How do I teach my second grade kid what AI is?

Cupid Chan

Disclaimer

- This presentation session was prepared or accomplished by Cupid Chan in his personal capacity. The opinions expressed in this talk are the author's own and do not reflect the view of any United States Government Agency or Department.

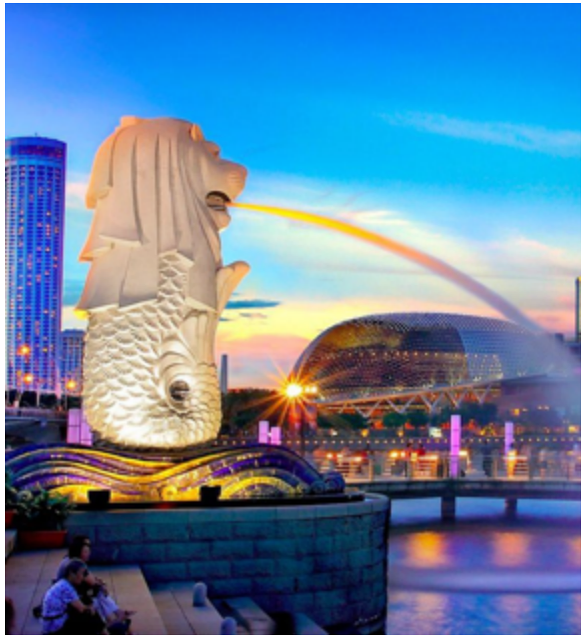
Cupid Chan

- CTO of Index Analytics
- Board of Directors and TSC, Linux Foundation ODPi
- Organizer of a Big Data In Action Meetup in Washington/Baltimore area with 1600 members

www.linkedin.com/in/cupidchan/

@cupidckchan

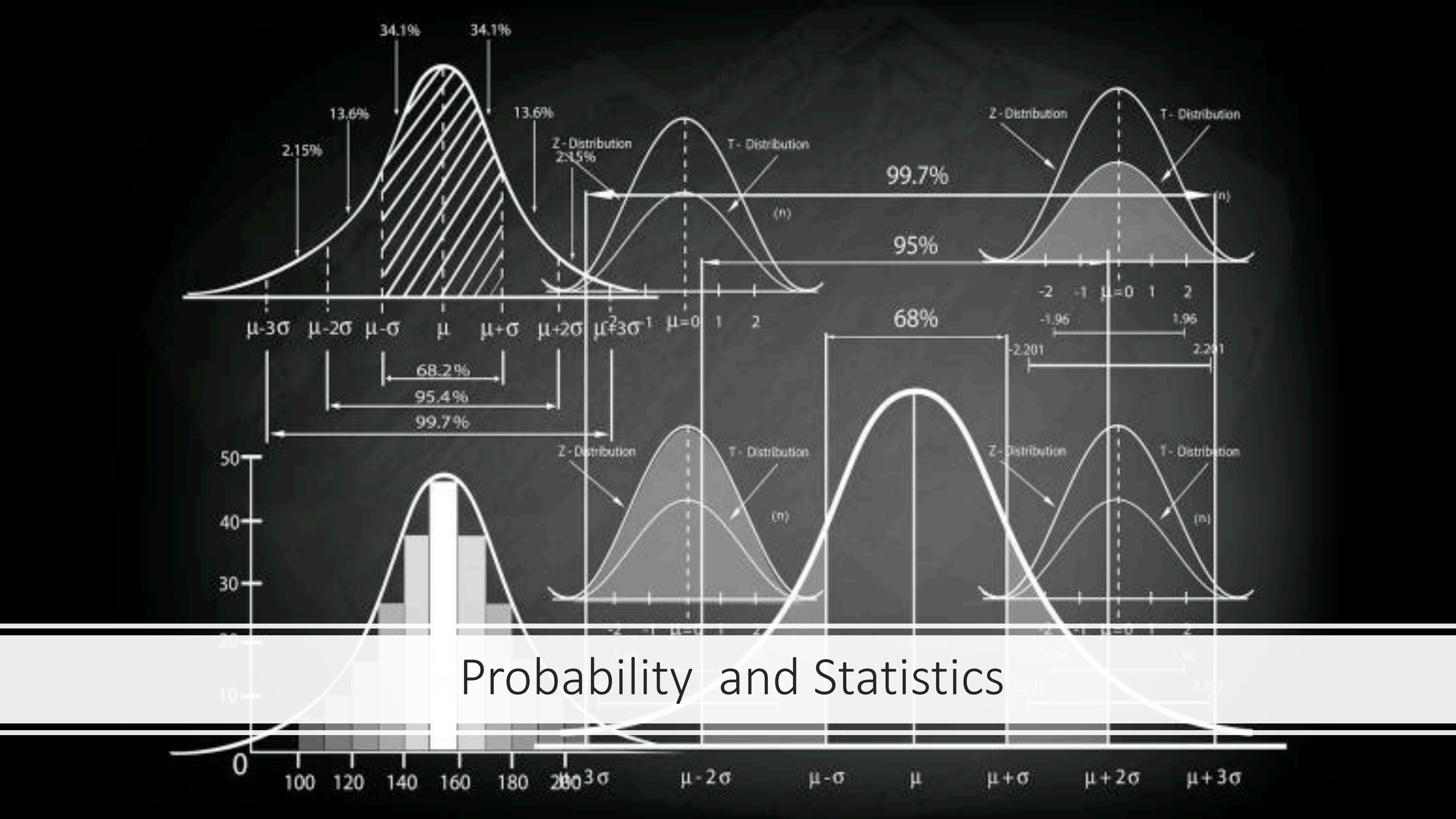
Strata
DATA CONFERENCE
PRESENTED BY
O'REILLY cloudera



Let’s play a game

Probability to guess the 3 cards correctly

$$\frac{1}{132{,}600}$$



Probability and Statistics

$x + (-x) = 0$.   6) multiplication, for all scalars $a \in F$, and all vectors $x \in X$,
7) For any $x \in X$, $1x = x$ (for scalar 1).
8) For any two scalars $a \in F$ and $b \in F$ and any $x \in X$, $a(bx) = (ab)x$.
9) $(a + b)x = ax + bx$.   10) $a(x + y) = ax + ay$.

Linear Independence: Consider n vectors $\{x_1, x_2, \dots, x_n\}$. If there exists n scalars $a_1, a_2, \dots, a_n$, at least one of which is nonzero, such that $a_1x_1 + a_2x_2 + \dots + a_nx_n = 0$, then the $\{x_i\}$ are linearly dependent.

Spanning a Space:
Let X be a linear vector space and let $\{u_1, u_2, \dots, u_m\}$ be a subset of vectors in X. This subset spans X if and only if for every vector $x \in X$ there exist scalars $x_1, x_2, \dots, x_m$ such that $x = x_1u_1 + x_2u_2 + \dots + x_mu_m$.

Inner Product: (x, y) for any scalar function of x and y.
1. $(x,y) = (y,x)$ 2. $(x, ay_1 + by_2) = a(x,y_1) + b(x,y_2)$
3. $(x,x) \geq 0$, where equality holds iff x is the zero vector.

Norm: A scalar function $\|x\|$ is called a norm if it satisfies:
1. $\|x\| \geq 0$ 2. $\|x\| = 0$ if and only if $x = 0$.
3. $\|ax\| = |a|\|x\|$ 4. $\|x + y\| \leq \|x\| + \|y\|$

Angle: The angle θ bet. 2 vectors x and y is defined by $\cos\theta = \frac{(x,y)}{\|x\|\,\|y\|}$

Orthogonality: 2 vectors $x, y \in X$ are said to be orthogonal if $(x,y) = 0$.

Gram Schmidt Orthogonalization:
Assume that we have n independent vectors $y_1, y_2, \dots, y_n$. From these vectors we will obtain n orthogonal vectors $v_1, v_2, \dots, v_n$.

$$v_1 = y_1, \quad v_k = y_k - \sum_{i=1}^{k-1} \frac{(v_i, y_k)}{(v_i, v_i)} v_i,$$

where $\frac{(v_i, y_k)}{(v_i, v_i)} v_i$ is the projection of y_k on v_i

Vector Expansions:

$$x = \sum_{i=1}^{n} x_i v_i = x_1v_1 + x_2v_2 + \dots + x_nv_n,$$

$$\text{for orthogonal vectors, } x_j = \frac{(v_j, x)}{(v_j, v_j)}$$

Reciprocal Basis Vectors:

$$(r_i, v_j) = \begin{cases} 0 & i \neq j \\ 1 & i = j \end{cases}, \quad x_j = (r_j, x)$$

To compute the reciprocal basis vectors: set $\mathbf{B} = [\mathbf{v}_1\ \mathbf{v}_2 \dots \mathbf{v}_n]$,
$\mathbf{R} = [\mathbf{r}_1\ \mathbf{r}_2 \dots \mathbf{r}_n]$, $\mathbf{R}^T = \mathbf{B}^{-1}$ In matrix form: $\mathbf{x}^v = \mathbf{B}^{-1}\mathbf{x}^s$

Transformations:
A *transformation* consists of three parts:
domain: $X = \{x_i\}$, range: $Y = \{y_i\}$, and a rule relating each $x_i \in X$ to an element $y_i \in Y$.

