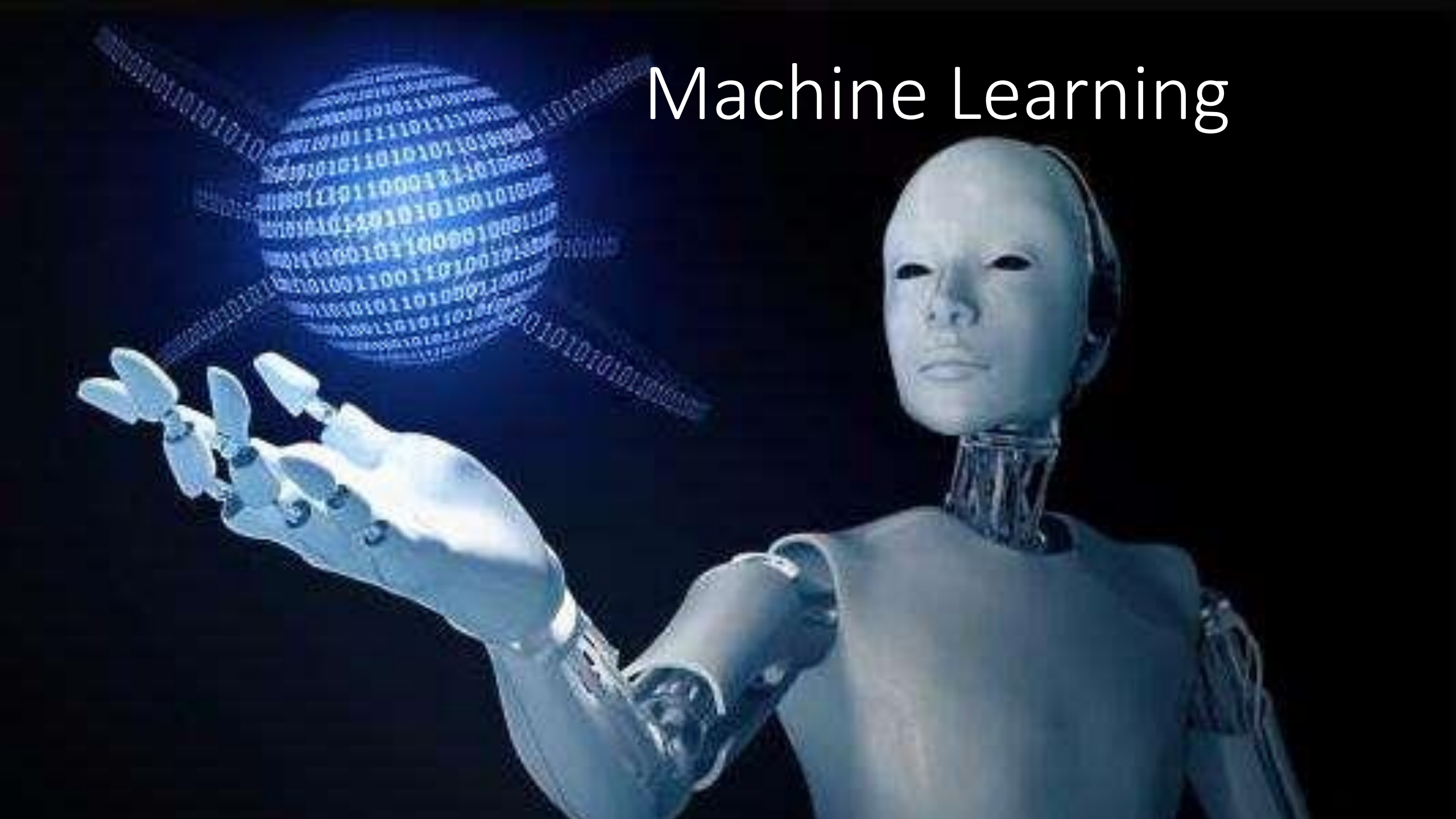
Machine Learning



Outline

- What is Machine Learning
- Categories of Machine Learning
- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning

Machine Learning

- The term machine learning was first coined by Arthur Lee Samuel：

Machine learning is the field of study that gives computers the ability **to learn without being explicitly programmed**.

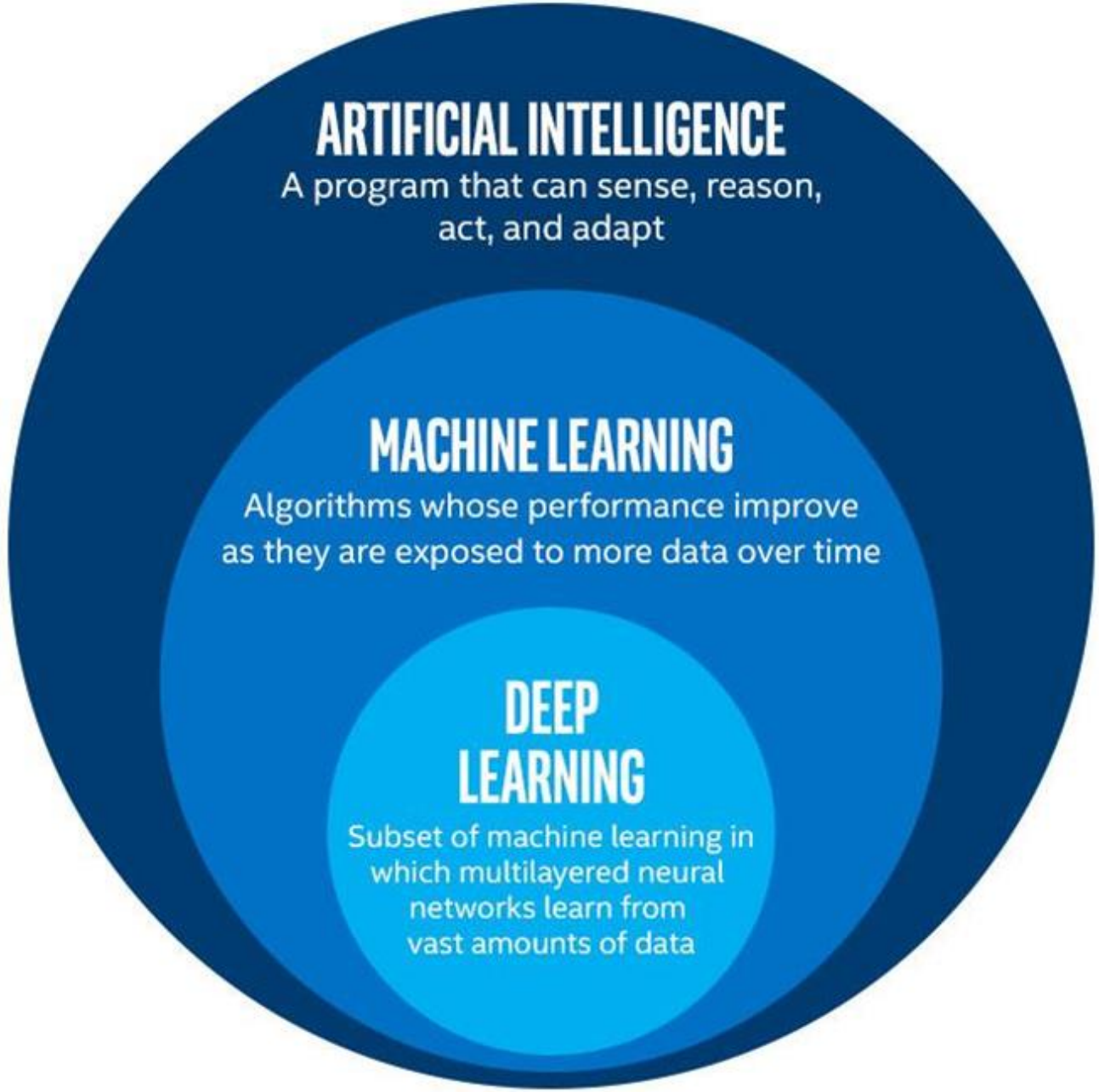
- Arthur L. Samuel, AI pioneer, 1959

Machine Learning

A computer program is said to **learn** from experience E with respect to some class of tasks T and performance measure P, if its **performance at tasks in T, as measured by P, improves with experience E**.

- Tom Mitchell, Machine Learning Professor at Carnegie Mellon University

Tom M Mitchell et al. “Machine learning. 1997". In: Burr Ridge, IL: McGraw Hill 45.37 (1997), pp. 870-877

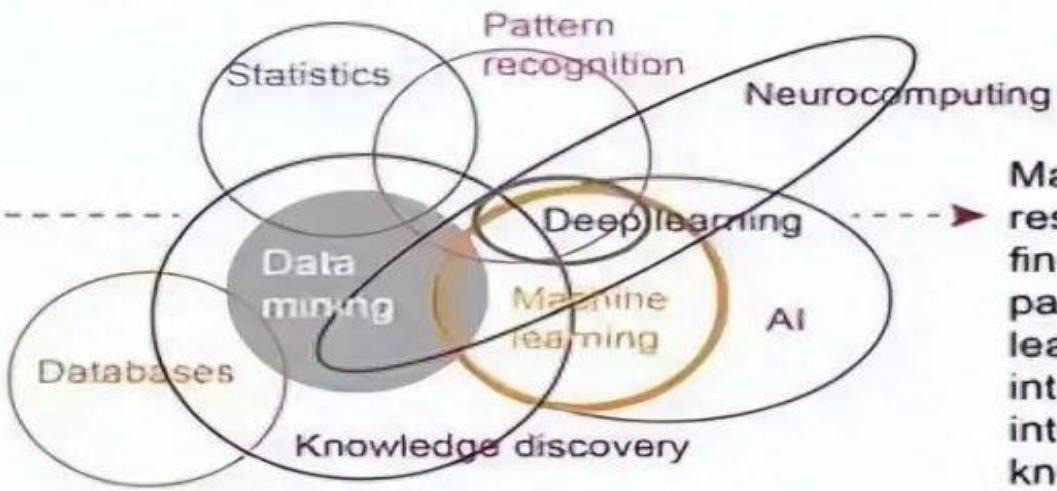

ARTIFICIAL INTELLIGENCE
A program that can sense, reason, act, and adapt
MACHINE LEARNING
Algorithms whose performance improve as they are exposed to more data over time
DEEP LEARNING
Subset of machine learning in which multilayered neural networks learn from vast amounts of data

A look at
Machine learning

What is it?

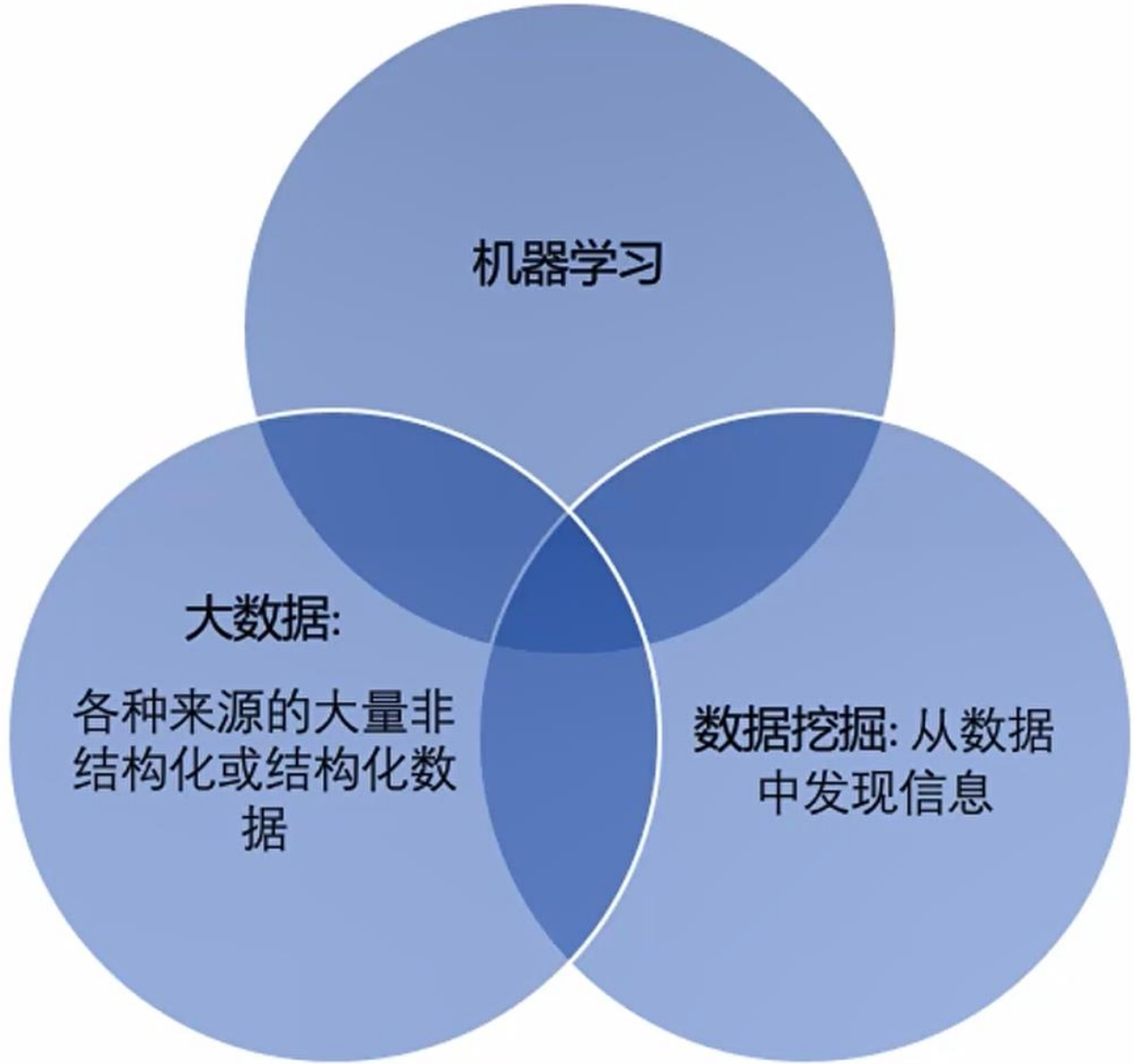
Machines can "learn" by analyzing large amounts of data. For example, rather than being programmed to recognize a cat or human face, they can be trained with images from which to generalize and recognize specific objects.

How does machine learning relate to artificial intelligence?



Source: SAS, 2014 and PwC, 2016

Machine learning is a category of research and algorithms focused on finding patterns in data and using those patterns to make predictions. Machine learning falls within the artificial intelligence (AI) umbrella, which in turn intersects with the broader field of knowledge discovery and data mining.


机器学习
大数据:
各种来源的大量非结构化或结构化数据
数据挖掘: 从数据中发现信息

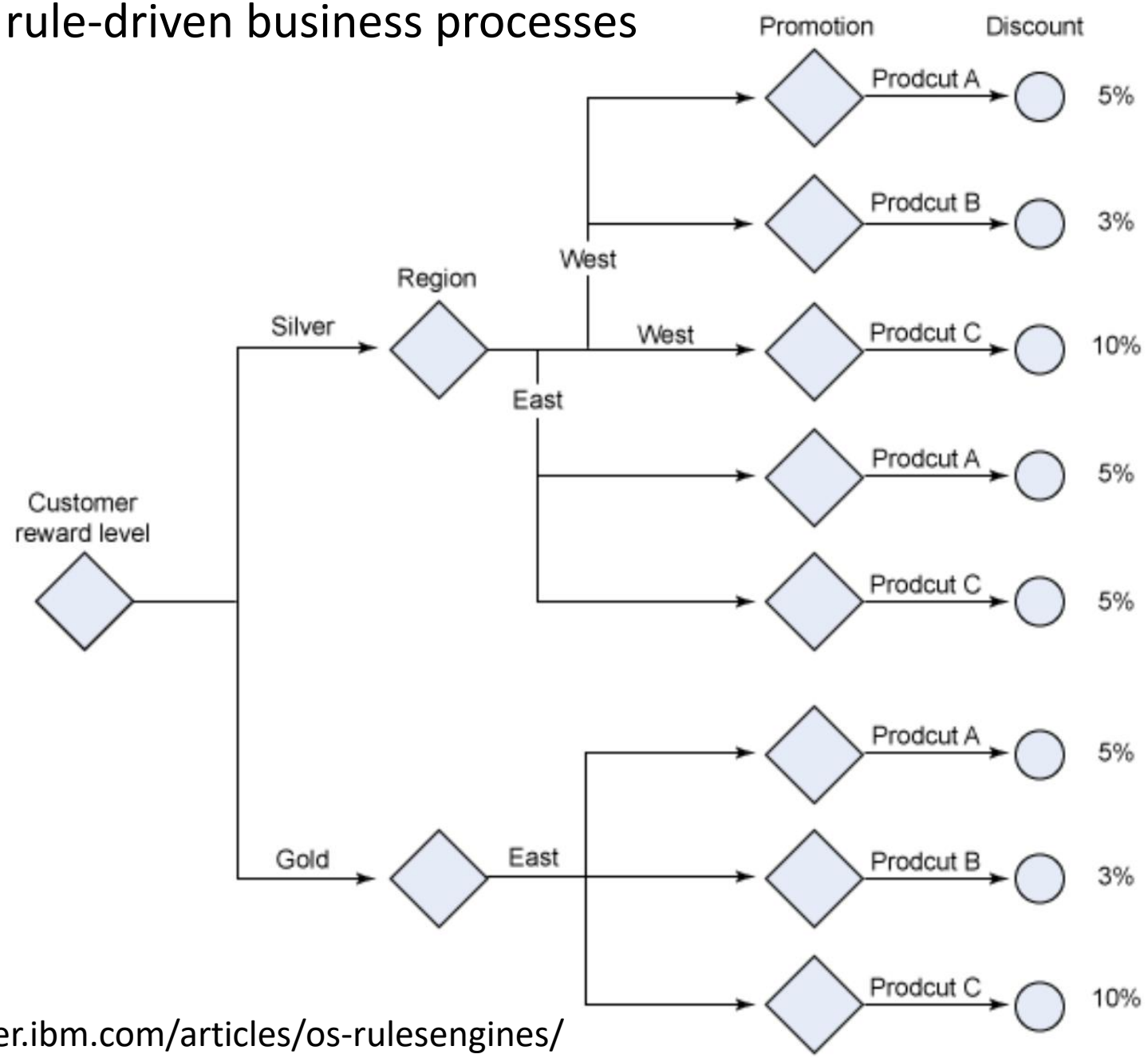
Outline

- What is Machine Learning
- Categories of Machine Learning
- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning

Machine Learning Algorithms

- Some common classes of machine learning algorithms:
  - Generalized linear models (e.g., logistic regression)
  - Support vector machines (e.g., linear SVM, RBF-kernel SVM)
  - Artificial neural networks (e.g., multi-layer perceptrons)
  - Tree- or rule-based models (e.g., decision trees)
  - Graphical models (e.g., Bayesian networks)
  - Ensembles (e.g., Random Forest)
  - Instance-based learners (e.g., K-nearest neighbors)

