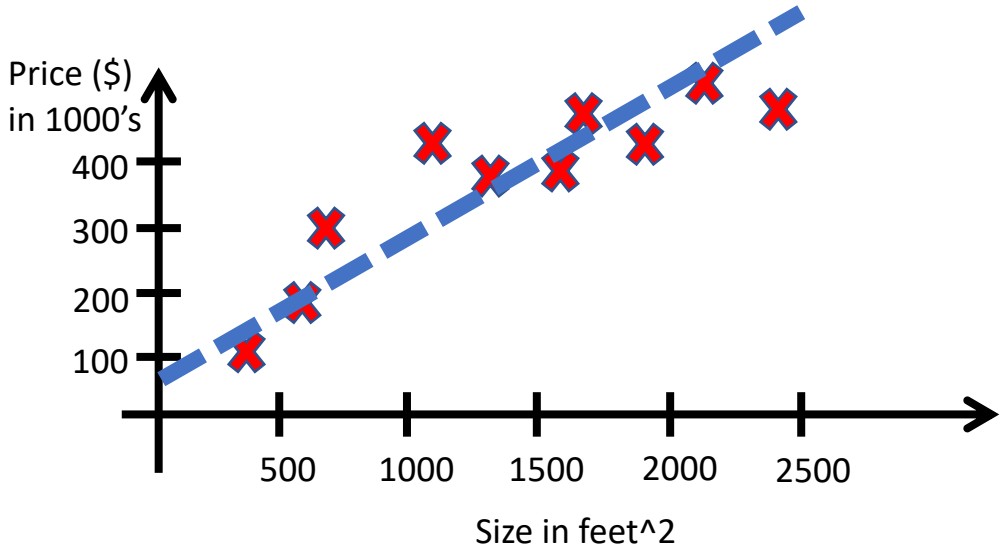
$x^{(2)} = 1216$ $y^{(2)} = 630$

Model Representation

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

Shorthand $h(x)$





Univariate linear regression

Linear Regression

Training Set

- Hypothesis

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

θ_0, θ_1: parameters/weights

How to choose θ_i's?

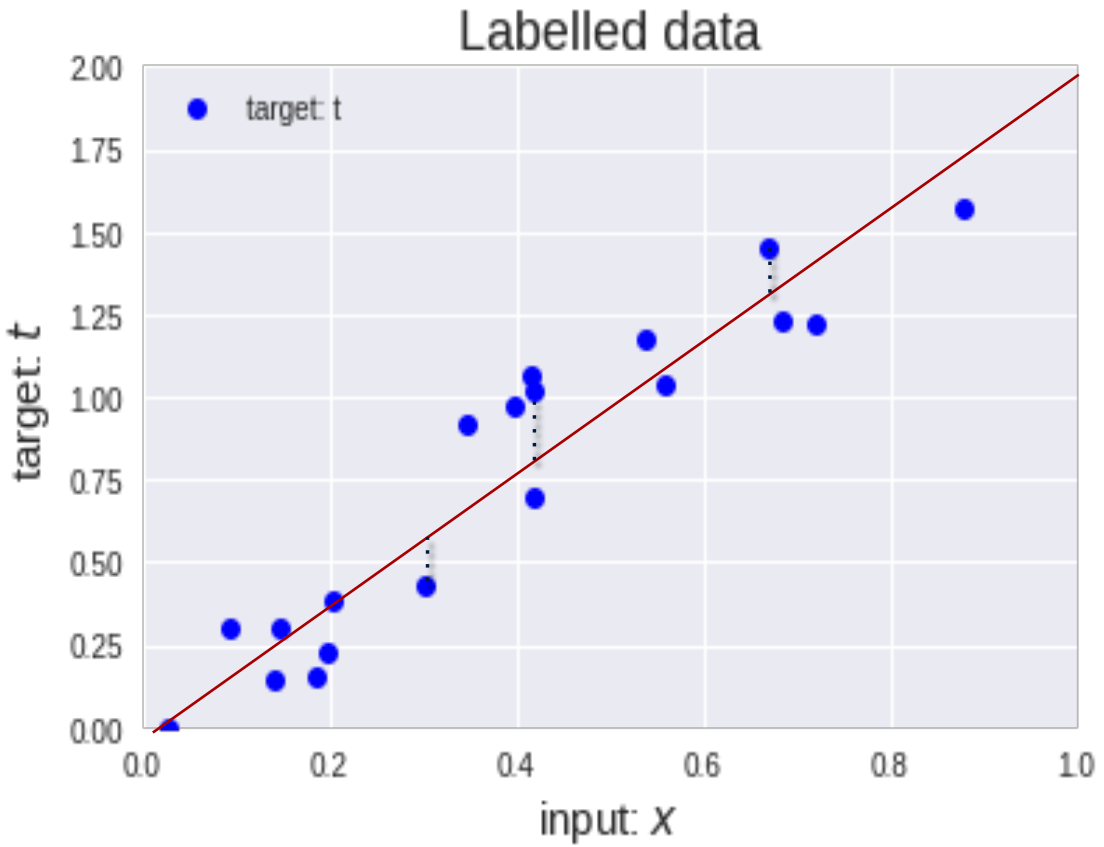
Size in feet^2 (x)	Price ($) in k (y)
2104	460
1416	232
1534	315
852	178
…	…

$m = 47$

Examples:

$$x^{(1)} = 2104$$
$$x^{(2)} = 1416$$
$$y^{(1)} = 460$$

Cost Function / Loss Function / Objective Function



Source: https://bit.ly/2IoAGzL

For this linear regression example, to determine the best *p* (slope of the line) for

$$y = x \cdot p$$

we can calculate the **cost function**, such as Mean Square Error, Mean absolute error, etc.

For this example, we'll use sum of squared absolute differences

$$cost = \sum | t - y |^2$$

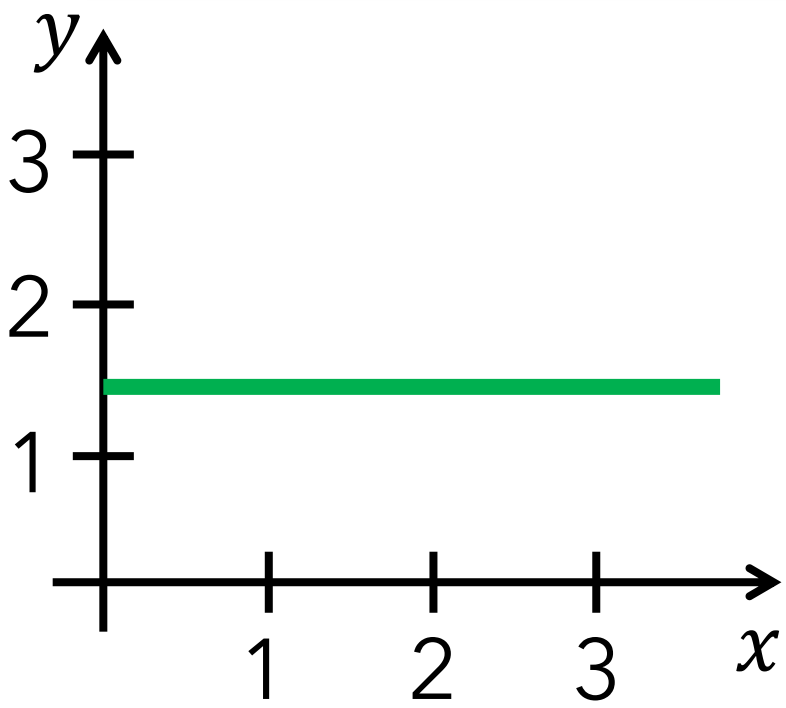
$$h_\theta(x) = \theta_0 + \theta_1 x$$

$y = h_\theta(x) = 2\,x + 1$ => ***Error$_1$=very high***

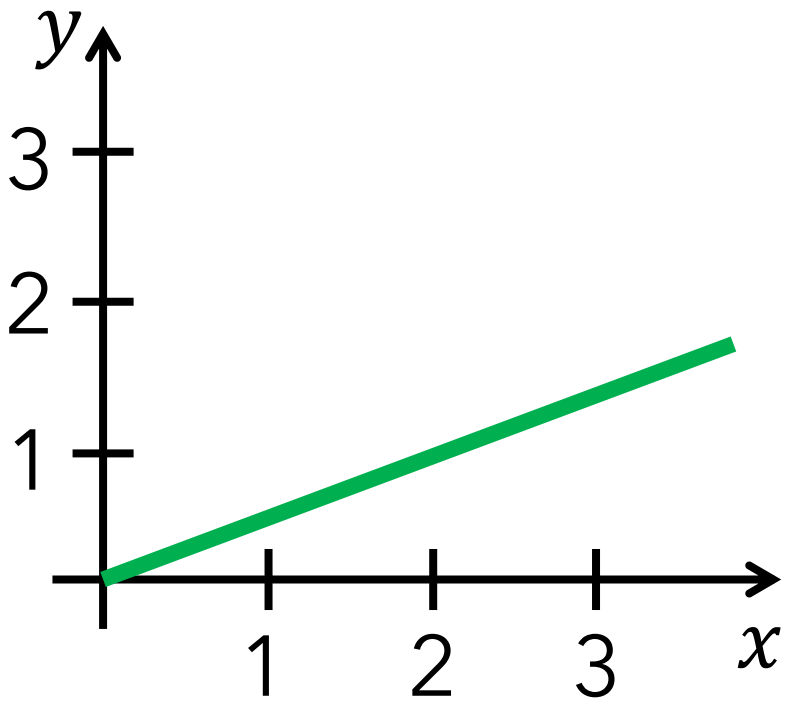
$y = h_\theta(x) = 2.5\,x + 2$ => ***Error$_2$= high***

....

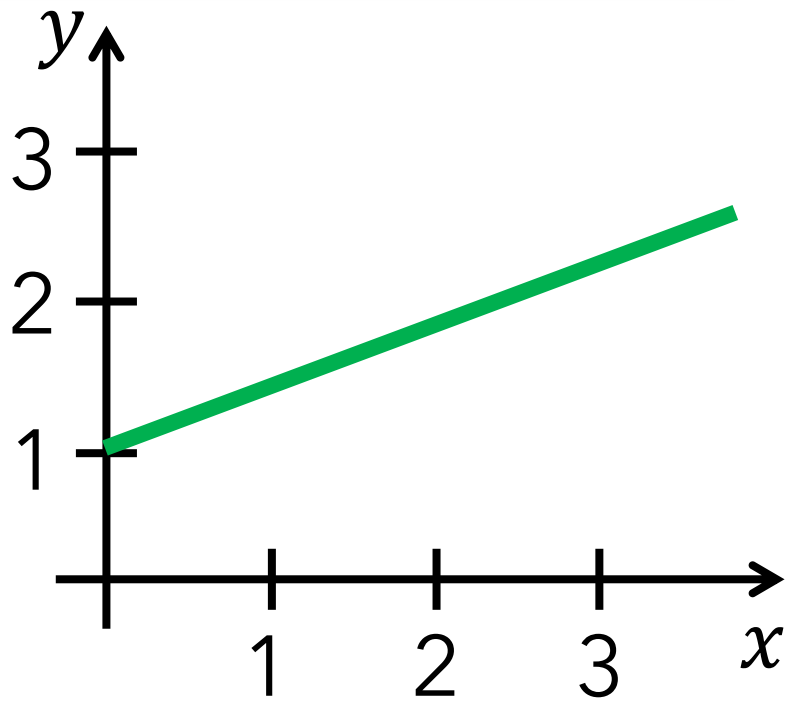
$y = h_\theta(x) = -8.9\,x + 5.5$ => ***Error$_n$= low***



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



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

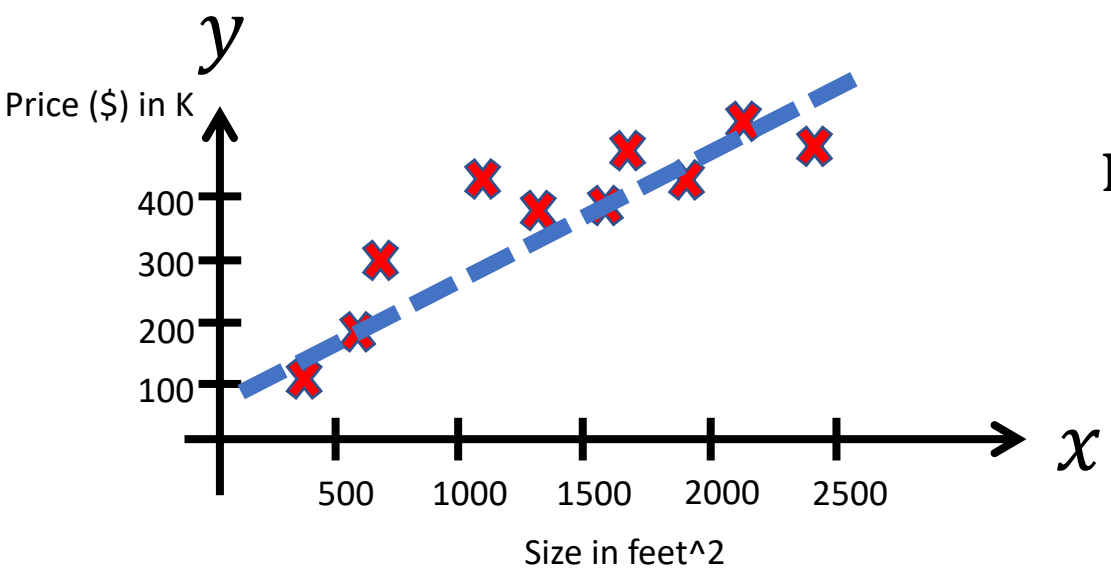


$\theta_0 = 1$
$\theta_1 = 0.5$

Cost Function

- **Idea:**

  Choose θ_0, θ_1 so that $h_\theta(x)$ is close to y for our training example (x, y)



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

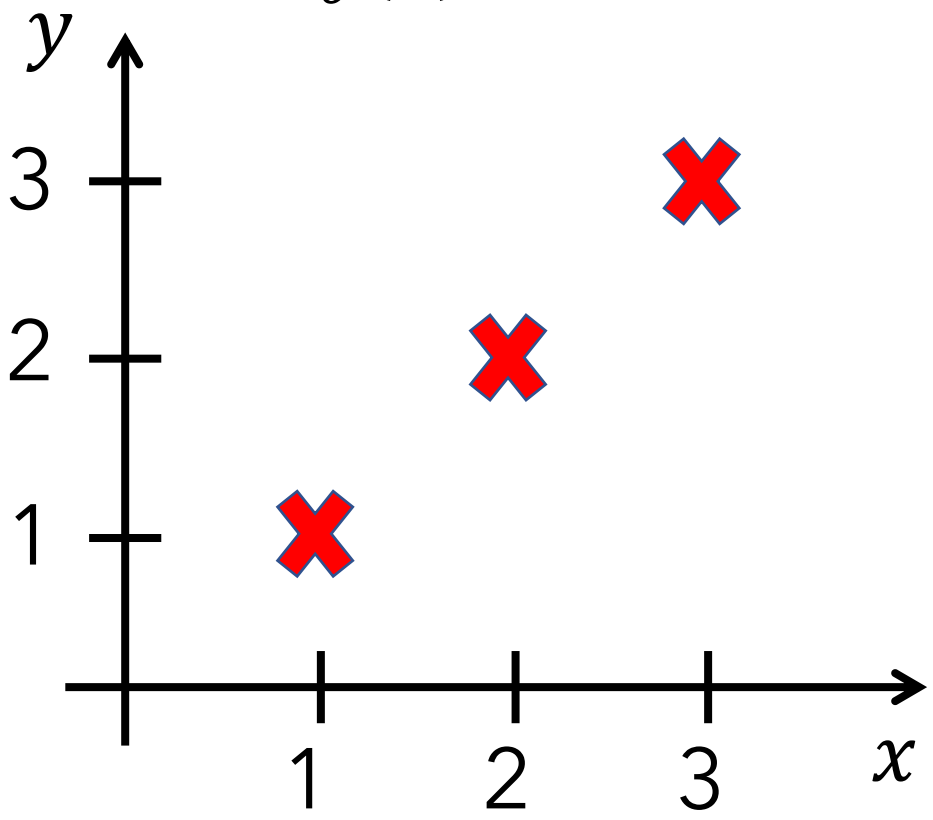
$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2$$

$$\underset{\theta_0, \theta_1}{\text{minimize}} \; J(\theta_0, \theta_1)$$ **Cost function**

