Supervised learning

Linear Regression

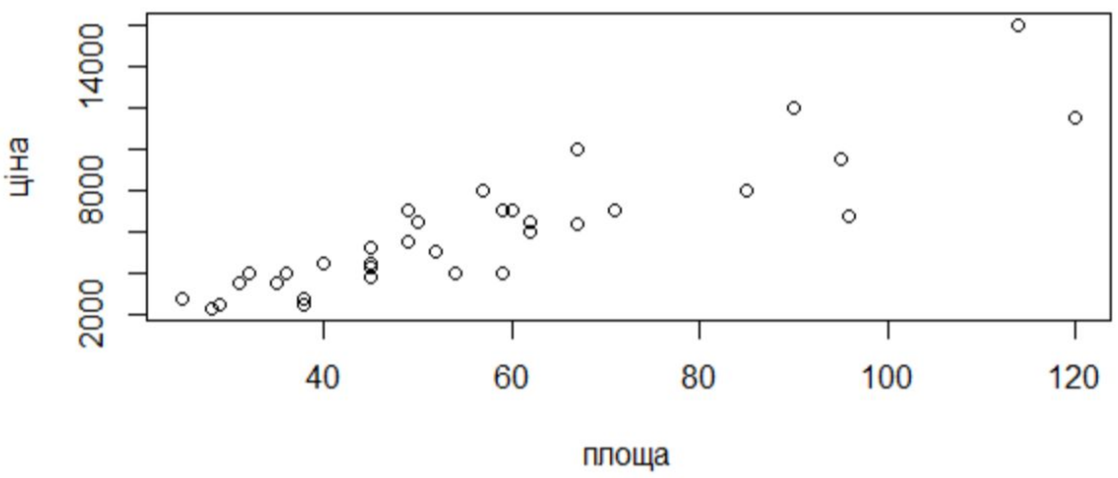
Predicting the house price

Площа, м2	Вартість, 1000 грн
50	1250
141	3000
75	1680
42	900
80	1700
...	...

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

m

training set



Hypothesis: *y* is a linear function of *x*

Площа, м2	Вік будинку	Вартість, 1000 грн
50	10	1250
141	25	3000
75	15	1680
42	5	900
80	1	1700
...		...

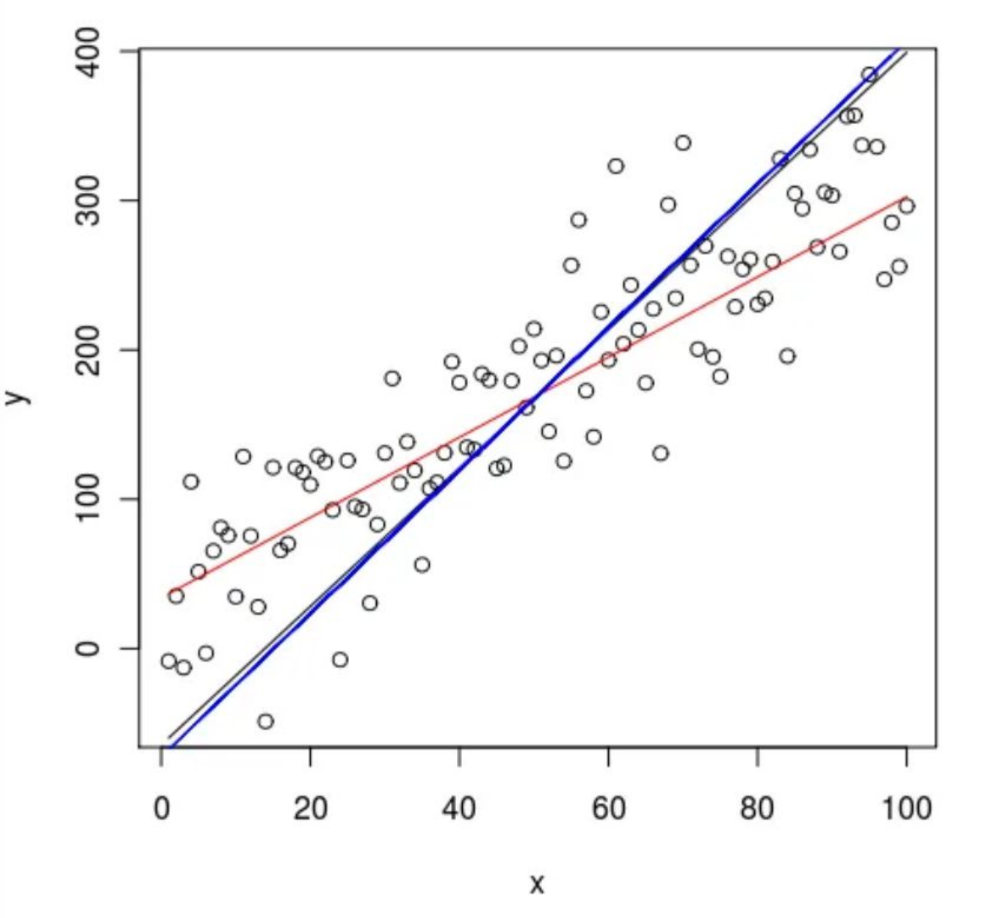
гіпотеза (hypothesis)

вектори

$$h(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 = \sum_{i=1}^{m} \theta_i x_i = \theta^T x$$

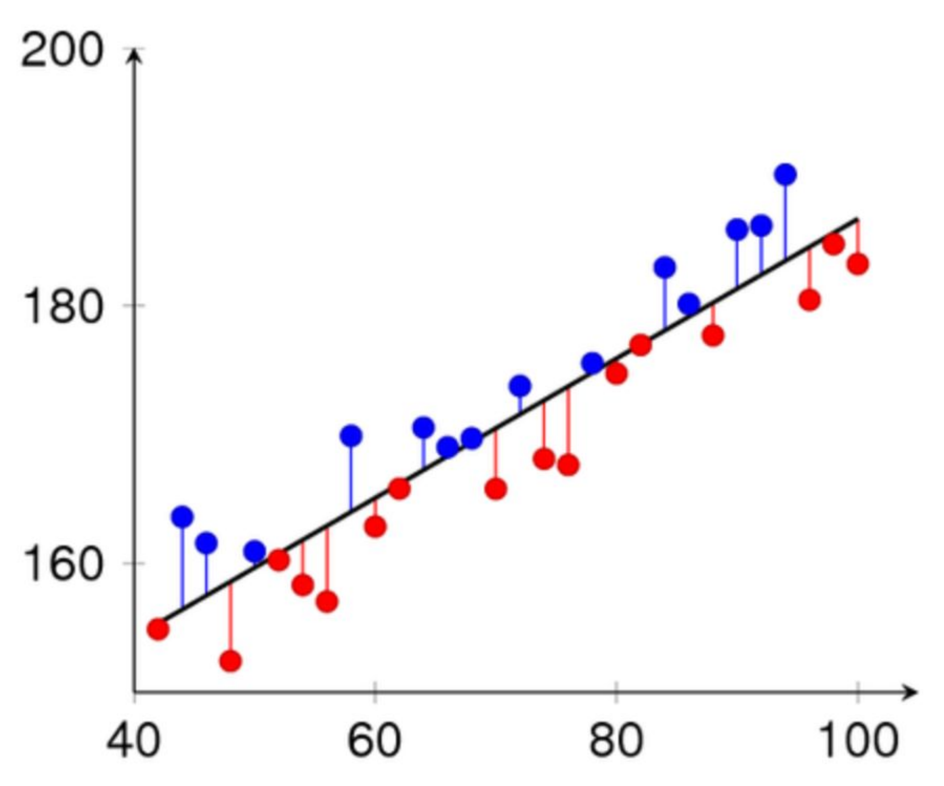
параметри|ваги (weights)

Learning parameters



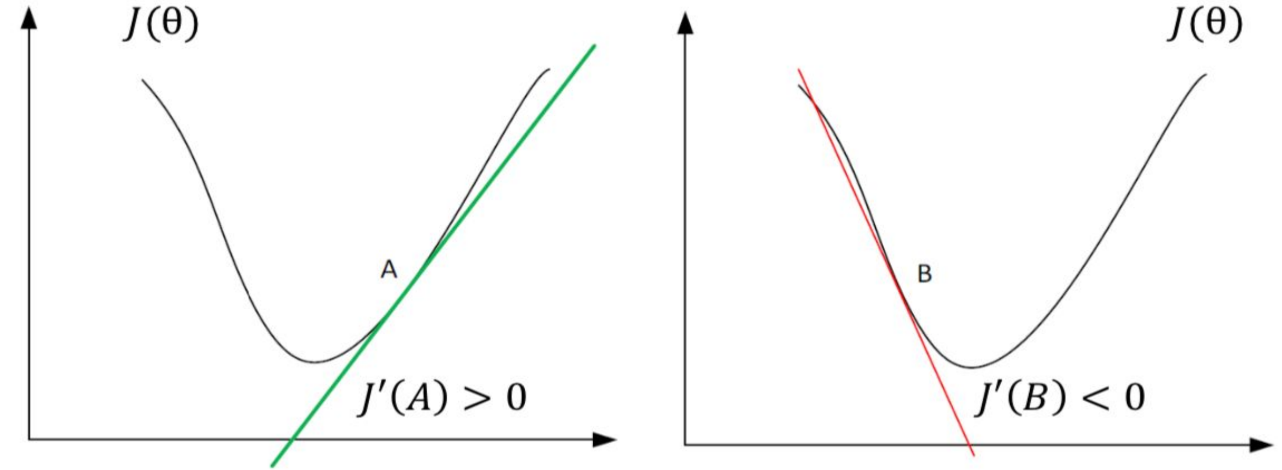
Which line is better?