An Example of rule-driven business processes



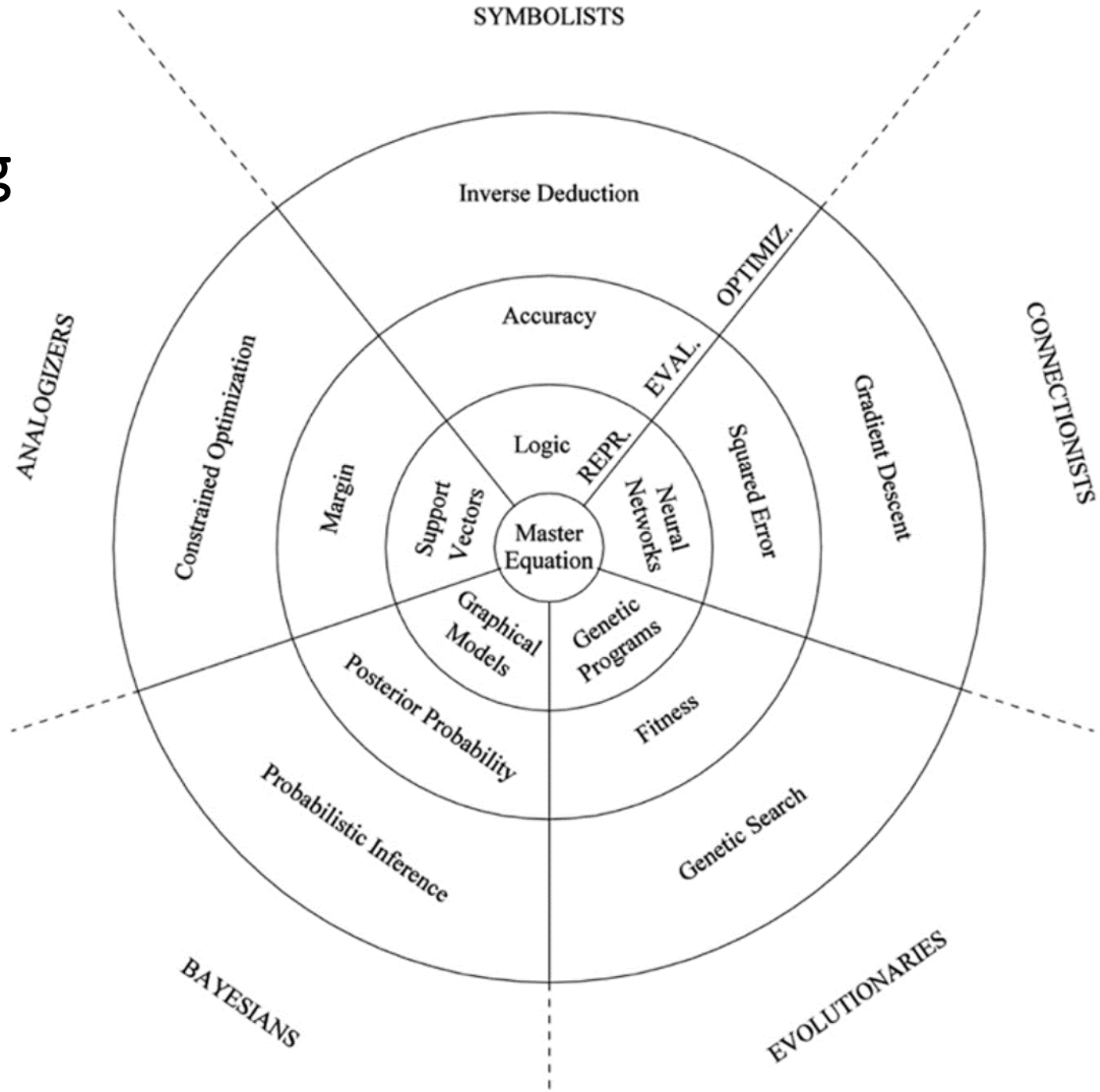
From https://developer.ibm.com/articles/os-rulesengines/

Tribes of Machine Learning

REPR.=Representation

EVAL.=Evaluation metric

OPTIMIZ.=Optimization algorithm.

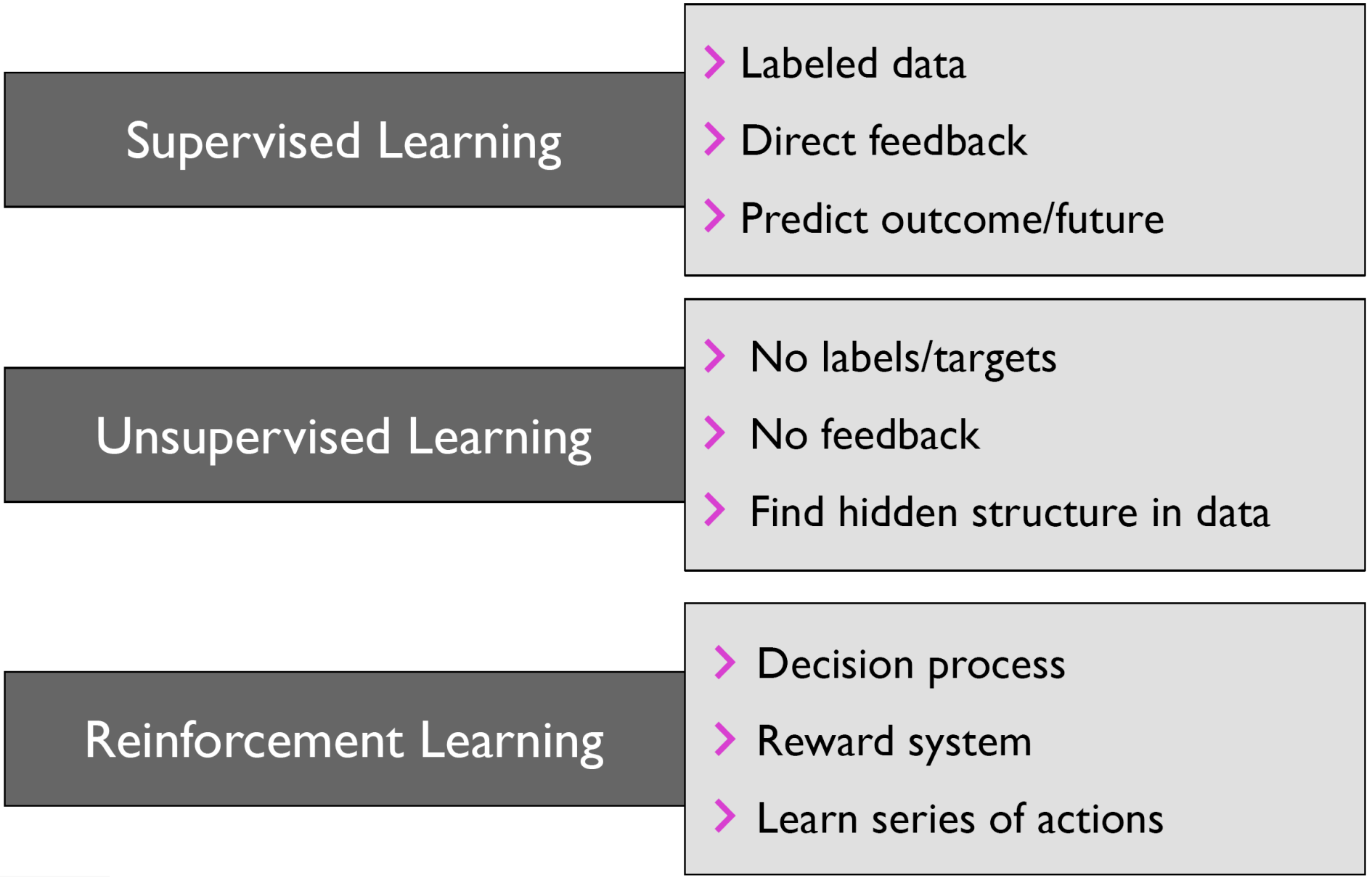


Pedro Domingos. The master algorithm: How the quest for the ultimate learning machine will remake our world. 2015.

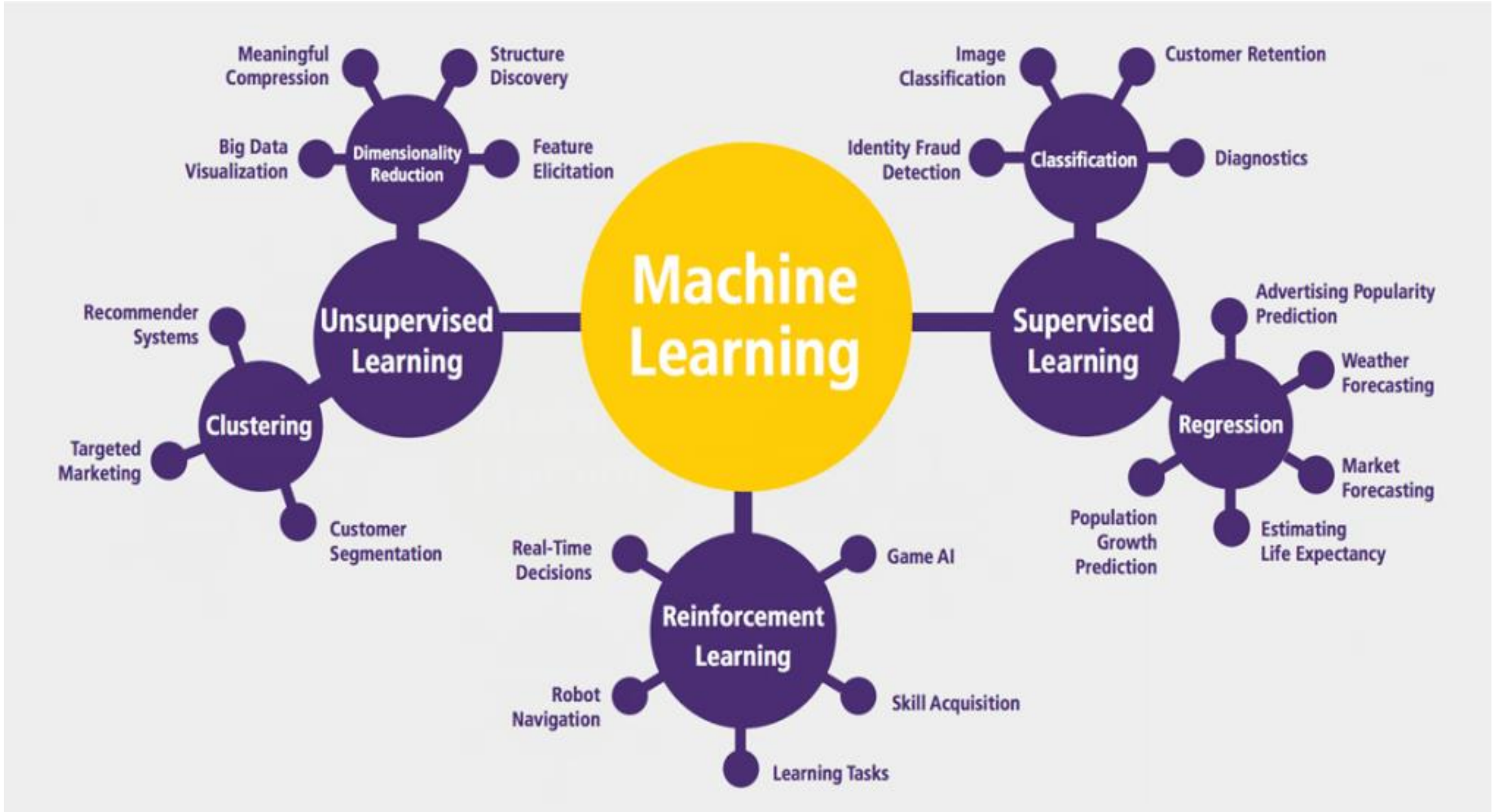
Categories of learning task

by Geoffrey Hinton

- Supervised Learning
  - Learn to predict an output when given an input vector
- Unsupervised Learning
  - Learn to discover a good internal representation of the input
- Reinforcement Learning
  - Learn to select an action to maximize payoff



Raschka & Mirjalili: Python Machine Learning, 3rd Ed.


Machine Learning
Unsupervised Learning
Dimensionality Reduction
Meaningful Compression
Structure Discovery
Big Data Visualization
Feature Elicitation
Clustering
Recommender Systems
Targeted Marketing
Customer Segmentation
Supervised Learning
Classification
Image Classification
Customer Retention
Identity Fraud Detection
Diagnostics
Regression
Advertising Popularity Prediction
Weather Forecasting
Market Forecasting
Estimating Life Expectancy
Population Growth Prediction
Reinforcement Learning
Real-Time Decisions
Game AI
Robot Navigation
Skill Acquisition
Learning Tasks

Yann Lecun's "Learning Cake"

"Pure" Reinforcement Learning (cherry)

- The machine predicts a scalar reward given once in a while.
- **A few bits for some samples**

Supervised Learning (icing)

- The machine predicts a category or a few numbers for each input
- Predicting human-supplied data
- **10→10,000 bits per sample**

Self-Supervised Learning (cake génoise)

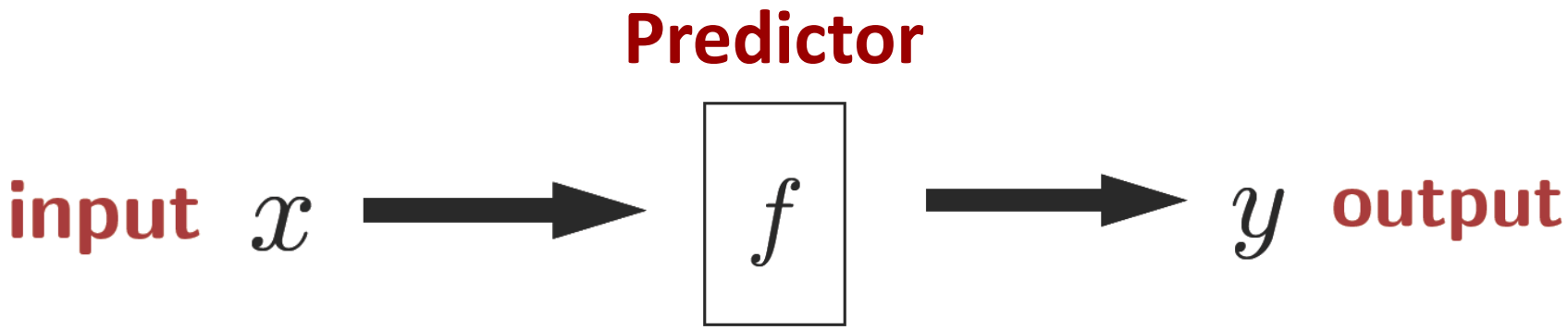
- The machine predicts any part of its input for any observed part.
- Predicts future frames in videos
- **Millions of bits per sample**

Outline

- What is Machine Learning
- Categories of Machine
- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning

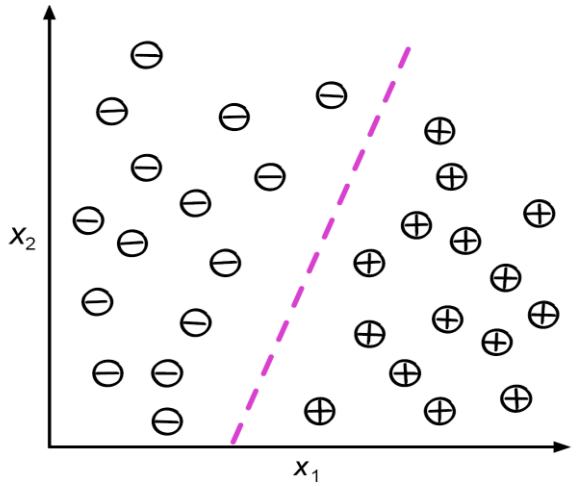
Supervised Learning

- Given a data set of **input-output pairs**, learn a function to map inputs to outputs.

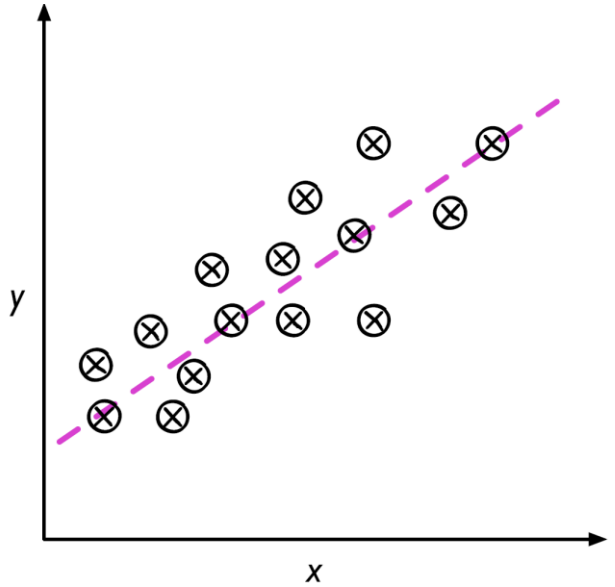


Different tasks of supervised learning

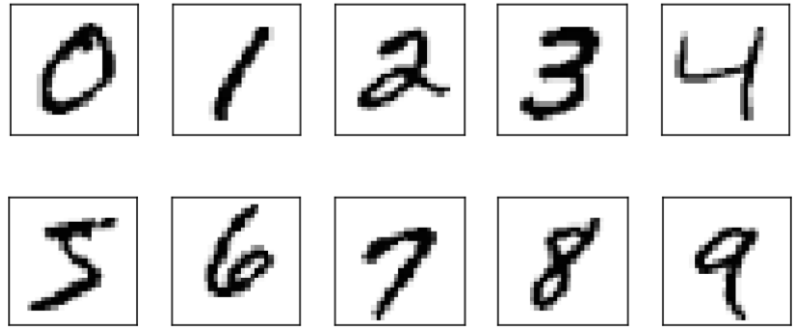
- Classification
  - learning a function mapping an input point to a discrete category



- Regression
  - learning a function mapping an input point to a continuous value



- Binary classification



- Multiclass classification

A generalization of binary classification where the output *y* could be one of *K* possible values.