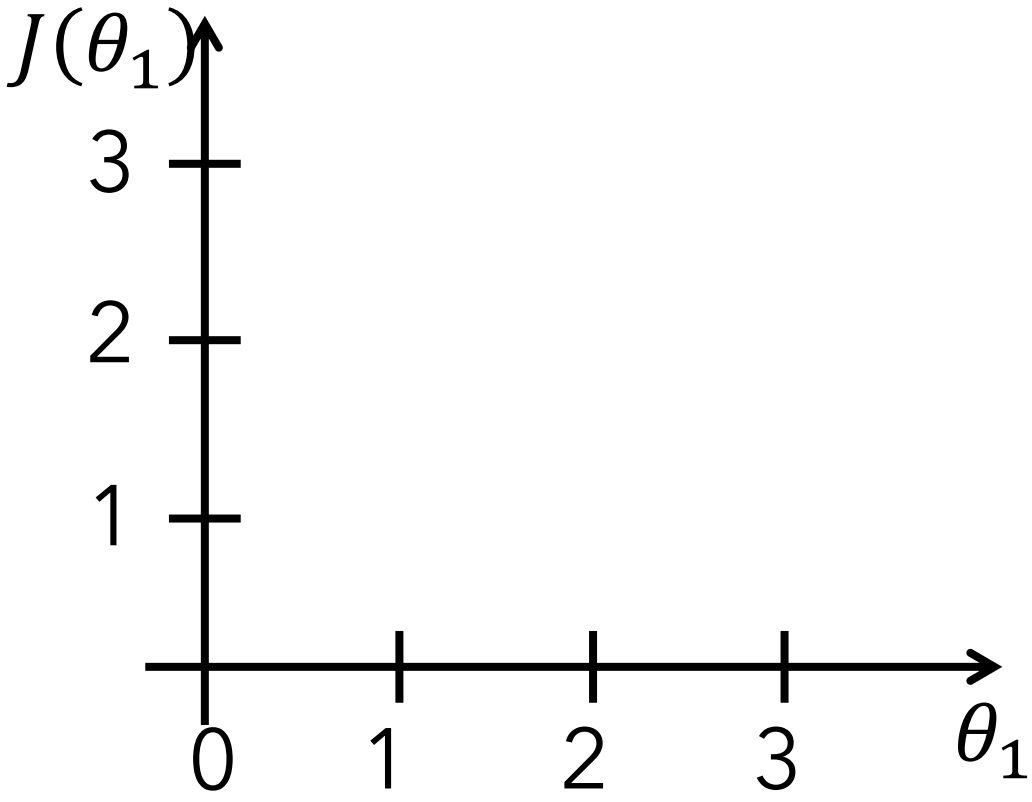
$$\underset{\theta_0, \theta_1}{\text{minimize}} \; \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2$$

Cost Function (Simplified)

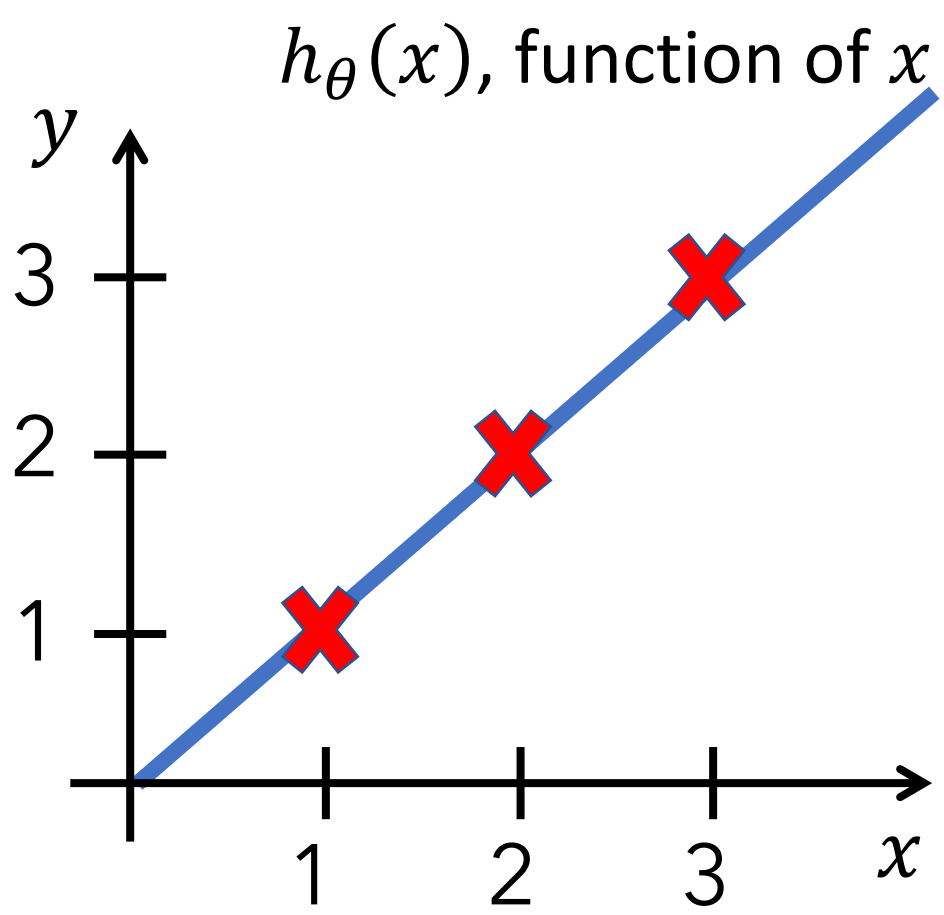
$h_\theta(x)$, function of x

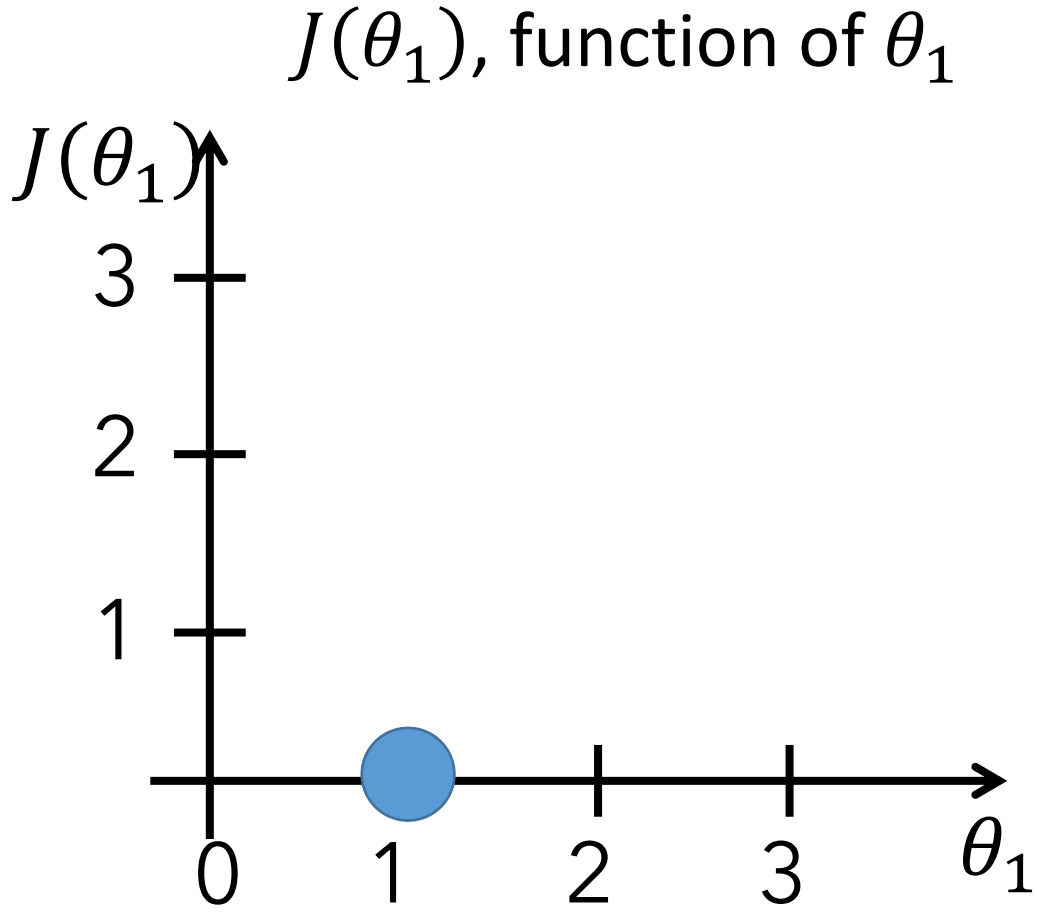


$J(\theta_1)$, function of θ_1

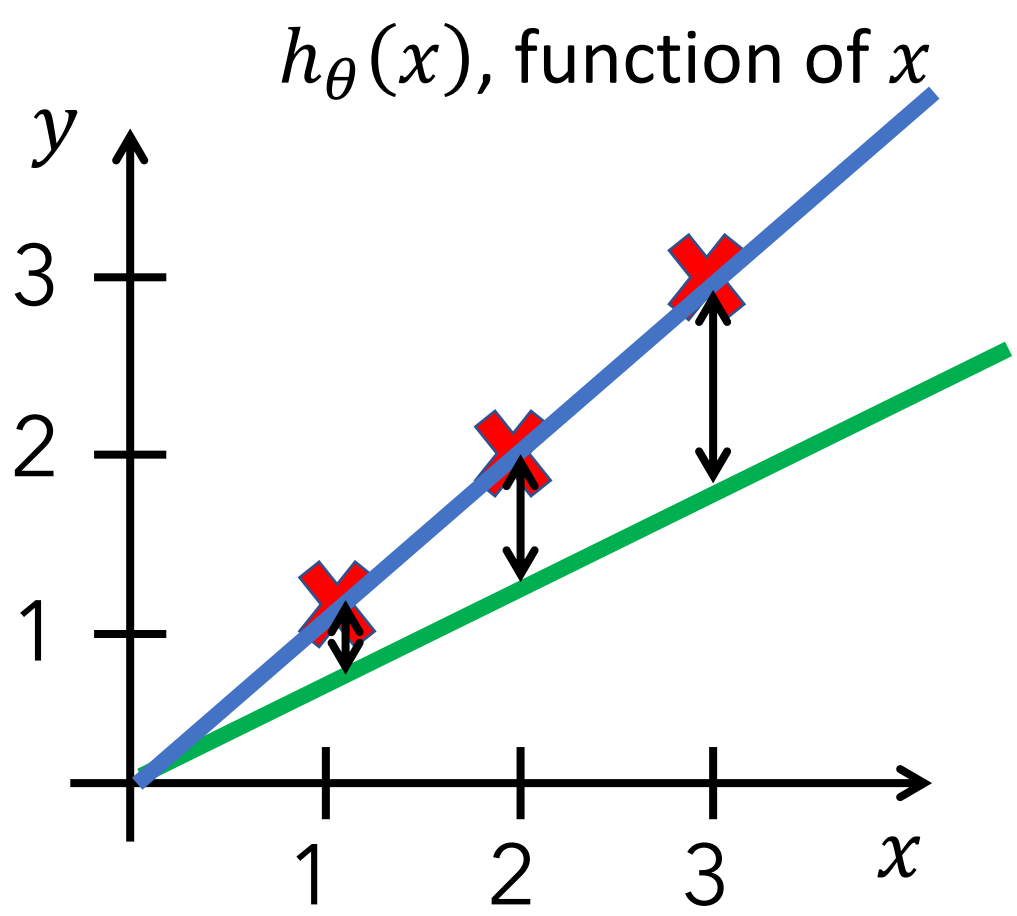


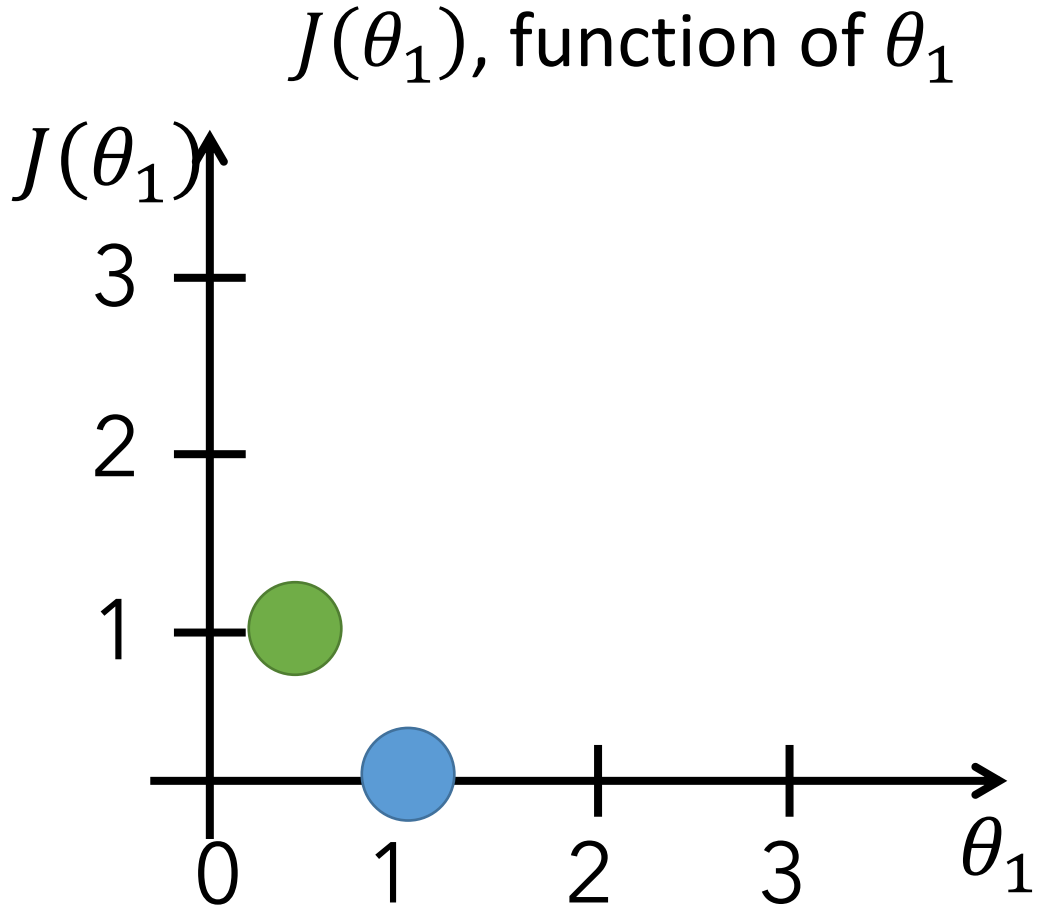
Cost Function (Simplified)