Cost function



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

Gradient Descent



один «крок» градієнтного спуску:

кінцеве положення

початкове положення

градієнт

$$\theta_j := \theta_j - \alpha \frac{dJ(\theta)}{d\theta_j}$$

темп (learning rate)

Supervised learning

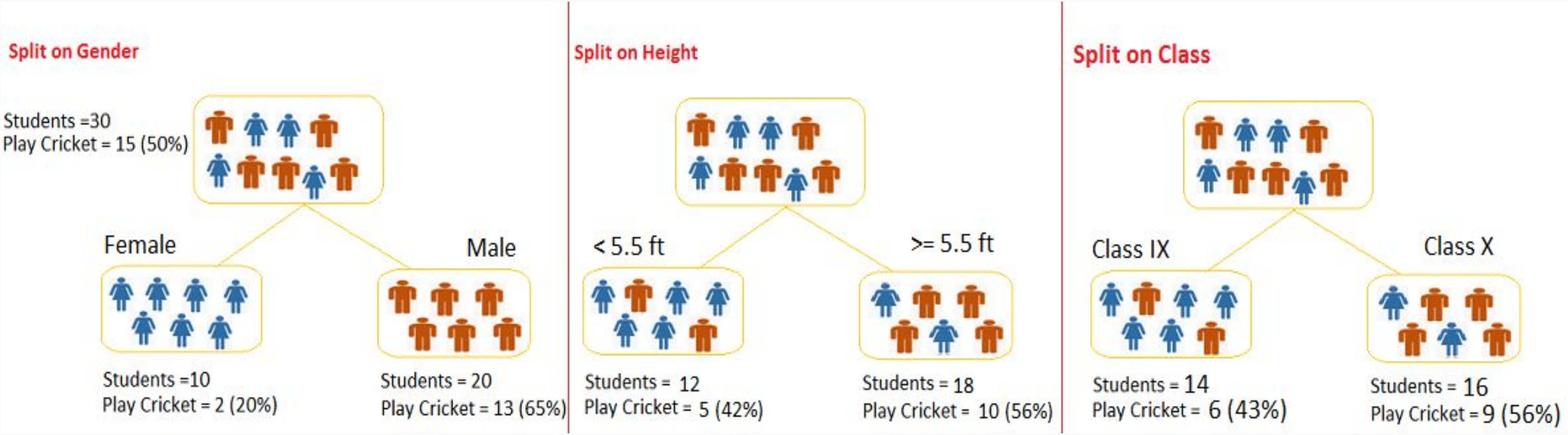
Decision Tree

Decision Tree: example

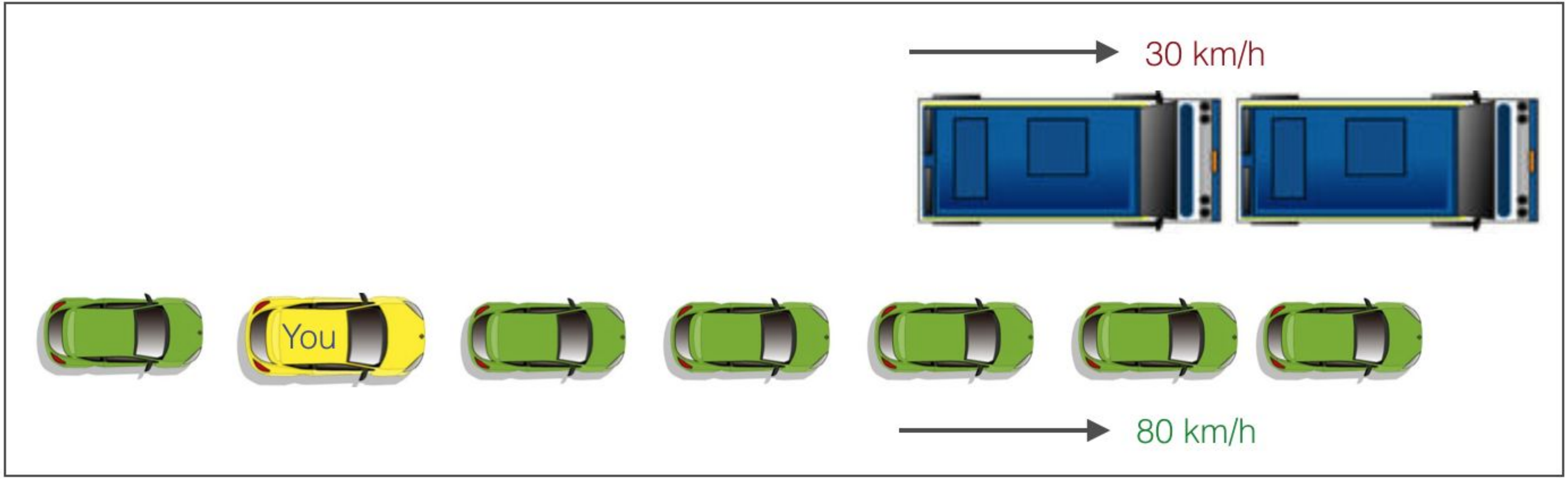
How does a tree decide where to split?