Classification

Humidity (relative humidity)	Pressure (sea level, mb)	Rain or not
93%	999.7	Rain
49%	1015.5	No Rain
79%	1031.1	No Rain
65%	984.9	Rain
90%	975.2	Rain

- f(humidity,pressure)

  f(93, 999.7) = Rain

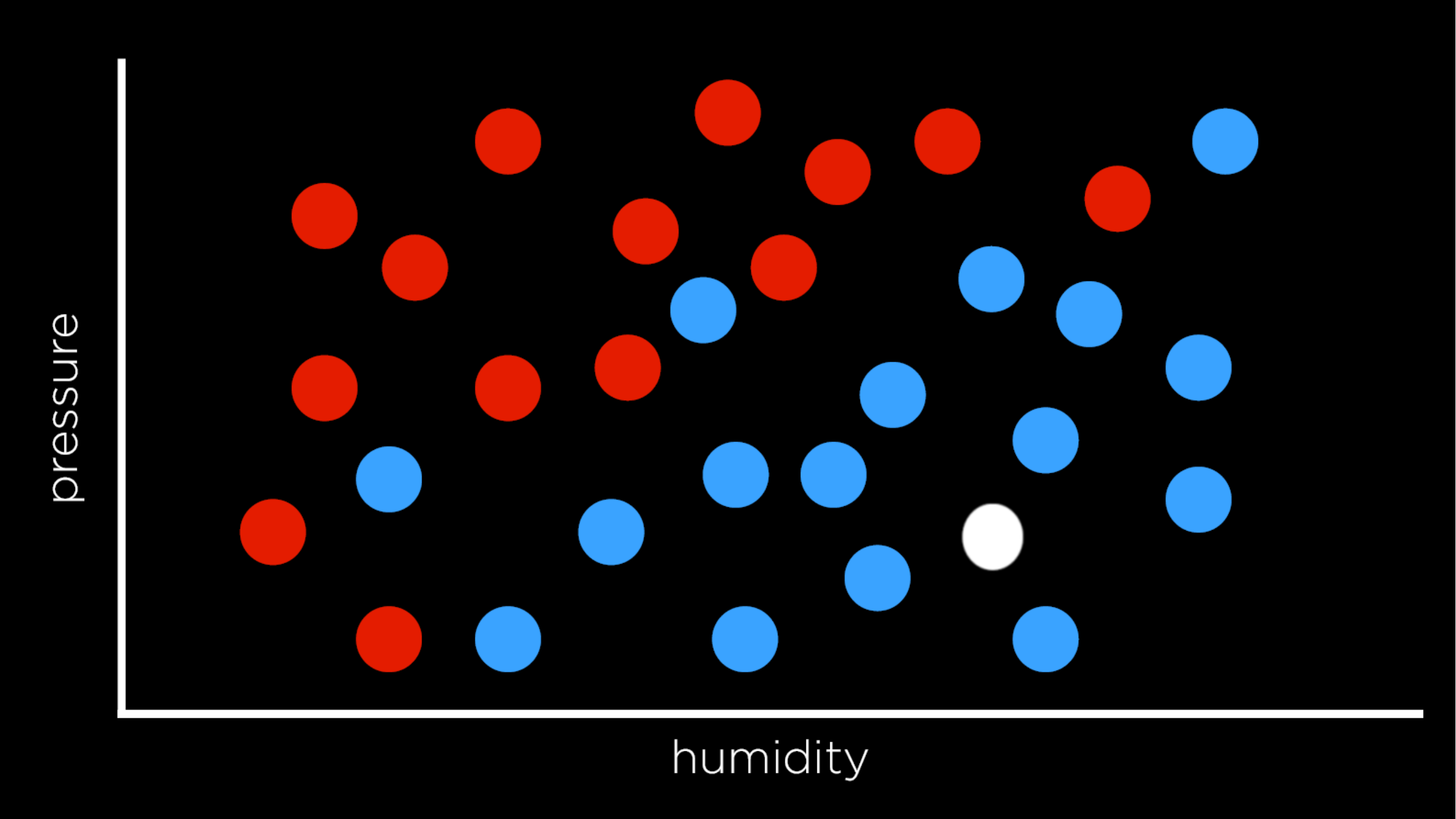
  f(49, 1015.5) = No Rain

  f(79, 1031.1) = No Rain

- h(humidity, pressure)

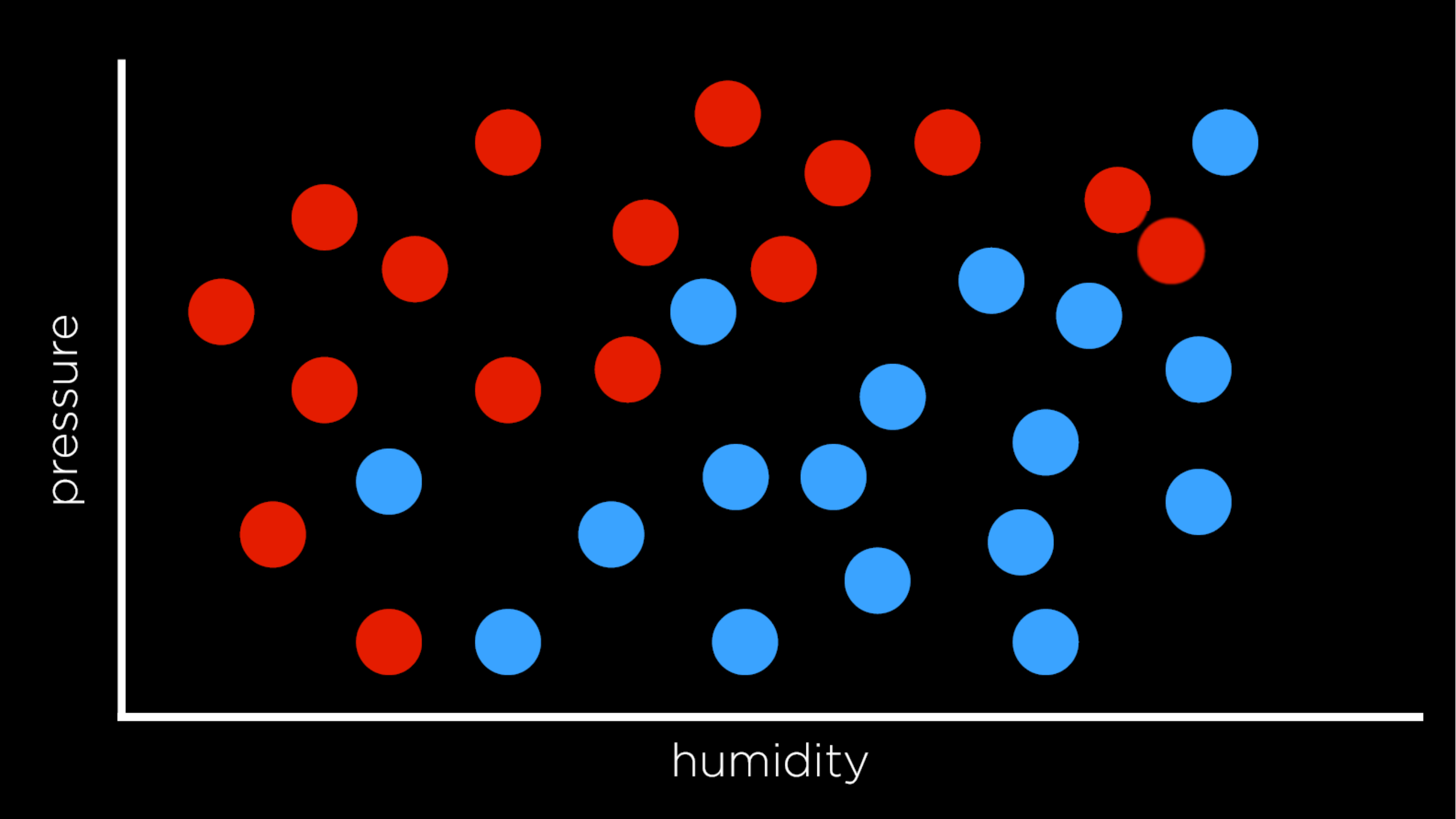
→ f(humidity,pressure)

Hypothesis : a certain function that we believe (or hope) is similar to the true function.


pressure
humidity

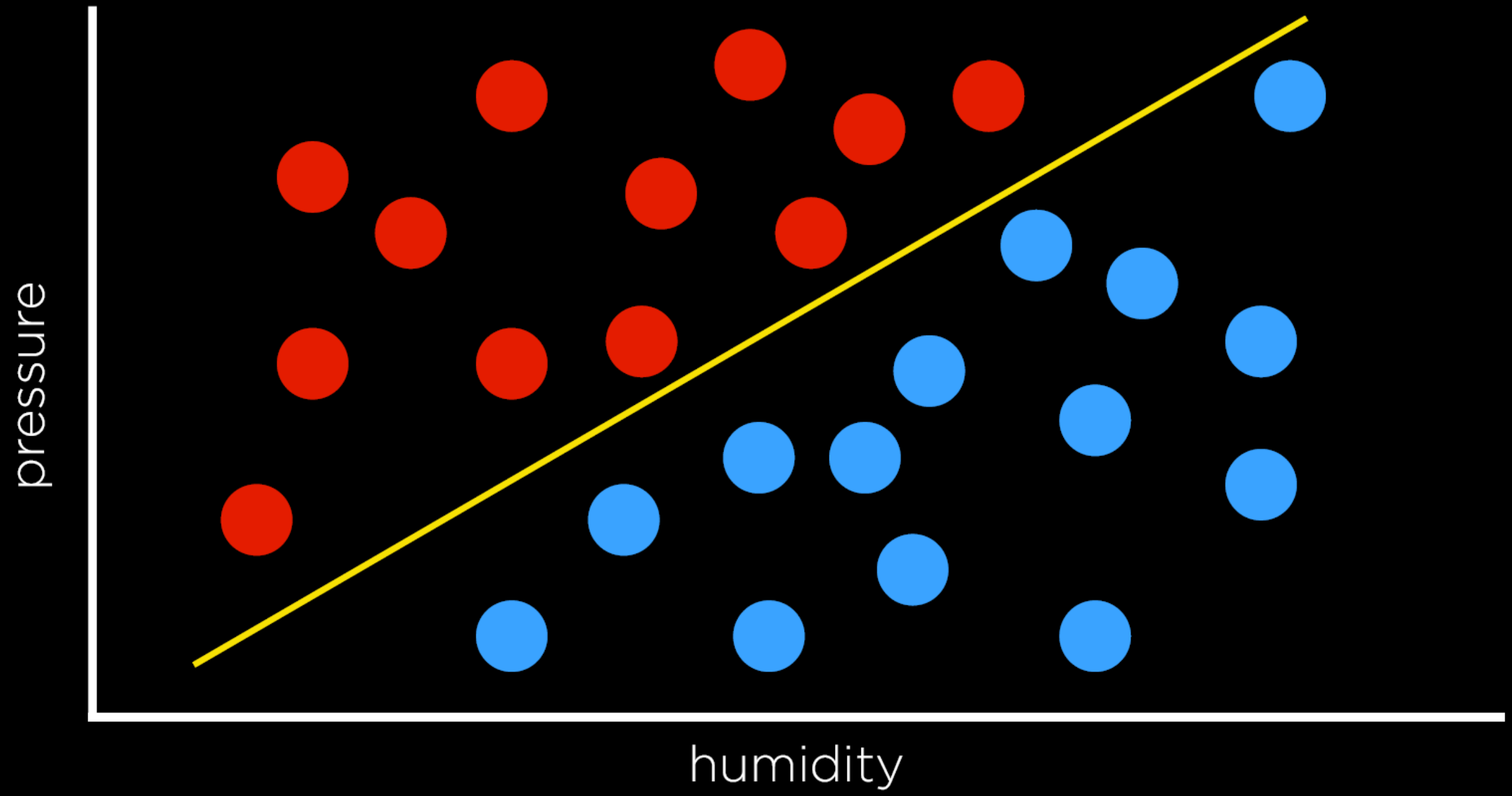
Nearest-neighbor classification

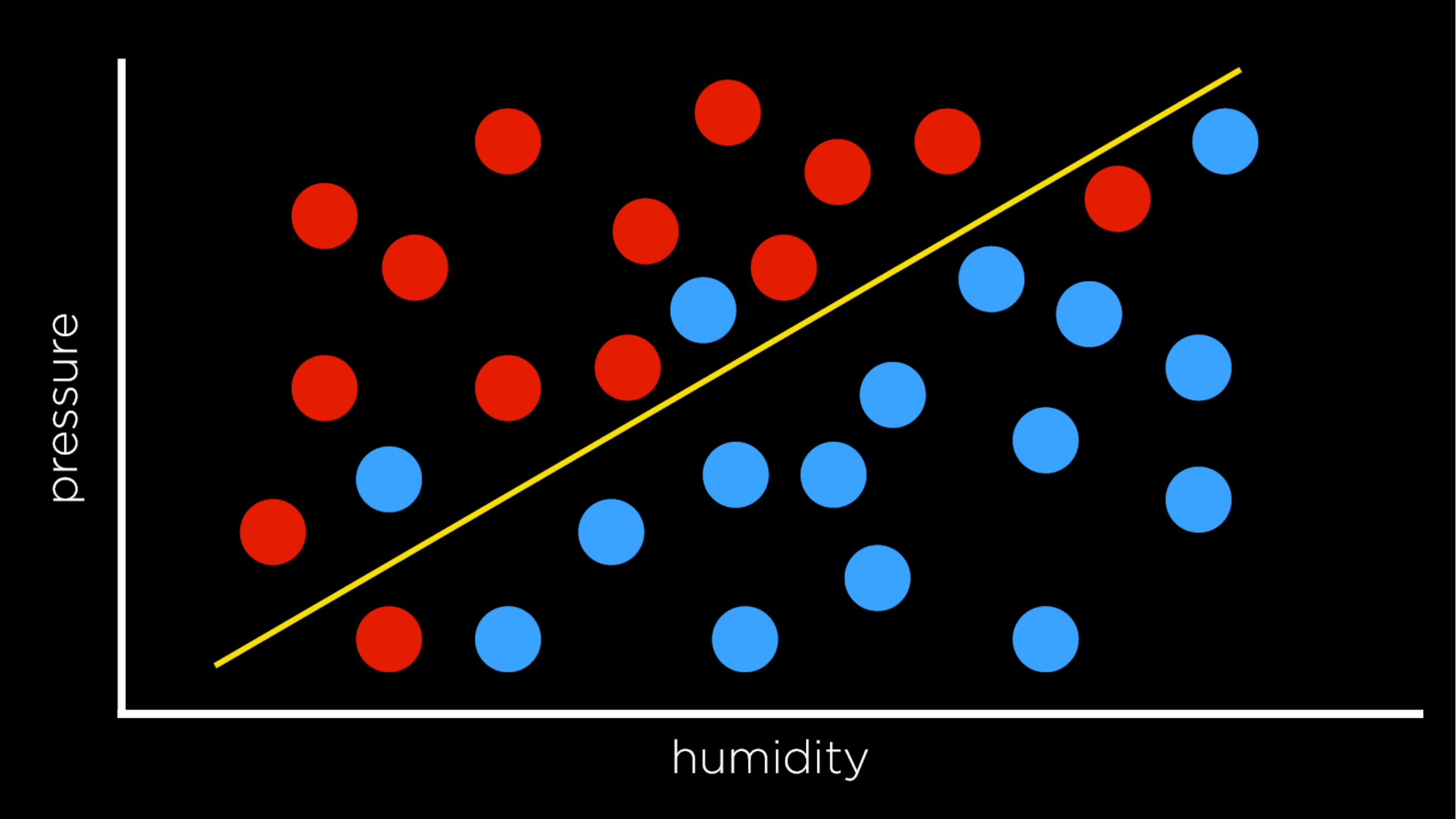
- given an input, chooses the class of the nearest data point to that input


pressure
humidity

K-nearest-neighbor classification

- given an input, chooses the most common class out of the k nearest data points to that input


pressure
humidity


pressure
humidity

- $h(x1, x2) = \begin{cases} \text{Rain}, & \text{if } w0 + w1x1 + w2x2 \geq 0 \\ \text{No Rain}, & \text{otherwise} \end{cases}$

$x1$ = Humidity

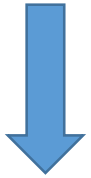
$x2$ = Pressure

Weight Vector ***W***: (w0, w1, w2)

Input Vector ***X***: (1, x1, x2)

- $\mathbf{W} \cdot \mathbf{X}$: $w0 + w1x1 + w2x2$

$$h(x1, x2) = \begin{cases} \text{Rain}, & \text{if } w0 + w1x1 + w2x2 \geq 0 \\ \text{No Rain}, & \text{otherwise} \end{cases}$$



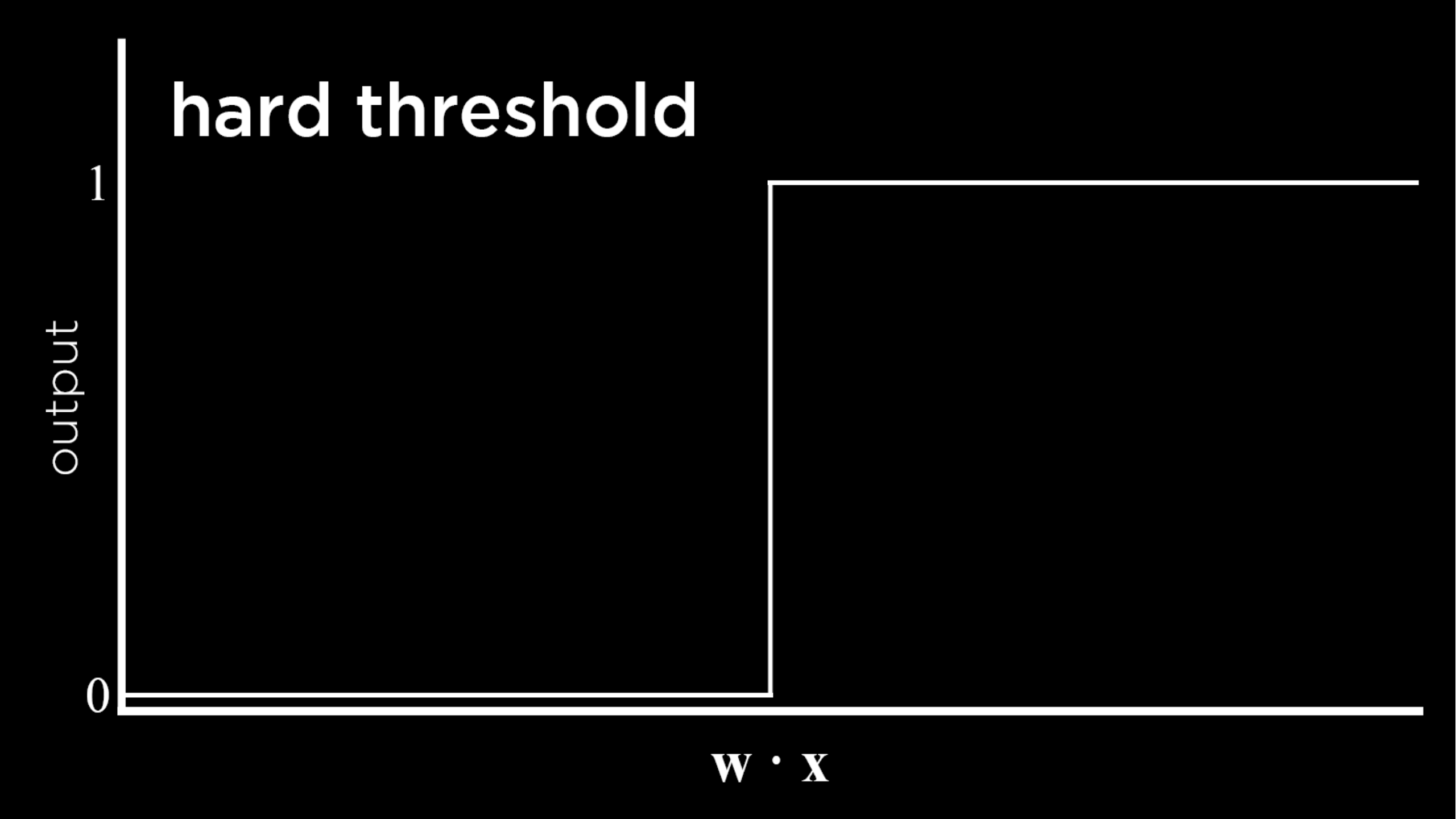
- $$h(x1, x2) = \begin{cases} 1, & \text{if } \mathbf{W} \cdot \mathbf{X} \geq 0 \\ 0, & \text{otherwise} \end{cases}$$