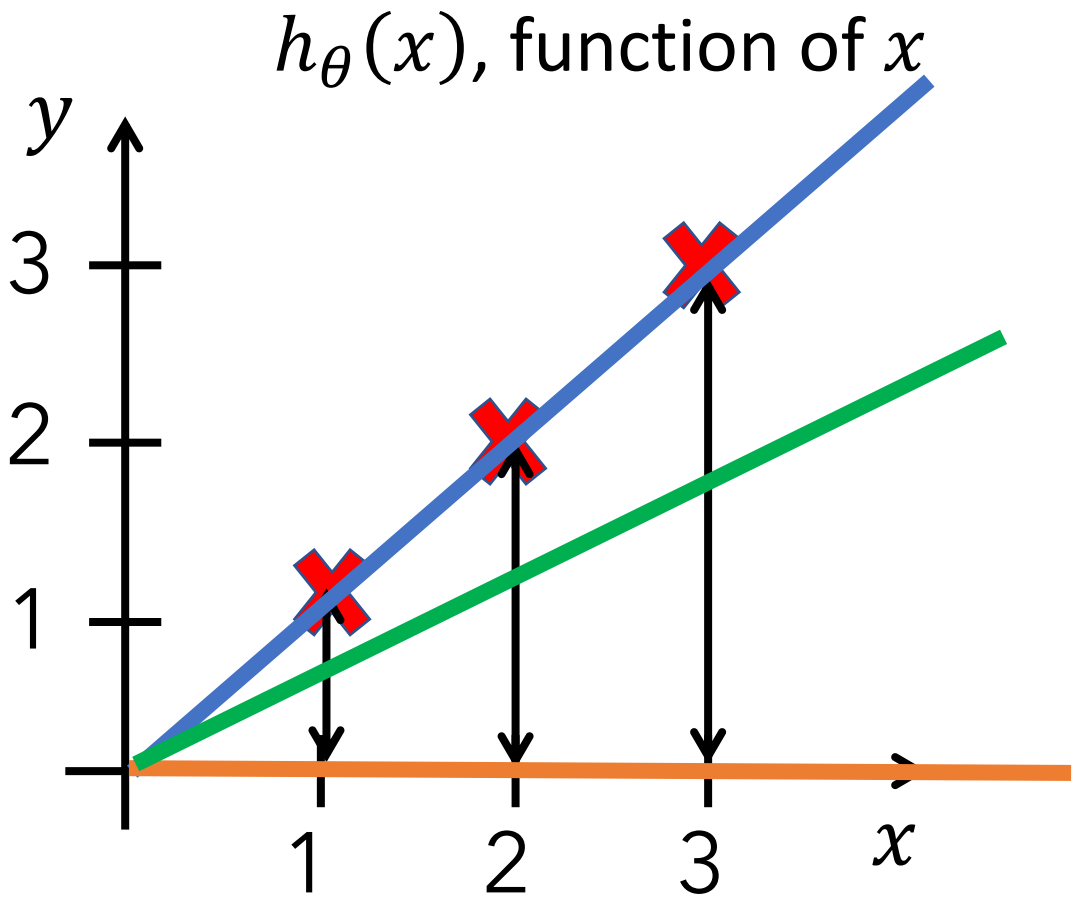


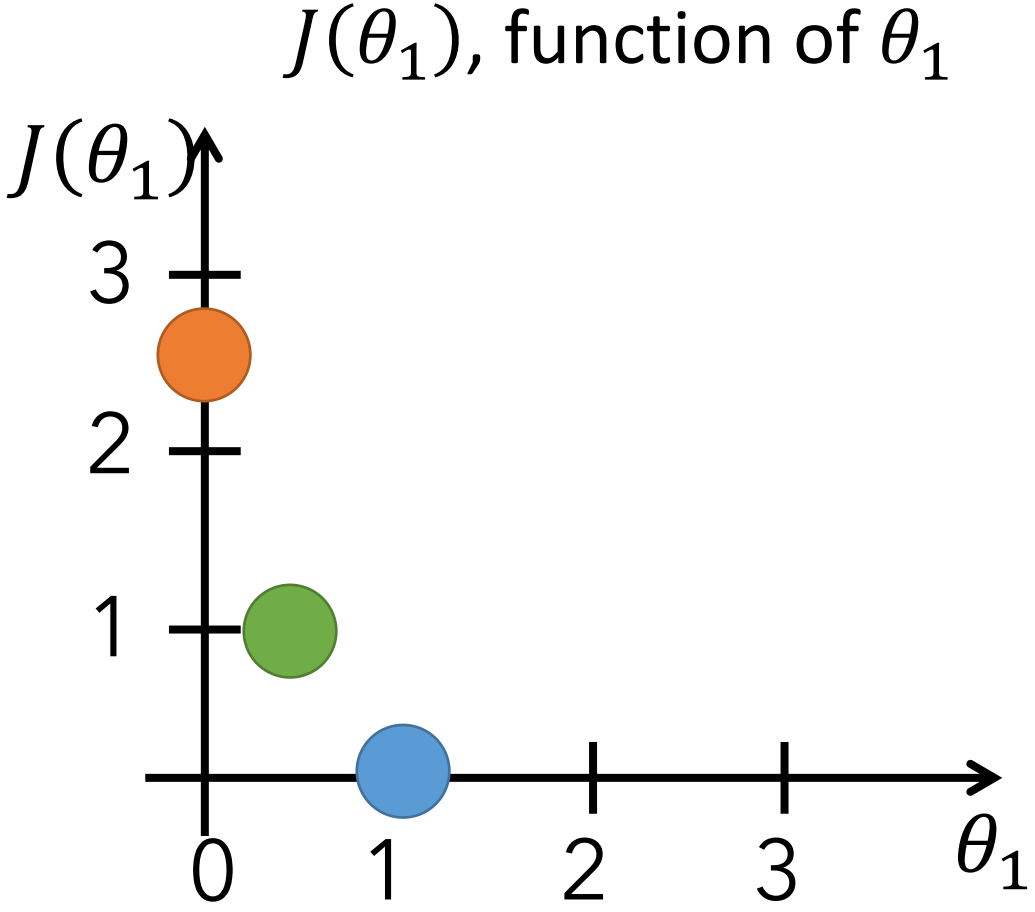
Cost Function (Simplified)



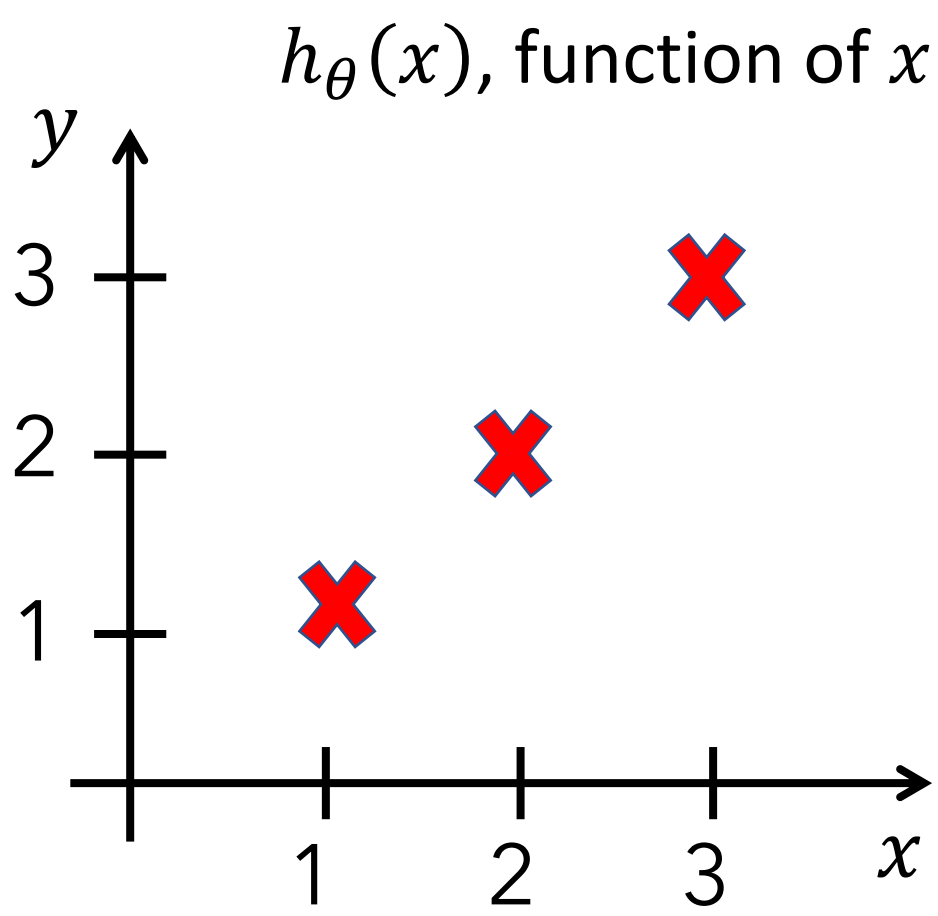


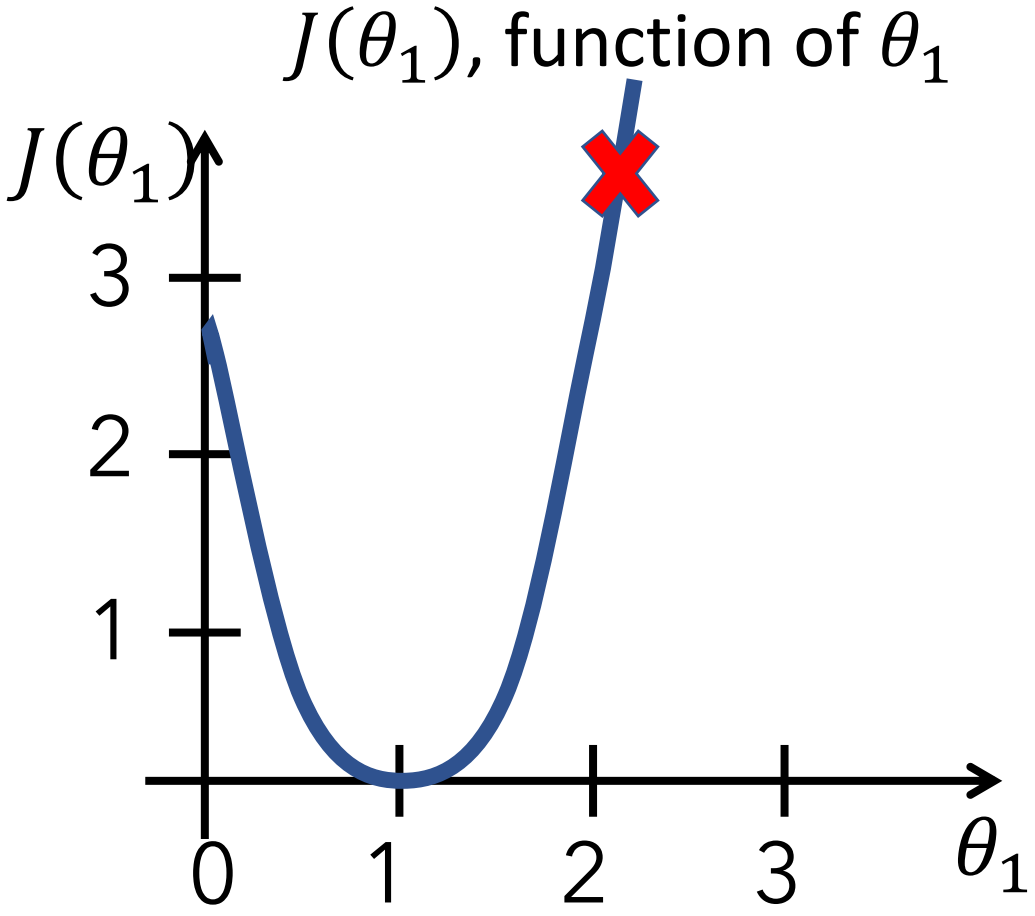
Cost Function (Simplified)



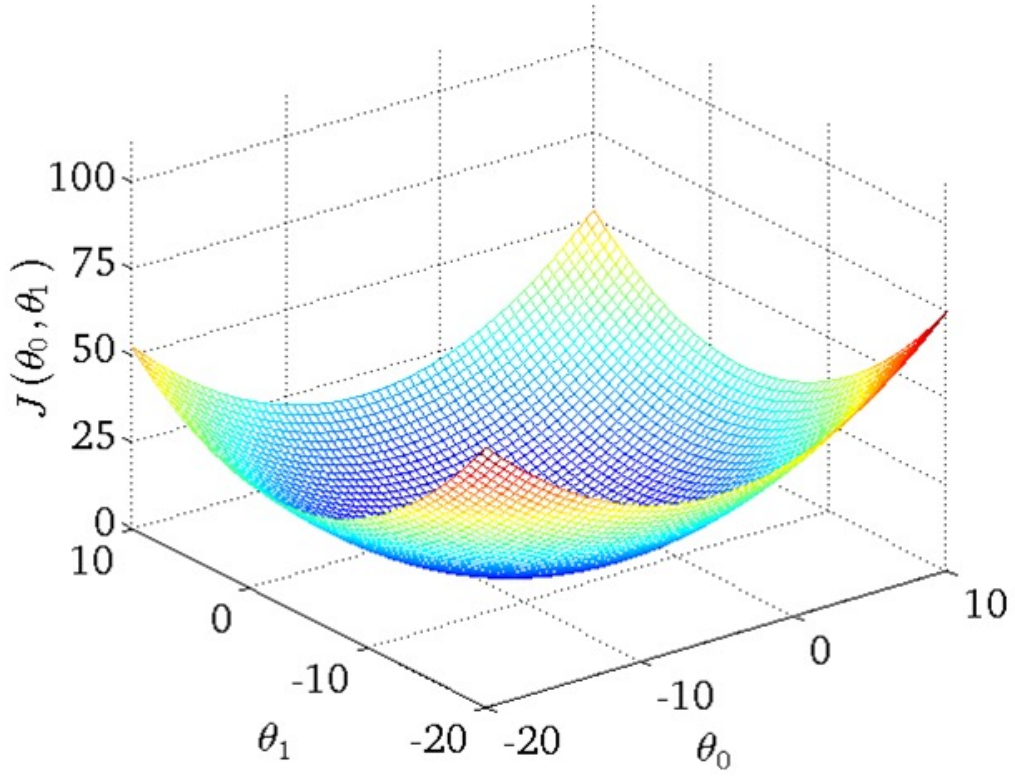


Cost Function (Simplified)





Cost Function (Simplified)

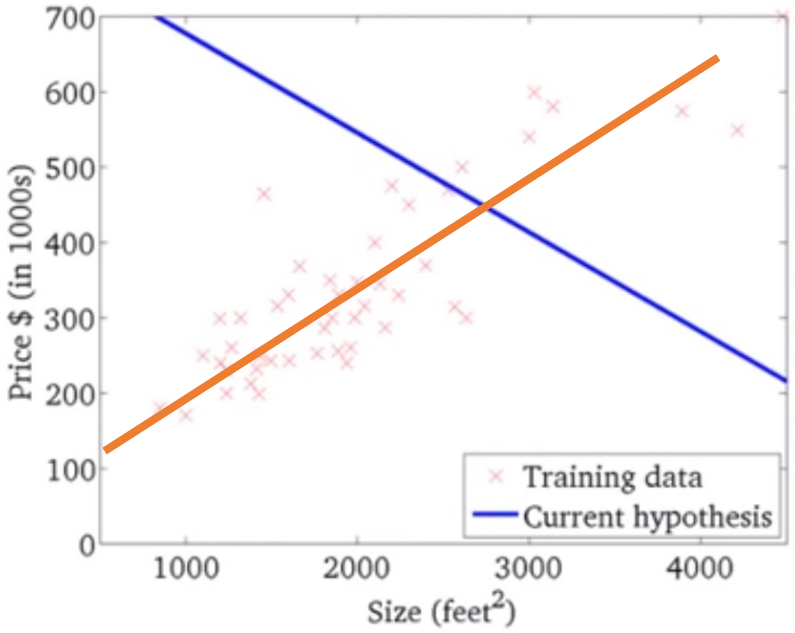


Cost Function (Simplified)

How do we find good θ_0, θ_1 that minimize $J(\theta_0, \theta_1)$?