30 students
3 variables: Gender, Class, Height
15 play cricket

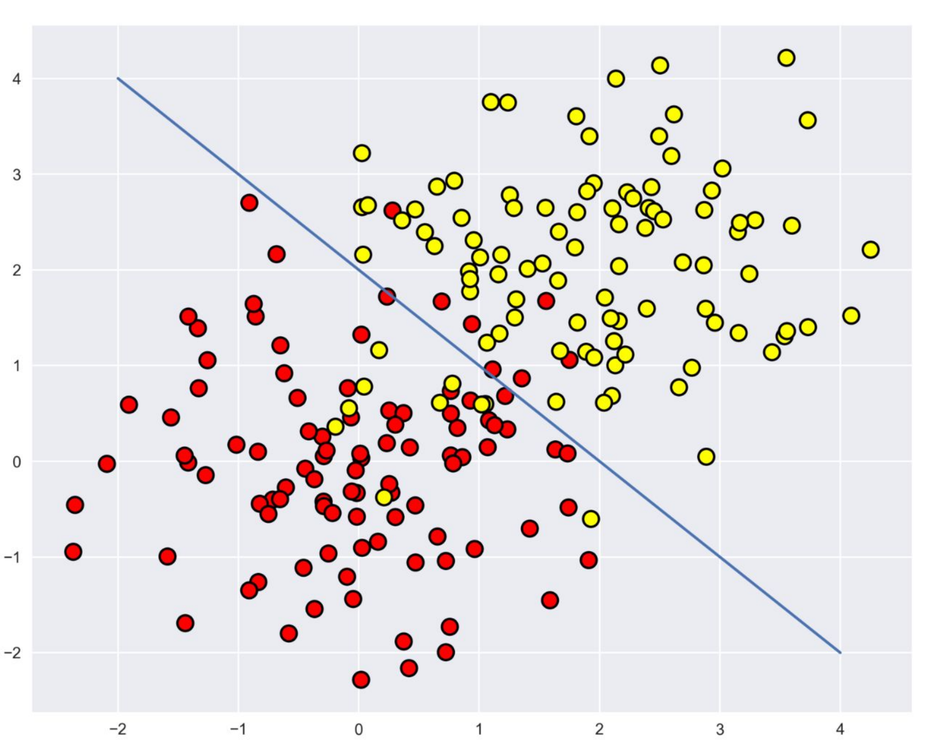
Goal: predict who will play cricket

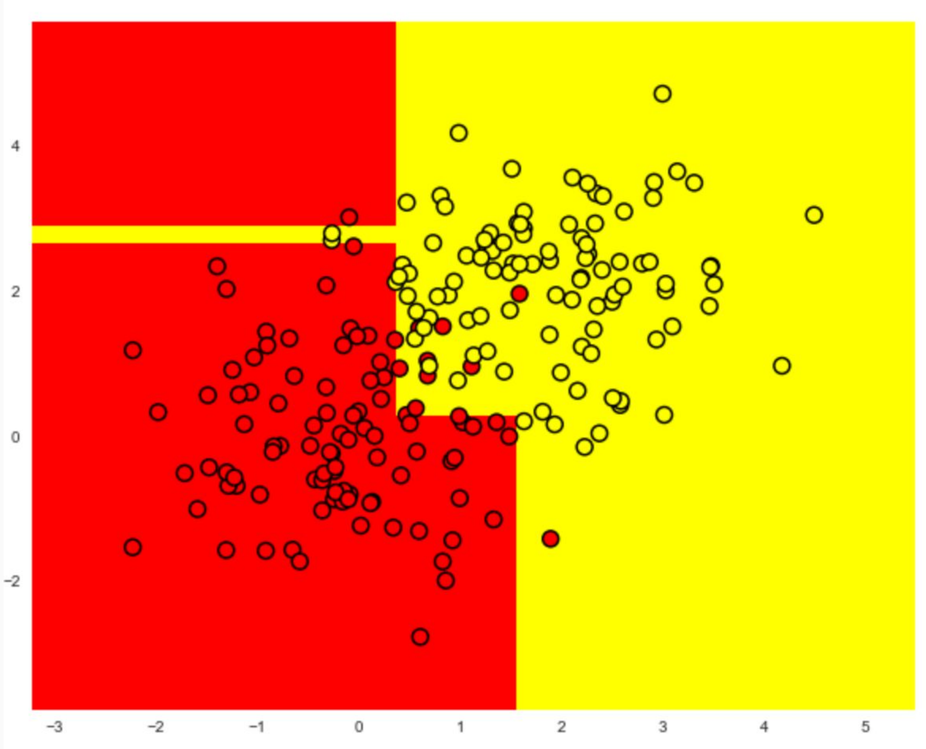


Greedy approach

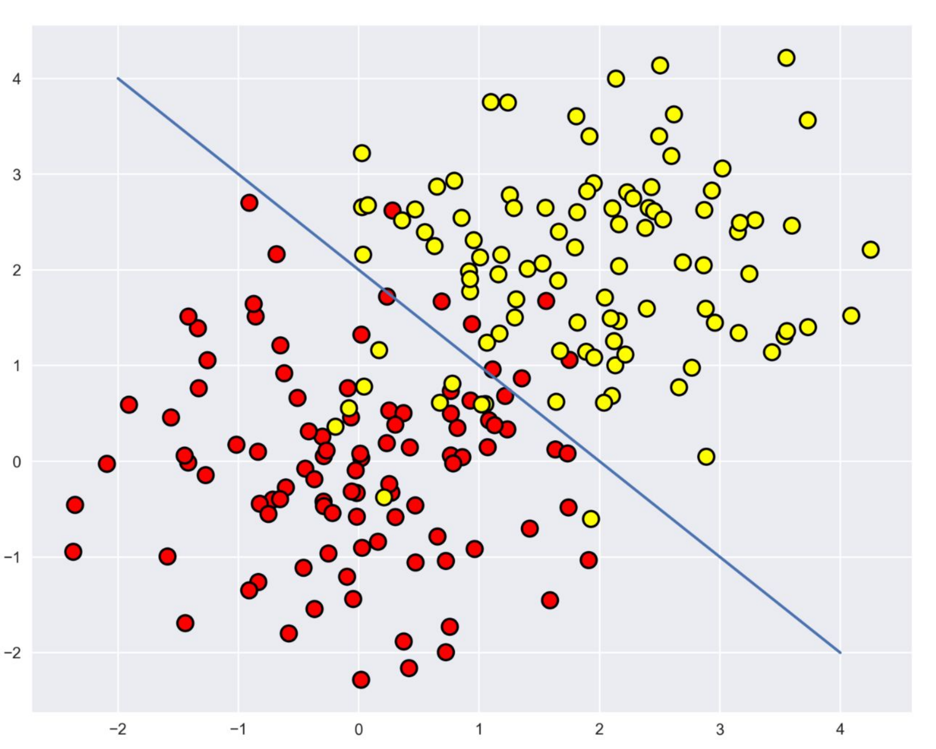


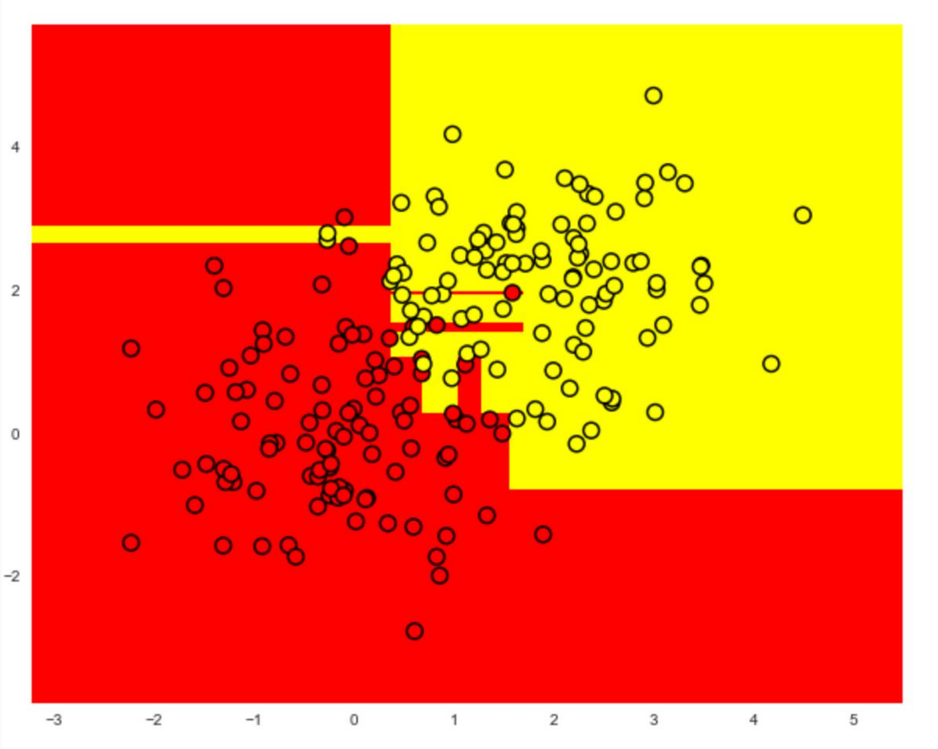
Example



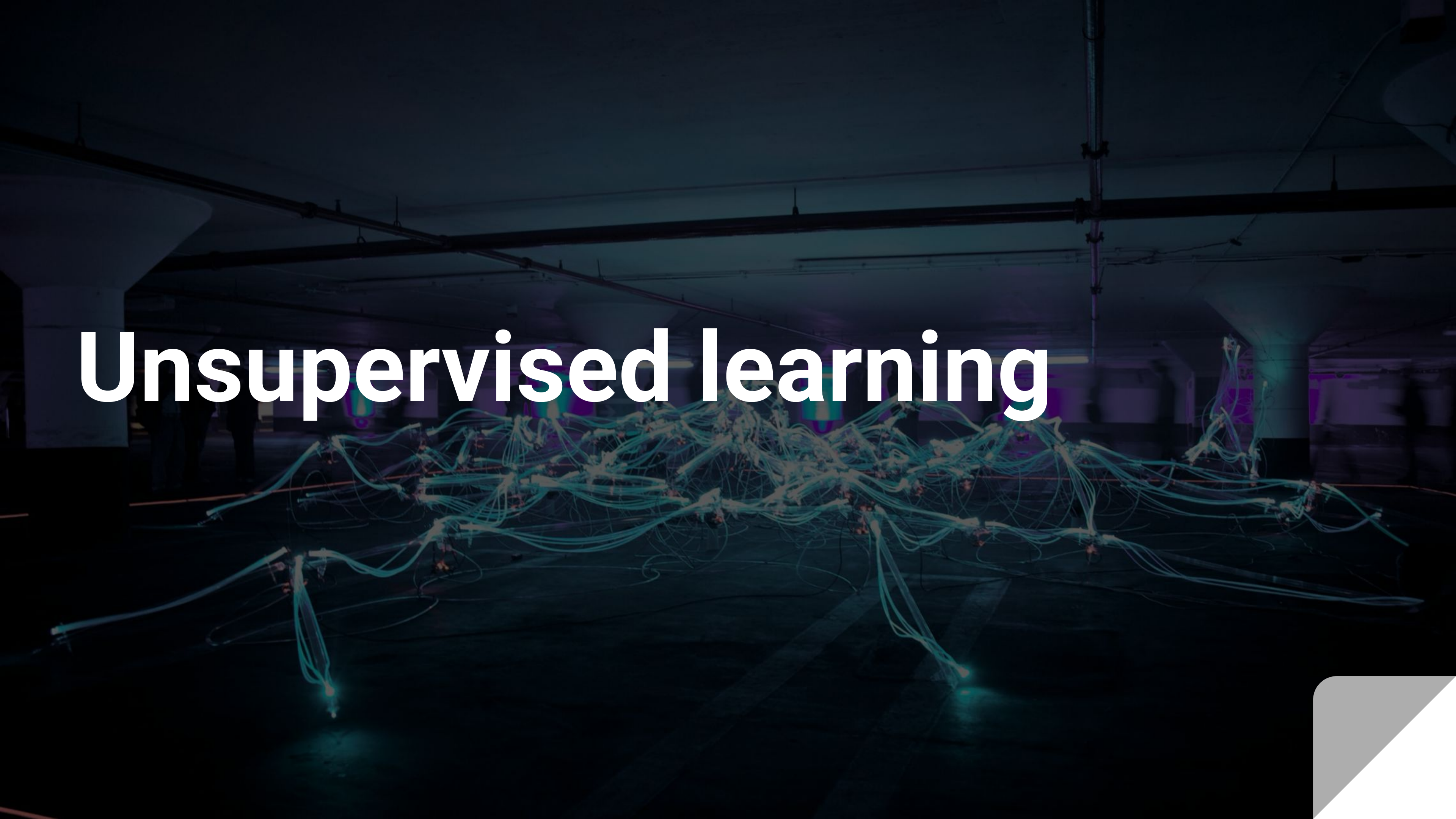


Overfitting

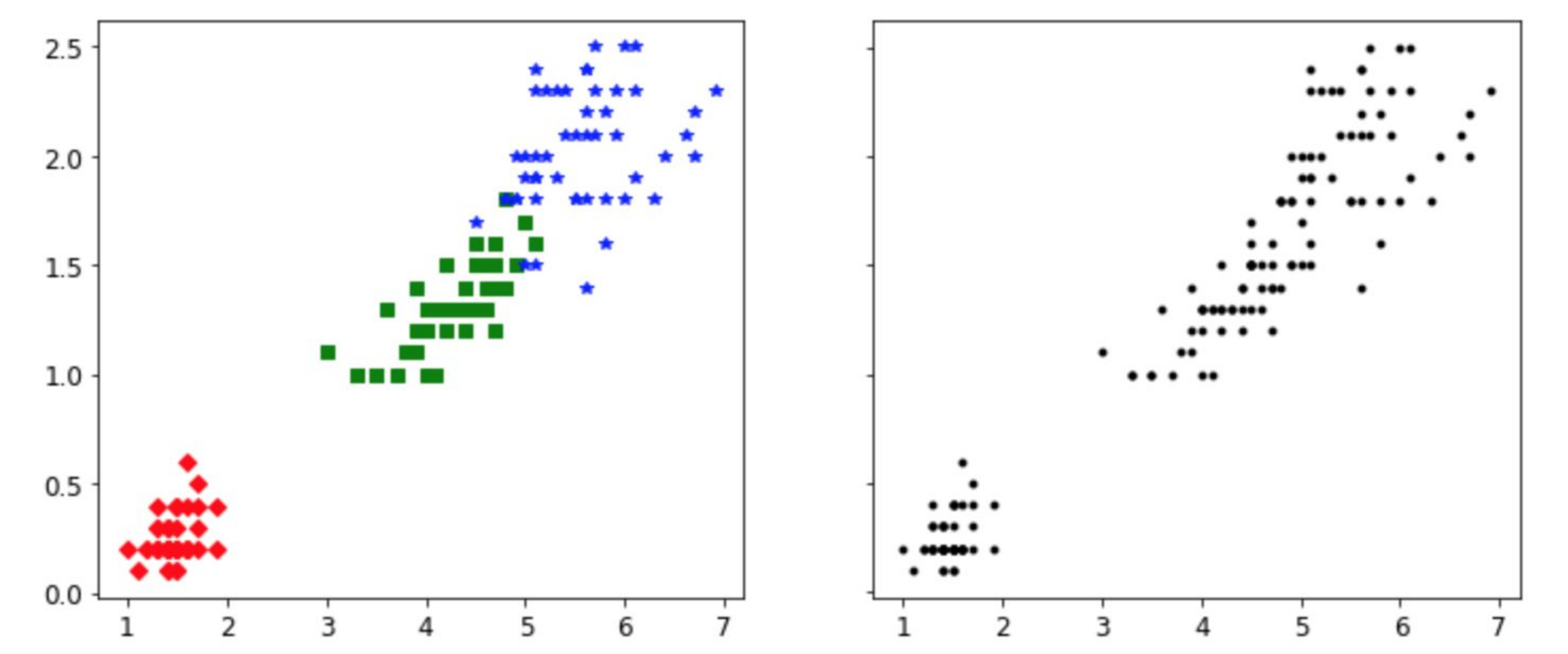




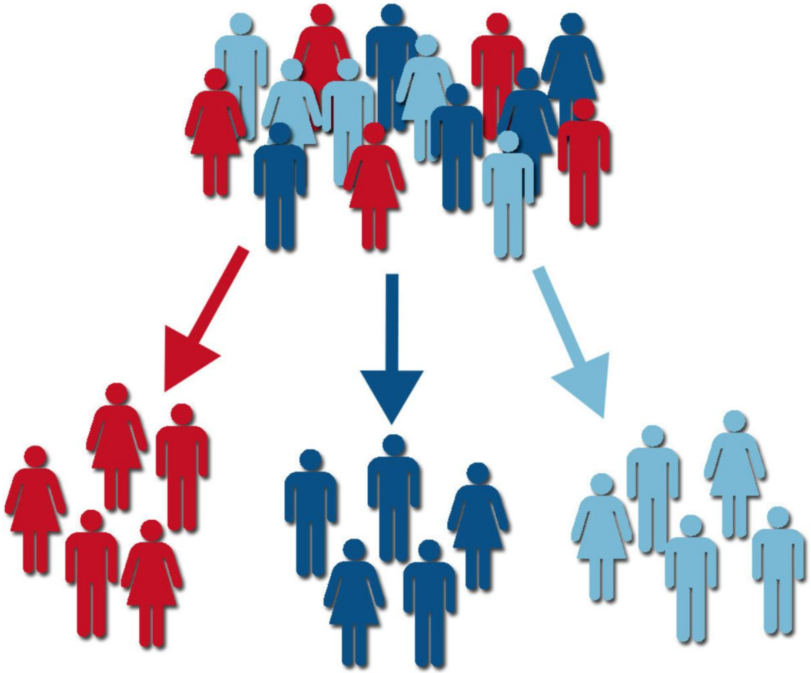
Unsupervised learning



Unsupervised learning - data without labels



Example: Customer Segmentation Analysis



Goal:

- Divide customers into groups sharing the same properties

Benefits:

- More targeted marketing campaigns.
- Reducing marketing spendings.
- Personalized marketing messages.

Unsupervised learning tasks

- Clustering
- Anomaly Detection
- Dimensionality Reduction
- Density estimation

Clustering

Goal: Identify similar instances and assign them to clusters.

- Customer Segmentation
- Data Analysis
- Dimensionality reduction
- Fraud detection
- Semi-supervised learning
- Search engines
- Image segmentation

K-means

Given a training set $\{x^{(1)}, \ldots, x^{(m)}\} \in \mathbb{R}^n$ K-means algorithm is as follows:

1. Initialize **cluster centroids** $\mu_1, \mu_2, \ldots, \mu_k \in \mathbb{R}^n$ randomly.

2. Repeat until convergence: {

   For every i, set

   $$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2.$$

   For each j, set

   $$\mu_j := \frac{\sum_{i=1}^m 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^m 1\{c^{(i)} = j\}}.$$

   }

Is k_means guaranteed to converge?

Let's define the **distortion** function to be

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

- k-means is coordinate descent on J
- J monotonically decrease -> value of J converges
- Usually c, μ converge too
- J is a non-convex, not guaranteed to converge to the global minimum.

Centroid Initialization Methods

- **Run k-means many times** (with different random initial values for centroids). Pick the one that gives the lowest distortion J.

- **K-Means++** Select centroid that are distant from one another.

K-Means improvements

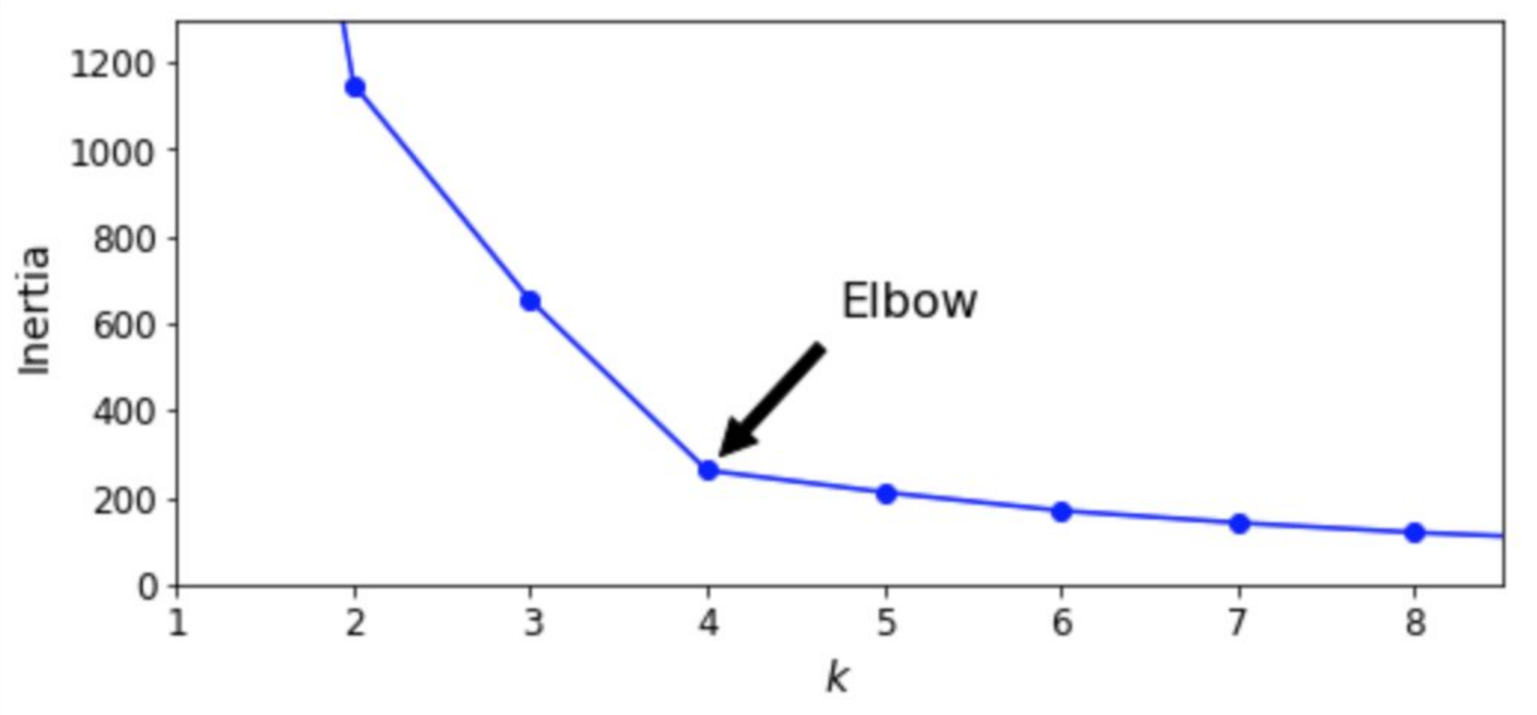
- **Accelerated K-Means**

  Considerably accelerates the algorithm by avoiding many unnecessary distance calculations