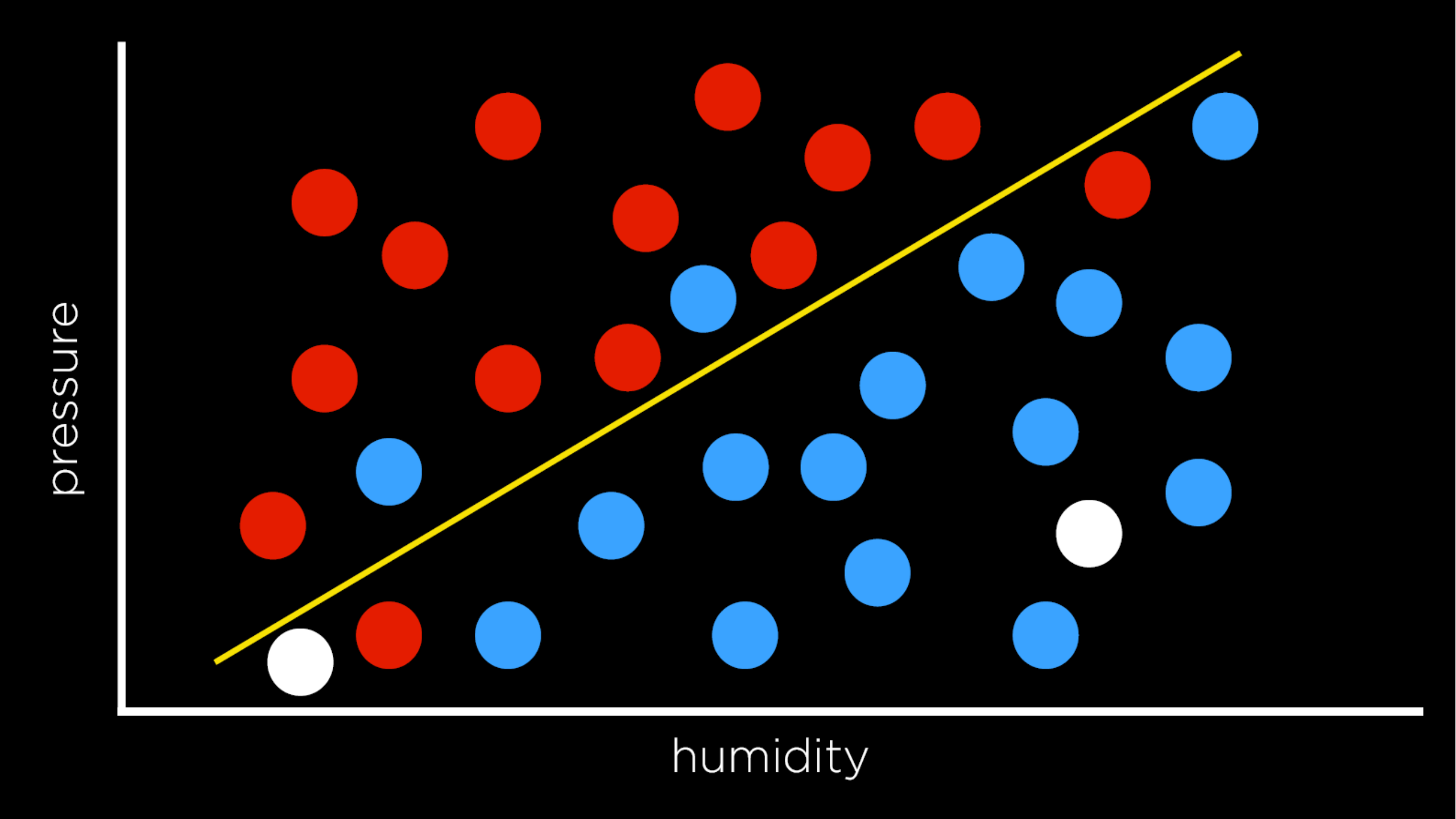
Perceptron learning rule

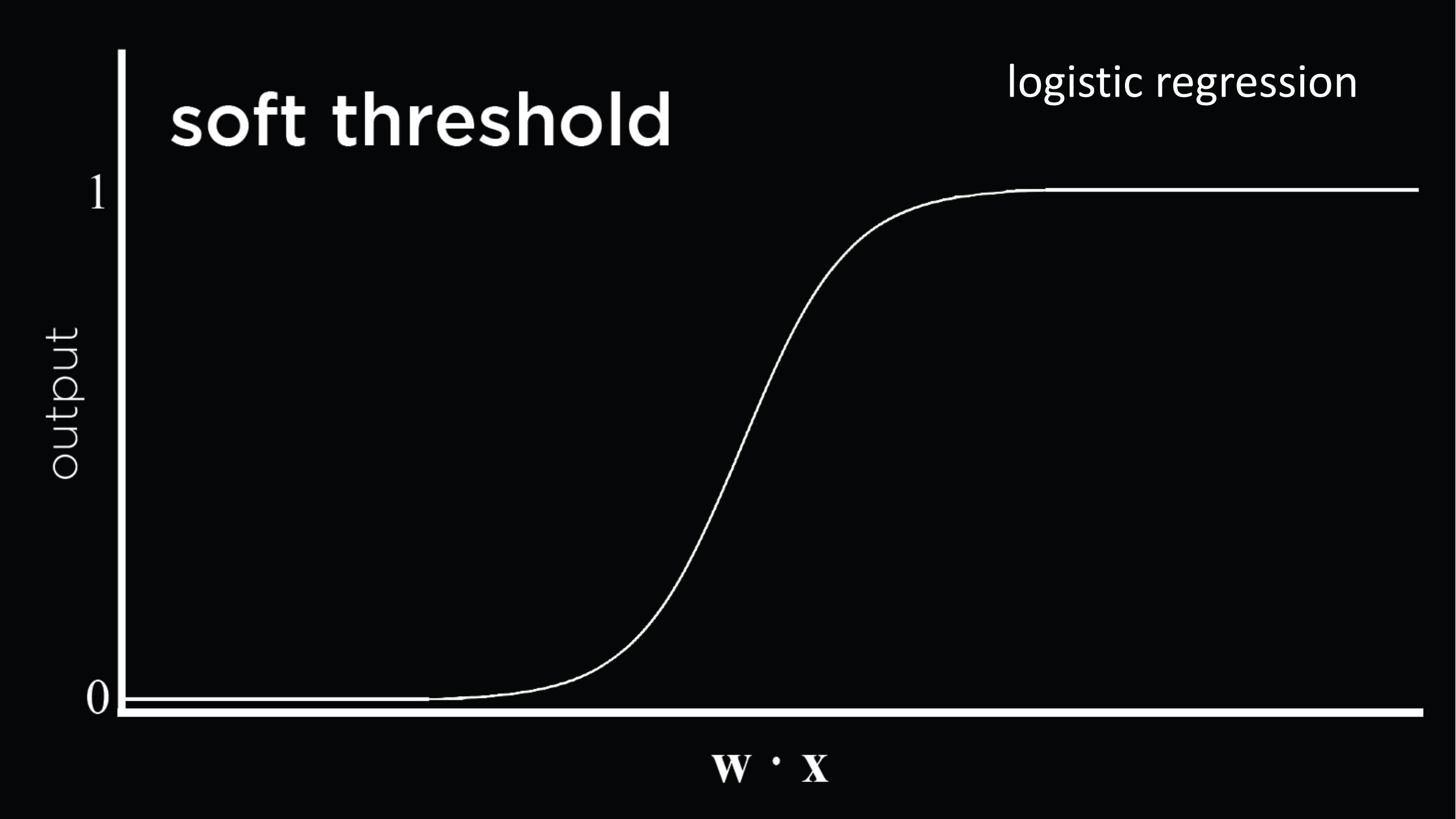
- Given data point (x, y), update each weight according to

$$w_i = w + \alpha(y - h_w(x)) \times x_i$$

$$w_i = w + \alpha(\textit{actual value} - \textit{estimate}) \times x_i$$

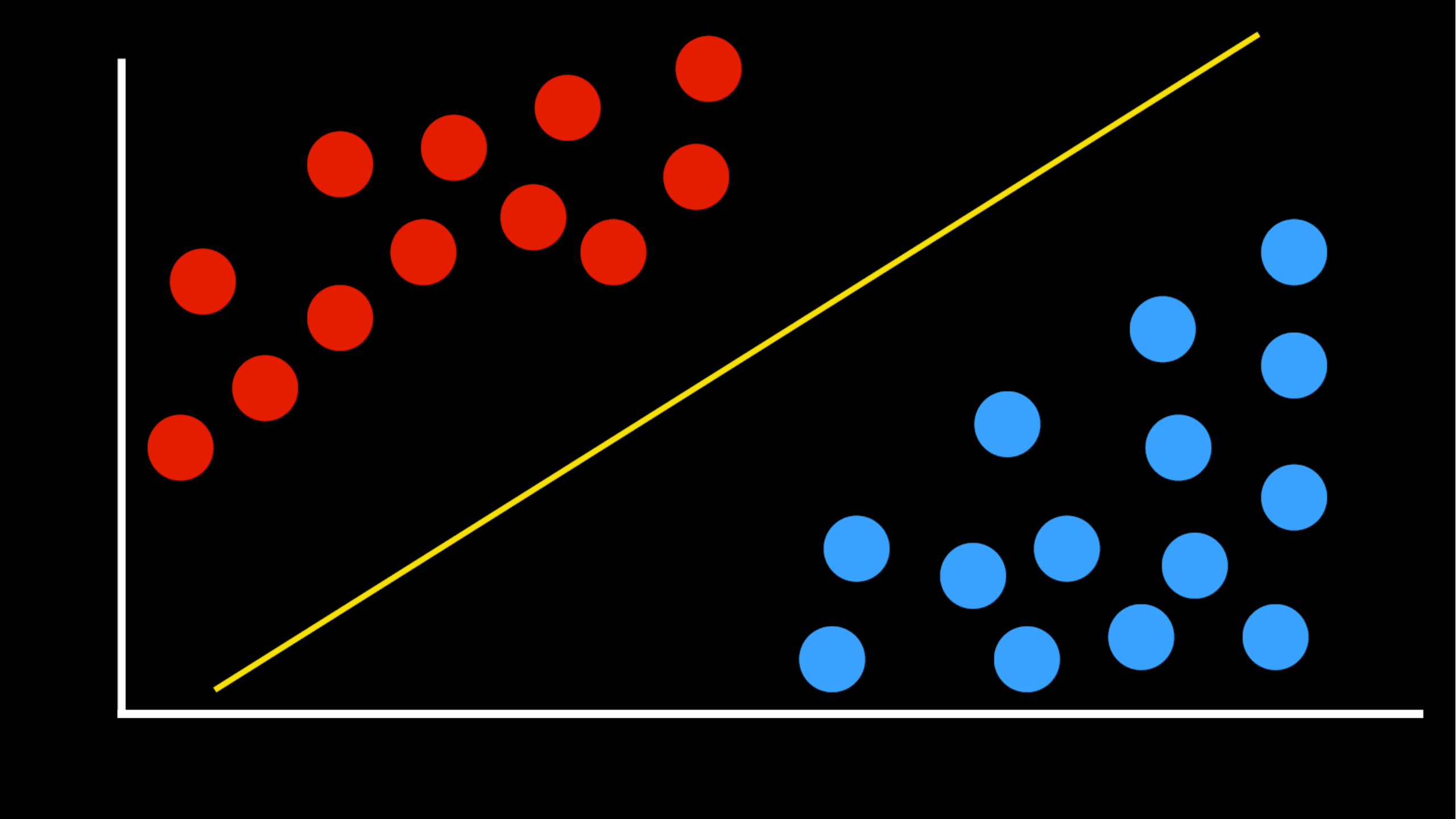

hard threshold
1
0
output
w · x

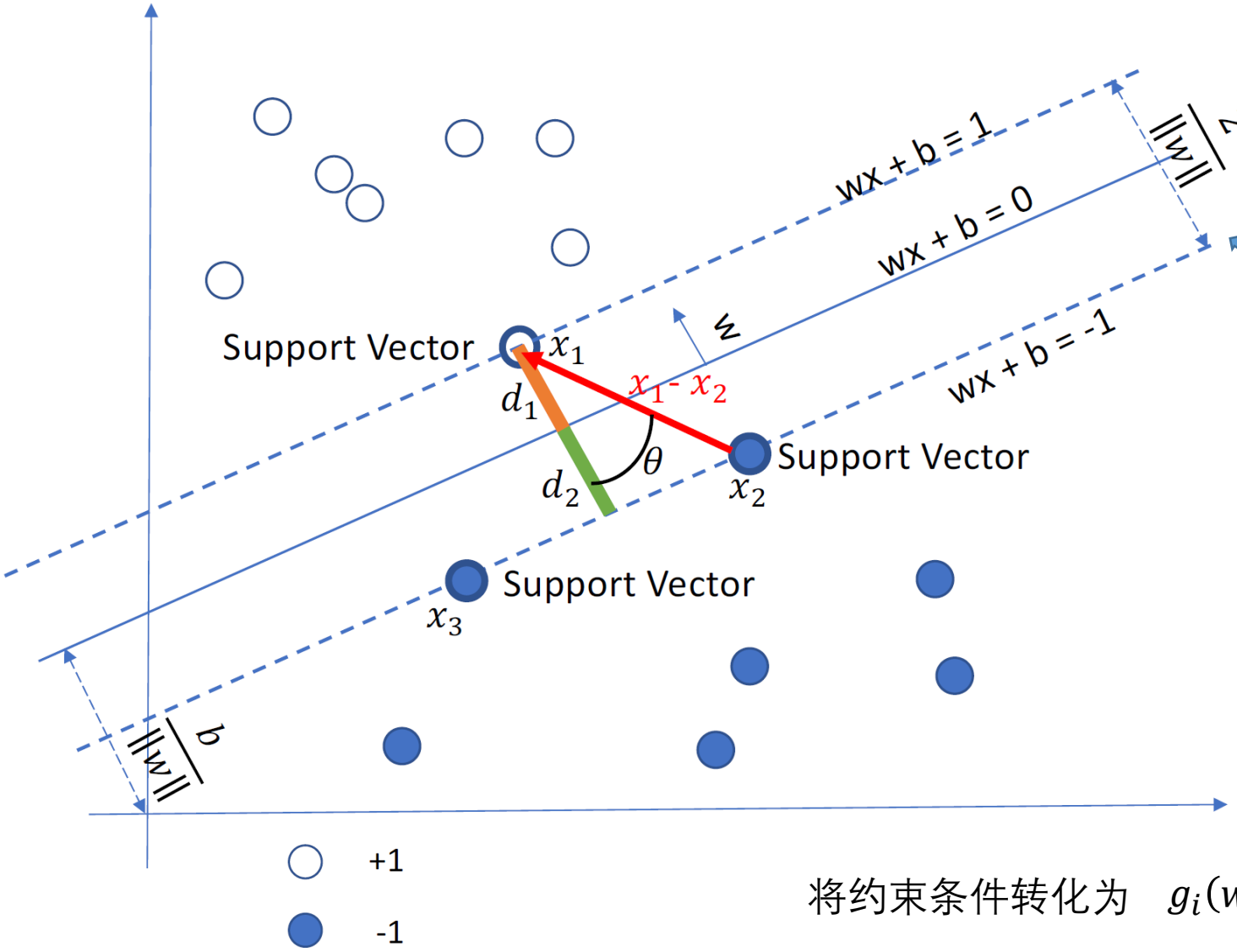

pressure
humidity


soft threshold
logistic regression
1
0
output
w · x

Support Vector Machines

- maximum margin separator
- boundary that maximizes the distance between any of the data points





$$w^T x_1 + b = 1 \quad \overset{sub}{\Longrightarrow} \quad w^T(x_1 - x_2) = 2$$

$$w^T x_2 + b = -1$$

$$w^T(x_1 - x_2) = \|w\|_2 \|x_1 - x_2\|_2 \cos\theta = 2$$

$$\|x_1 - x_2\|_2 \cos\theta = \frac{2}{\|w\|_2}$$

$$d_1 + d_2 = \|x_1 - x_2\|_2 \cos\theta = \frac{2}{\|w\|_2}$$

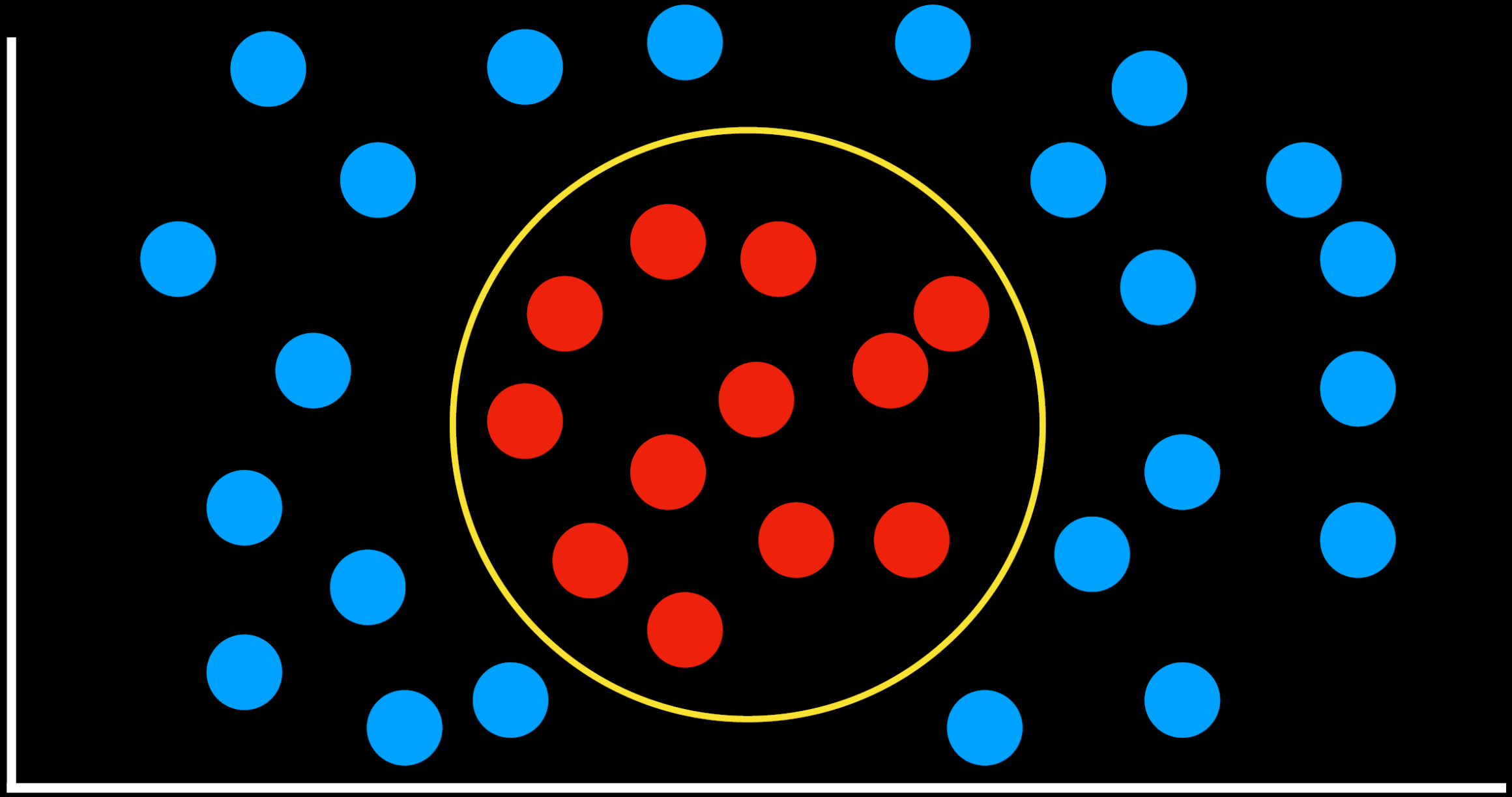
SVM的数学模型:

$$\min_{w,b} \frac{1}{2}\|w\|^2$$

s.t. $y^{(i)}(w^T x^{(i)} + b) \geq 1, i = 1,\ldots,n$

将约束条件转化为 $g_i(w) = -y^{(i)}(w^T x^{(i)} + b) + 1 \leq 0, i = 1,\ldots,n$

可将约束问题拉格朗日化 $L(w,b,\alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} \alpha_i \left[y^{(i)}(w^T x^{(i)} + b) - 1 \right]$