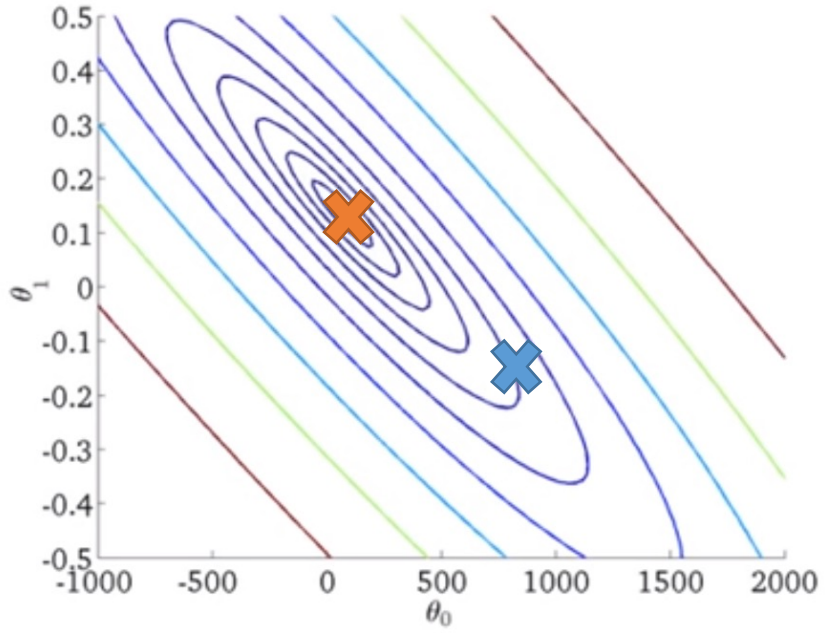
$h_\theta(x)$

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

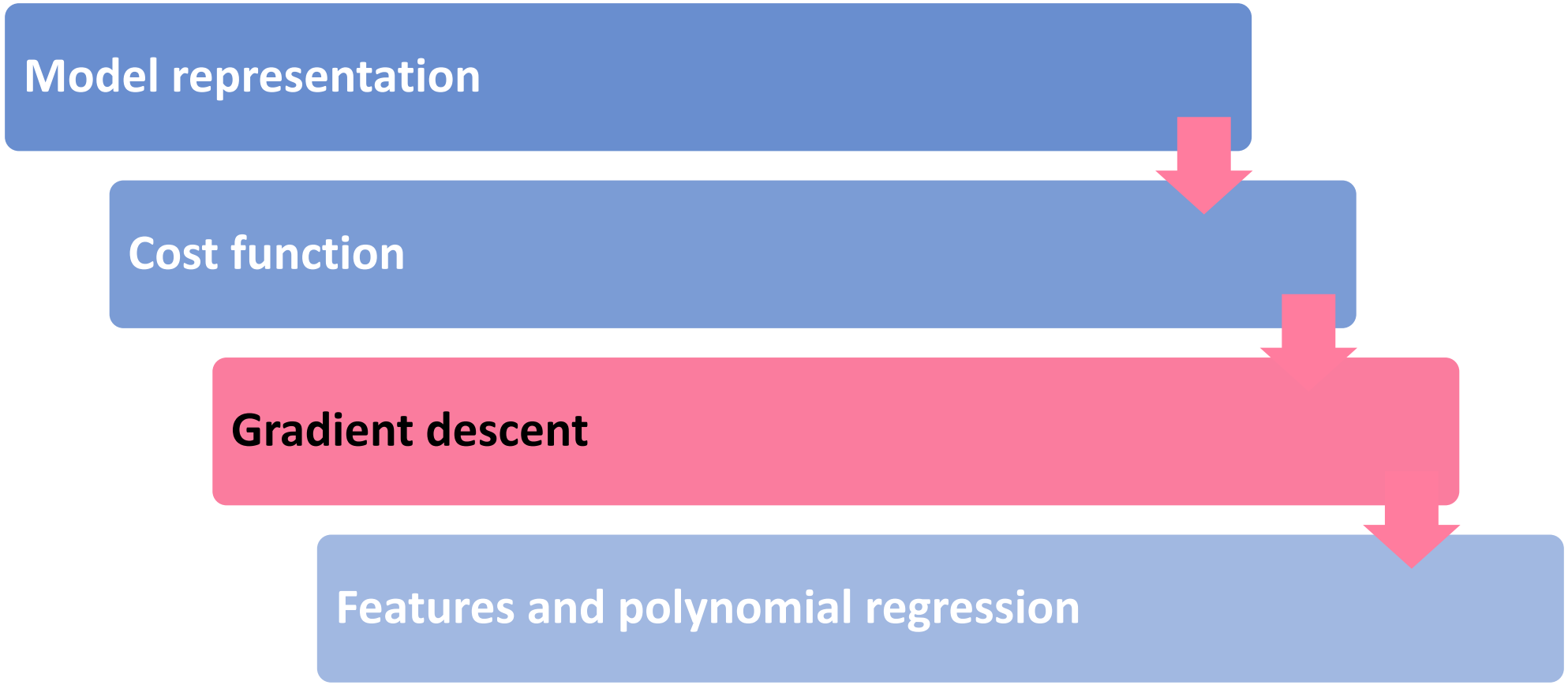


$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)



Linear Regression



Gradient Decent

Arg max
In mathematics, the arguments of the maxima are the points, or elements, of the domain of some function at which the function values are maximized.

Have some function $J(\theta_0, \theta_1)$

Want $\underset{}{\text{argmin}}\ J(\theta_0, \theta_1)$

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

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 minimum

Cost Function (Simplified)

- **Hypothesis:**

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

- **Paramecters:**

  θ_0, θ_1

- **Cost function:**

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

- **Goal**:
  minimize $J(\theta_0, \theta_1)$

- **Hypothesis:**

  $h_\theta(x) = \theta_1 x$ $\quad\quad$ $\theta_0 = 0$

- **Parameters:**

  θ_1

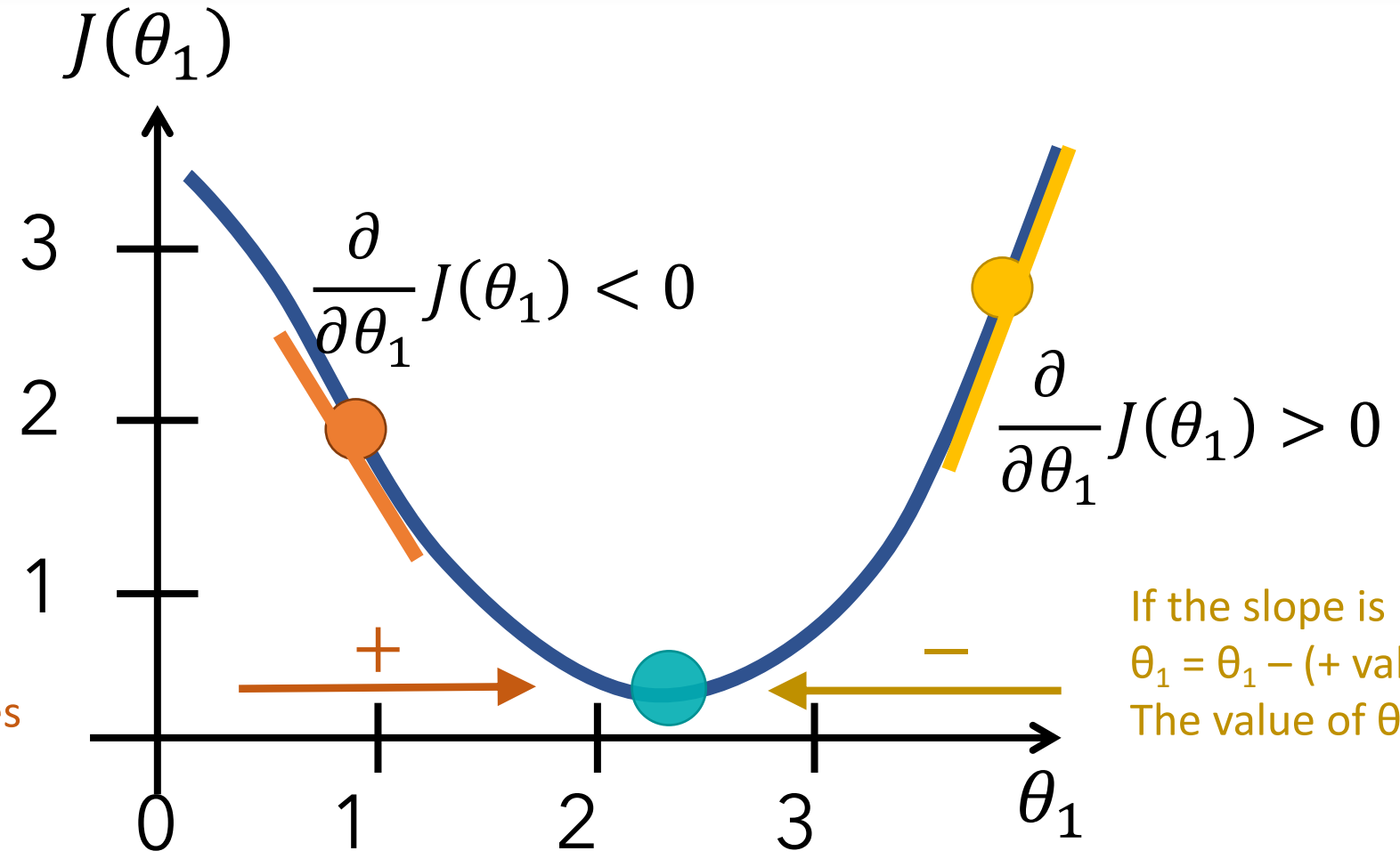
- **Cost function:**

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

- **Goal**:
  minimize $J(\theta_1)$

Gradient Decent

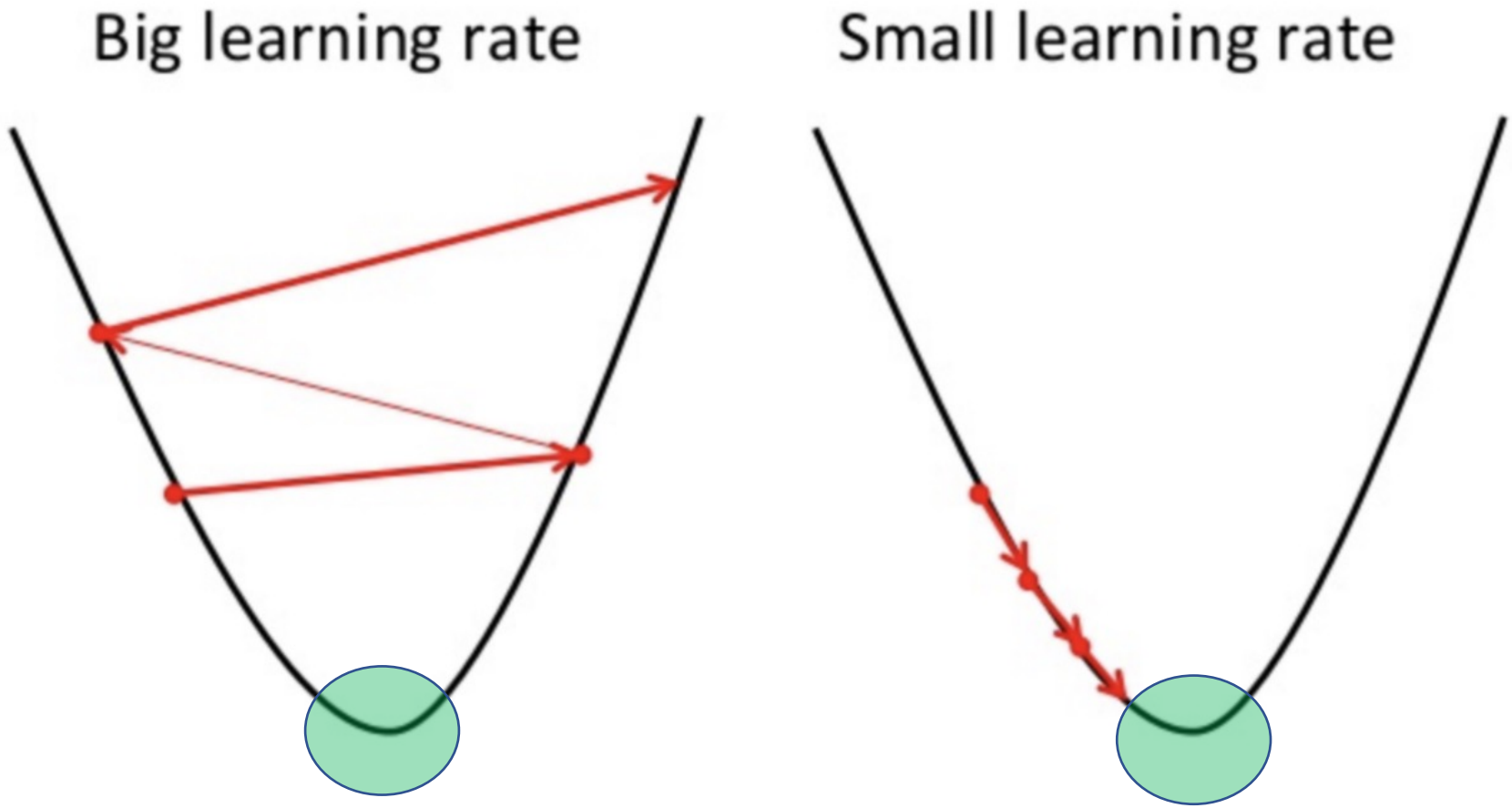
$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$



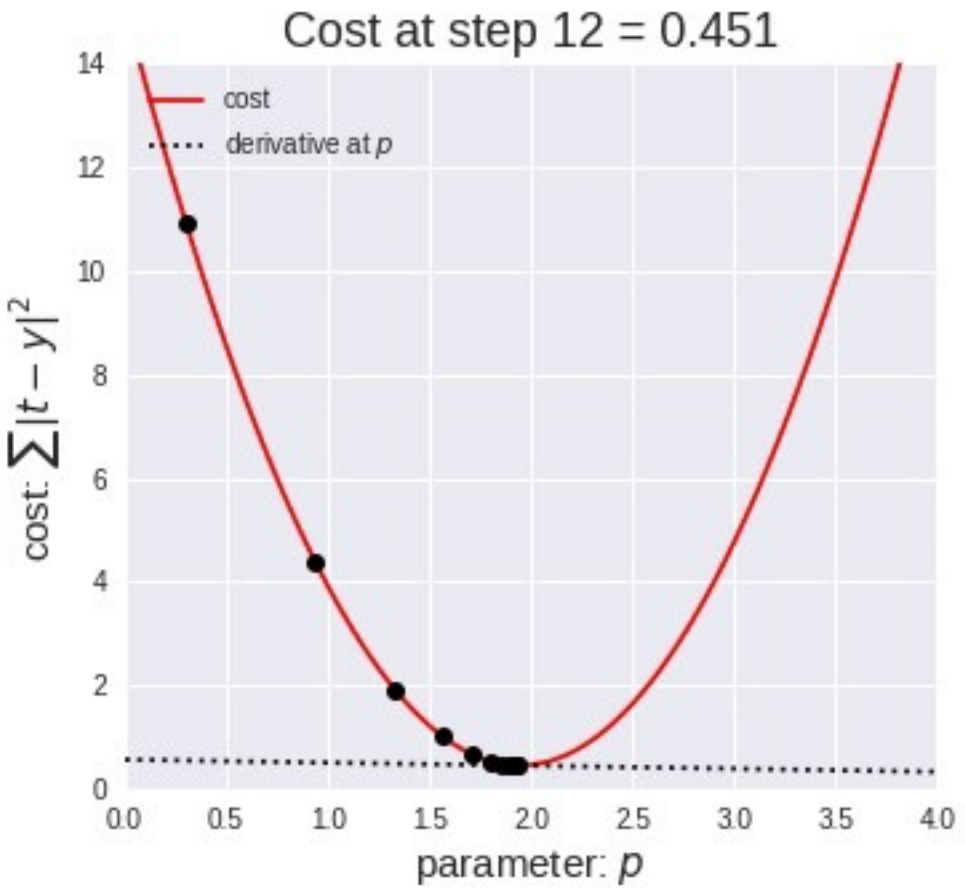
If the slope is negative,
$\theta_1 = \theta_1 - (-\text{value})$
The value of θ_1 increases

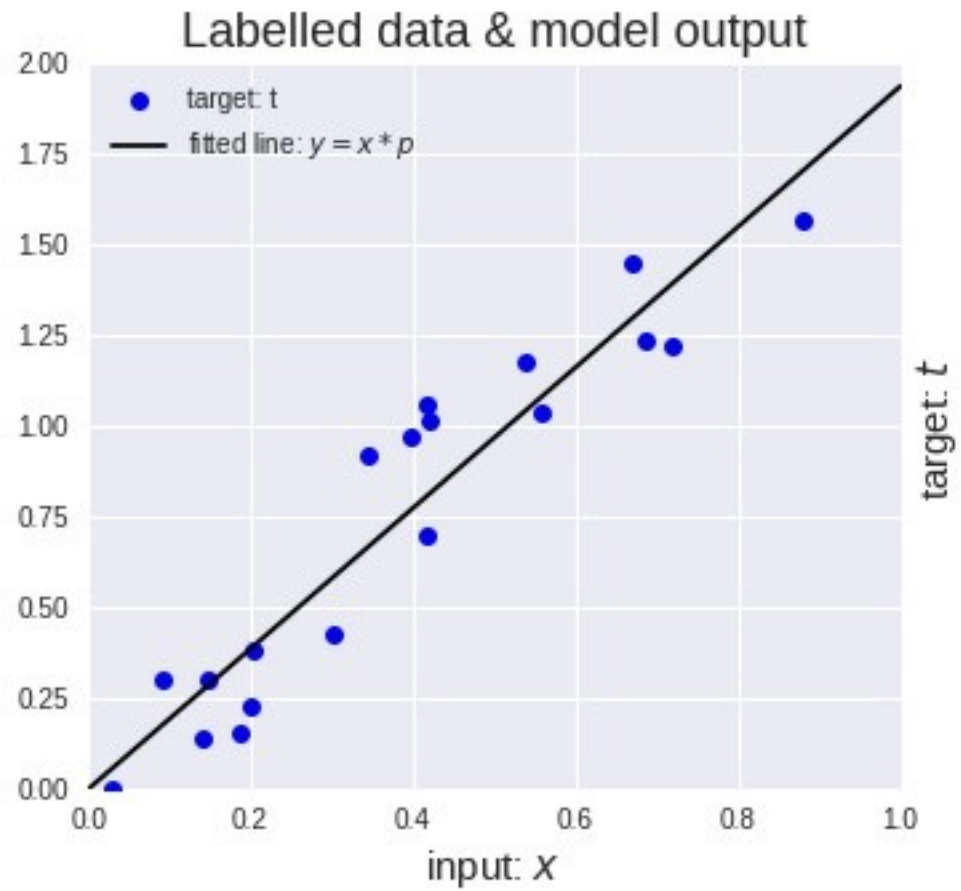
If the slope is positive,
$\theta_1 = \theta_1 - (+\text{value})$
The value of θ_1 decreases