Linear Transformations: transformation A is *linear* if:
1. for all $x_1, x_2 \in X$, $A(x_1 + x_2) = A(x_1) + A(x_2)$
2. for all $x \in X$, $a \in R$, $A(ax) = aA(x)$

Matrix Representations:
Let $\{v_1, v_2, \dots, v_n\}$ be a basis for vector space X, and let $\{u_1, u_2, \dots, u_m\}$ be a basis for vector space Y. Let A be a linear transformation with domain X and range Y: $A(x) = y$
The coefficients of the matrix representation are obtained from

$$A(v_j) = \sum_{i=1}^{m} a_{ij} u_i$$

Change of Basis: $\mathbf{B}_t = [\mathbf{t}_1\ \mathbf{t}_2 \dots \mathbf{t}_n]$, $\mathbf{B}_w = [\mathbf{w}_1\ \mathbf{w}_2 \dots \mathbf{w}_n]$

$$\mathbf{A}' = [\mathbf{B}_w^{-1}\mathbf{A}\mathbf{B}_t]$$

Eigenvalues & Eigenvectors: $\mathbf{A}\mathbf{z} = \lambda\mathbf{z}$, $|[\mathbf{A} - \lambda\mathbf{I}]| = 0$

Diagonalization: $\mathbf{B} = [\mathbf{z}_1\ \mathbf{z}_2 \dots \mathbf{z}_n]$,

Single-layer perceptrons can only classify linearly separable vectors.

Perceptron Learning Rule

$$\mathbf{W}^{new} = \mathbf{W}^{old} + \mathbf{e}\mathbf{p}^T, \mathbf{b}^{new} = \mathbf{b}^{old} + \mathbf{e},$$
$$\text{where } \mathbf{e} = \mathbf{t} - \mathbf{a}$$

Hebb's Postulate: "When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased."

Linear Associator: $\mathbf{a} = purelin(\mathbf{W}\mathbf{p})$

The Hebb Rule: *Supervised Form:* $w_{ij}^{new} = w_{ij}^{old} + t_{qi}p_{qj}$

$$\mathbf{W} = t_1P_1^T + t_2P_2^T + \dots + t_QP_Q^T$$

$$\mathbf{W} = [\mathbf{t}_1\ \mathbf{t}_2 \dots \mathbf{t}_Q] \begin{bmatrix} \mathbf{p}_1^T \\ \mathbf{p}_2^T \\ \vdots \\ \mathbf{p}_Q^T \end{bmatrix} = \mathbf{T}\mathbf{P}^T$$

Pseudoinverse Rule: $\mathbf{W} = \mathbf{T}\mathbf{P}^+$
When the number, R, of rows of $\mathbf{P}$ is greater than the number of columns, Q, of $\mathbf{P}$ and the columns of $\mathbf{P}$ are independent, then the pseudoinverse can be computed by $\mathbf{P}^+ = (\mathbf{P}^T\mathbf{P})^{-1}\mathbf{P}^T$

Variations of Hebbian Learning:
Filtered Learning (Ch 10): $\mathbf{W}^{new} = (1 - \gamma)\mathbf{W}^{old} + \alpha\mathbf{t}_q\mathbf{p}_q^T$
Delta Rule (Ch 10): $\mathbf{W}^{new} = \mathbf{W}^{old} + \alpha(\mathbf{t}_q - \mathbf{a}_q)\mathbf{p}_q^T$
Unsupervised Hebb (Ch 13): $\mathbf{W}^{new} = \mathbf{W}^{old} + \alpha\mathbf{a}_q\mathbf{p}_q^T$

Taylor: $F(\mathbf{x}) = F(\mathbf{x}^*) + \nabla F(\mathbf{x})^T|_{\mathbf{x}=\mathbf{x}^*}(\mathbf{x} - \mathbf{x}^*) + \frac{1}{2}(\mathbf{x} - \mathbf{x}^*)\nabla^2 F(\mathbf{x})^T|_{\mathbf{x}=\mathbf{x}^*}(\mathbf{x} - \mathbf{x}^*) + \cdots$

Grad $\nabla F(\mathbf{x}) = \left[\frac{\partial}{\partial x_1}F(\mathbf{x})\ \ \frac{\partial}{\partial x_2}F(\mathbf{x})\ \dots\ \frac{\partial}{\partial x_n}F(\mathbf{x})\right]^T$

Hessian: $\nabla^2 F(\mathbf{x}) =$

$$\begin{bmatrix} \frac{\partial^2}{\partial x_1^2}F(\mathbf{x}) & \frac{\partial^2}{\partial x_1 \partial x_2}F(\mathbf{x}) \dots & \frac{\partial^2}{\partial x_1 \partial x_n}F(\mathbf{x}) \\ \frac{\partial^2}{\partial x_2 \partial x_1}F(\mathbf{x}) & \frac{\partial^2}{\partial x_2^2}F(\mathbf{x}) \dots & \frac{\partial^2}{\partial x_2 \partial x_n}F(\mathbf{x}) \\ \vdots & \vdots & \vdots \\ \frac{\partial^2}{\partial x_n \partial x_1}F(\mathbf{x}) & \frac{\partial^2}{\partial x_n \partial x_2}F(\mathbf{x}) \dots & \frac{\partial^2}{\partial x_n^2}F(\mathbf{x}) \end{bmatrix}$$

Directional Derivatives:
1st Dir.Der.: $\frac{\mathbf{p}^T\nabla F(\mathbf{x})}{\|\mathbf{p}\|}$, **2nd Dir.Der.:** $\frac{\mathbf{p}^T\nabla^2 F(\mathbf{x})\mathbf{p}}{\|\mathbf{p}\|^2}$

Minima:
Strong Minimum: if a scalar $\delta > 0$ exists, such that $F(x) < F(x + \Delta x)$ for all Δx such that $\delta > \|\Delta x\| > 0$.
Global Minimum: if $F(x) < F(x + \Delta x)$ for all $\Delta x \neq 0$
Weak Minimum: if it is not a strong minimum, and a scalar $\delta > 0$ exists, such that $F(x) \leq F(x + \Delta x)$ for all Δx such that $\delta > \|\Delta x\| > 0$.

Necessary Conditions for Optimality:
1st-Order Condition: $\nabla F(\mathbf{x})|_{\mathbf{x}=\mathbf{x}^*} = 0$ (Stationary Points)
2nd-Order Condition: $\nabla^2 F(\mathbf{x})|_{\mathbf{x}=\mathbf{x}^*} \geq 0$ (Positive Semi-definite Hessian Matrix).

Quadratic fn.: $F(x) = \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x} + \mathbf{d}^T\mathbf{x} + c$

Stable Learning Rate: ($\alpha_k = \alpha$, constant) $\alpha < \frac{2}{\lambda_{max}}$
$\{\lambda_1, \lambda_2, \dots, \lambda_n\}$ Eigenvalues of Hessian matrix A

Learning Rate to Minimize Along the Line:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k\mathbf{p}_k \overset{by}{\Rightarrow} \alpha_k = -\frac{\mathbf{g}_k^T\mathbf{p}_k}{\mathbf{p}_k^T\mathbf{A}\mathbf{p}_k} \text{ (For quadratic fn.)}$$

After Minimization Along the Line:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k\mathbf{p}_k \Rightarrow \mathbf{g}_{k+1}^T\mathbf{p}_k = 0$$

ADALINE: $\mathbf{a} = purelin(\mathbf{W}\mathbf{p} + \mathbf{b})$

Mean Square Error: *(for ADALINE it is a quadratic fn.)*
$F(\mathbf{x}) = E[e^2] = E[(t - a)^2] = E[(t - \mathbf{x}^T\mathbf{z})^2]$
$F(\mathbf{x}) = c - 2\mathbf{x}^T\mathbf{h} + \mathbf{x}^T\mathbf{R}\mathbf{x}$,
$c = E[t^2]$, $\mathbf{h} = E[t\mathbf{z}]$ and $\mathbf{R} = E[\mathbf{z}\mathbf{z}^T] \Rightarrow \mathbf{A} = 2\mathbf{R}, \mathbf{d} = -2\mathbf{h}$
Unique minimum, if it exists, is $\mathbf{x}^* = \mathbf{R}^{-1}\mathbf{h}$,
where $\mathbf{x} = \begin{bmatrix} {}_1\mathbf{w} \\ b \end{bmatrix}$ and $\mathbf{z} = \begin{bmatrix} \mathbf{p} \\ 1 \end{bmatrix}$

LMS Algorithm: $\mathbf{W}(k + 1) = \mathbf{W}(k) + 2\alpha\,\mathbf{e}(k)\,\mathbf{p}^T(k)$

$$\mathbf{b}(k + 1) = \mathbf{b}(k) + 2\alpha\,\mathbf{e}(k)$$

Convergence Point: $\mathbf{x}^* = \mathbf{R}^{-1}\mathbf{h}$
Stable Learning Rate: $0 < \alpha < 1/\lambda_{max}$ where λ_{max} is the maximum eigenvalue of $\mathbf{R}$

Adaptive Filter ADALINE:

$$a(k) = purelin(\mathbf{W}\mathbf{p}(k) + b) = \sum_{i=1}^{R} w_{1,i}\,y(k - i + 1) + b$$