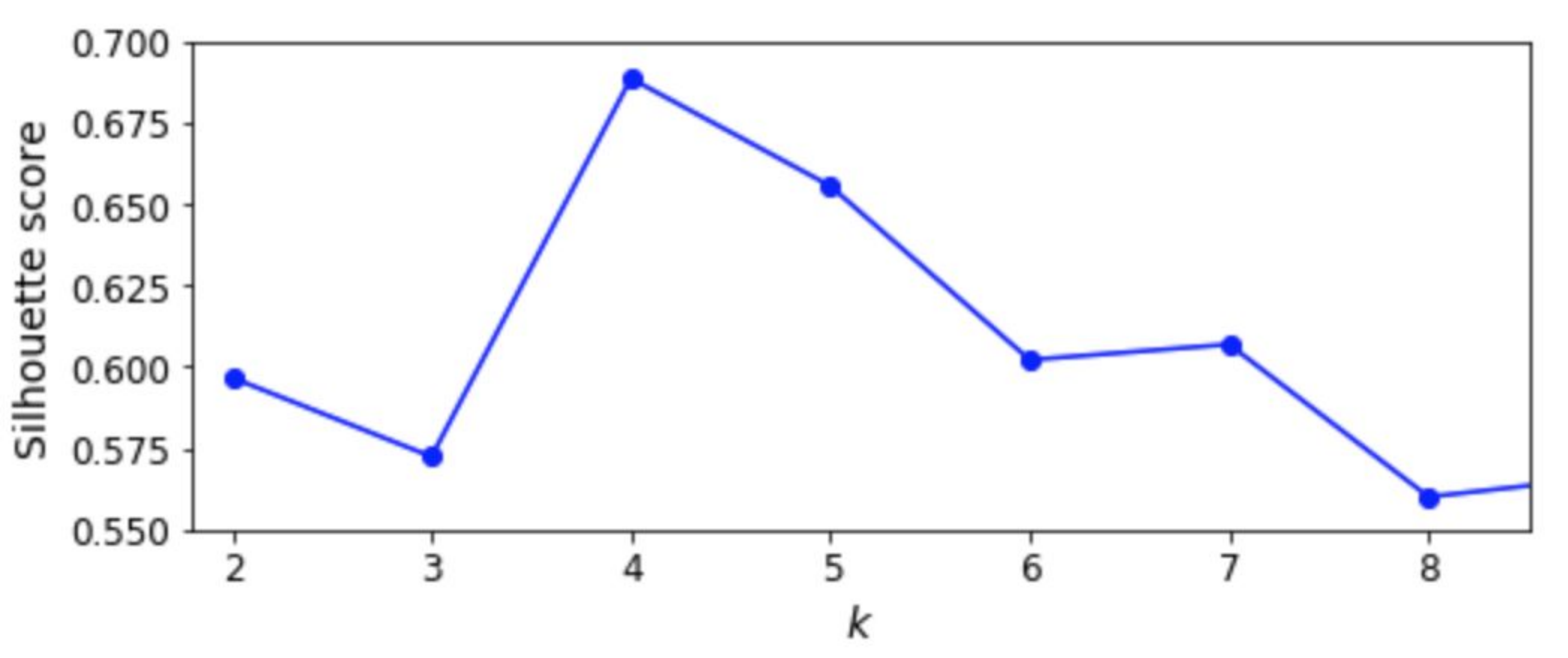
- **Mini-batch K-Means**

  Algorithm uses mini-batches, moving centroids just slightly at each iteration

Optimal number of clusters. Elbow rule



Optimal number of clusters. Silhouette score



Pros and Cons of k-means

K-Means **is good** for:

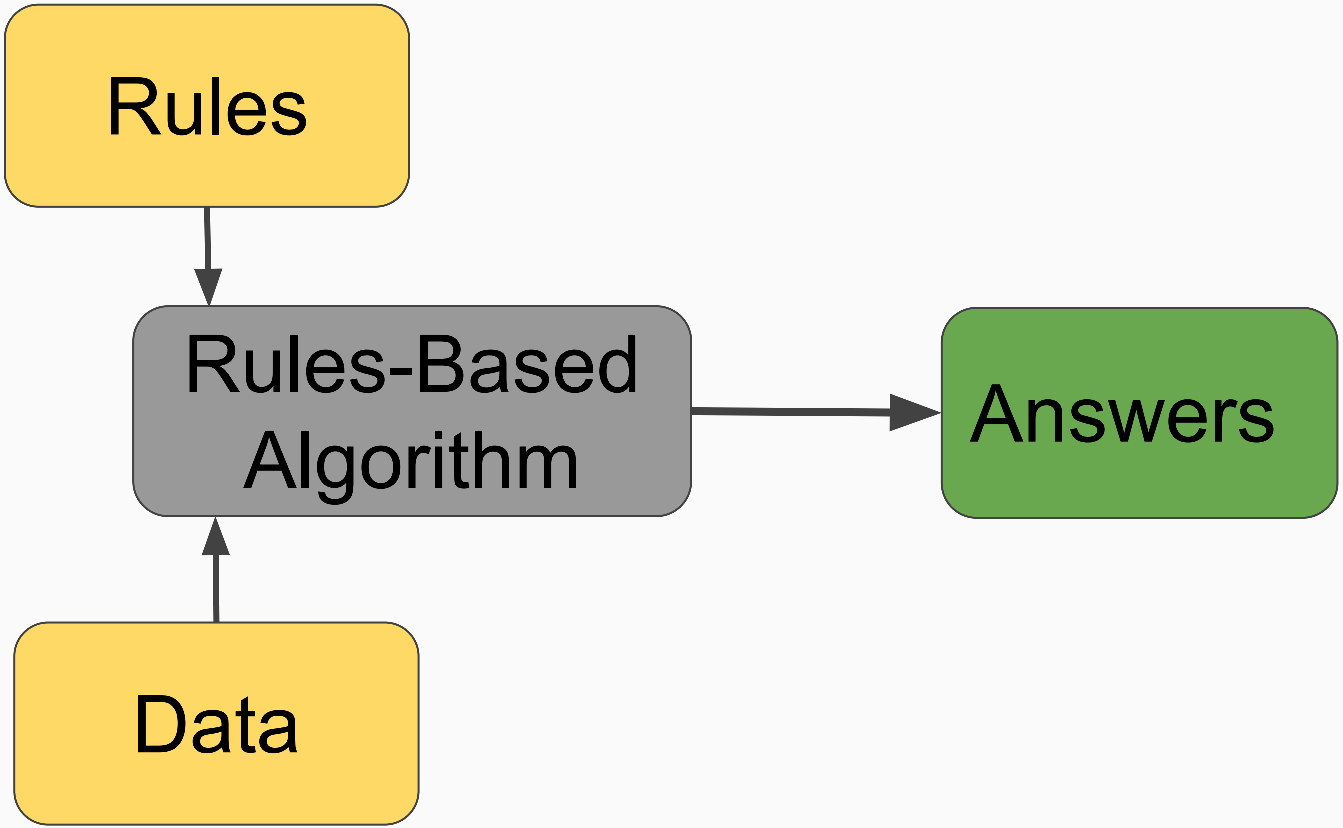
1. Scalability
2. Simplicity

K-Means is **not good** for clusters of:

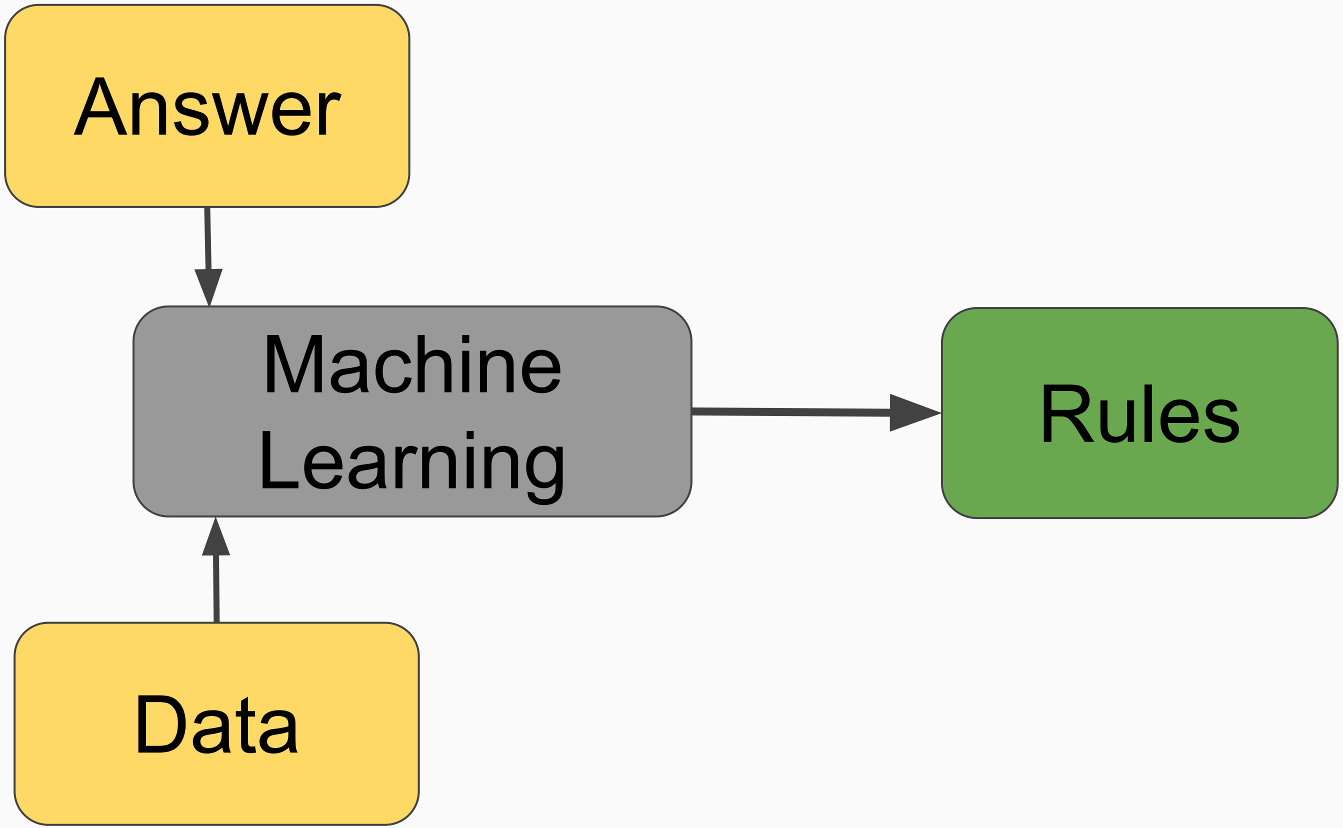
1. Different densities
2. Non-spherical shapes
3. Varying sizes