Gradient Decent



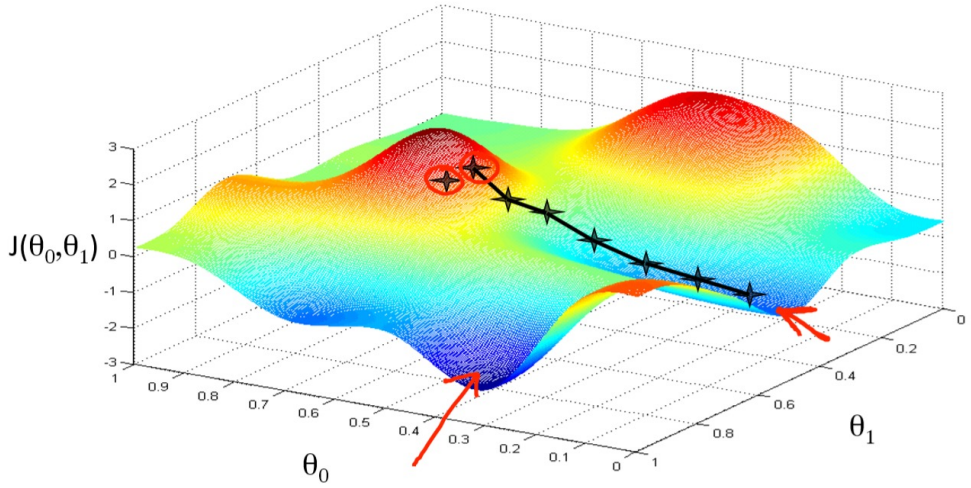
Gradient Decent

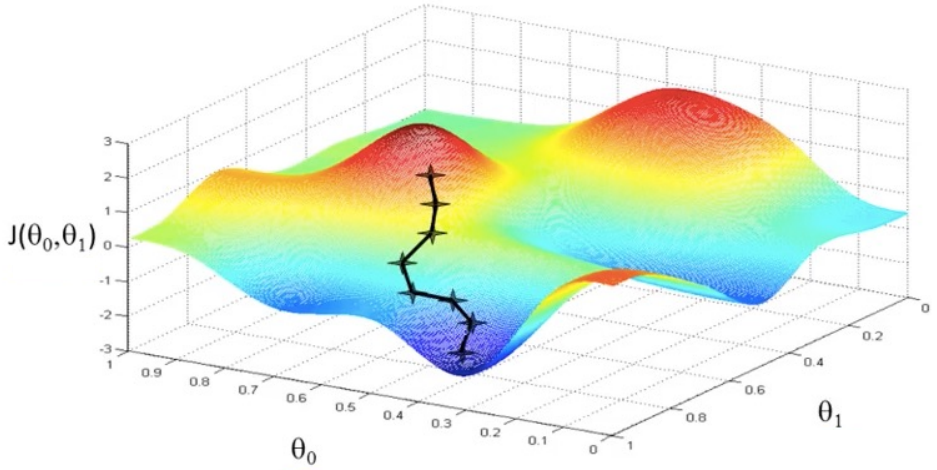




Source: https://bit.ly/2IoAGzL

Gradient Decent





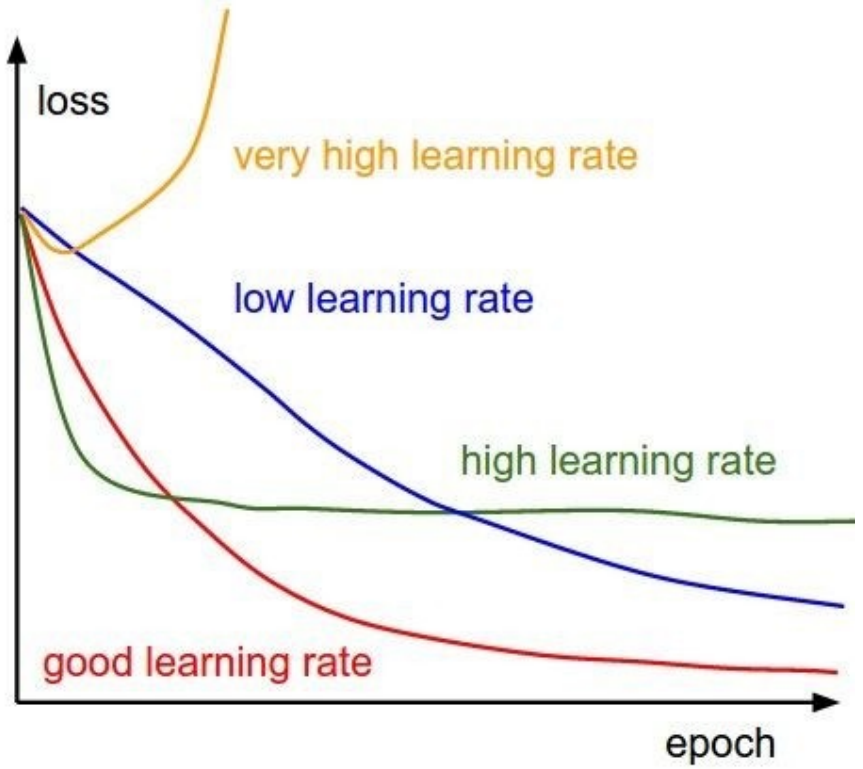
Gradient Decent

- We'll update the weights
- Move in direction opposite to gradient:

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla E(\mathbf{w}^{(\tau)})$$

Time

Learning rate

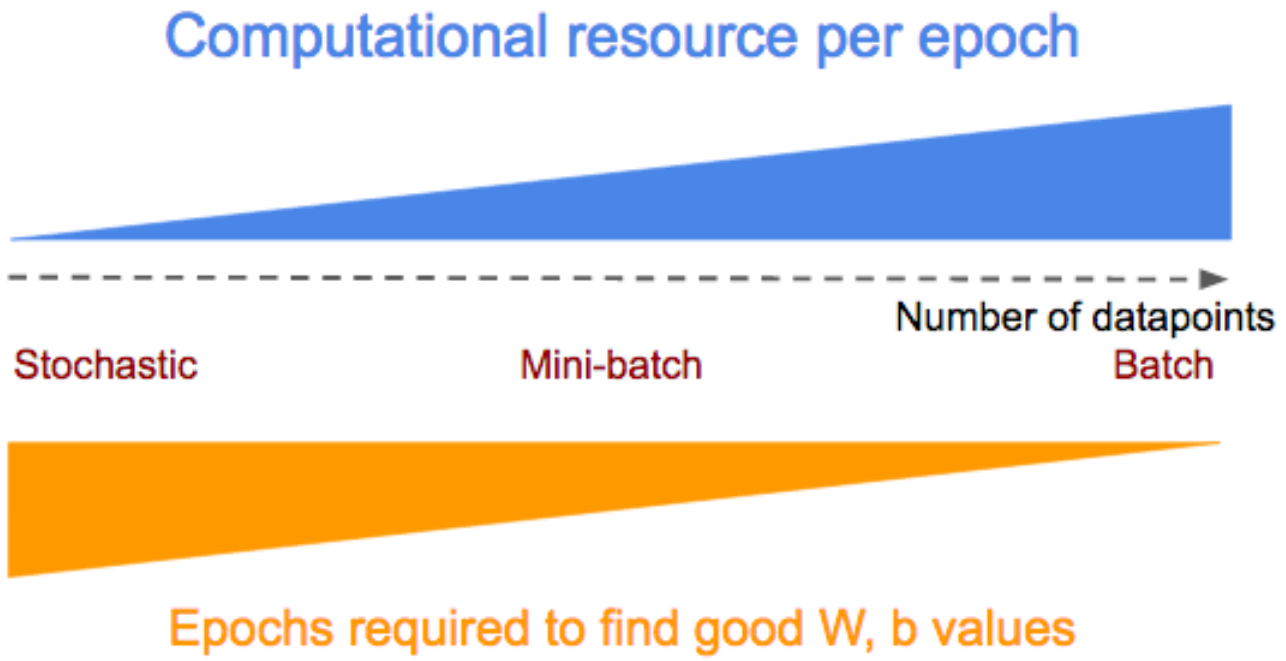


GD variants

In **Gradient Descent** or Batch **Gradient Descent**, we use the whole training data per epoch.

In **Stochastic Gradient Descent**, we use only single training example per epoch.

Finally, Mini-batch **Gradient Descent** lies in between of these two extremes, in which we can use a mini-batch(small portion) of training data per epoch.



Linear Regression vs Neural Network

Simple Linear Regression

$$y = b_0 + b_1 x_1$$

Multiple Linear Regression

$$y = b_0 + b_1 x_1 + b_2 x_2 + \ldots + b_n x_n$$

Polynomial Linear Regression

$$y = b_0 + b_1 x_1 + b_2 x_1^2 + \ldots + b_n x_1^n$$

Multiple Features

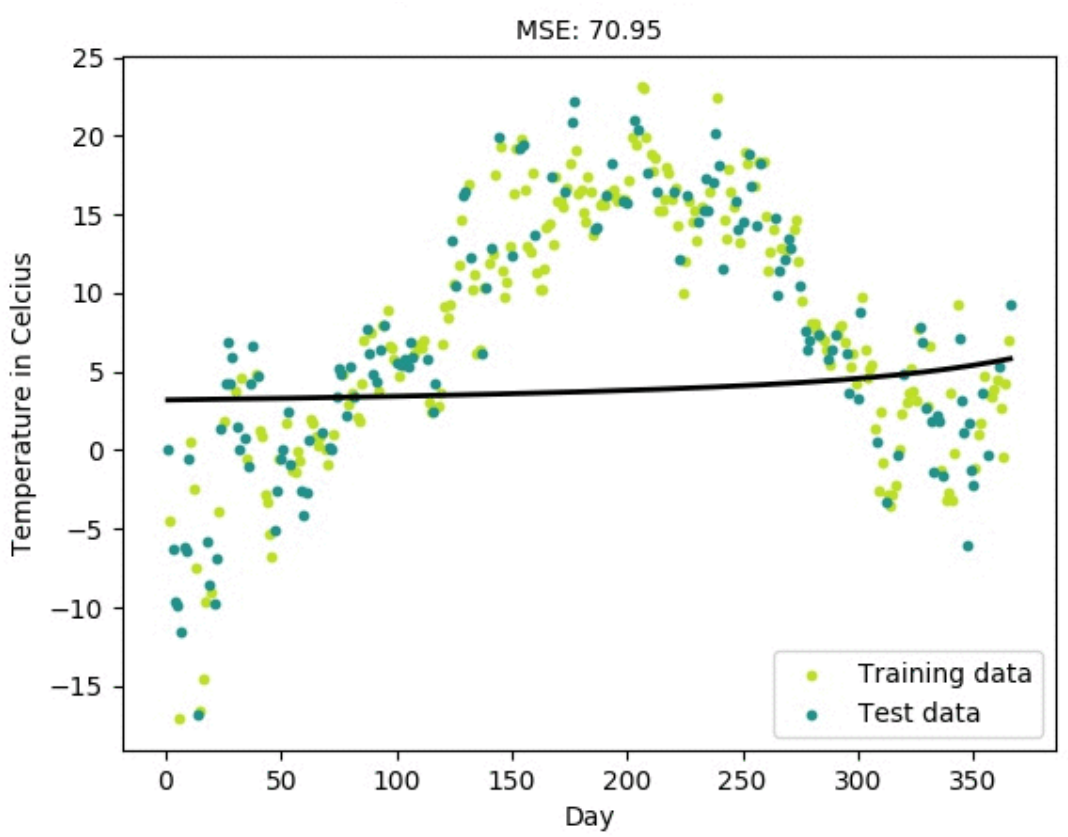
Size in feet^2 (x_1)	Number of bedrooms (x_2)	Number of floors (x_3)	Age of home (years) (x_4)	Price ($) in 1000's (y)
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	1	36	178
…				…

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

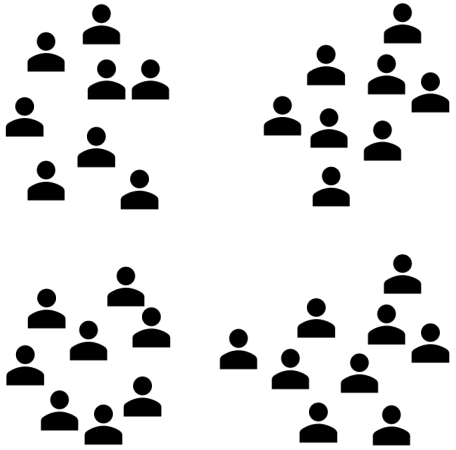
Now: $h_\theta(x) = \boldsymbol{\theta_0} + \boldsymbol{\theta_1} x_1 + \boldsymbol{\theta_2} x_2 + \boldsymbol{\theta_3}\, x_3 + \boldsymbol{\theta_4} x_4$

Area in feet ^{2} — Number of bedrooms — Location — Number of bathroom

Polynomial Regression (During training)



Clustering Applications (Customer segmentation)



Who is your:

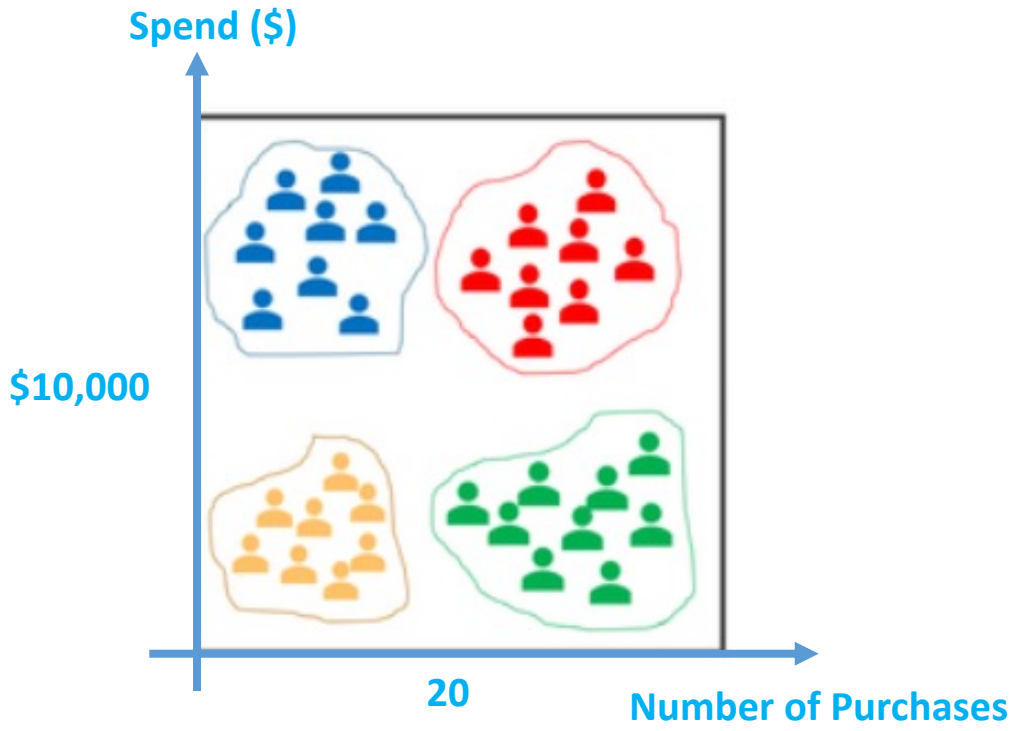
Most valued customer?

At the risk of churn?

Will be interested in a particular promotion/product?