Backpropagation Algorithm:
Performance Index:
Mean Square error: $F(\mathbf{x}) = E[\mathbf{e}^T\mathbf{e}] = E[(\mathbf{t} - \mathbf{a})^T(\mathbf{t} - \mathbf{a})]$
Approximate Performance Index: (single sample)
$\hat{F}(x) = \mathbf{e}^T(k)\mathbf{e}(k) = (\mathbf{t}(k) - \mathbf{a}(k))^T(\mathbf{t}(k) - \mathbf{a}(k))$

Sensitivity: $\mathbf{s}^m = \frac{\partial\hat{F}}{\partial\mathbf{n}^m} = \left[\frac{\partial\hat{F}}{\partial n_1^m}\ \ \frac{\partial\hat{F}}{\partial n_2^m}\ \ \dots\ \ \frac{\partial\hat{F}}{\partial n_{S^m}^m}\right]^T$

Forward Propagation: $\mathbf{a}^0 = \mathbf{p}$,

$$\mathbf{a}^{m+1} = \mathbf{f}^{m+1}(\mathbf{W}^{m+1}\mathbf{a}^m + \mathbf{b}^{m+1}) \text{ for } m = 0, 1, \dots, M - 1$$
$$\mathbf{a} = \mathbf{a}^M$$

Backward Propagation: $\mathbf{s}^M = -2\dot{\mathbf{F}}^M(\mathbf{n}^M)(\mathbf{t} - \mathbf{a})$,

$\mathbf{s}^m = \dot{\mathbf{F}}^m(\mathbf{n}^m)(\mathbf{W}^{m+1})^T\mathbf{s}^{m+1}$ for $m = M - 1, \dots, 2, 1$, where

$$\dot{\mathbf{F}}^m(\mathbf{n}^m) = \mathrm{diag}([\dot{f}^m(n_1^m)\ \ \dot{f}^m(n_2^m)\ \ \dots\ \ \dot{f}^m(n_{S^m}^m)])$$
$$\dot{f}^m(n_j^m) = \frac{\partial f^m(n_j^m)}{\partial n_j^m}$$

Weight Update (Approximate Steepest Descent):

$$\mathbf{W}^m(k + 1) = \mathbf{W}^m(k) - \alpha\mathbf{s}^m(\mathbf{a}^{m-1})^T$$
$$\mathbf{b}^m(k + 1) = \mathbf{b}^m(k) - \alpha\mathbf{s}^m$$

Backpropagation with Momentum (MOBP):

$$\Delta\mathbf{W}^m(k) = \gamma\Delta\mathbf{W}^m(k - 1) - (1 - \gamma)\alpha\,\mathbf{s}^m(\mathbf{a}^{m-1})^T$$
$$\Delta\mathbf{b}^m(k) = \gamma\Delta\mathbf{b}^m(k - 1) - (1 - \gamma)\alpha\,\mathbf{s}^m$$

Variable Learning Rate Backpropagation (VLBP)
1. If the squared error (over the entire training set) increases by more than some set percentage ζ (typically one to five percent) after a weight update, then the weight update is discarded, the learning rate is multiplied by some factor $\rho < 1$, and the momentum coefficient γ (if it is used) is set to zero.
2. If the squared error decreases after a weight update, then the weight update is accepted and the learning rate is multiplied by some factor $\eta > 1$. If γ has been previously set to zero, it is reset to its original value.
3. If the squared error increases by less than ζ, then the weight update is accepted but the learning rate and the momentum coefficient are unchanged.

Association: $\mathbf{a} = hardlim(\mathbf{W}^0\mathbf{p}^0 + \mathbf{W}\mathbf{p} + b)$
An association is a link between the inputs and outputs of a network so that when a stimulus A is presented to the network, it will output a response B.

Associative Learning Rules:
Unsupervised Hebb Rule:

$$\mathbf{W}(q) = \mathbf{W}(q - 1) + \alpha\,\mathbf{a}(q)\mathbf{p}^T(q)$$

Hebb with Decay:

$$\mathbf{W}(q) = (1 - \gamma)\mathbf{W}(q - 1) + \alpha\,\mathbf{a}(q)\mathbf{p}^T(q)$$

Instar: $\mathbf{a} = hardlim(\mathbf{W}\mathbf{p} + b)$, $\mathbf{a} = hardlim({}_1\mathbf{w}^T\mathbf{p} + b)$
The instar is activated for ${}_1\mathbf{w}^T\mathbf{p} = \|{}_1\mathbf{w}\|\|\mathbf{p}\|\cos\theta \geq -b$
where θ is the angle between $\mathbf{p}$ and ${}_1\mathbf{w}$.

Instar Rule:

$${}_i\mathbf{w}(q) = {}_i\mathbf{w}(q - 1) + \alpha\,a_i(q)(\mathbf{p}(q) - {}_i\mathbf{w}(q - 1))$$
$${}_i\mathbf{w}(q) = (1 - \alpha)\,{}_i\mathbf{w}(q - 1) + \alpha\,\mathbf{p}(q), \text{ if } (a_i(q) = 1)$$

Kohonen Rule:

$${}_i\mathbf{w}(q) = {}_i\mathbf{w}(q - 1) + \alpha(\mathbf{p}(q) - {}_i\mathbf{w}(q - 1)) \text{ for } i \in X(q)$$

Outstar Rule: $\mathbf{a} = satlins(\mathbf{W}\mathbf{p})$

$$\mathbf{w}_j(q) = \mathbf{w}_j(q - 1) + \alpha\left(\mathbf{a}(q) - \mathbf{w}_j(q - 1)\right)p_j(q)$$

Competitive Layer: $\mathbf{a} = compet(\mathbf{W}\mathbf{p}) = compet(\mathbf{n})$
Competitive Learning with the Kohonen Rule:

$${}_{i^*}\mathbf{w}(q) = {}_{i^*}\mathbf{w}(q - 1) + \alpha\left(\mathbf{p}(q) - {}_{i^*}\mathbf{w}(q - 1)\right)$$
$$= (1 - \alpha)\,{}_{i^*}\mathbf{w}(q - 1) + \alpha\,\mathbf{p}(q)$$

${}_i\mathbf{w}(q) = {}_i\mathbf{w}(q - 1)$, $i \neq i^*$ where i^* is the winning neuron.

Self-Organizing with the Kohonen Rule:

$${}_i\mathbf{w}(q) = {}_i\mathbf{w}(q - 1) + \alpha\left(\mathbf{p}(q) - {}_i\mathbf{w}(q - 1)\right)$$
$$= (1 - \alpha)\,{}_i\mathbf{w}(q - 1) + \alpha\,\mathbf{p}(q), \ i \in N_{i^*}(d)$$
$$N_i(d) = \{j, d_{i,j} \leq d\}$$

LVQ Network: $(w_{k,i}^2 = 1) \Rightarrow$ subclass i is a part of class k

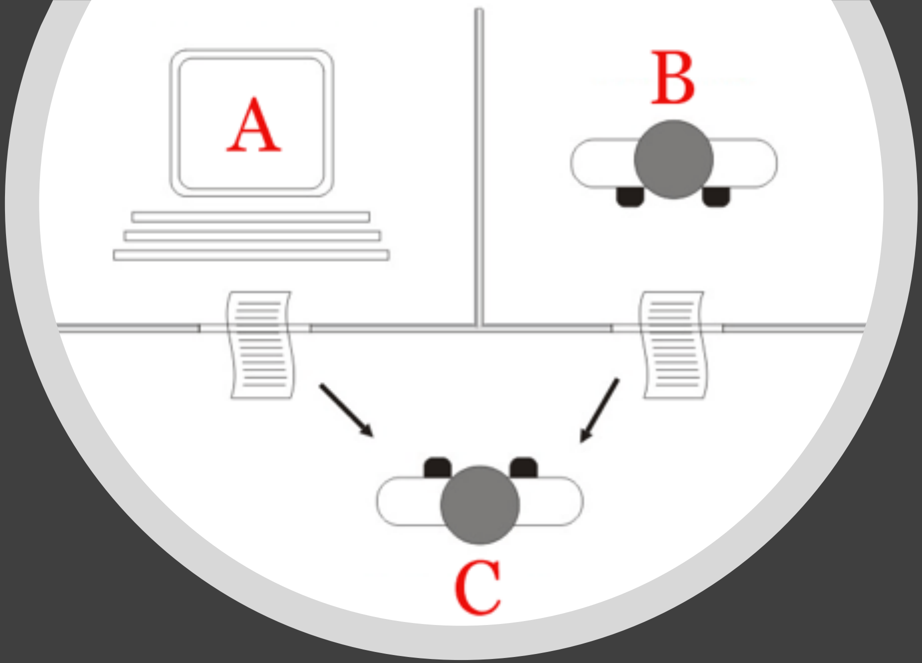
$$n_i^1 = -\|{}_i\mathbf{w}^1 - \mathbf{p}\|, \mathbf{a}^1 = compet(\mathbf{n}^1), \ \mathbf{a}^2 = \mathbf{W}^2\mathbf{a}^1$$

LVQ Network Learning with the Kohonen Rule:

$${}_{i^*}\mathbf{w}^1(q) = {}_{i^*}\mathbf{w}^1(q - 1) + \alpha\left(\mathbf{p}(q) - {}_{i^*}\mathbf{w}^1(q - 1)\right),$$
$$\text{if } a_{k^*}^2 = t_{k^*} = 1$$
$${}_{i^*}\mathbf{w}^1(q) = {}_{i^*}\mathbf{w}^1(q - 1) - \alpha\left(\mathbf{p}(q) - {}_{i^*}\mathbf{w}^1(q - 1)\right),$$
$$\text{if } a_{k^*}^2 = 1 \neq t_{k^*} = 0$$

hardlim: $a = \begin{cases} 0 & n < 0 \\ 1 & n \geq 0 \end{cases}$, ***hardlims:*** $a = \begin{cases} -1 & n < 0 \\ +1 & n \geq 0 \end{cases}$, ***purelin:*** $a = n$, ***logsig:*** $a = \frac{1}{1 + e^{-n}}$, ***tansig:*** $a = \frac{e^n - e^{-n}}{e^n + e^{-n}}$, ***poslin:*** $a = \begin{cases} 0 & n < 0 \\ n & n \geq 0 \end{cases}$

"Dad, what are you doing at work?"