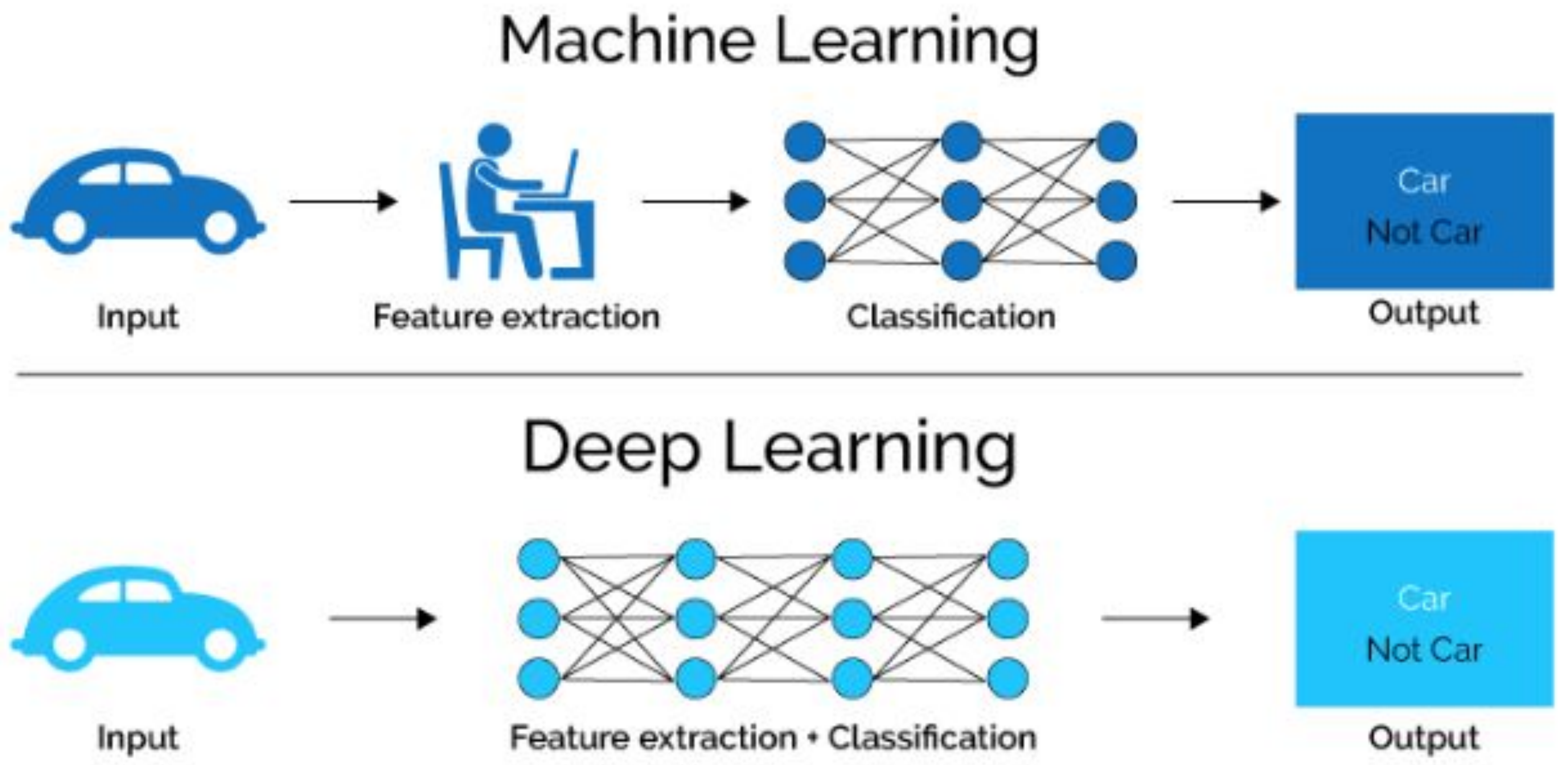
Deep Learning

Automated AI



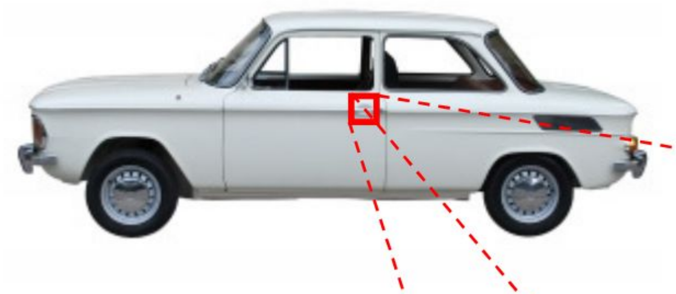
Intelligence AI




Machine Learning
Car
Not Car
Input
Feature extraction
Classification
Output
Deep Learning
Car
Not Car
Input
Feature extraction + Classification
Output

What is this?

You see this:



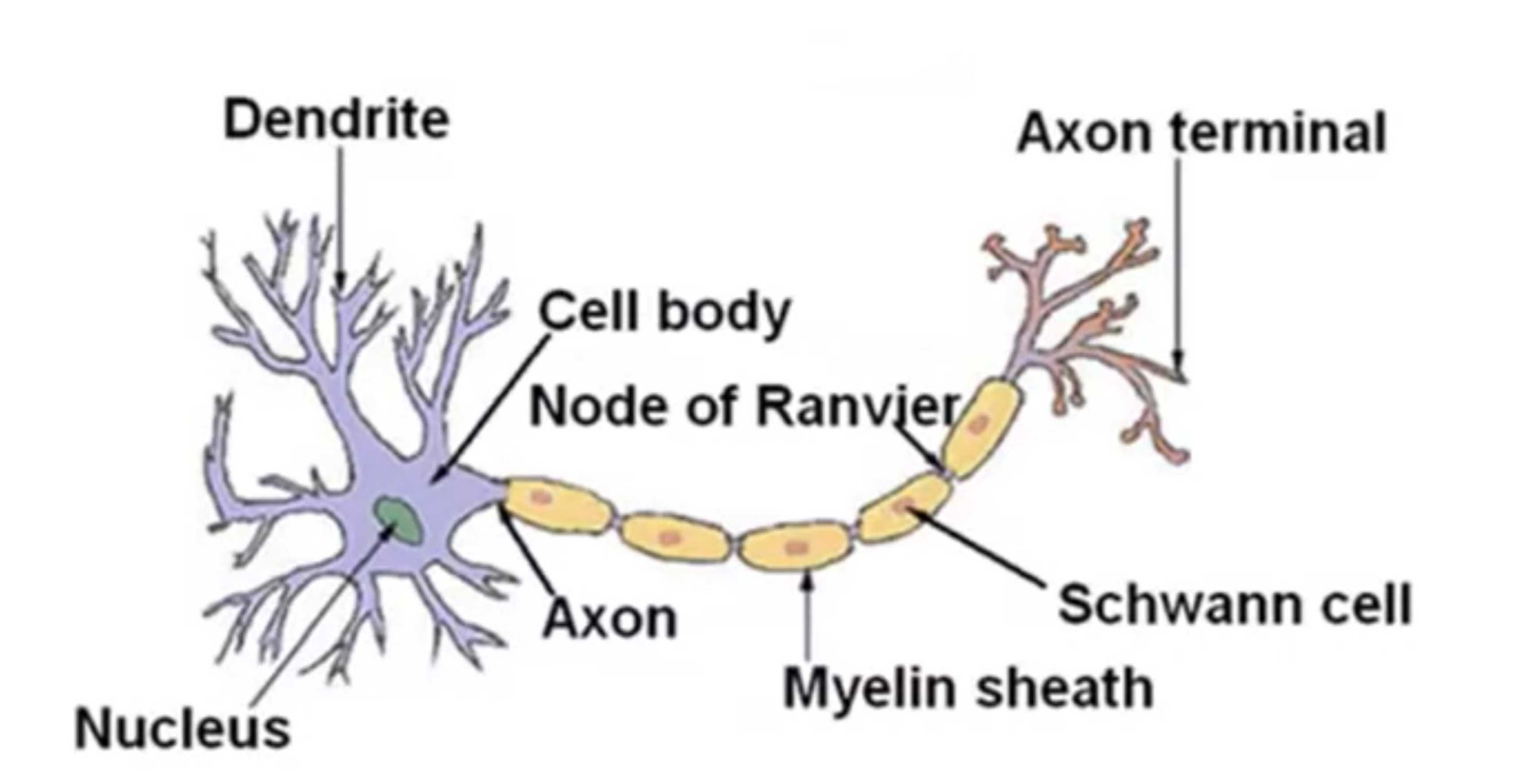
But the camera sees this:

194	210	201	212	199	213	215	195	178	158	182	209
180	189	190	221	209	205	191	167	147	115	129	163
114	126	140	188	176	165	152	140	170	106	78	88
87	103	115	154	143	142	149	153	173	101	57	57
102	112	106	131	122	138	152	147	128	84	58	66
94	95	79	104	105	124	129	113	107	87	69	67
68	71	69	98	89	92	98	95	89	88	76	67
41	56	68	99	63	45	60	82	58	76	75	65
20	43	69	75	56	41	51	73	55	70	63	44
50	50	57	69	75	75	73	74	53	68	59	37
72	59	53	66	84	92	84	74	57	72	63	42
67	61	58	65	75	78	76	73	59	75	69	50

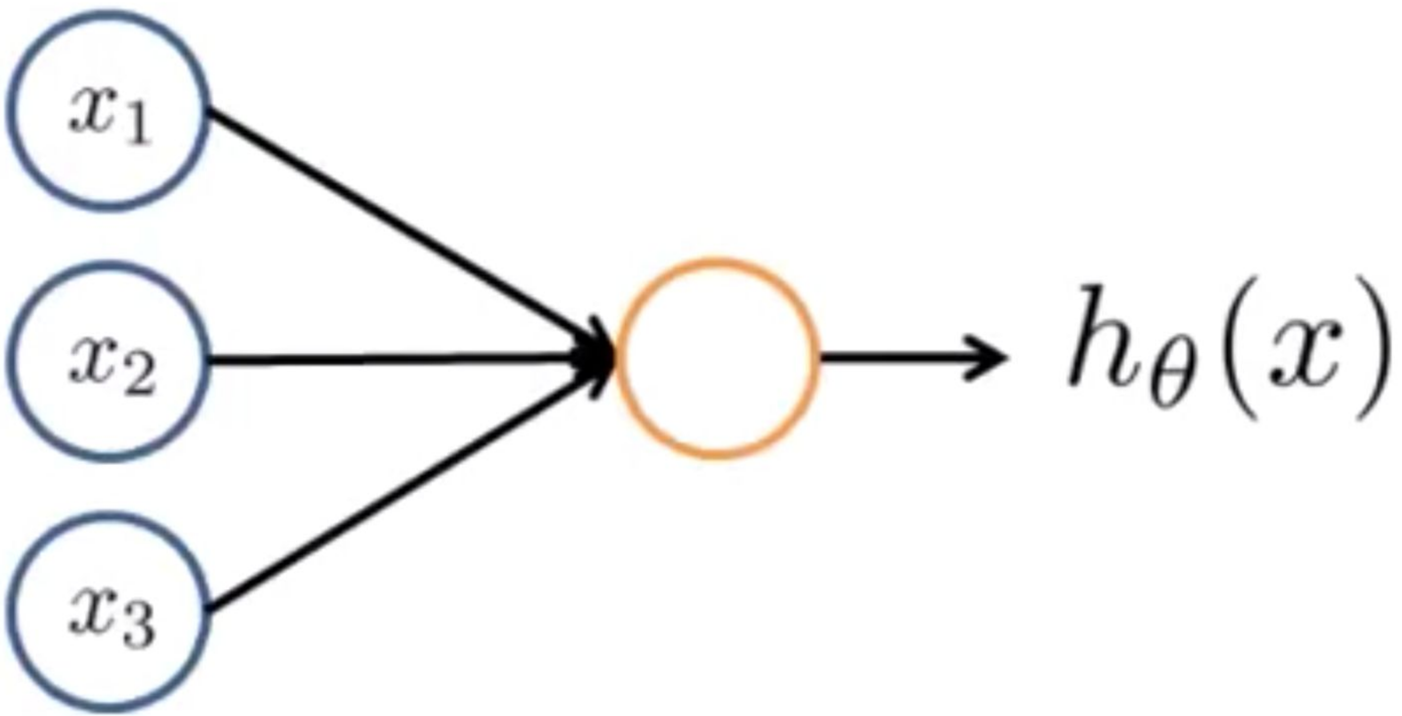
Neural Networks

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

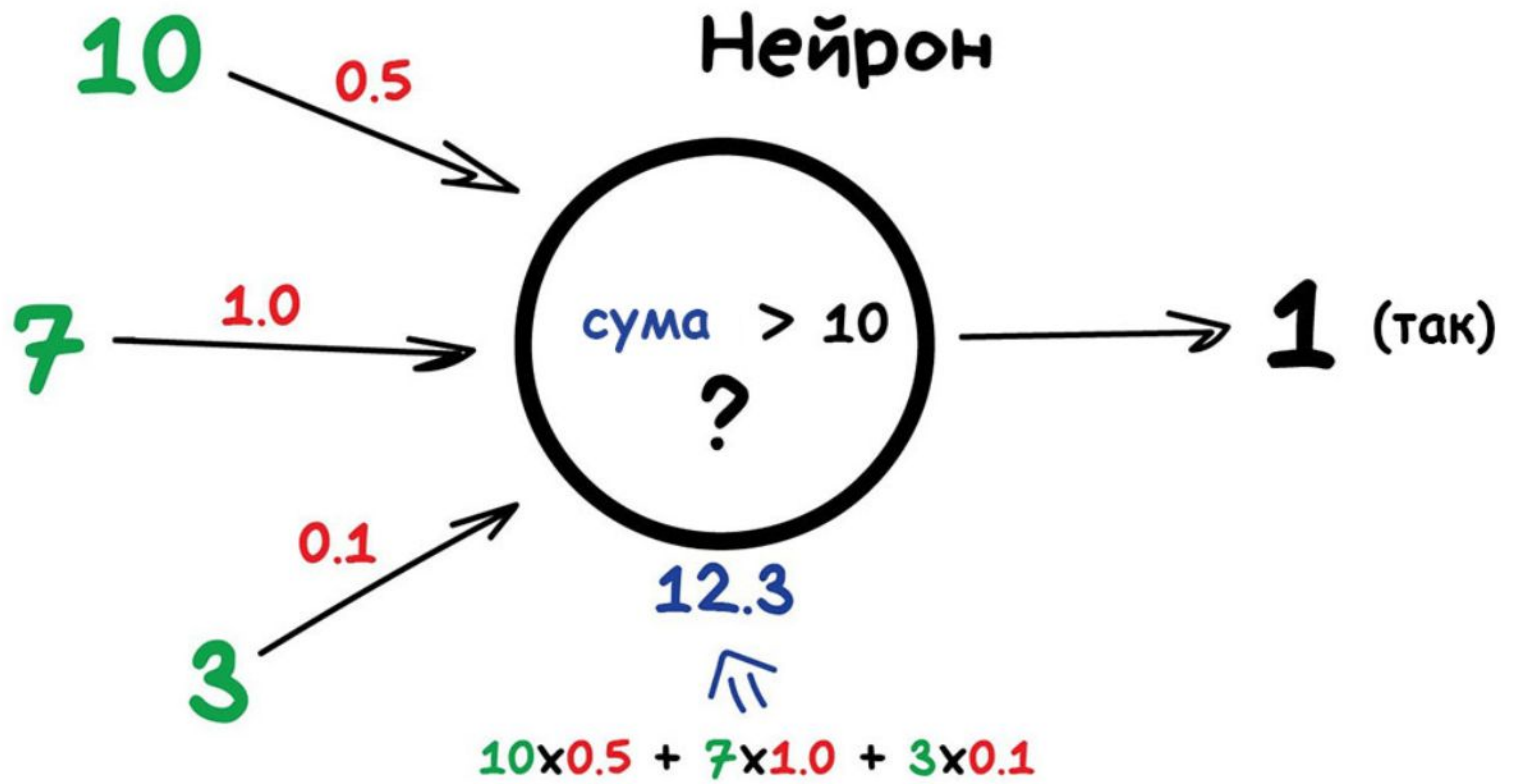
Neural of the Brain



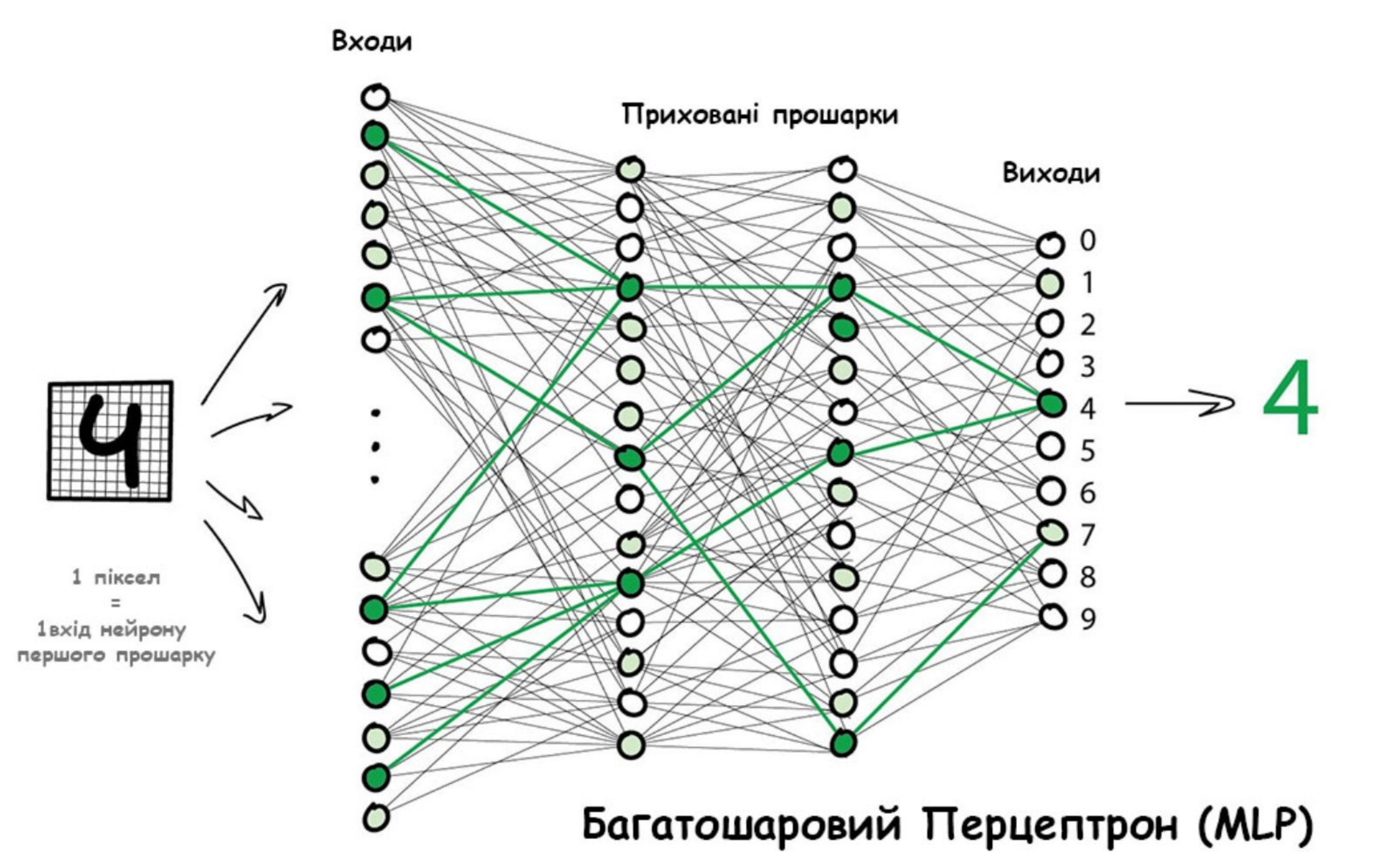
Neural model: Logistic Unit



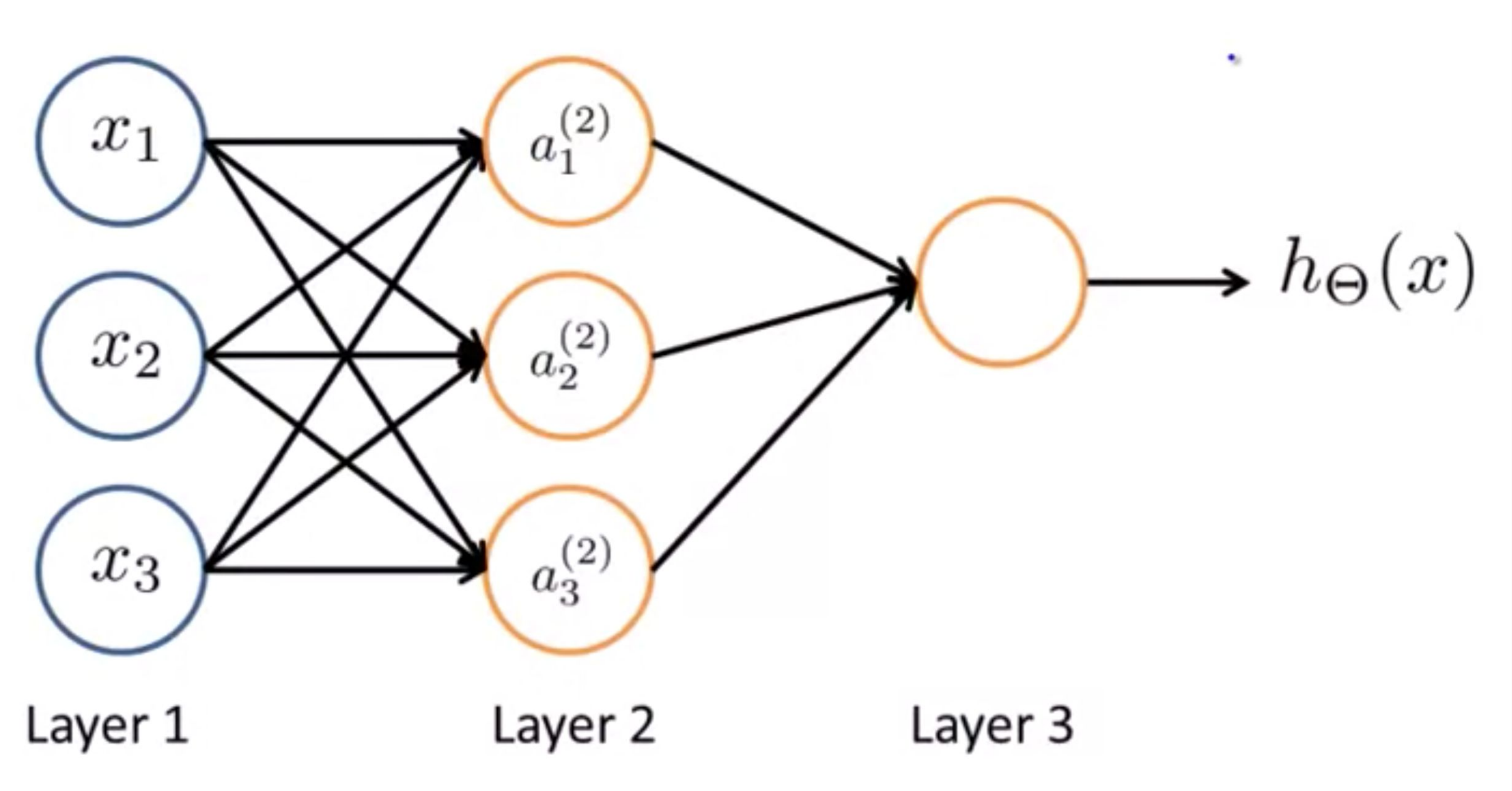
Perceptron



MLP