……

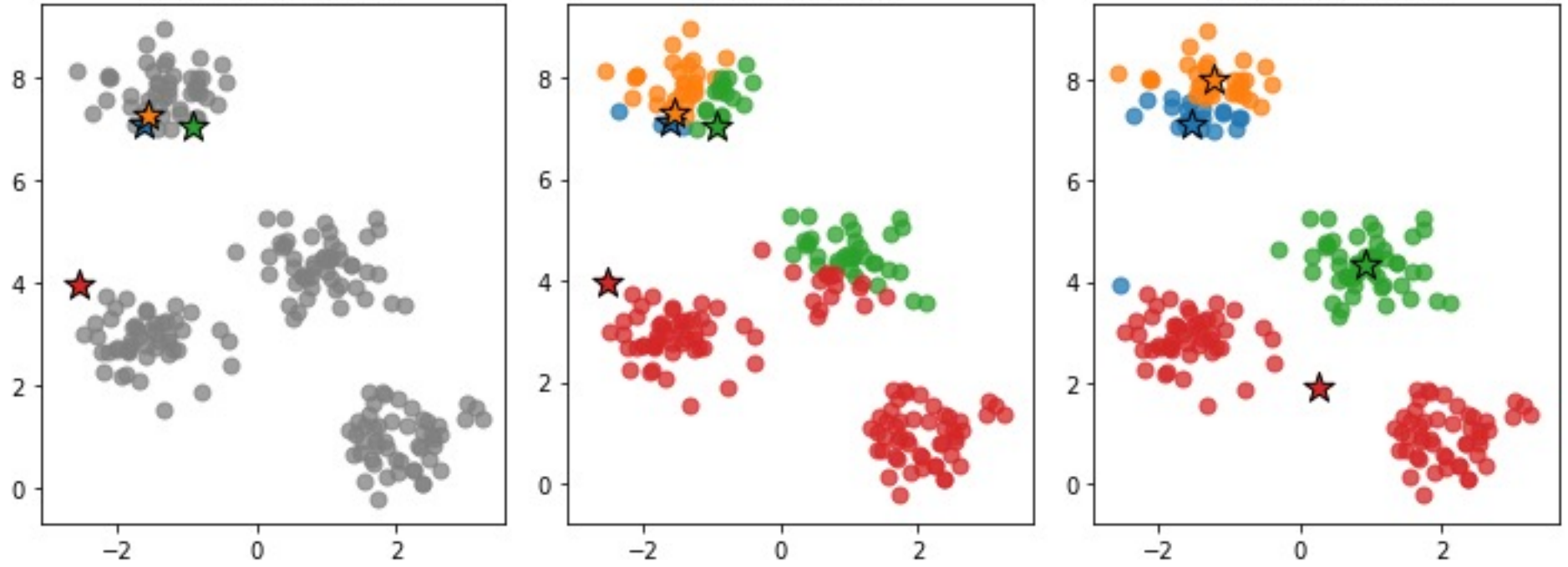
Unsupervised Learning (clustering)



Sell specific products to the targeted group of audience

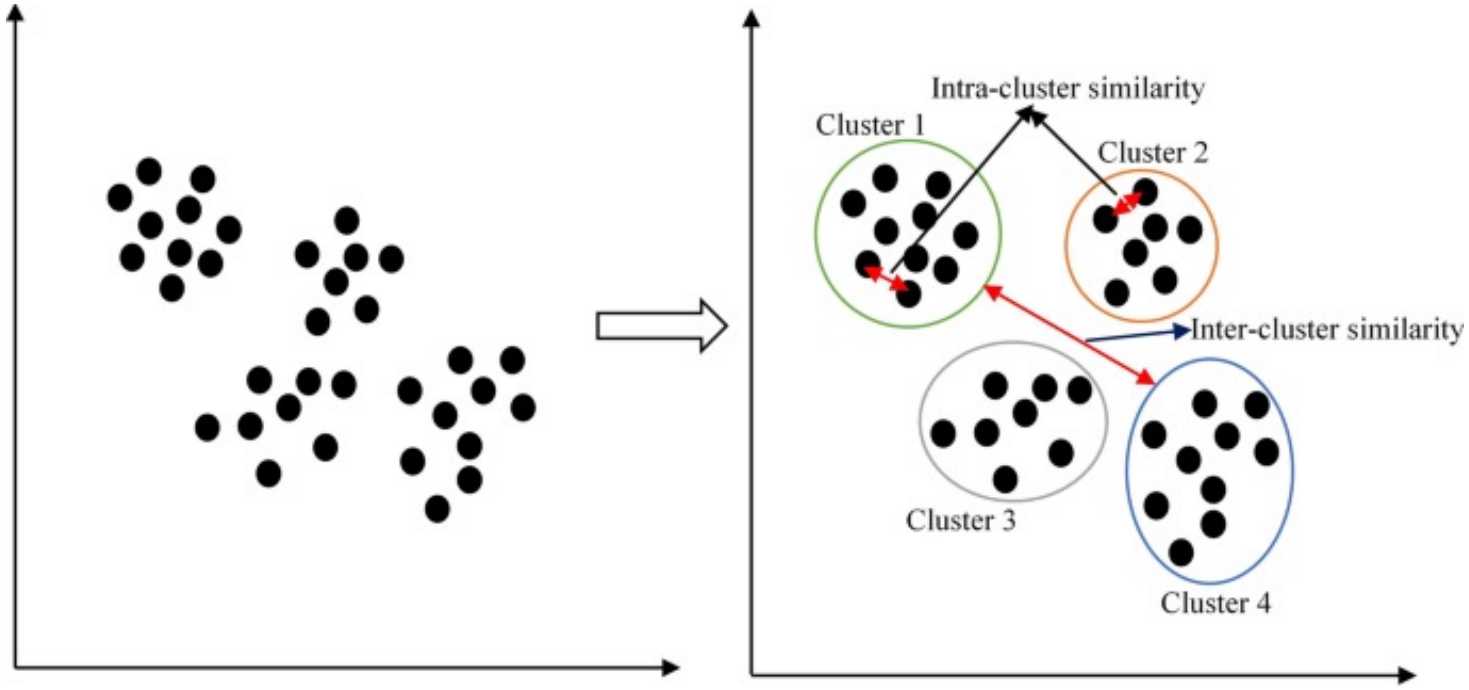
What makes a GOOD clustering?

Is this clustering “good”?



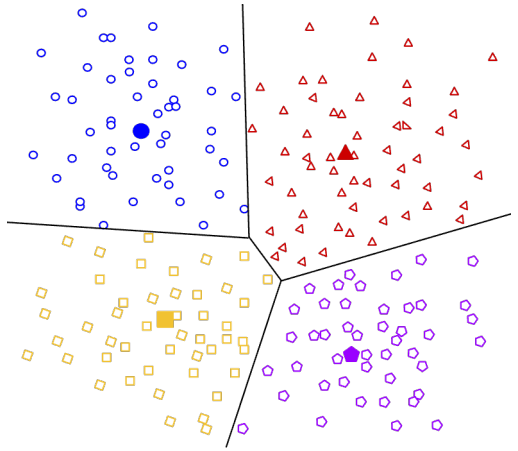
What makes a good clustering?

- Organizing data into clusters such that there is:
  - high intra-cluster similarity (more distinct clusters)
  - low inter-cluster similarity (better association)

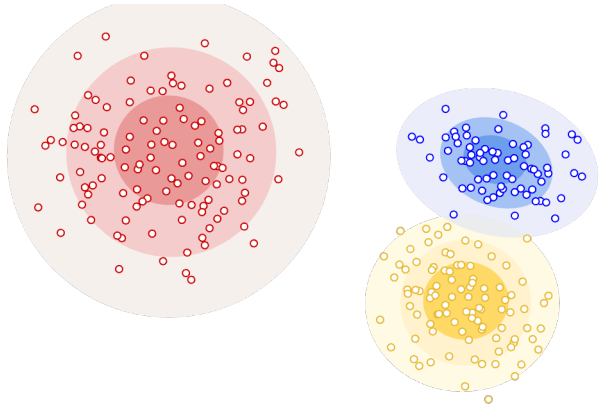


Types of Clustering Methods

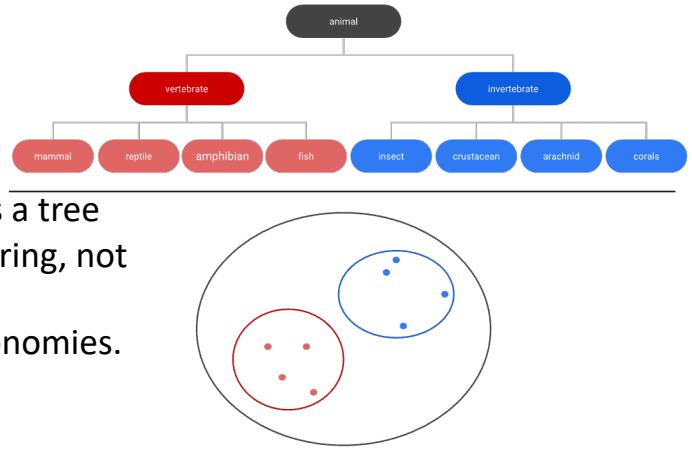
Centroid-based Clustering:
k-means is the most widely-used centroid-based clustering algorithm.



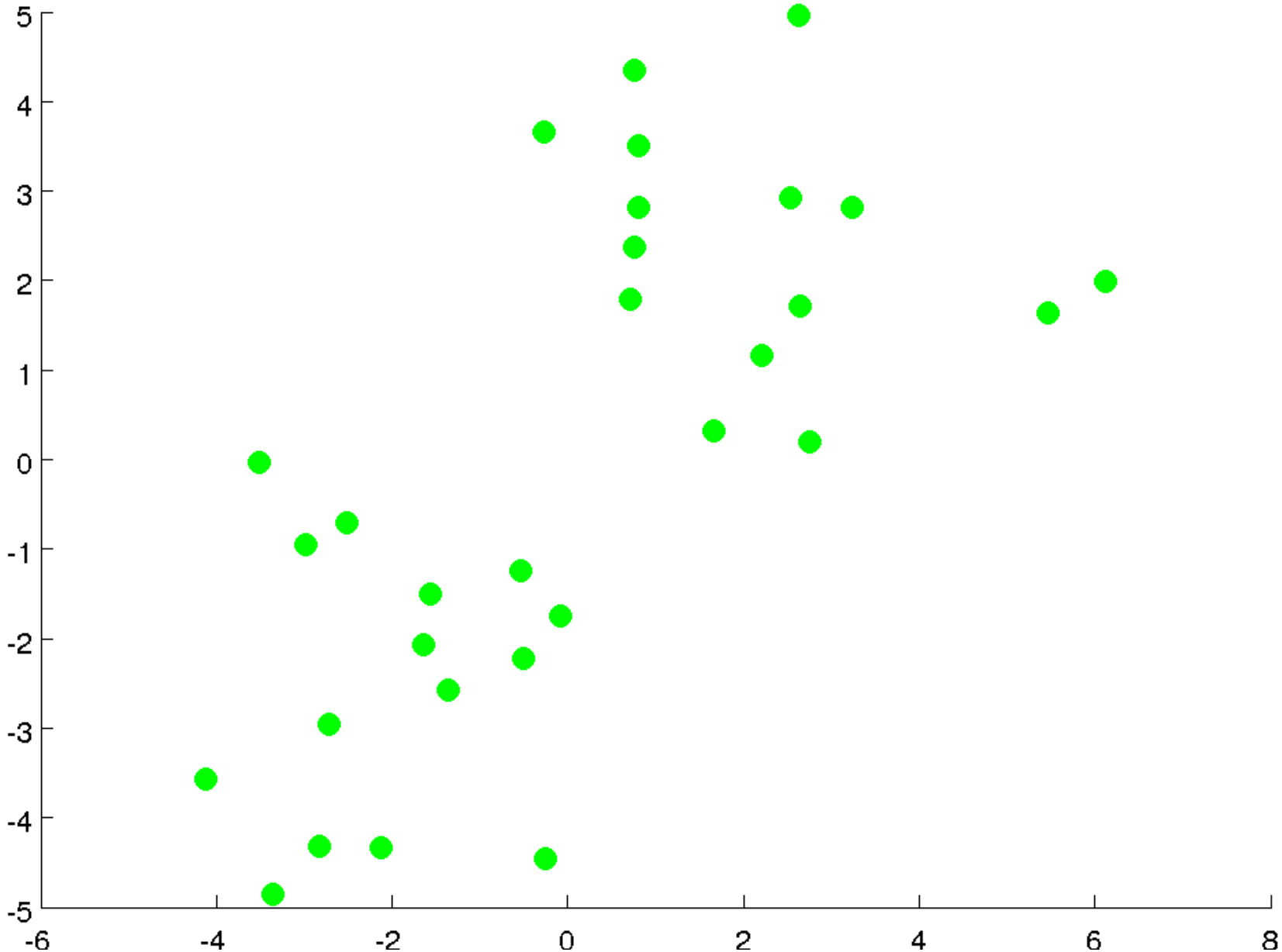
Distribution-based Clustering
This clustering approach assumes data is composed of distributions, such as [Gaussian distributions].



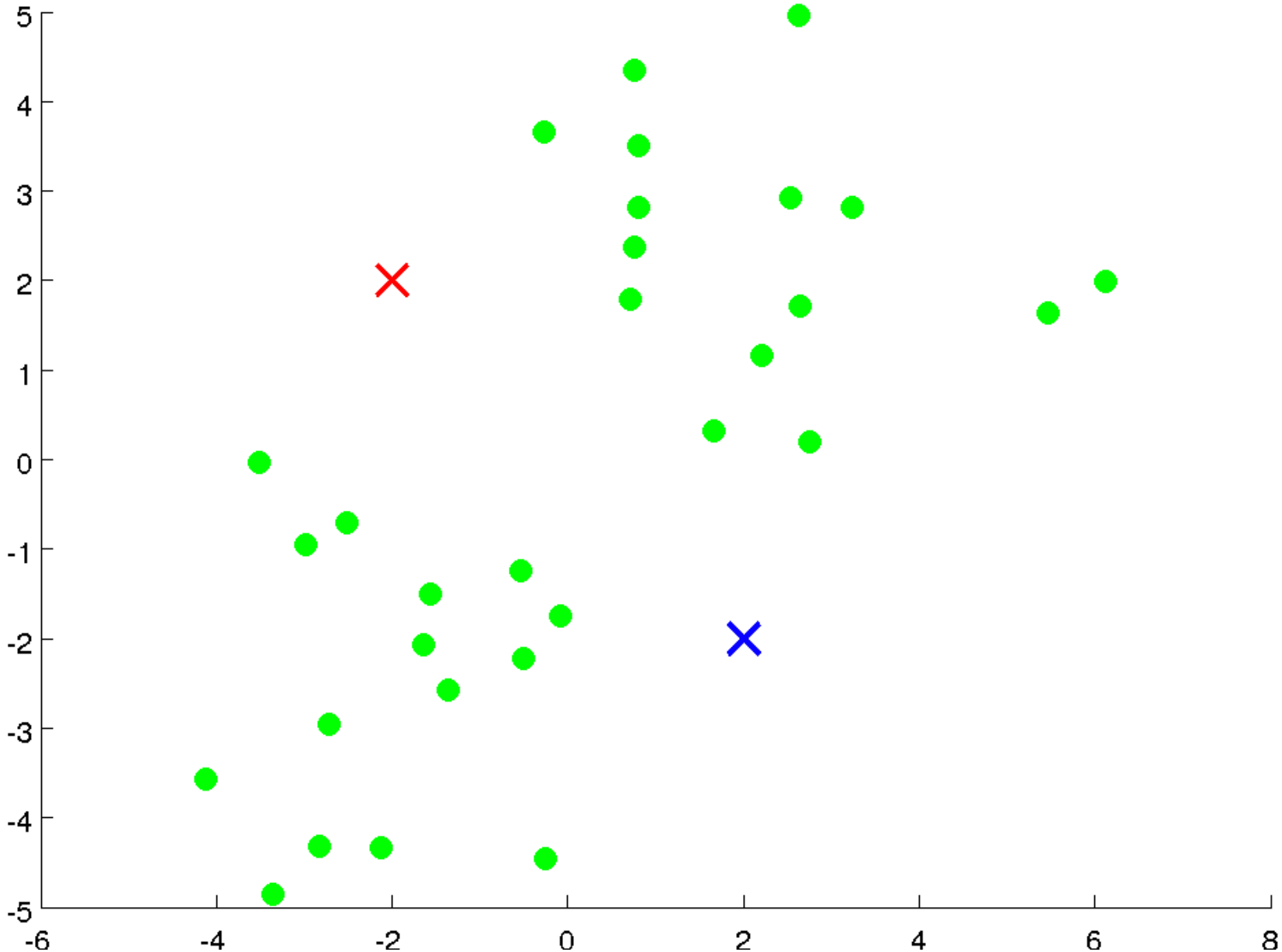
Hierarchical Clustering
Hierarchical clustering creates a tree of clusters. Hierarchical clustering, not surprisingly, is well suited to hierarchical data, such as taxonomies.