Computer can help you to do something for you and people will think you are doing that.

Based on Turing Test…

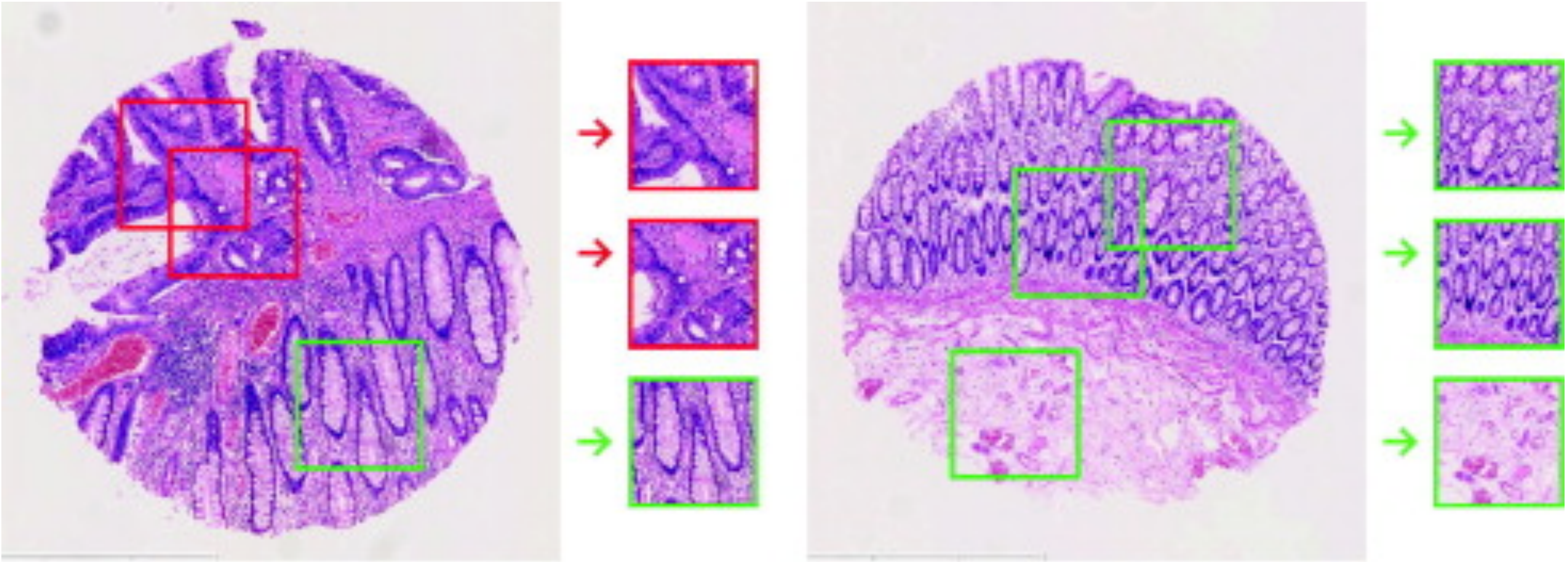


How did you teach your younger sister?

Supervised Learning - Maps an input to an output based on example input-output pairs

ピコ太郎

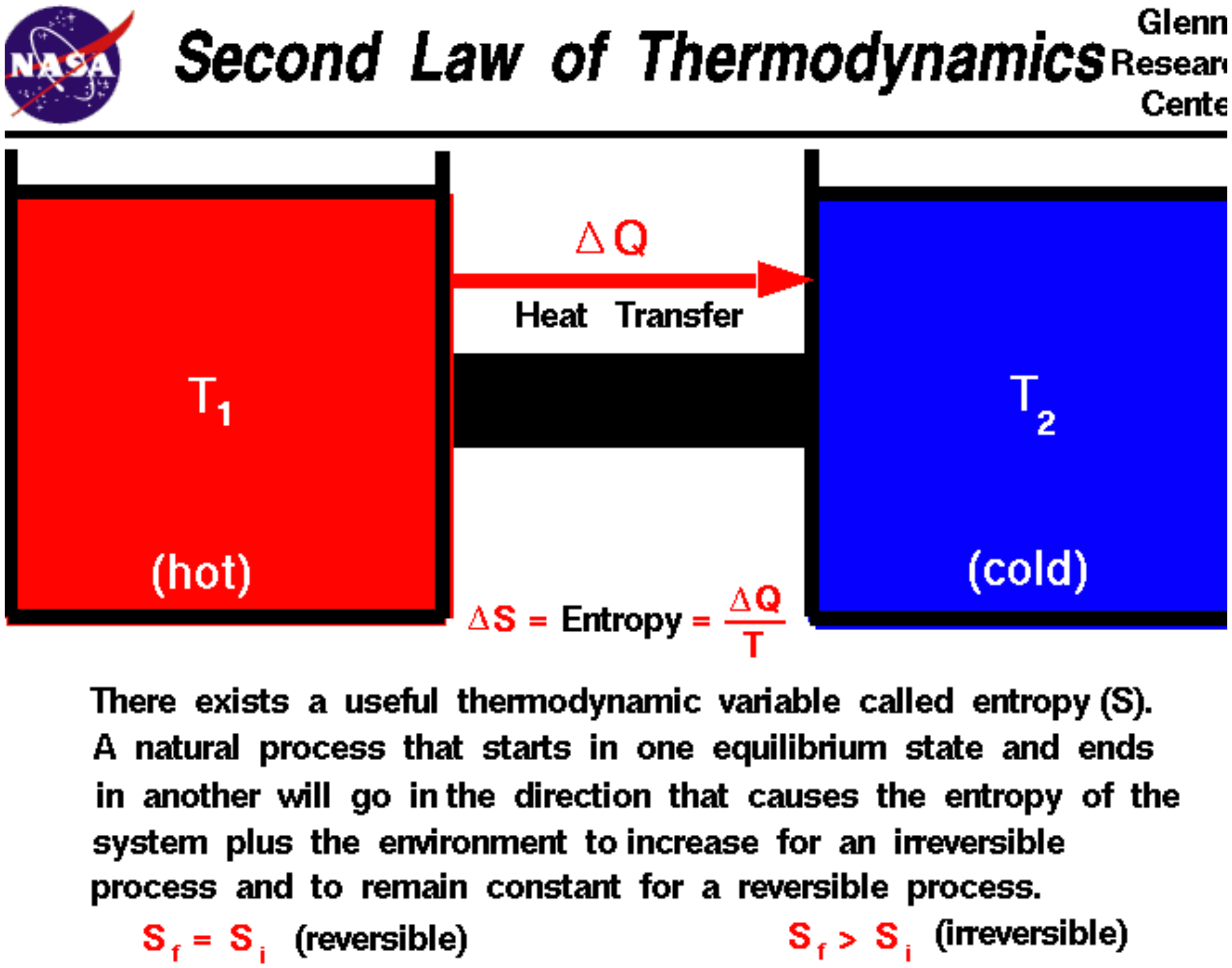
Supervised Learning Example – Detecting Cancer