Входи
Приховані прошарки
Виходи
0
1
2
3
4
5
6
7
8
9
4
1 піксел
=
1вхід нейрону
першого прошарку
Багатошаровий Перцептрон (MLP)

Neural network



Neural network

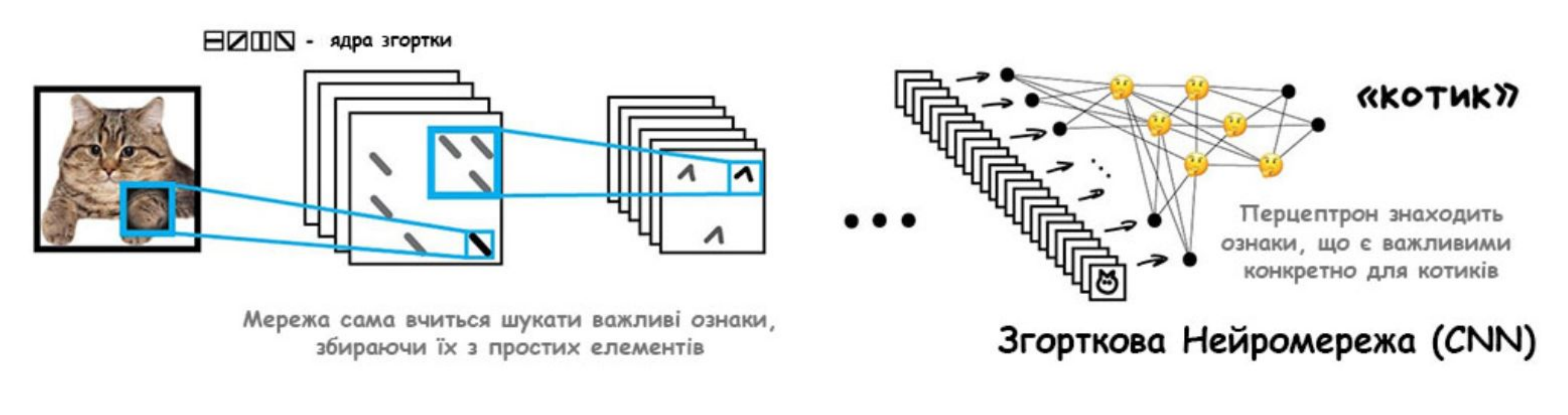
$$a_1^{(2)} = g(\Theta_{10}^{(1)} x_0 + \Theta_{11}^{(1)} x_1 + \Theta_{12}^{(1)} x_2 + \Theta_{13}^{(1)} x_3)$$

$$a_2^{(2)} = g(\Theta_{20}^{(1)} x_0 + \Theta_{21}^{(1)} x_1 + \Theta_{22}^{(1)} x_2 + \Theta_{23}^{(1)} x_3)$$

$$a_3^{(2)} = g(\Theta_{30}^{(1)} x_0 + \Theta_{31}^{(1)} x_1 + \Theta_{32}^{(1)} x_2 + \Theta_{33}^{(1)} x_3)$$

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

CNN


- ядра згортки
«котик»
Перцептрон знаходить ознаки, що є важливими конкретно для котиків
Мережа сама вчиться шукати важливі ознаки, збираючи їх з простих елементів
Згорткова Нейромережа (CNN)