Different tasks of supervised learning

- Classification
  - Learning a function mapping an input point to a discrete category
- Regression
  - learning a function mapping an input point to a continuous value

Regression

$$x \longrightarrow \boxed{f} \longrightarrow y \in \mathbb{R} \text{ response}$$

e.g. The prediction of Housing Price

The arriving time of delivery

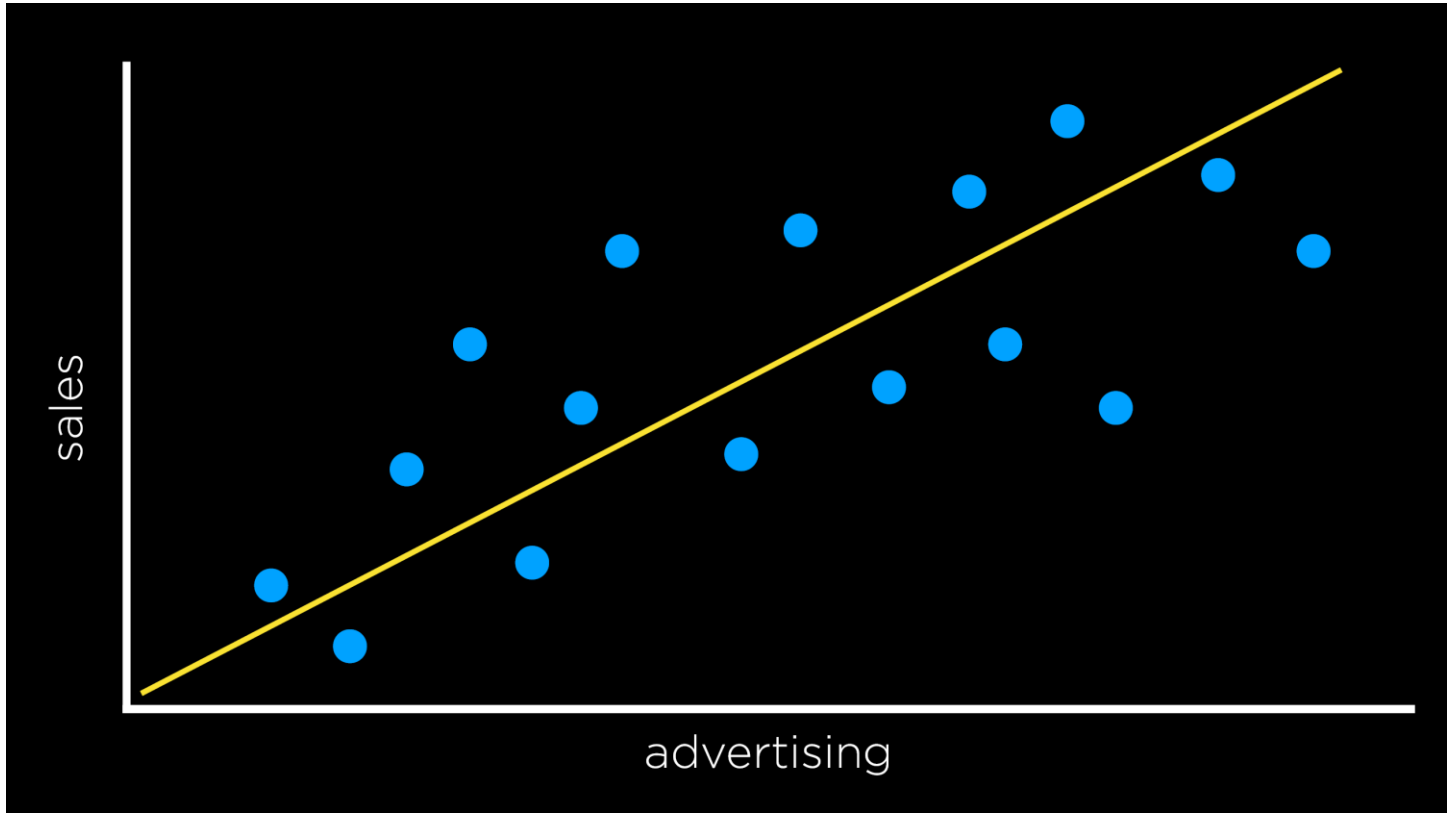
- f(advertising)

  f(1200) = 5800

  f(2800) = 13400

  f(1800) = 8400

- h(advertising)



Evaluating Hypotheses

- How to tell whether our hypothesis is good/bad?
- loss function /cost function / objective
- A loss function tells how poorly our hypothesis performs
  - high： a poor job
  - low： doing well

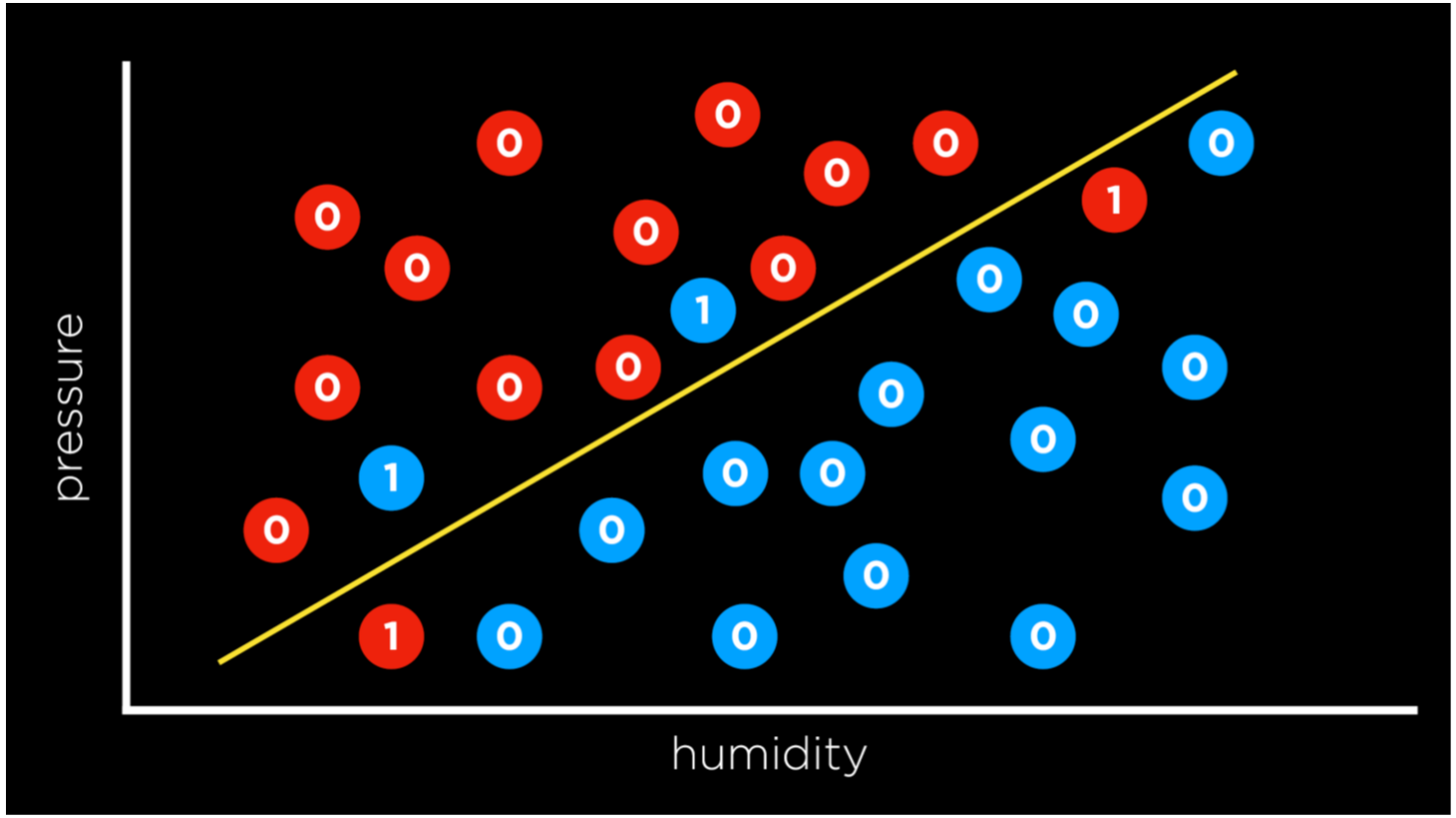
Loss function

- Quantifying what it means to have a “good” W

- **Loss function：**
  - Measure the quality of a particular set of parameters W
  - Based on how well the induced scores agreed with the ground truth labels in the training data

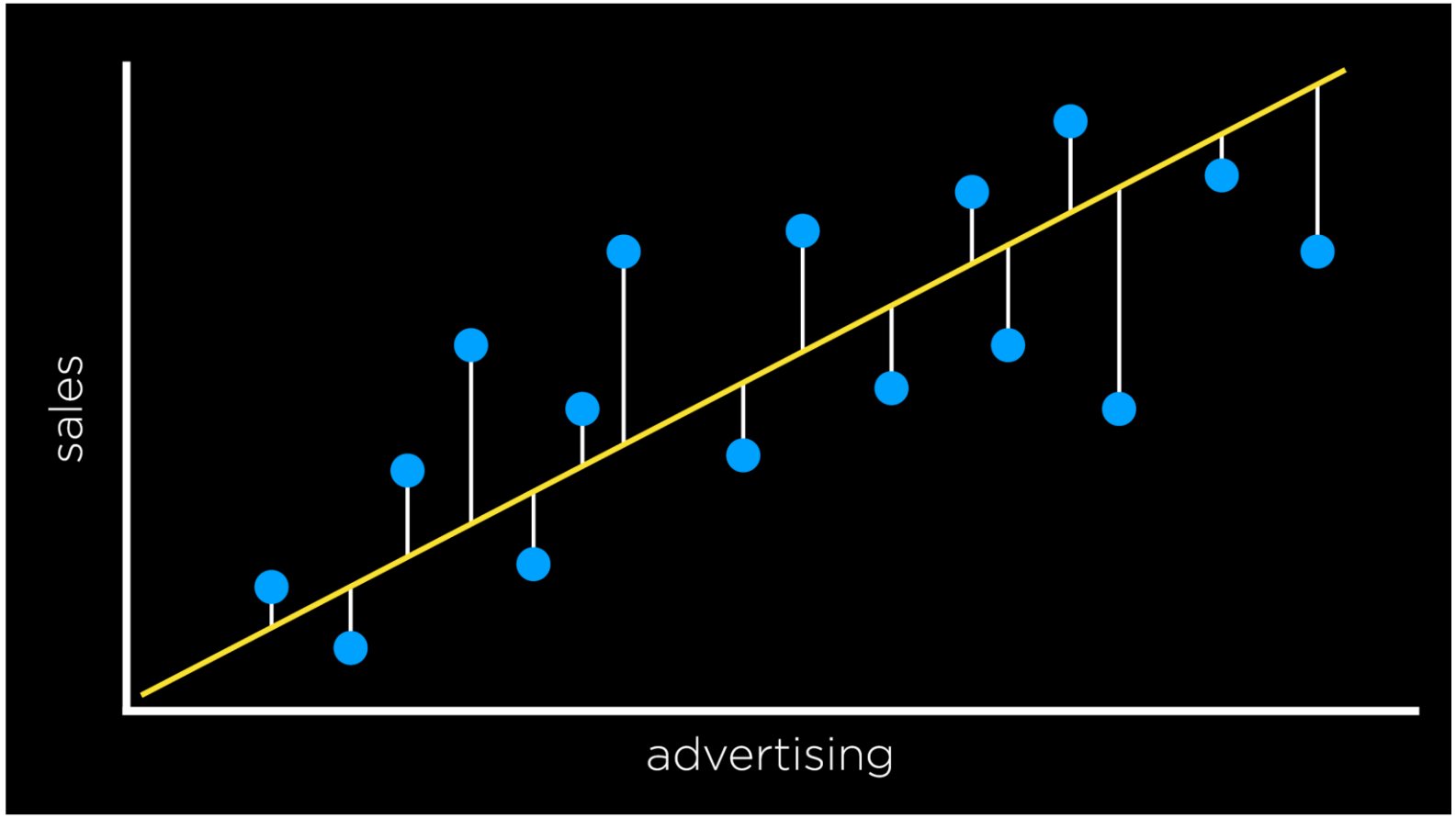
0-1 loss function

- L(actual, predicted) = $\begin{cases} 0, & \text{if actual = predicted} \\ 1, & \text{otherwise} \end{cases}$



L1 loss function

- L(actual, predicted) = | actual - predicted |



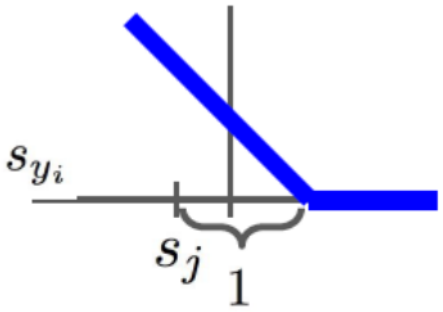
L2 loss function

- $L(\text{actual}, \text{predicted}) = (\text{actual} - \text{predicted})^2$

MSE loss

- $L(\text{actual}, \text{predicted} = 1/n * \Sigma(\text{actual} - \text{predicted})^2$

- **Hinge loss**
  - Multiclass Support Vector Machine (SVM) Loss

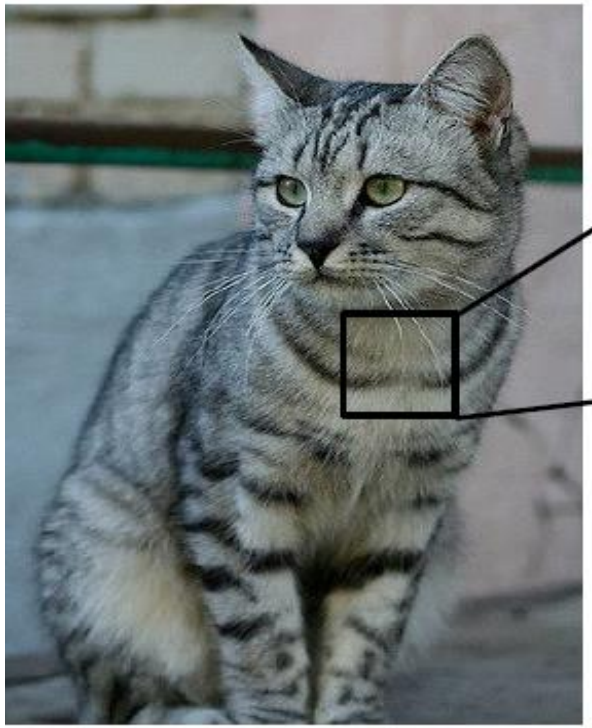


- **Cross-entropy loss**
  - Softmax classifier

An Image Classification Example

A core task in Computer Vision - Image Classification

Assigning a single label to an image from a fixed set of categories



```
[[105 112 108 111 104  99 106  99  96 103 112 119 104  97  93  87]
 [ 91  98 102 106 104  79  98 103  99 105 123 136 110 105  94  85]
 [ 76  85  90 105 128 105  87  96  95  99 115 112 106 103  99  85]
 [ 99  81  81  93 120 131 127 100  95  98 102  99  96  93 101  94]
 [106  91  61  64  69  91  88  85 101 107 109  98  75  84  96  95]
 [114 108  85  55  55  69  64  54  64  87 112 129  98  74  84  91]
 [133 137 147 103  65  81  80  65  52  54  74  84 102  93  85  82]
 [128 137 144 140 109  95  86  70  62  65  63  63  60  73  86 101]
 [125 133 148 137 119 121 117  94  65  79  80  65  54  64  72  98]
 [127 125 131 147 133 127 126 131 111  96  89  75  61  64  72  84]
 [115 114 109 123 150 148 131 118 113 109 100  92  74  65  72  78]
 [ 89  93  90  97 108 147 131 118 113 114 113 109 106  95  77  80]
 [ 63  77  86  81  77  79 102 123 117 115 117 125 125 130 115  87]
 [ 62  65  82  89  78  71  80 101 124 126 119 101 107 114 131 119]
 [ 63  65  75  88  89  71  62  81 120 138 135 105  81  98 110 118]
 [ 87  65  71  87 106  95  69  45  76 130 126 107  92  94 105 112]
 [118  97  82  86 117 123 116  66  41  51  95  93  89  95 102 107]
 [164 146 112  80  82 120 124 104  76  48  45  66  88 101 102 109]
 [157 170 157 120  93  86 114 132 112  97  69  55  70  82  99  94]
 [130 128 134 161 139 100 109 118 121 134 114  87  65  53  69  86]
 [128 112  96 117 150 144 120 115 104 107 102  93  87  81  72  79]
 [123 107  96  86  83 112 153 149 122 109 104  75  80 107 112  99]
 [122 121 102  80  82  86  94 117 145 148 153 102  58  78  92 107]
 [122 164 148 103  71  56  78  83  93 103 119 139 102  61  69  84]]
```

What the computer sees

Images are represented as 3D arrays of numbers, with integers between [0, 255]
0 - black 255 - white

eg： 248 x 400 x 3 = 297,600

(3 for 3 color channels RGB)

Linear Classifier

Array of 32x32x3 numbers
(3072 numbers total)

10x1

10x1

$f(x,W) = Wx + b$

Parameters or weights:

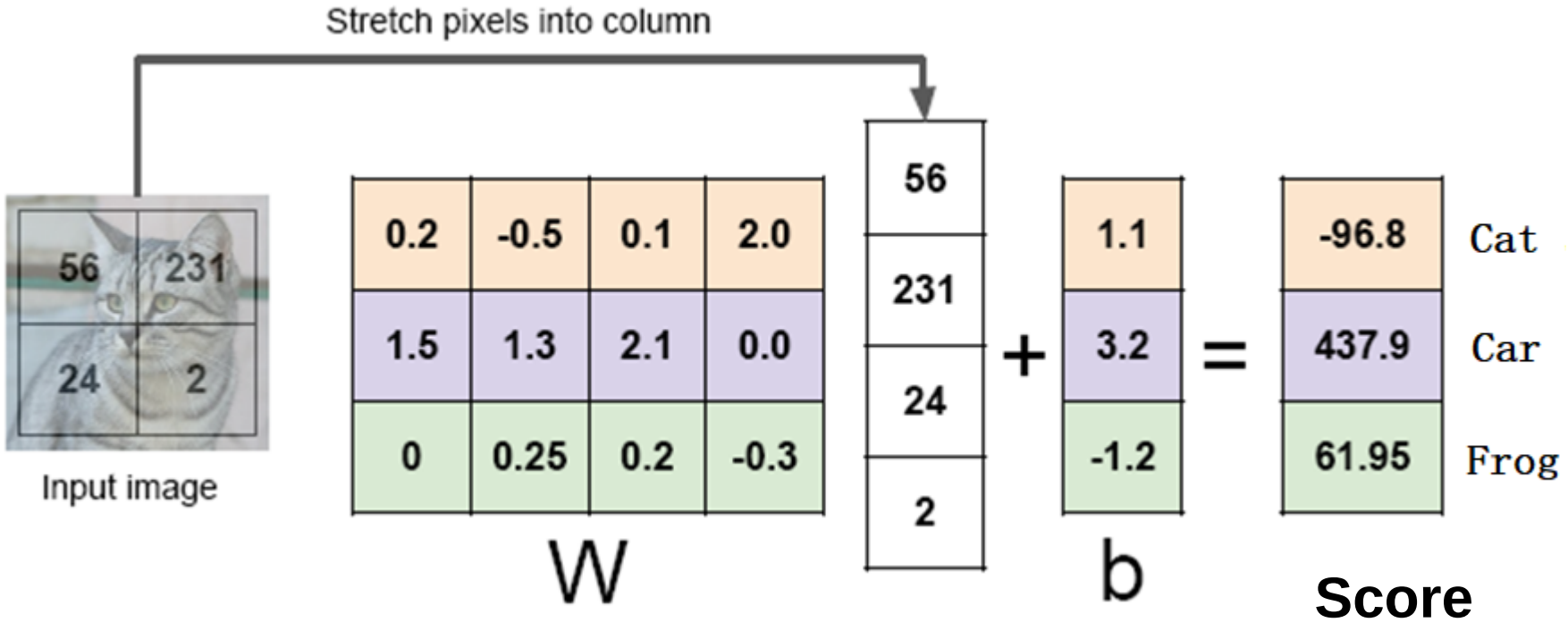
10x3072

Input 3072x1

class scores
(eg. 10 classes)

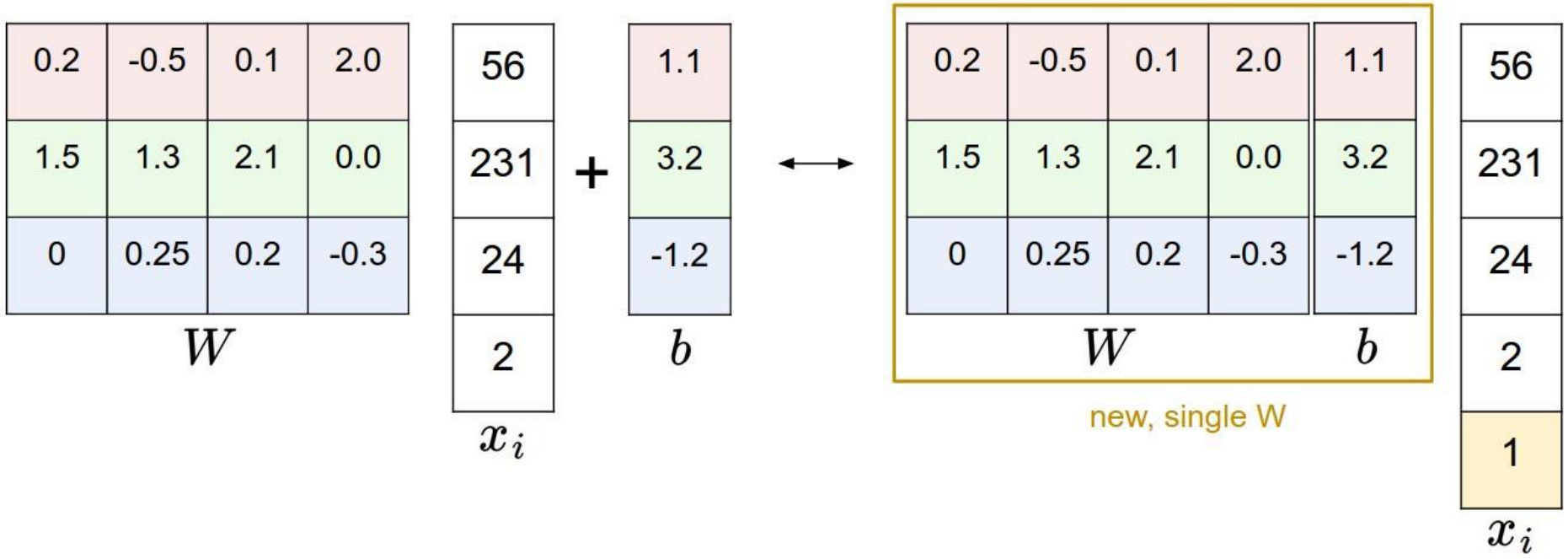
Example:

- An image with 4 pixels, and 3 classes (cat/car/frog)



Bias trick

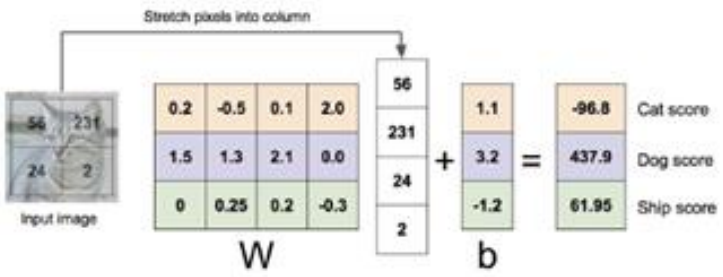
- Representing the two parameters W and b as one



Linear Classifier: Three Viewpoints

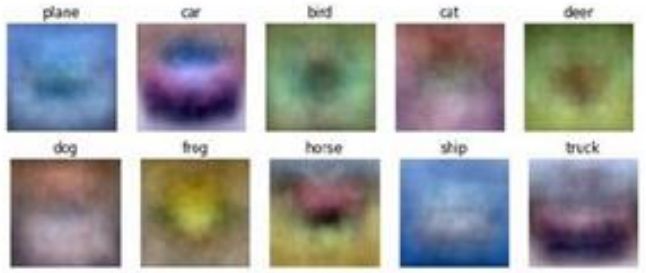
Algebraic Viewpoint

$$f(x,W)=Wx+b$$



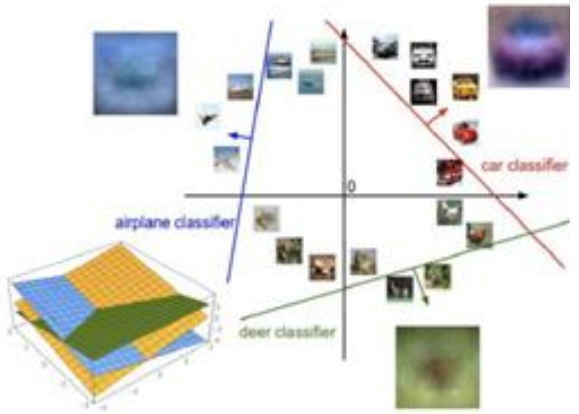
Visual Viewpoint

One template per class



Geometric Viewpoint

Hyperplanes cutting up space



Multiclass Support Vector Machine (SVM) Loss

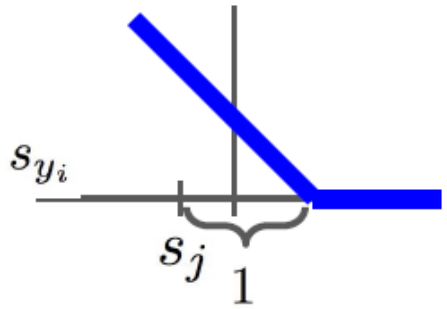
- Target： wants the correct class for each sample/data to a have a score higher than the incorrect classes by at least a margin of delta

- If any class has a score inside the red region (or higher), then there will be accumulated loss. Otherwise the loss will be zero.
- The objective will be to find the weights that will simultaneously satisfy this constraint for all the examples in the training data and give a total loss that is as low as possible.

Multiclass SVM Loss

$$L = \frac{1}{N}\sum_{i=1}^{N}\sum_{j\neq y_i}\max(0, f(x_i; W)_j - f(x_i; W)_{y_i} + 1)$$

（delta=1）





cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Losses:	2.9	0	12.9

$$L_i = \sum_{j\neq y_i}\max(0, s_j - s_{y_i} + 1)$$

= max(0, 2.2 - (-3.1) + 1)
+max(0, 2.5 - (-3.1) + 1)
= max(0, 6.3) + max(0, 6.6)
= 6.3 + 6.6
= 12.9

Loss over full dataset is :

$$L = \frac{1}{N}\sum_{i=1}^{N} L_i$$

L=(2.9+0+12.9)/3=5.27

- Q1: What happens to Loss if the scores of car change a bit?
- Q2： what is the min/max possible loss？
- Q3: At initialization W is small so all s ≈ 0, what is the loss?
- Q4:What if the sum was over all classes? (including j = yi)?
- Q5: What if we use $L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)^2$

Softmax Classifier (Multinomial Logistic Regression)

- Generalization of binary Logistic Regression classifier to multiple classes

Softmax Classifier (Multinomial Logistic Regression)

- Generalization of binary Logistic Regression classifier to multiple classes
- Interpret raw classifier scores as **probabilities**
- **score:** $s = f(x_i; W)$
  - **probability:** $P(Y = k|X = x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}$ Softmax Function
    - **loss:** $L_i = -\log P(Y = y_i|X = x_i)$
- This can be viewed as the **cross-entropy** between the "empirical" distribution $\hat{P}(c|x_i)$ and the "estimated" distribution $P_W(c|x_i)$: $-\sum_c \hat{P}(c|x_i) \log P_W(c|x_i)$

Example

Cat

Car

Frog

Probabilities must be >= 0

Probabilities must sum to 1

$$\frac{p_i}{\sum p_i}$$

$$s = f(x_i; W)$$

$p_i=\exp(s_i)$

$$L_i = -\log\left(\frac{e^{sy_i}}{\sum_j e^{s_j}}\right)$$

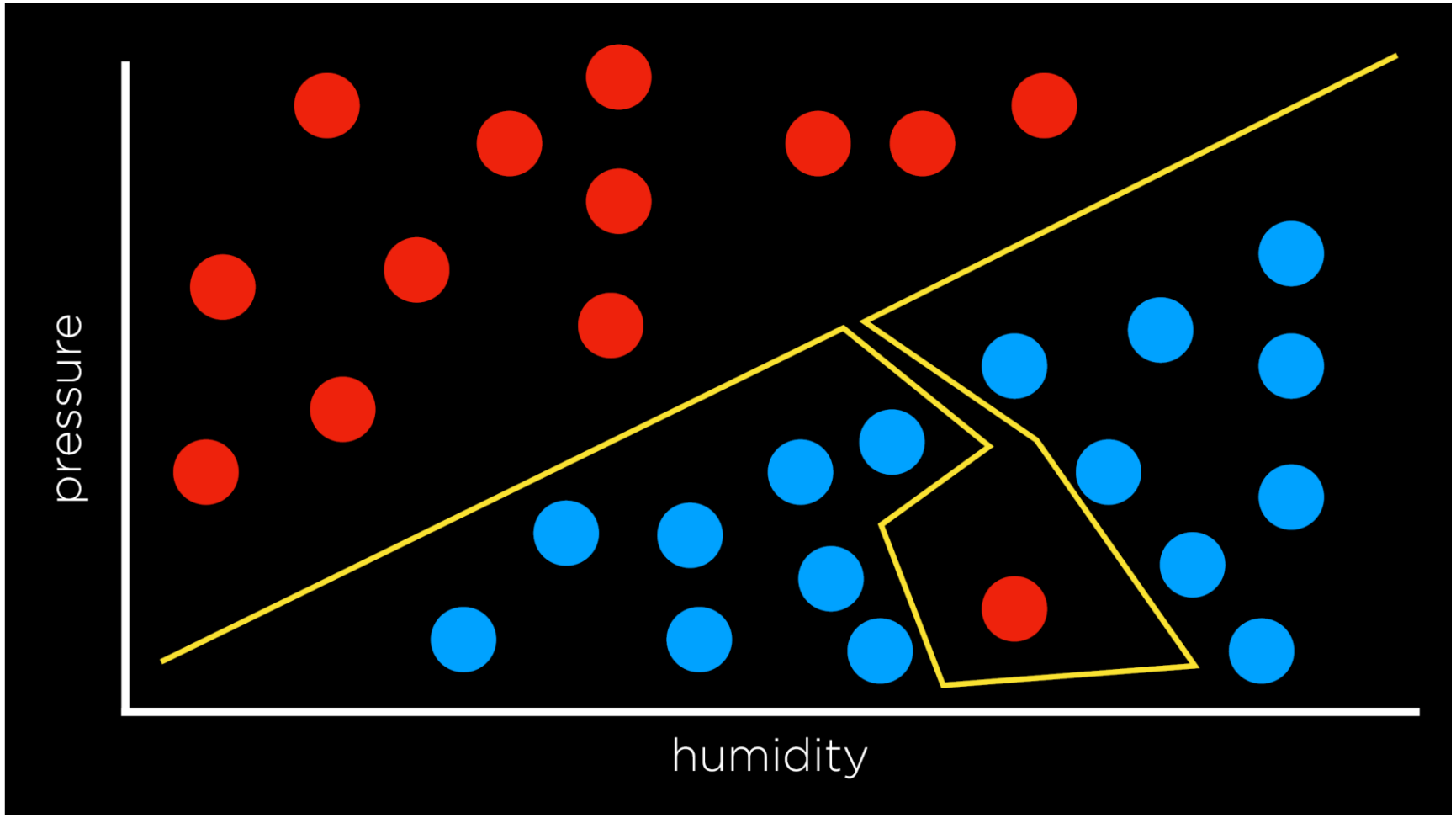
score	unnormalized probabilities	normalized probabilities
3.2	24.5	0.13
5.1	164.0	0.87
-1.7	0.18	0.00

Li = -log(0.13)
= 0.89

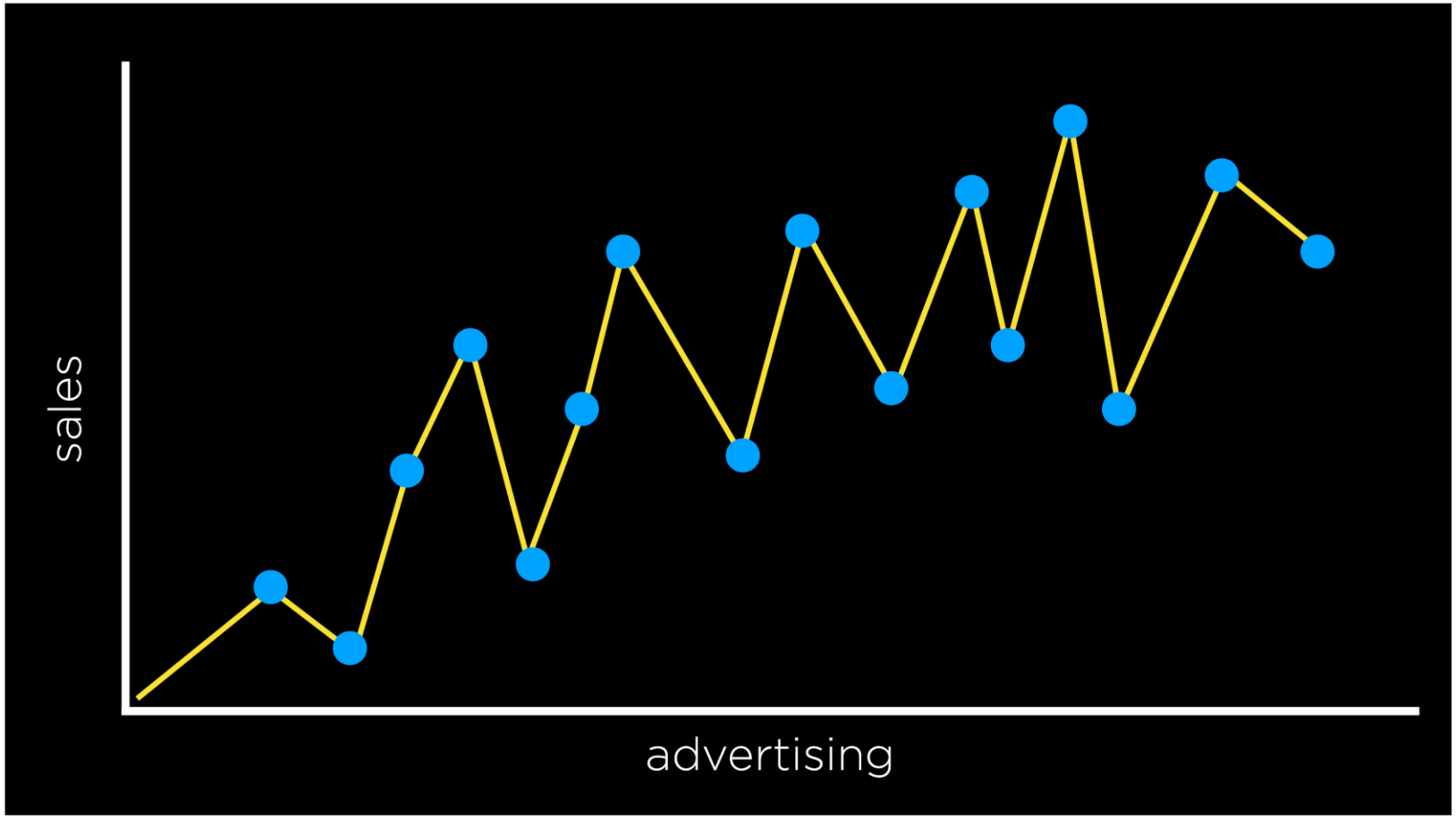
- Q1： what is the min/max possible loss of Li？

- Q2: At initialization W is small so all s ≈ 0 or all s will be approximately equal, what is the loss?