(a) cancer image (b) non-cancerimage

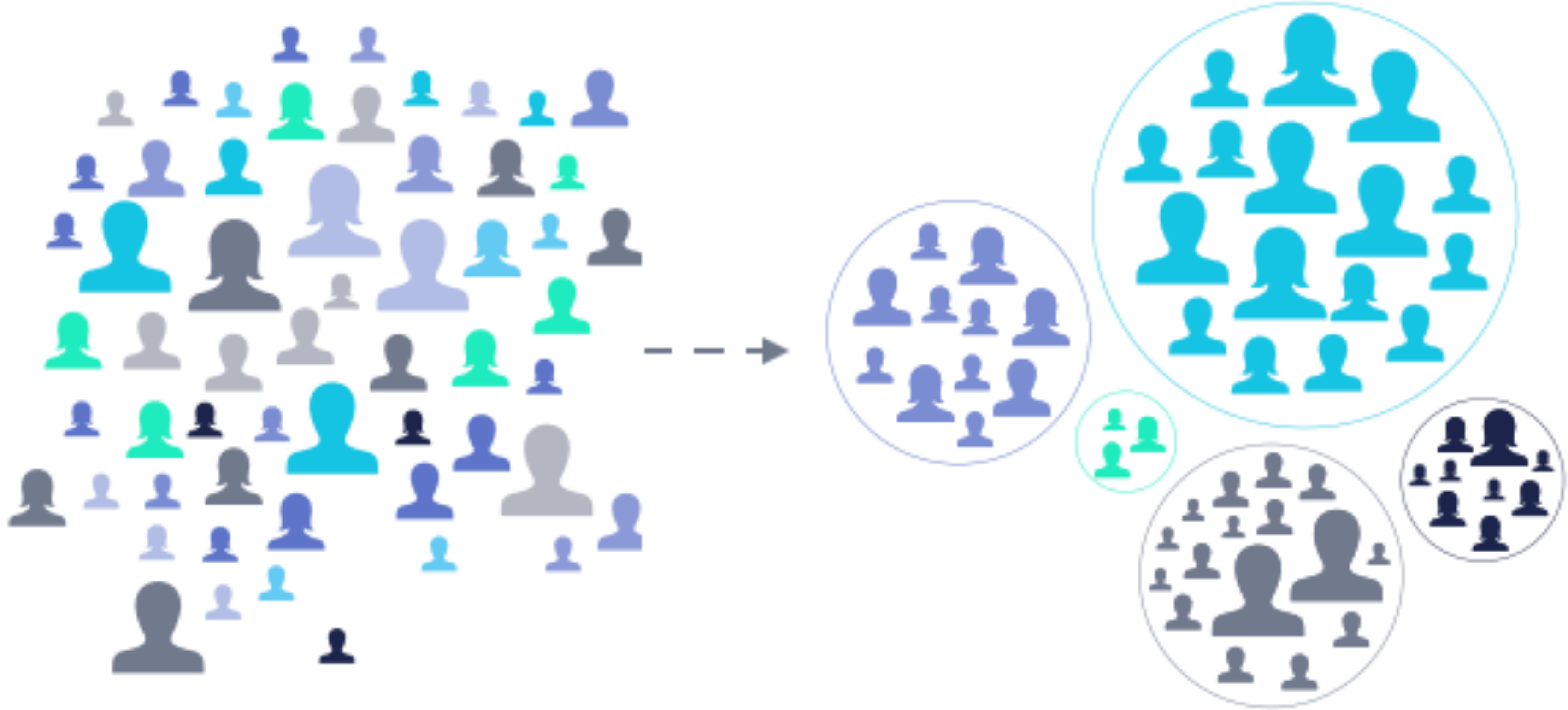
Law of Entropy

- The total entropy of an isolated system can never decrease over time



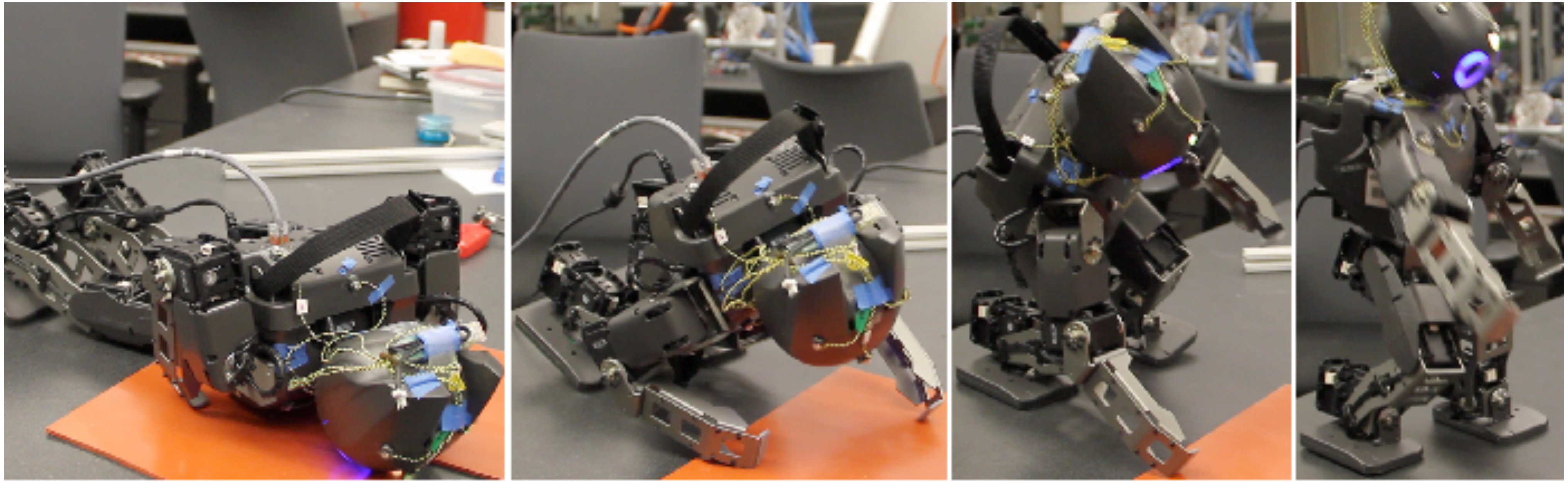
Unsupervised Learning – Uncovering hidden patterns from unlabeled data