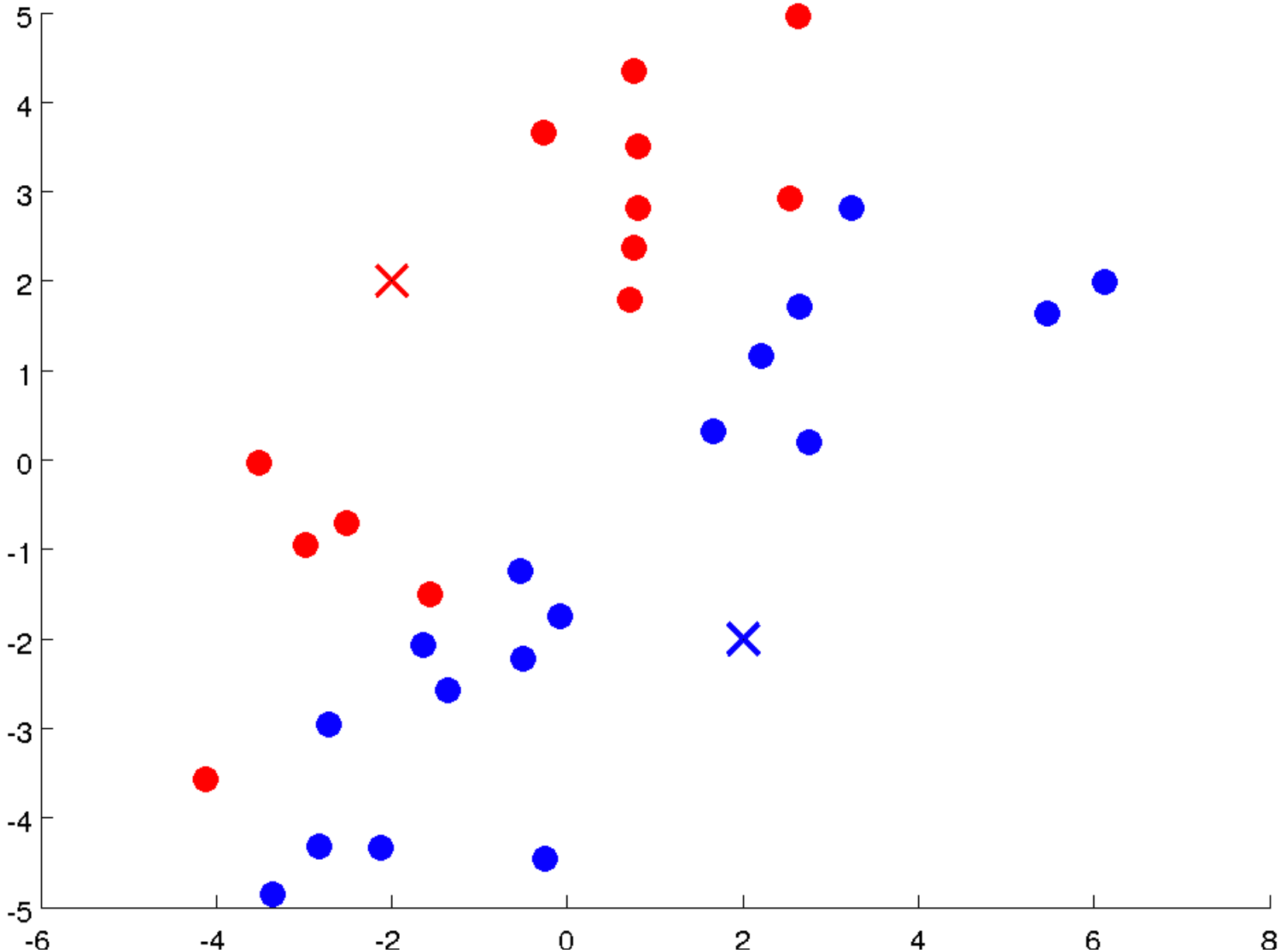
Photos taken from: https://developers.google.com/machine-learning/clustering/clustering-algorithms



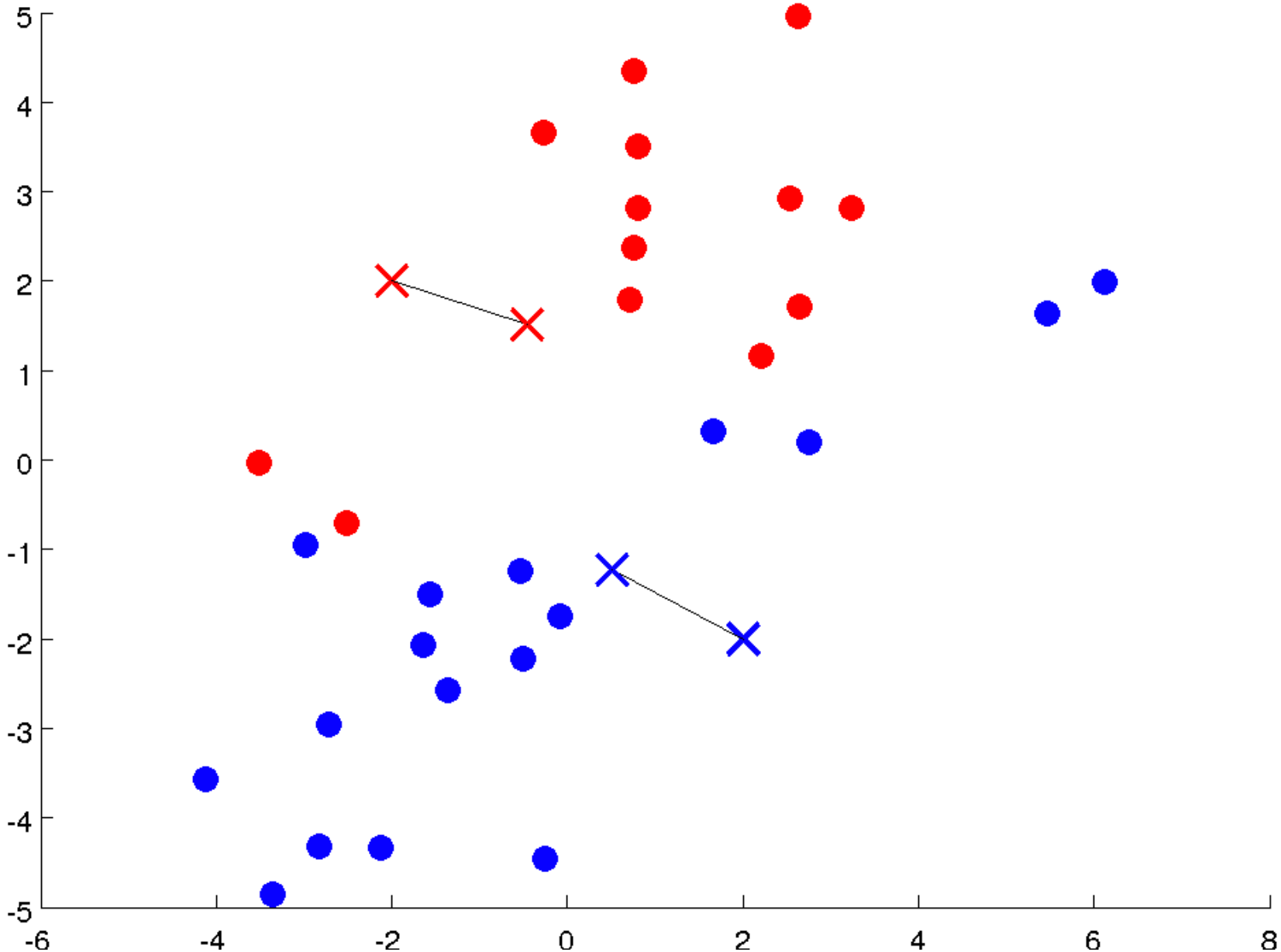
Slide credit: Andrew Ng



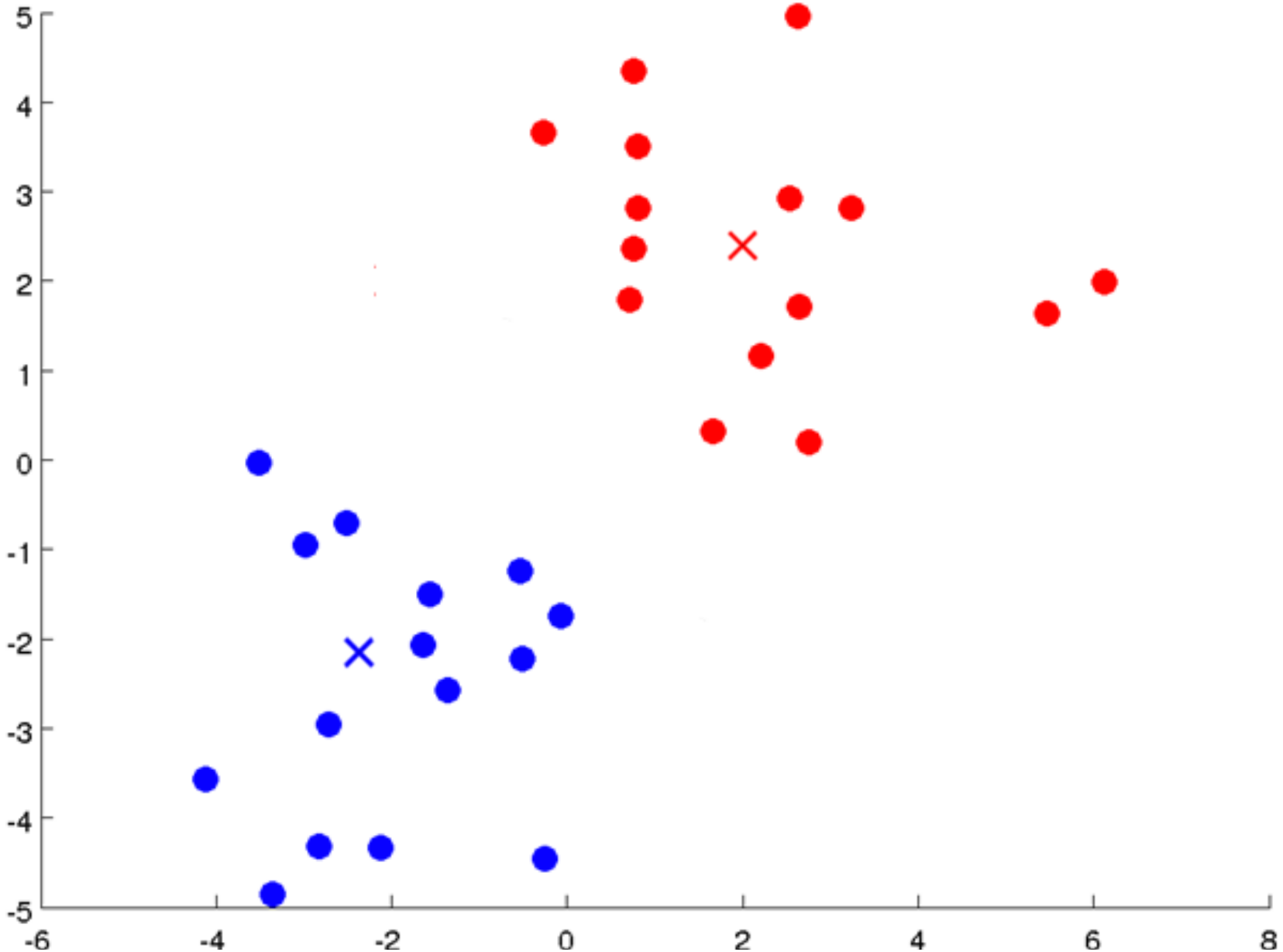
Slide credit: Andrew Ng



Slide credit: Andrew Ng



Slide credit: Andrew Ng



Slide credit: Andrew Ng

K-means Objective Function

- $c^{(i)}$ = Index of cluster (1, 2, … K) to which example $x^{(i)}$ is currently assigned
- μ_c = centroid of cluster c ($\mu_c \in \mathbb{R}^n$)
- S_c = a set of examples currently in cluster c
- $\mu_{c^{(i)}}$ = centroid of cluster to which example $x^{(i)}$ has been assigned

Optimization objective:

$$J\left(c^{(1)}, \cdots, c^{(m)}, \mu_1, \cdots, \mu_K\right) = \frac{1}{m}\sum_{i=1}^{m}\left\|x^{(i)} - \mu_{c^{(i)}}\right\|^2$$

Inertia

$$\min_{\substack{c^{(1)},\cdots,c^{(m)} \\ \mu_1,\cdots,\mu_K}} \sum_{c=1}^{k} \sum_{x^{(i)} \in S_c} \left\|x^{(i)} - \mu_{c^{(i)}}\right\|^2$$

K-means Objective Function

Randomly initialize K cluster centroids $\mu_1, \mu_2, \cdots, \mu_K \in \mathbb{R}^n$

Repeat{

for i = 1 to m

$c^{(i)}$:= index (from 1 to K) of cluster centroid closest to $x^{(i)}$

Cluster assignment step

$$J(c^{(1)}, \cdots, c^{(m)}, \mu_1, \cdots, \mu_K) = \frac{1}{m}\sum_{i=1}^{m} \|x^{(i)} - \mu_{c^{(i)}}\|^2$$

for k = 1 to K

μ_k := average (mean) of points assigned to cluster k

Centroid update step

$$J(c^{(1)}, \cdots, c^{(m)}, \mu_1, \cdots, \mu_K) = \frac{1}{m}\sum_{i=1}^{m} \|x^{(i)} - \mu_{c^{(i)}}\|^2$$

}

Slide credit: Andrew Ng

Time Complexity

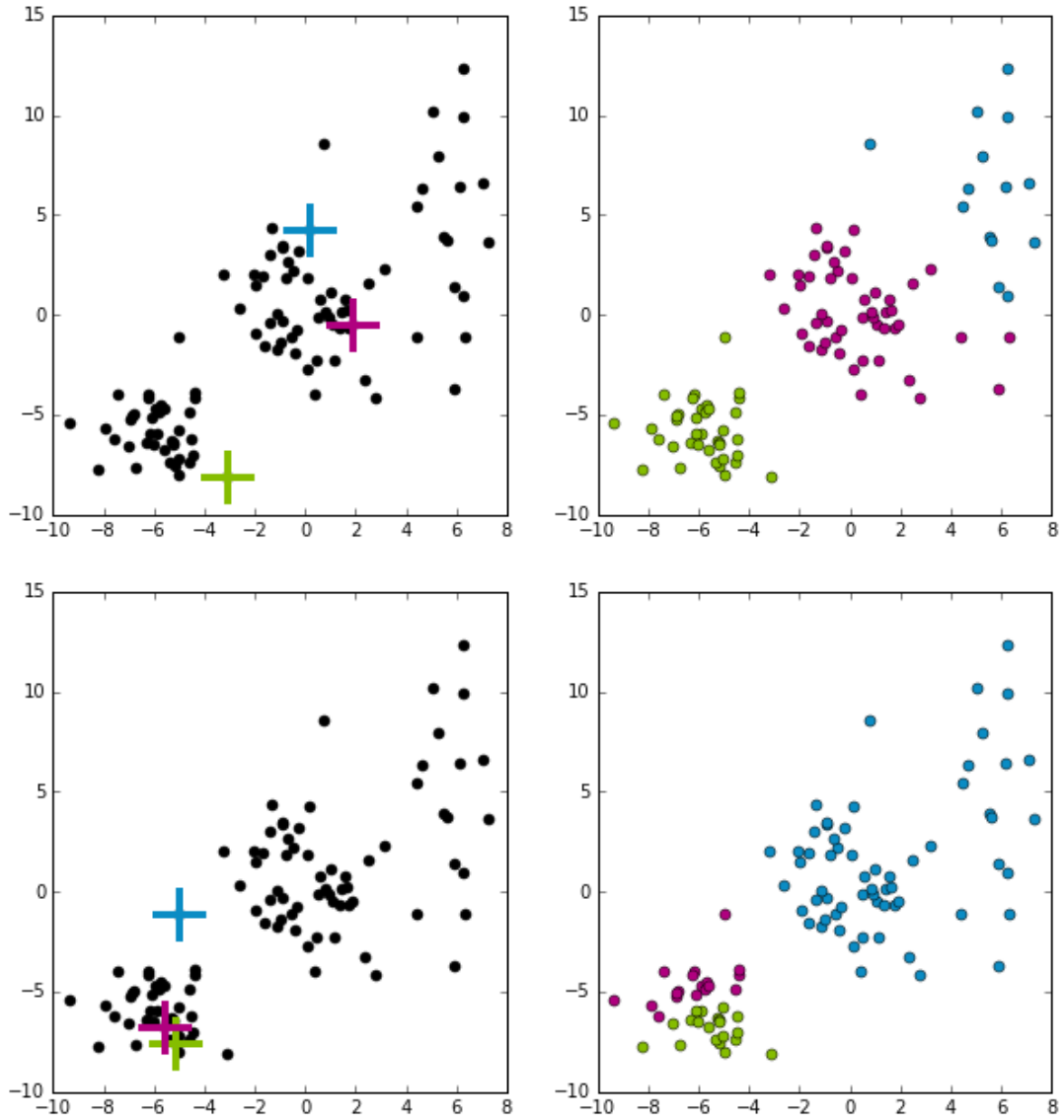
However, running a fixed number t of iterations of the standard algorithm takes only $O(t * k * n * d)$

For **n** number of (**d-dimensional**) points, where **k** is the number of centroids (or clusters).