Overfitting



Overfitting



- Suppose that we found a W such that L = 0. Is this W unique?
- **No! e.g. 2W is also has L = 0!**

W:

$$= \max(0, 1.3 - 4.9 + 1) + \max(0, 2.0 - 4.9 + 1)$$
$$= \max(0, -2.6) + \max(0, -1.9)$$
$$= 0 + 0$$
$$= 0$$

With W twice as large:

$$= \max(0, 2.6 - 9.8 + 1) + \max(0, 4.0 - 9.8 + 1)$$
$$= \max(0, -6.2) + \max(0, -4.8)$$
$$= 0 + 0$$
$$= 0$$

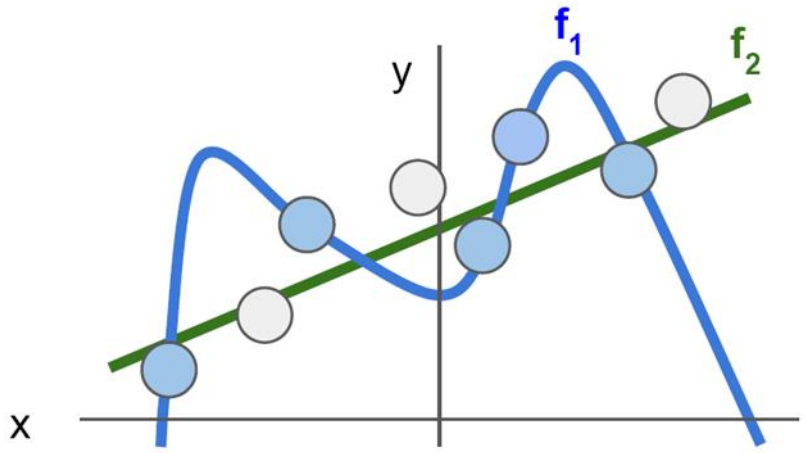
Regularization

- penalizing hypotheses that are more complex to favor simpler, more general hypotheses

$$cost(h) = loss(h) + \lambda complexity(h)$$

$$L = \frac{1}{N} \sum_i L_i(f(x_i, W), y_i)$$

- **Data loss**: Model predictions should match training data



Regularization

regularization strength (hyperparameter)

$$L(W) = \frac{1}{N}\sum_{i=1}^{N} L_i(f(x_i, W), y_i) + \lambda R(W)$$

Data loss: Model predictions should match training data

Regularization: Prevent the model from doing too well on training data

- Regularize
  - Express preferences over weights
  - Make the model simple so it can work on test data
  - Improve optimization by adding curvature

Regularization

- Simple examples
  - L2 regularization: $R(W) = \sum_k \sum_l W_{k,l}^2$
  - L1 regularization: $R(W) = \sum_k \sum_l |W_{k,l}|$
  - Elastic net (L1 + L2): $R(W) = \sum_k \sum_l \beta W_{k,l}^2 + |W_{k,l}|$
- More complex:
  - Dropout
  - Batch normalization
  - Stochastic depth, fractional pooling, etc

$$x = [1, 1, 1, 1]$$

$$w_1 = [1, 0, 0, 0]$$

$$w_2 = [0.25, 0.25, 0.25, 0.25]$$

$$w_1^T x = w_2^T x = 1$$

Use L2 Regularization $R(W) = \sum_k \sum_l W_{k,l}^2$

Which W will be chosen?

- L2 regularization prefers w2 to w1, because it likes to “spread out” the weights.

Basic supervised learning framework

Training Set: $\mathscr{D} = \{\langle \mathbf{x}^{[i]}, y^{[i]} \rangle, i = 1, \ldots, n\}$

Unknown function $f(\mathbf{x}) = y$

Hypothesis: $h(\mathbf{x}) = \hat{y}$

- **Training** (or **learning**): given a *training set* of labeled examples $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, instantiate a predictor *h*
- **Testing** (or **inference**): apply *h* to a new *test example* $\mathbf{x}$ and output the predicted value $y = h(\mathbf{x})$

Formalization

- Given: training data $\{(x_i, y_i), i = 1, \dots, n\}$
- Find $y = h(x) \in \mathcal{H}$
- S.t. h works well on *test* data

Formalization

- Given: training data $\{(x_i, y_i), i = 1, \dots, n\}$ *i.i.d.* from distribution $\boldsymbol{D}$
- **F**ind $y = h(x) \in \mathcal{H}$
- S.t. h works well on *test* data *i.i.d.* from distribution $\boldsymbol{D}$

Have the same distribution

i.i.d.: independently identically distributed

Formalization

- Given: training data $\{(x_i, y_i), i = 1, \dots, n\}$ i.i.d. from distribution $\boldsymbol{D}$
- Find $y = h(x) \in \mathcal{H}$
- S.t. h works well on *test* data *i.i.d.* from distribution $\boldsymbol{D}$

What kind of performance measure?

Formalization

- Given: training data $\{(x_i, y_i), i = 1, \dots, n\}$ i.i.d. from distribution D
- Find $y = h(x) \in \mathcal{H}$
- S.t. the *expected loss* is small:

$$L(h) = \mathbb{E}_{(x,y)\sim D}[l(h, x, y)]$$

$\rightarrow$ minimizes $\hat{L}(h) = \frac{1}{n}\sum_{i=1}^{n} l(h, x_i, y_i)$ —— Empirical loss

Challenges of Supervised Learning

- Need labelled data
- A crucial assumption in supervised learning：

  The training examples have the same distribution as the test (future) examples.
- In real-world applications, this assumption is oftentimes violated, which is one of the common challenges in the field.

Outline

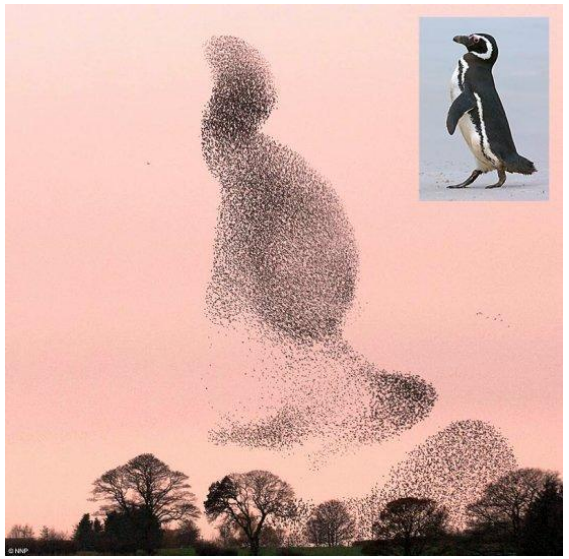
- What is Machine Learning
- Categories of Machine
- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning

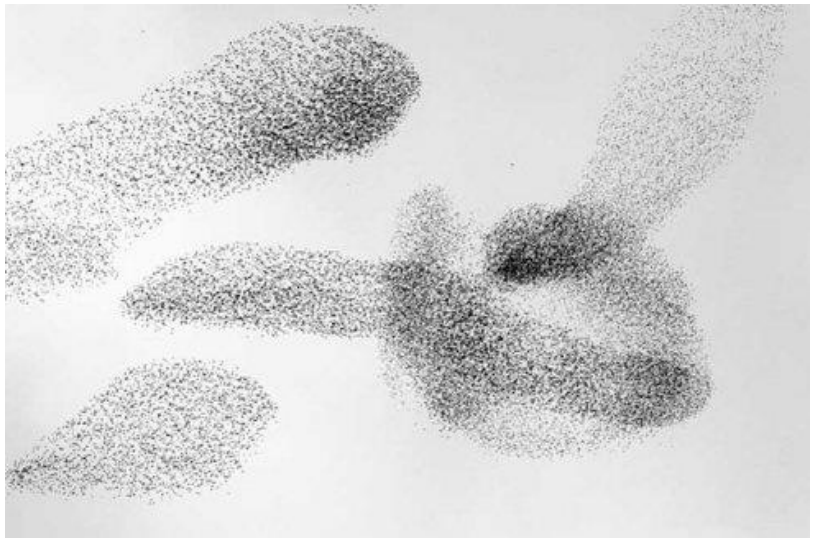
Unsupervised Learning

- given input data without any additional feedback, learn patterns

clustering

- organizing a set of objects into groups in such a way that similar objects tend to be in the same group





Some Clustering Applications

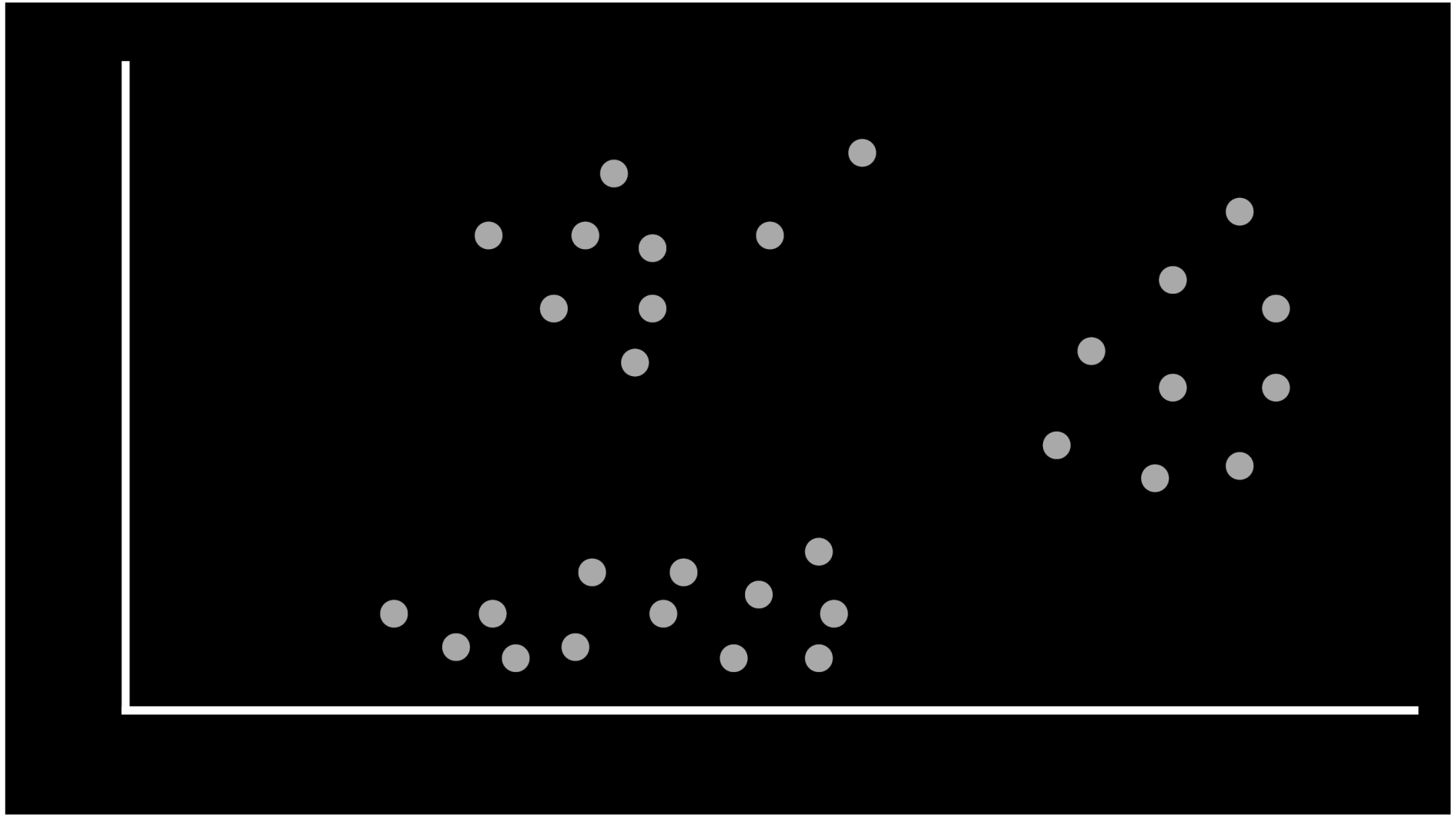
- Genetic research
- Image segmentation
- Market research
- Social network analysis

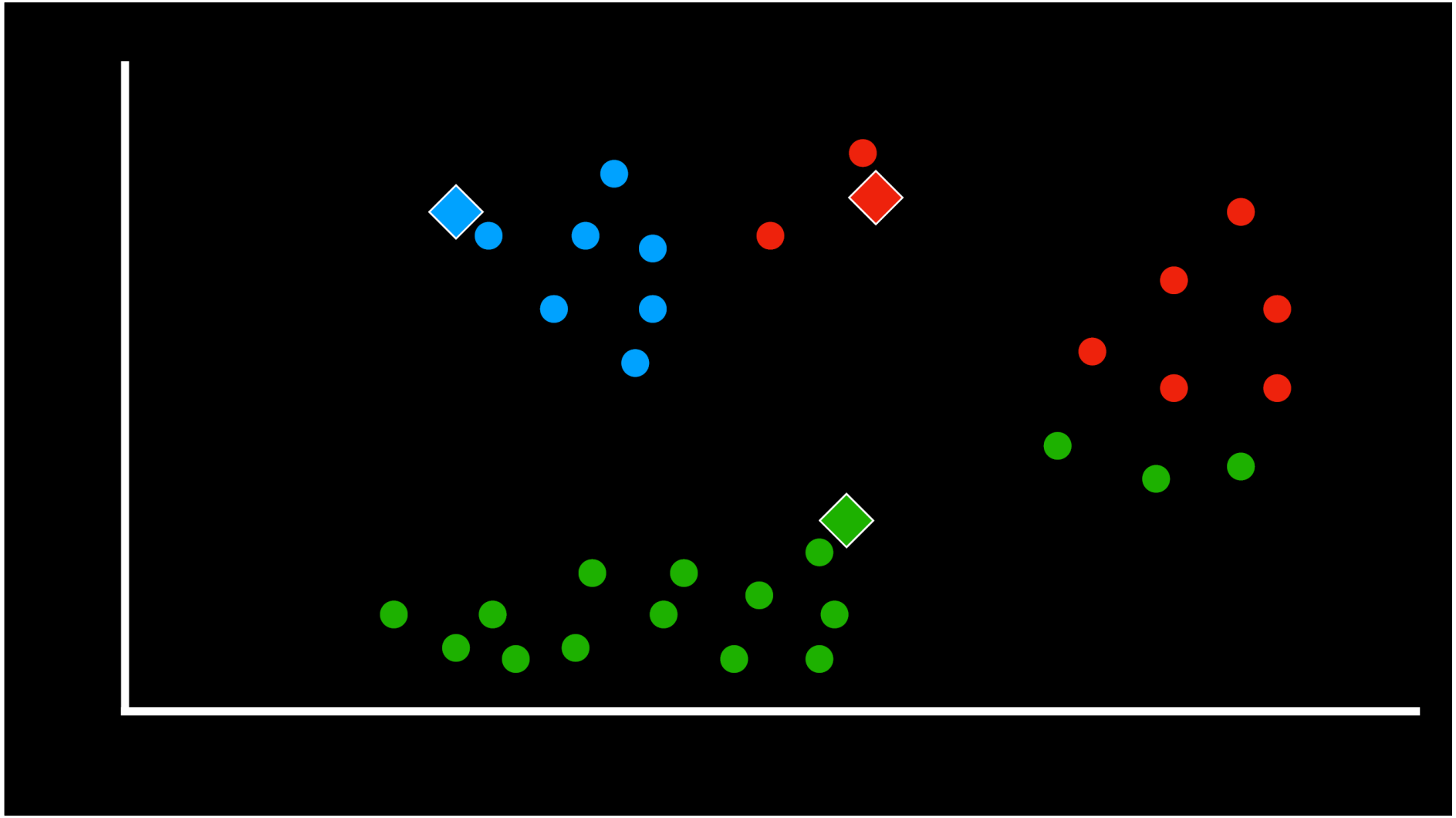
……

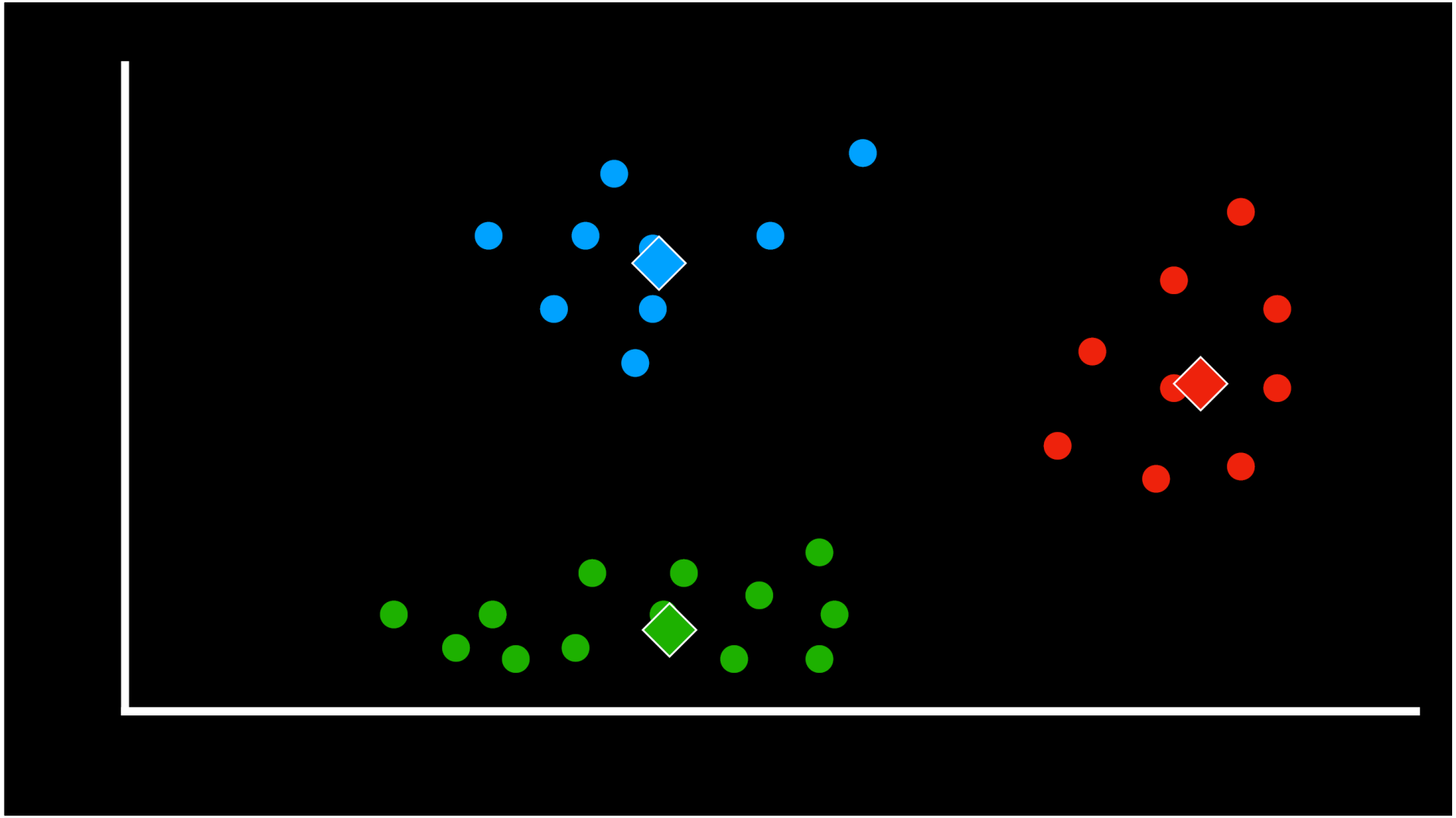
k-means clustering

- algorithm for clustering data based on repeatedly assigning points to clusters and updating those clusters' centers

Visualizing： https://www.naftaliharris.com/blog/visualizing-k-means-clustering/







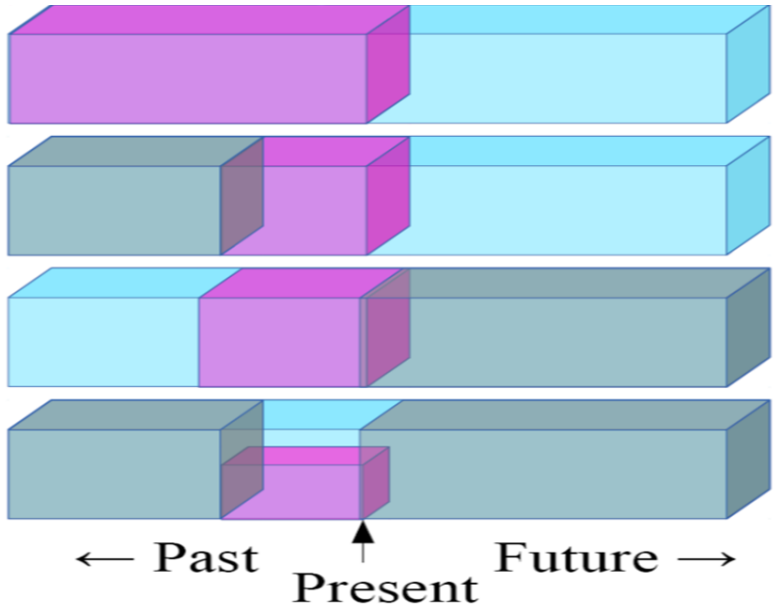
Self-supervised vs. Semi-supervised Leaning

- Self-supervised learning
  - a general learning framework that relies on surrogate (pretext) tasks that can be formulated using only unsupervised data
- Semi-supervised learning
  - describes a class of algorithms that seek to learn from both unlabeled and labeled samples, typically assumed to be sampled from the same or similar distributions

S^{4}L: Self-Supervised Semi-Supervised Learning, Google Research, Brain Team

Self-Supervised Learning

- Predict any part of the input from any other part.
- Predict the future from the past.
- Predict the future from the recent past.
- Predict the past from the present.
- Predict the top from the bottom.
- Predict the occluded from the visible.
- Pretend there is a part of the input you don't know and predict that.