Unsupervised Learning Example – Customer Segmentation



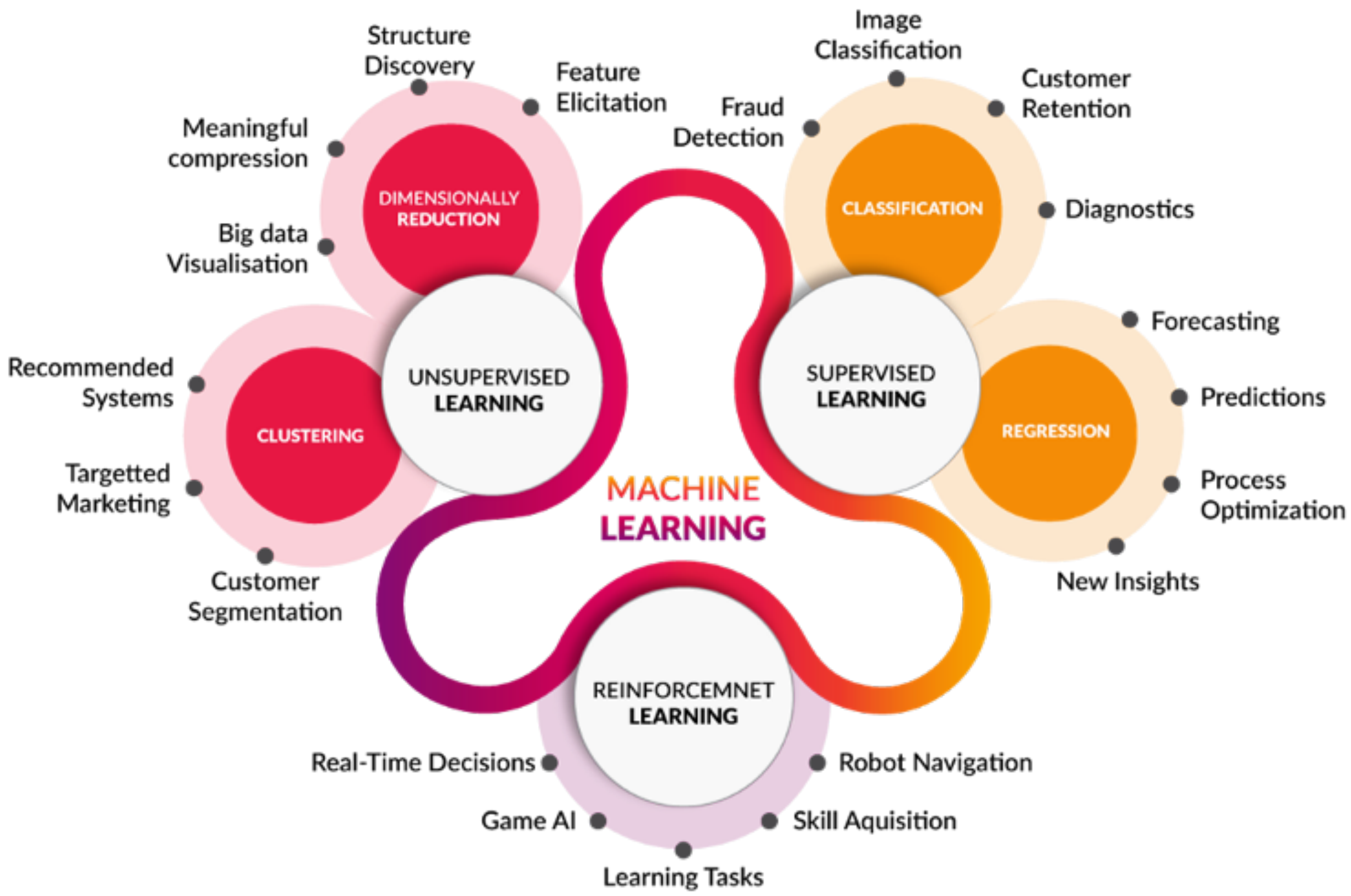
Reinforcement Learning

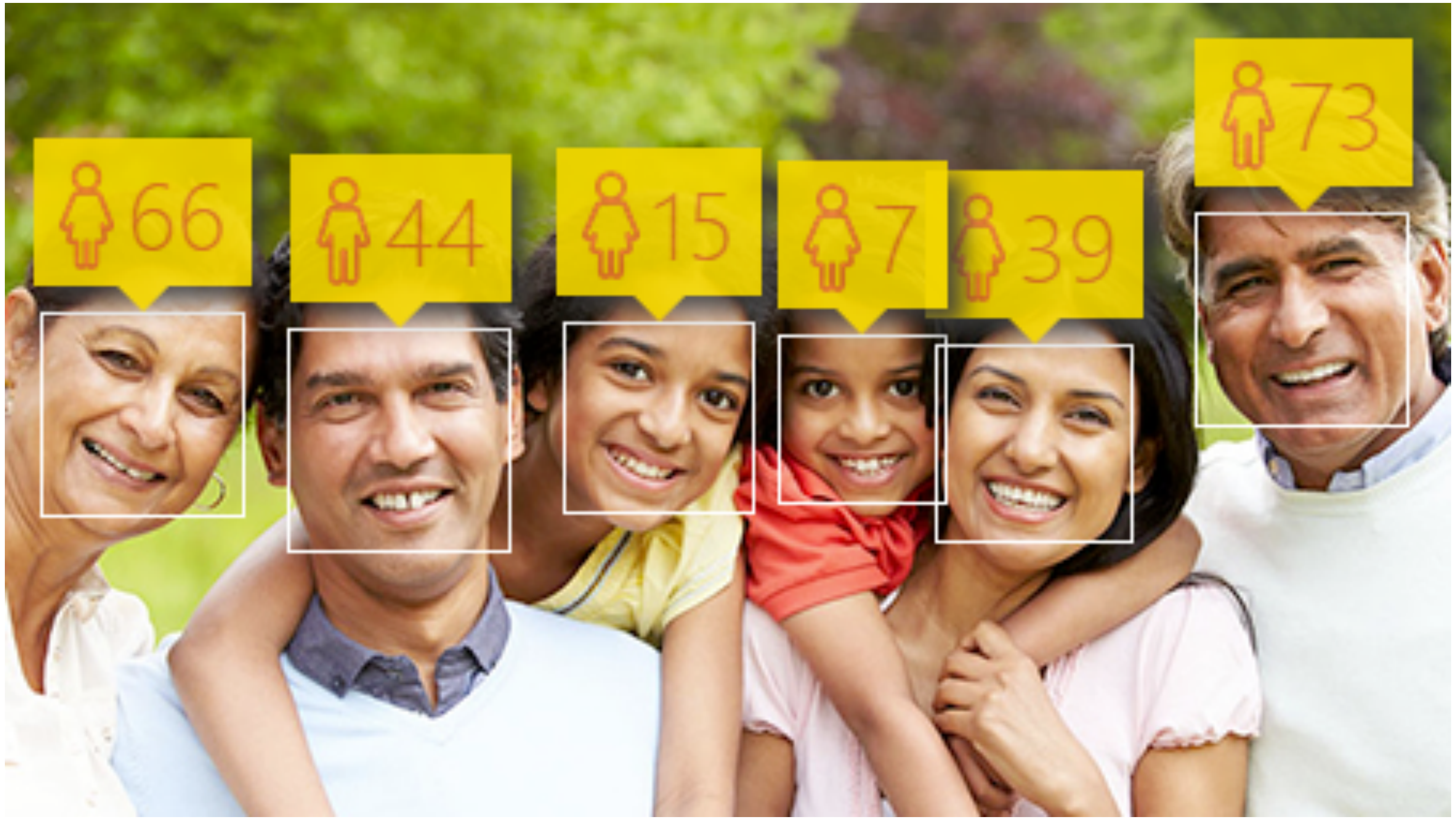
Take actions in an environment so as to maximize some notion of cumulative reward