This what practical implementations do (often with random restarts between the iterations). The standard algorithm only approximates a local optimum of the above function.

What does it mean for something to converge to a local optima?

Initial settings will greatly impact results!



Getting smarter with Initialization

K-Means ++

Making sure the initialized centroids are “good” is critical to finding quality local optima. Our purely random approach was wasteful since it’s very possible that initial centroids start close together.

Idea: Try to select a set of points farther away from each other.

k-means++ does a slightly smarter random initialization

1. Choose first cluster μ_1 from the data uniformly at random
2. For the current set of centroids (starting with just μ_1), compute the distance between each datapoint and its closest centroid
3. Choose a new centroid from the remaining data points with probability of x_i being chosen proportional to $d(x_i)^2$
4. Repeat 2 and 3 until we have selected k centroids

K-Mans ++

Pros / Cons

Pros

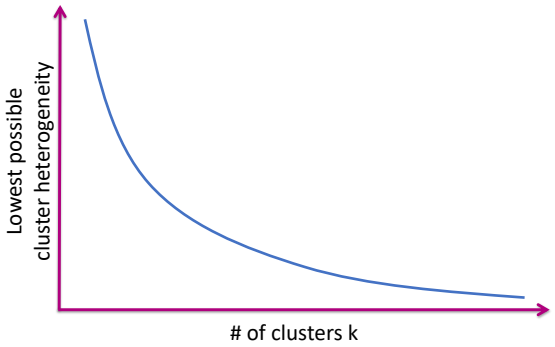
- Improves quality of local minima
- Faster convergence to local minima

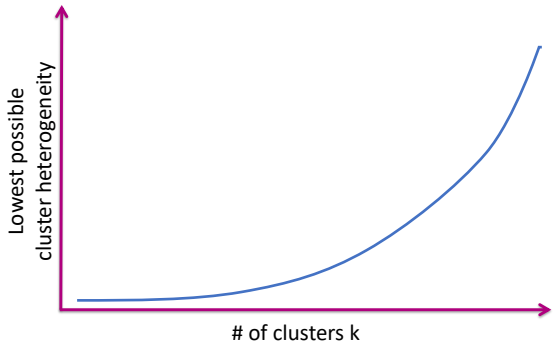
Cons

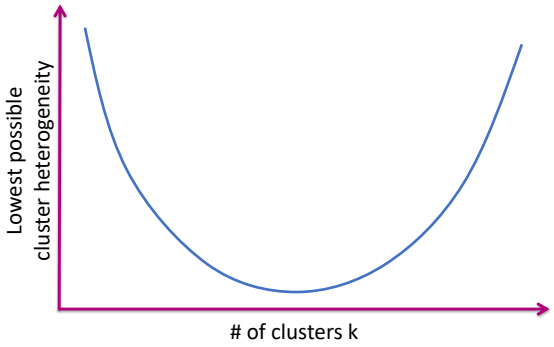
- Computationally more expensive at beginning when compared to simple random initialization

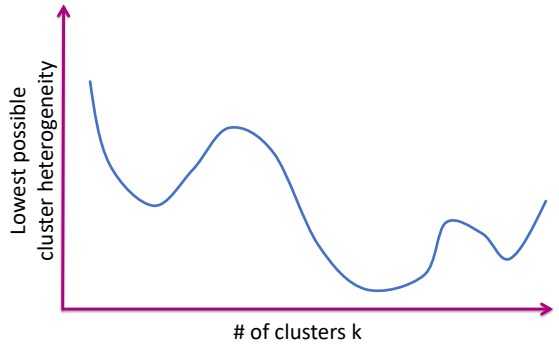
Question

Which of the following graphs shows how the globally optimal heterogeneity changes for each value of k?





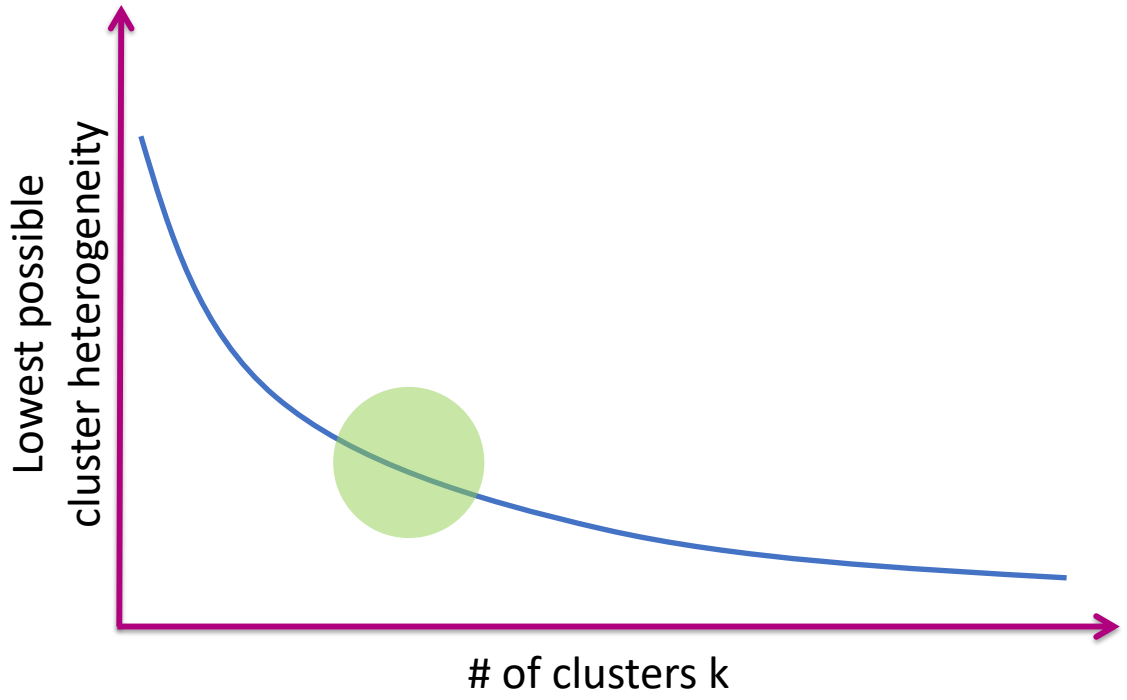




How to choose k?

No right answer! Depends on your application.

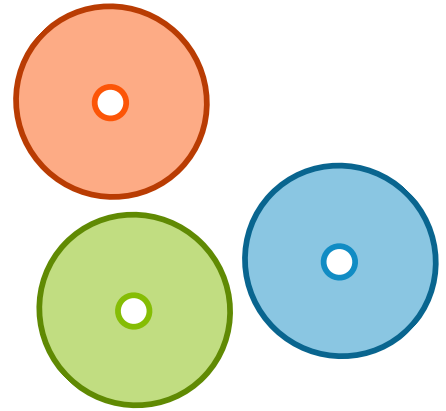
General, look for the “elbow” in the graph

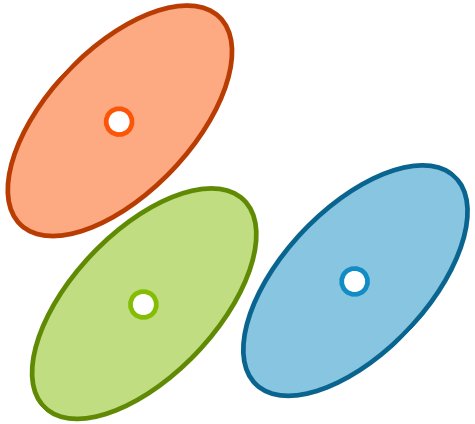


Note: You will usually have to run k-means multiple times for each k

Problems with k-means

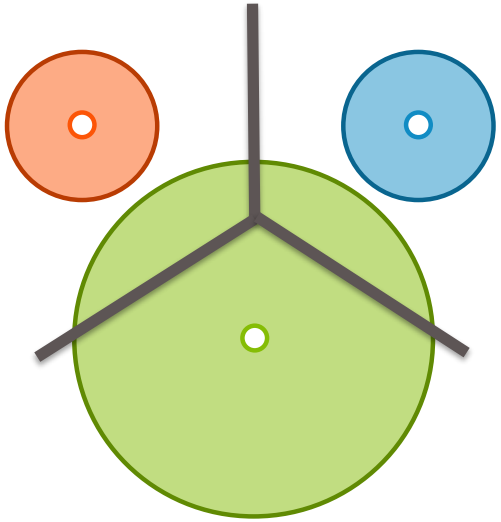
- In real life, cluster assignments are not always clear cut
  - E.g. The moon landing: Science? World News? Conspiracy?
- Because we minimize Euclidean distance, k-means assumes all the clusters are spherical
- We can change this with weighted Euclidean distance
  - Still assumes every cluster is the same shape/orientation



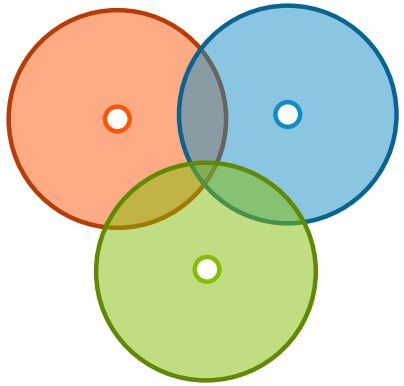


Failure Modes of k-means

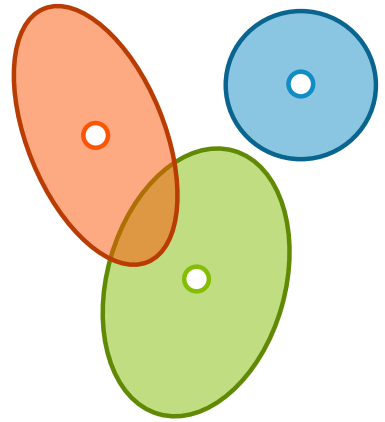
- If we don't meet the assumption of spherical clusters, we will get unexpected results



disparate cluster sizes

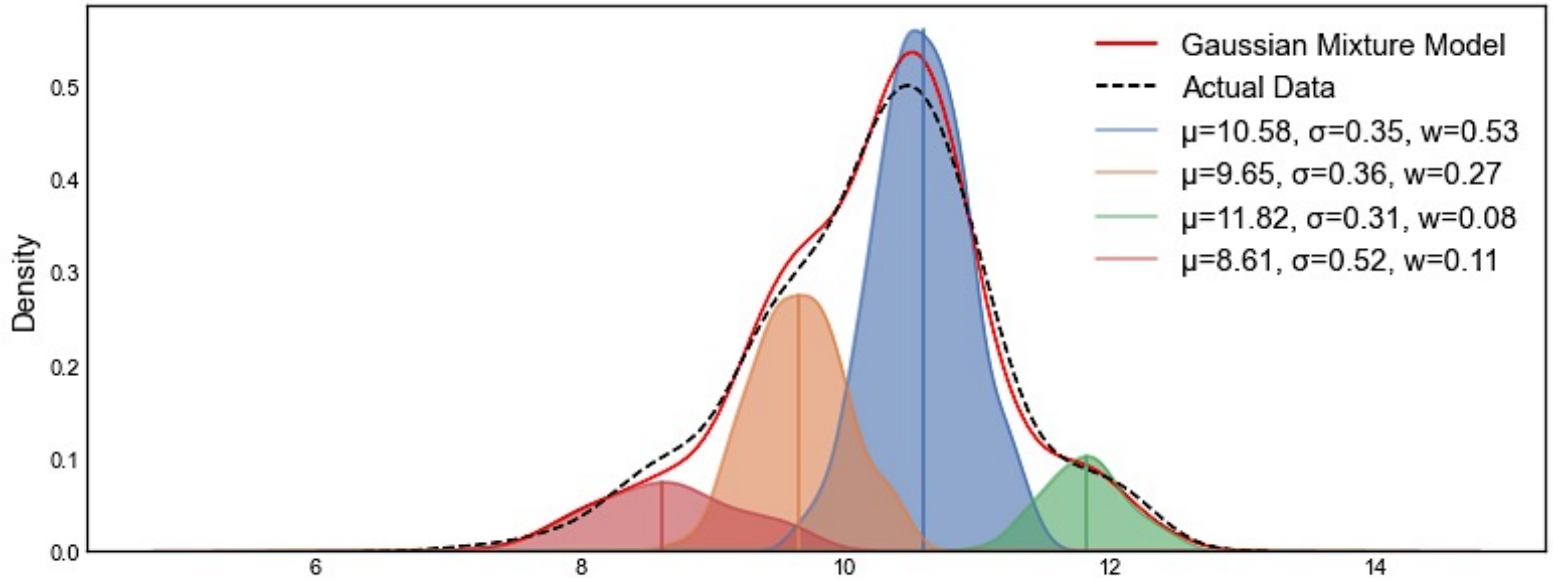


overlapping clusters



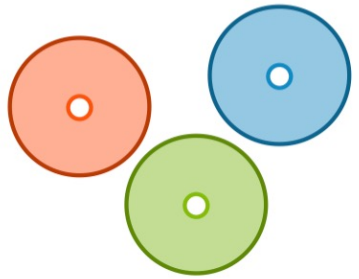
different shaped/oriented clusters

Mixture of Gaussians

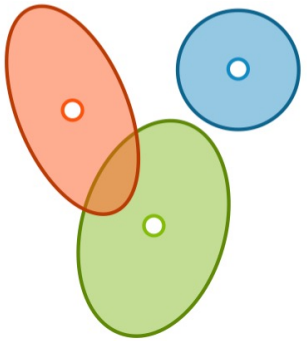


Discovering Shapes

k-means



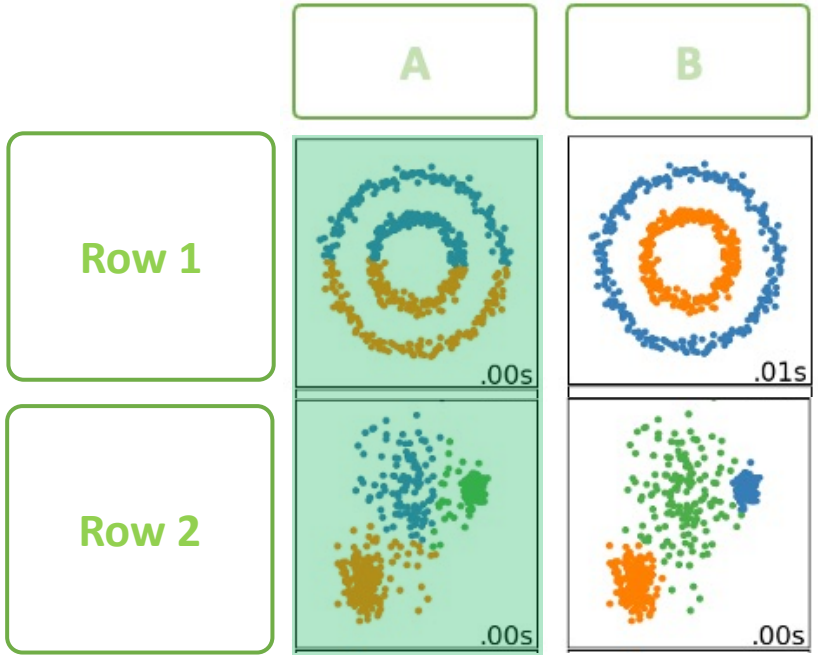
Mixture Models



Hierarchical Clustering

Week 12 Quiz – Q3

Recall the algorithm behind **k-means** clustering. Based on the characteristics of clusters in the following figure, guess which one column (for each of two data distributions) is achieved as the result of applying the k-means method:
For each row (data distribution), mention A or B.



Questions?

Northeastern University