Self-Supervised Learning: Filling in the Blanks

Unsupervised learning

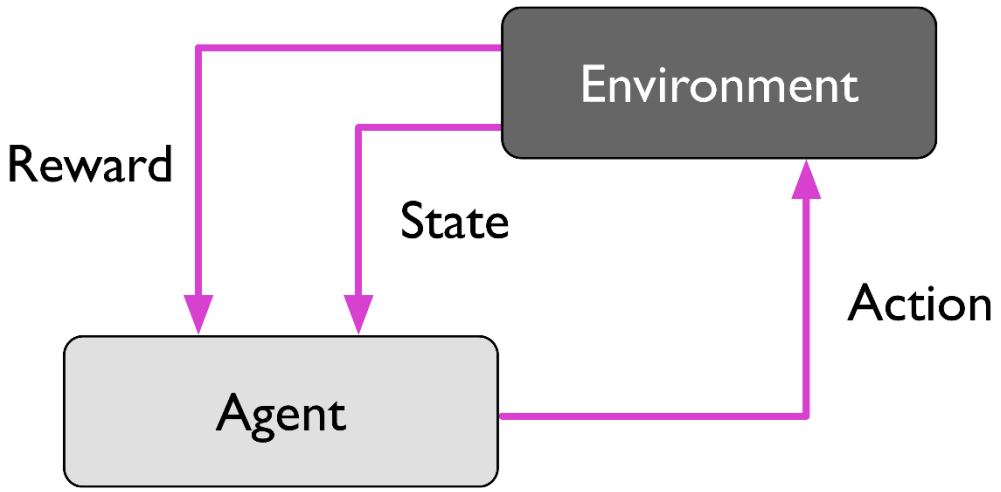
- Unsupervised learning use cases:
  - Data exploration and discovery
  - Providing representations to downstream supervised learning

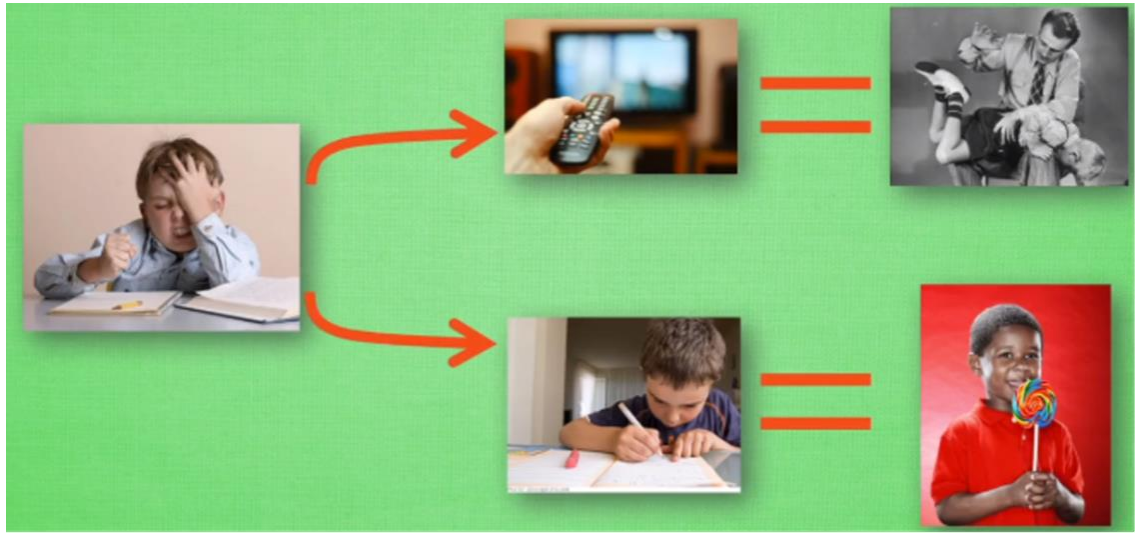
Outline

- What is Machine Learning
- Categories of Machine
- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning

Reinforcement Learning

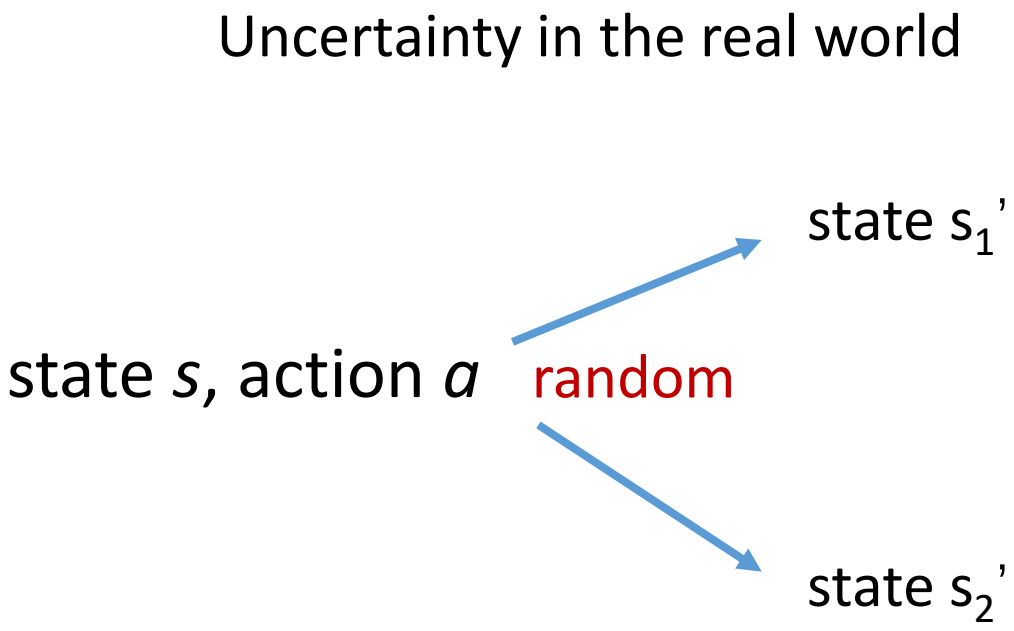
- given a set of rewards or punishments, learn what actions to take in the future

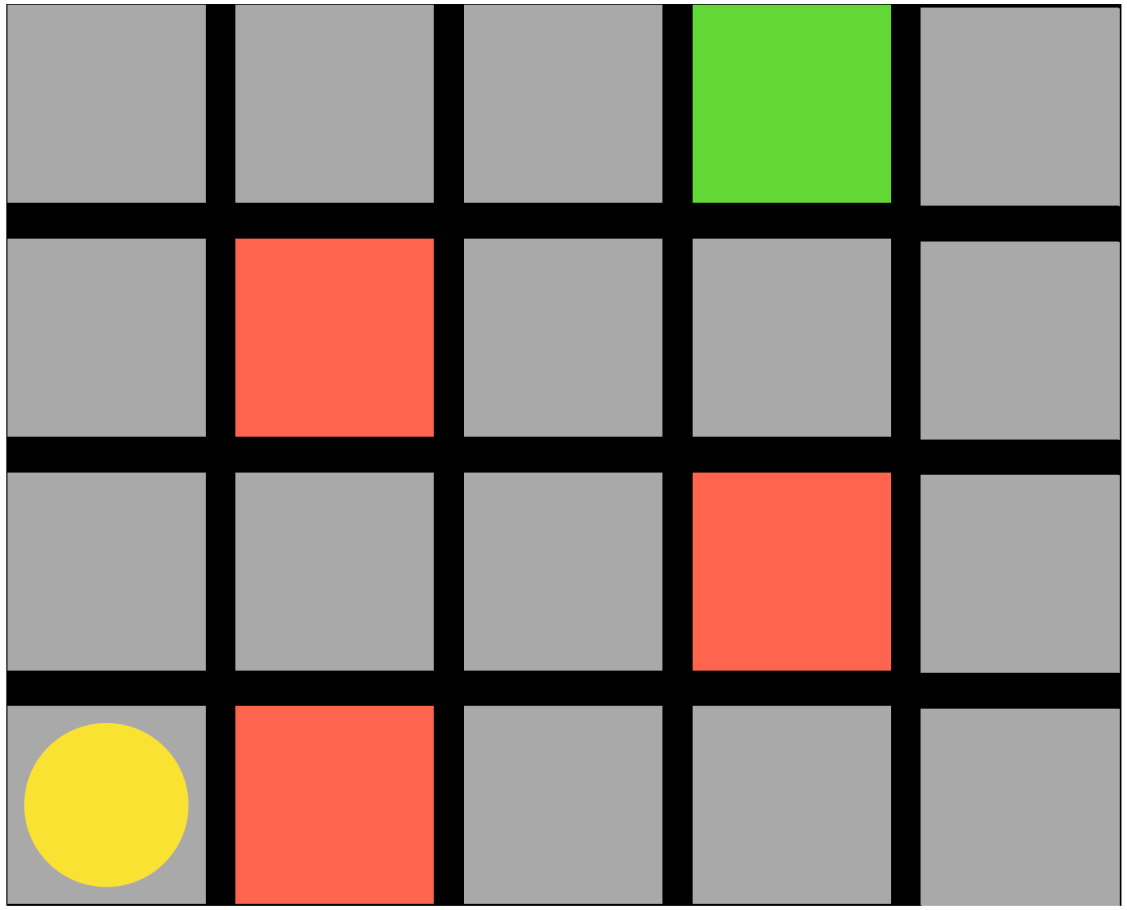


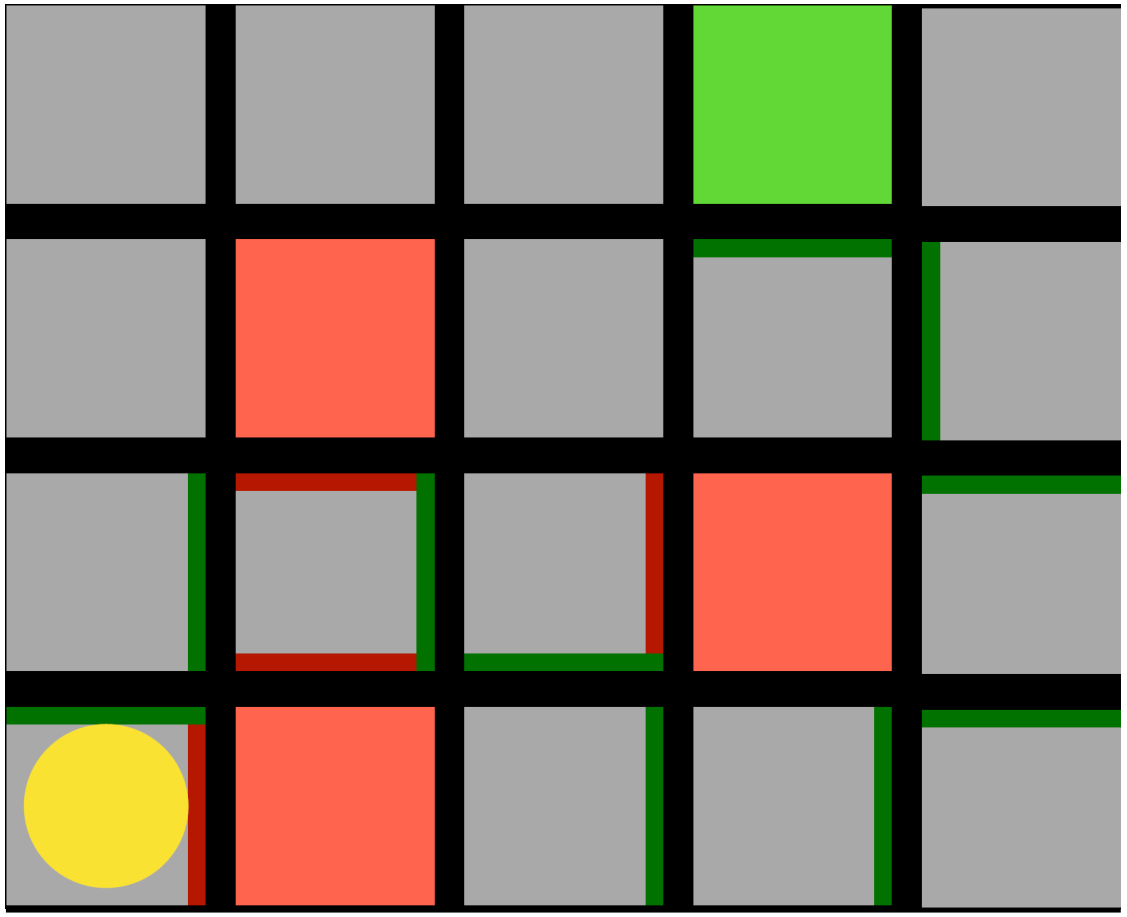


Markov Decision Process （MDP）

- model for decision-making, representing states, actions, and their rewards
  - Set of states *S*
  - Set of actions *Actions(s)*
  - Transition model *P(s'|s,a)*
  - Reward function *R(s,a,s')*







Q-learning

- method for learning a function *Q(s, a)*
- estimate of the value of performing action *a* in state *s*

Q-learning Overview

- Start with $Q(s,a)=0$ for all s, a
- When we take an action and receive a reward:
  - Estimate the value of $Q(s,a)$ based on current reward and expected future rewards
  - Update $Q(s,a)$ to take into account old estimate as well as our new estimate

$$Q(s,a) \leftarrow Q(s,a) + \alpha \text{ (new value estimate – old value estimate)}$$

Q-learning

- Start with $Q(s,a)$=0 for all s, a
- Every time we take an action a in state s and observe a reward r, we update:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left(\text{new value estimate} - Q(s,a) \right)$$

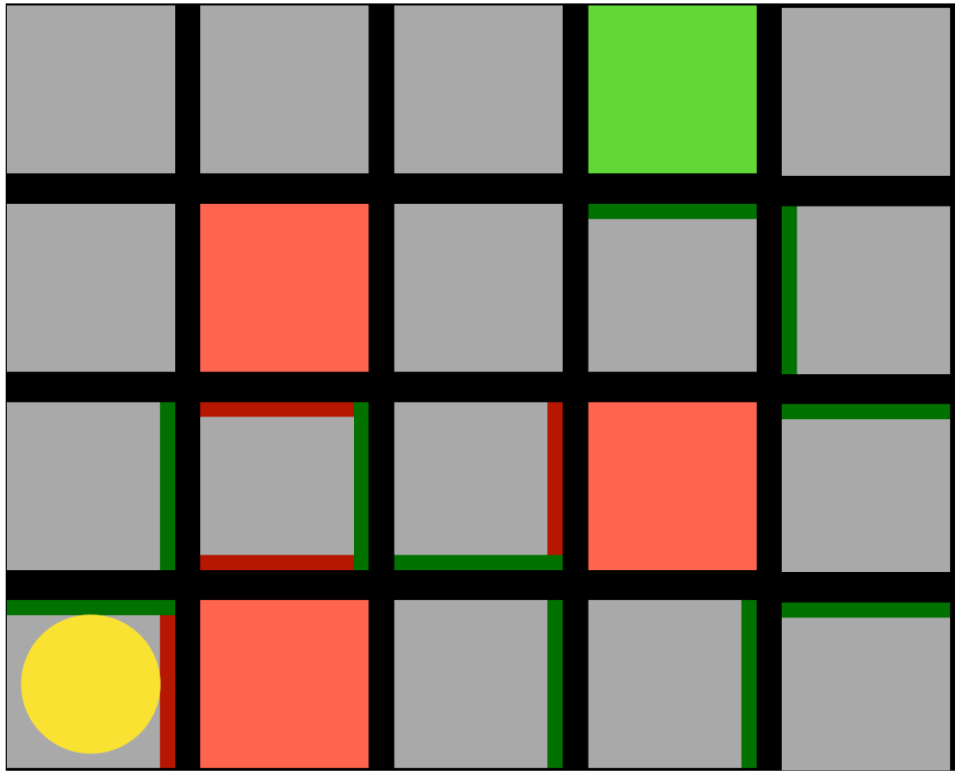
$$Q(s,a) \leftarrow Q(s,a) + \alpha \left((r + \text{future reward estimate}) - Q(s,a) \right)$$

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left((r + \max_{a'} Q(s',a')) - Q(s,a) \right)$$

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left((r + \gamma \max_{a'} Q(s',a')) - Q(s,a) \right)$$

Greedy Decision-Making

- When in state *s*, choose action *a* with the highest *Q(s,a)*



ε-greedy

Explore vs. Exploit

- Set ε equal to how often we want to move randomly.
  - With probability 1- ε , choose estimated best move.
  - With probability ε , choose a random move.

Function approximation

- approximating *Q(s,a)*, often by a function combining various features, rather than storing one value for every state-action pair

Summary

- What is Machine Learning： T、P、E
- Categories of Machine Learning
  - Supervised Learning
  - Unsupervised Learning
  - Reinforcement Learning

Q & A