Reinforcement Learning Example - Robotics

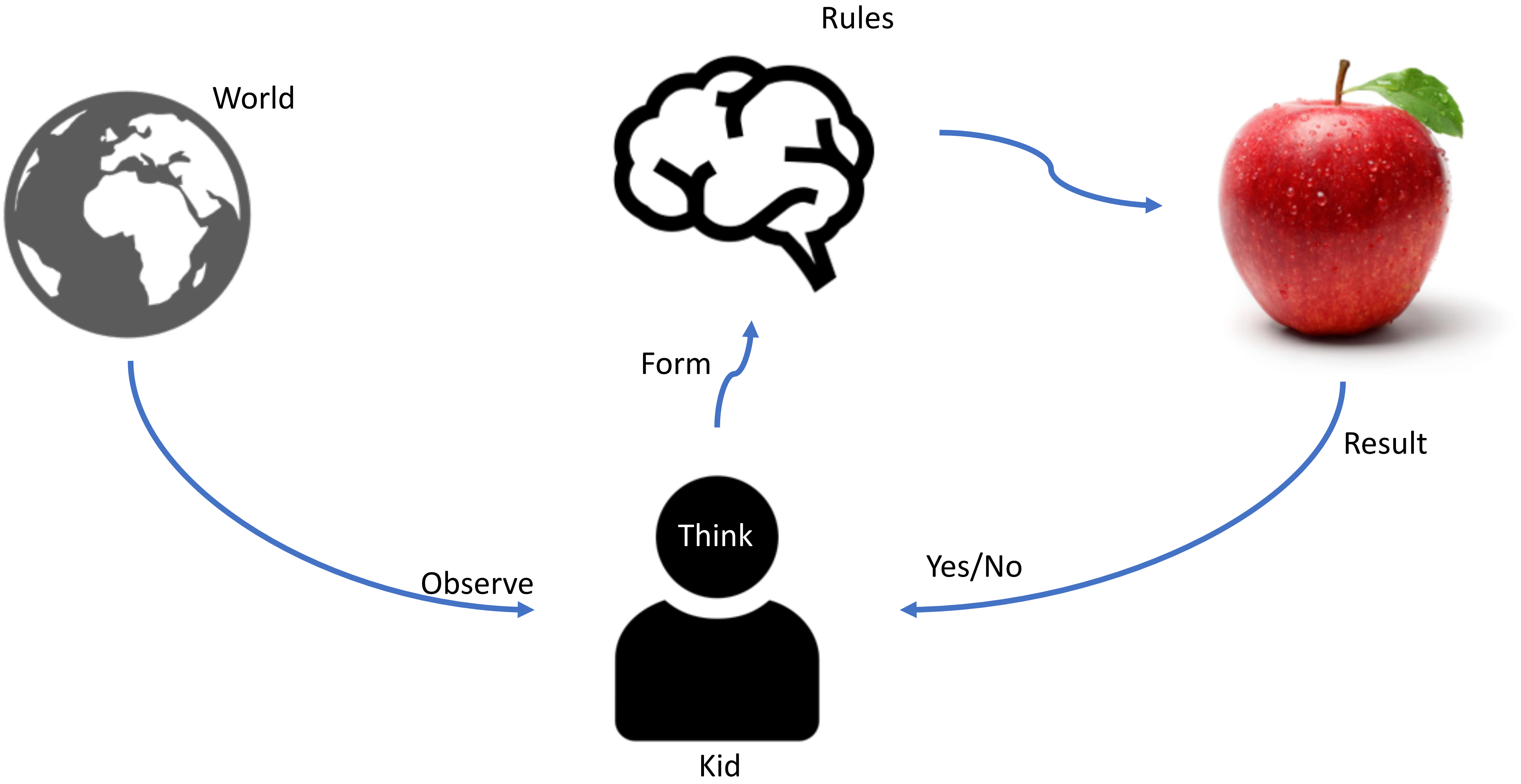
Major Types of AI



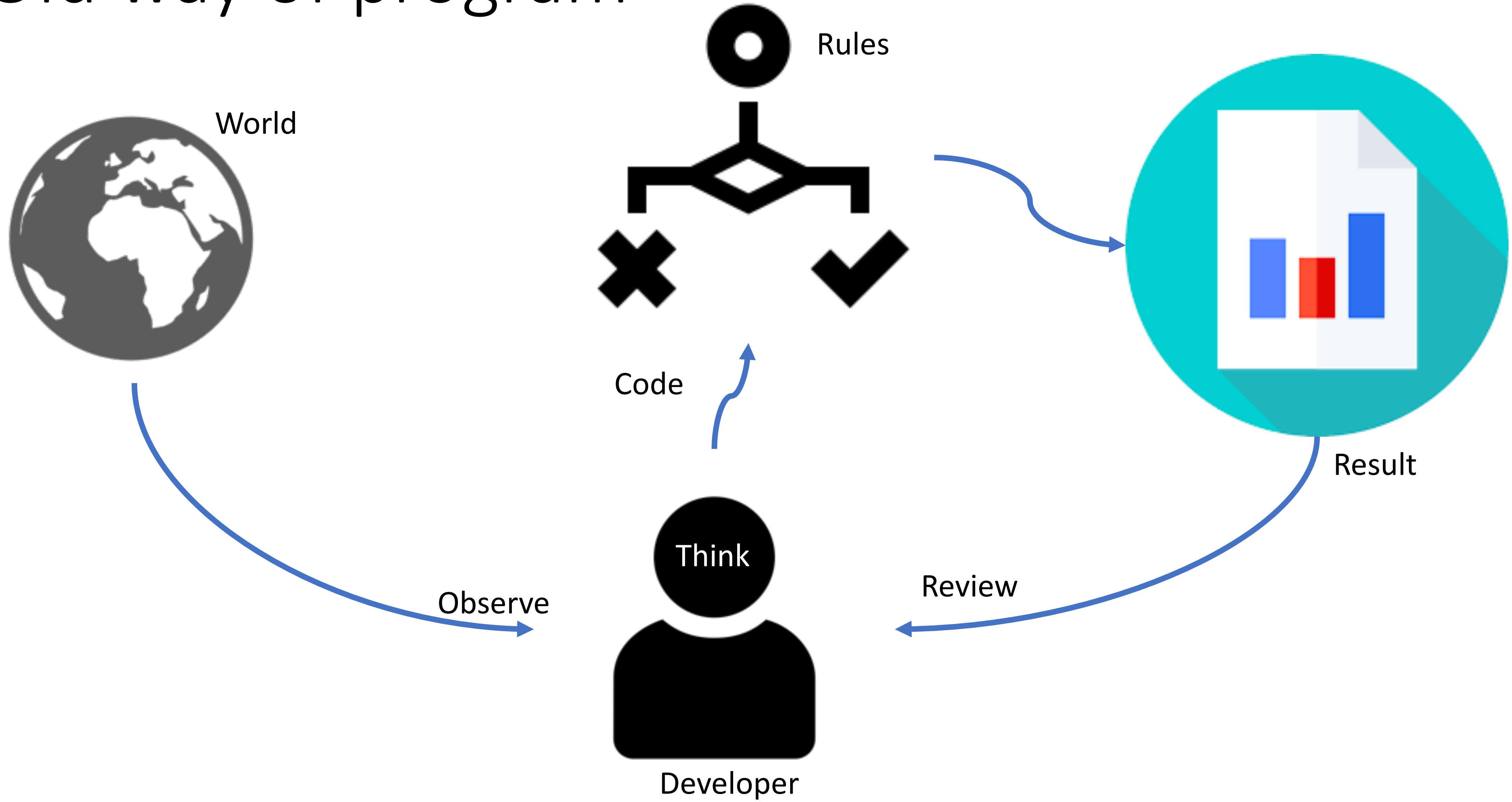


Regression – a prediction method with a real number as an output
Classification – a prediction method with a predefined category

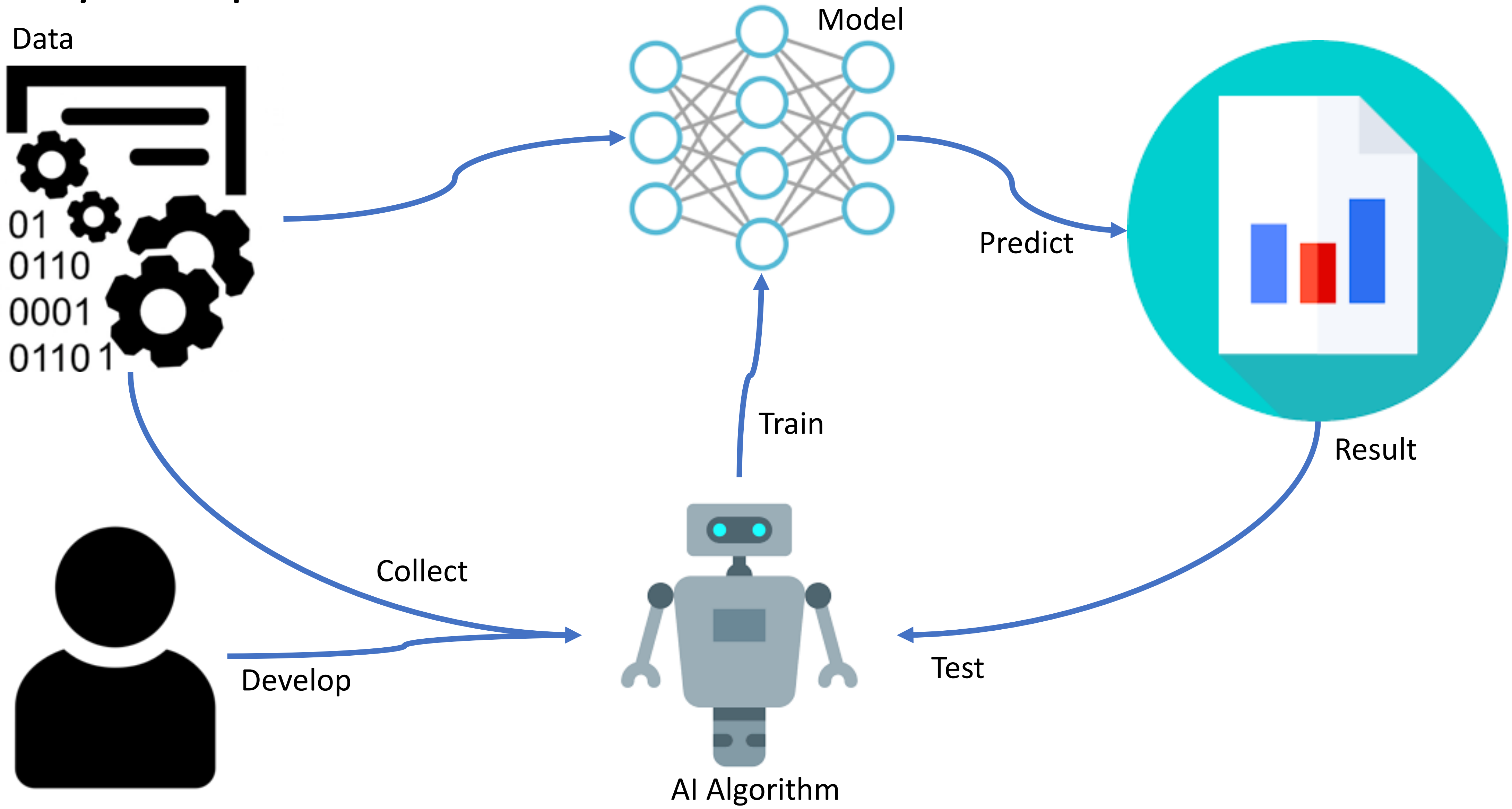
How a kid learn?



Old way of program



Key Steps of AI

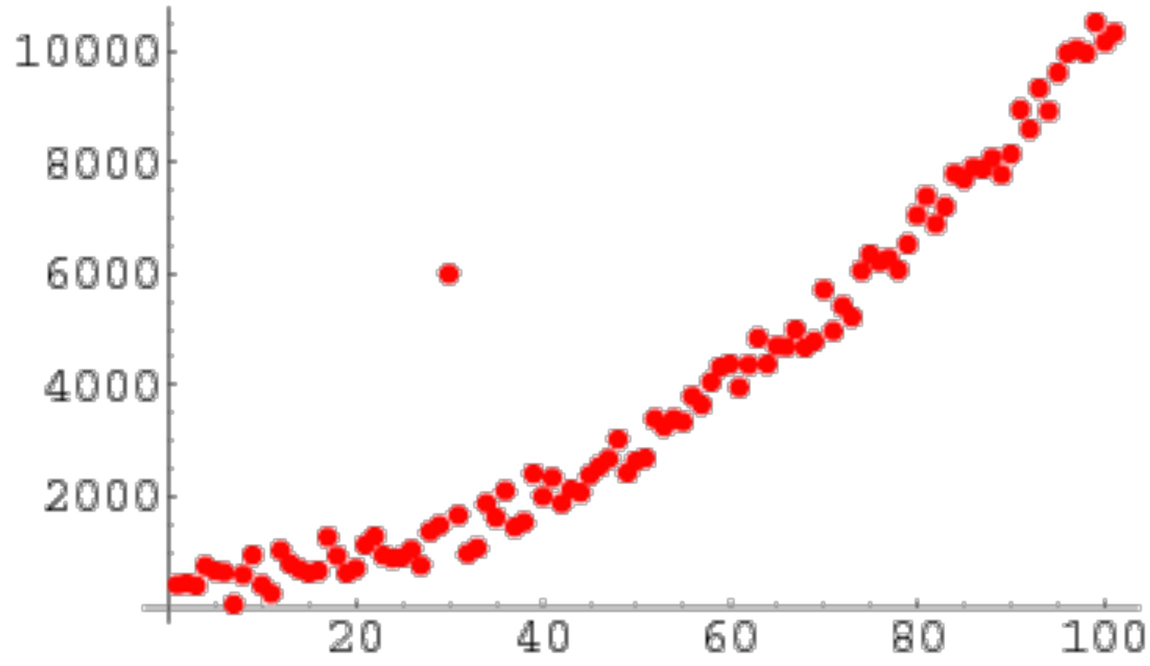


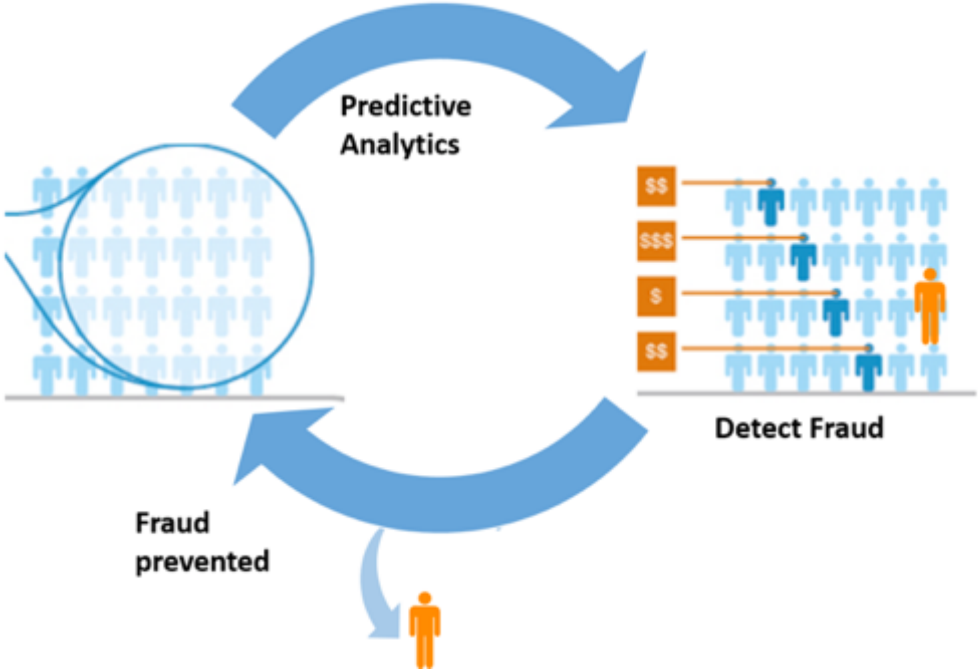

Give a Man a Fish, and You Feed Him for a Day…
Teach a Man To Fish, and You Feed Him for a Lifetime – **Proverb**

Give a Computer a Code, and You Help It Solving for one Situation…
Teach a Computer To Code, and You Help It Solving for a Collection of Situations – **Cupid Proverb**

Memorize VS Generalize

- What is 2 + 1?
- How about 1 + 2?





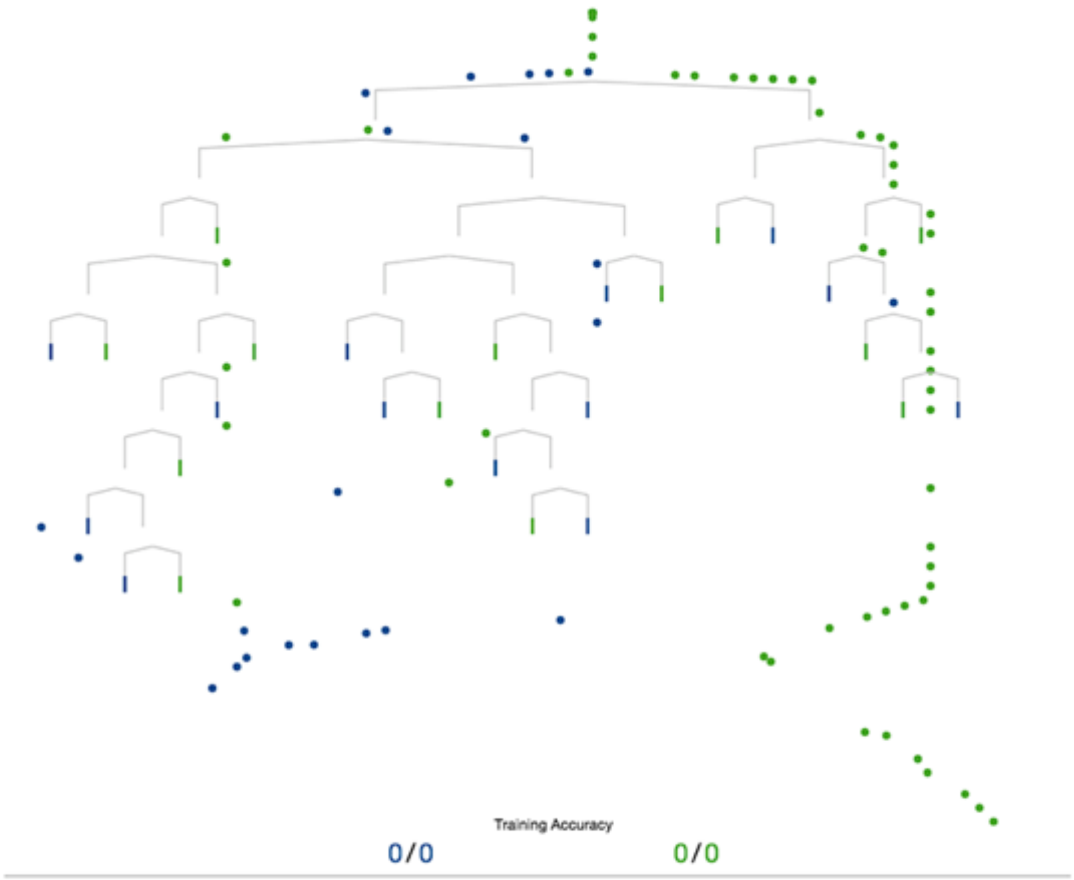
Fraud Detection

McDonald “Guess Who?”

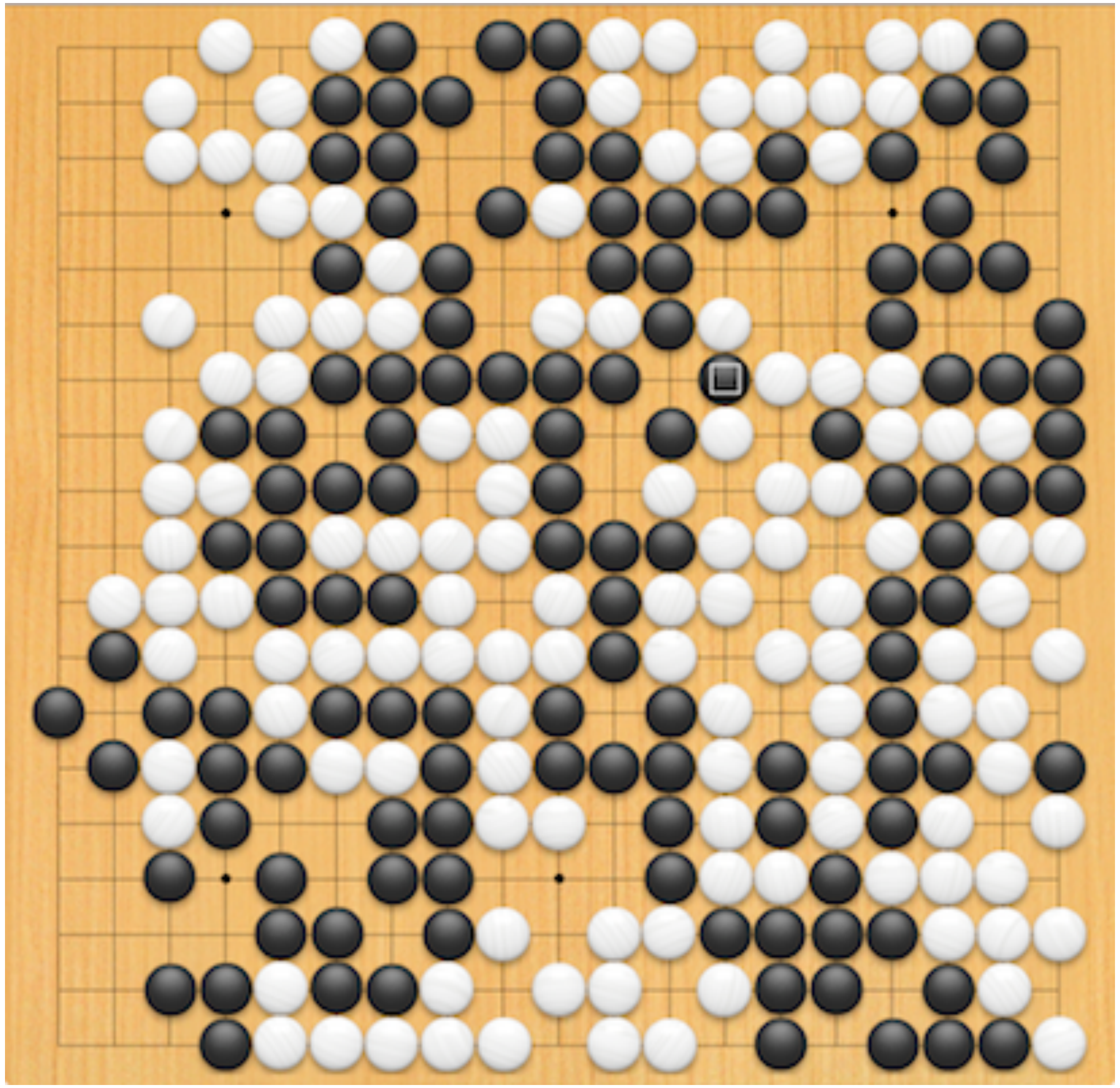
Decision Tree

Making predictions

The newly-trained decision tree model determines whether a home is in San Francisco or New York by running each data point through the branches.




ORIGINAL
MASTERMIND
Game



Main Algorithms of AI

TESLA

UBER

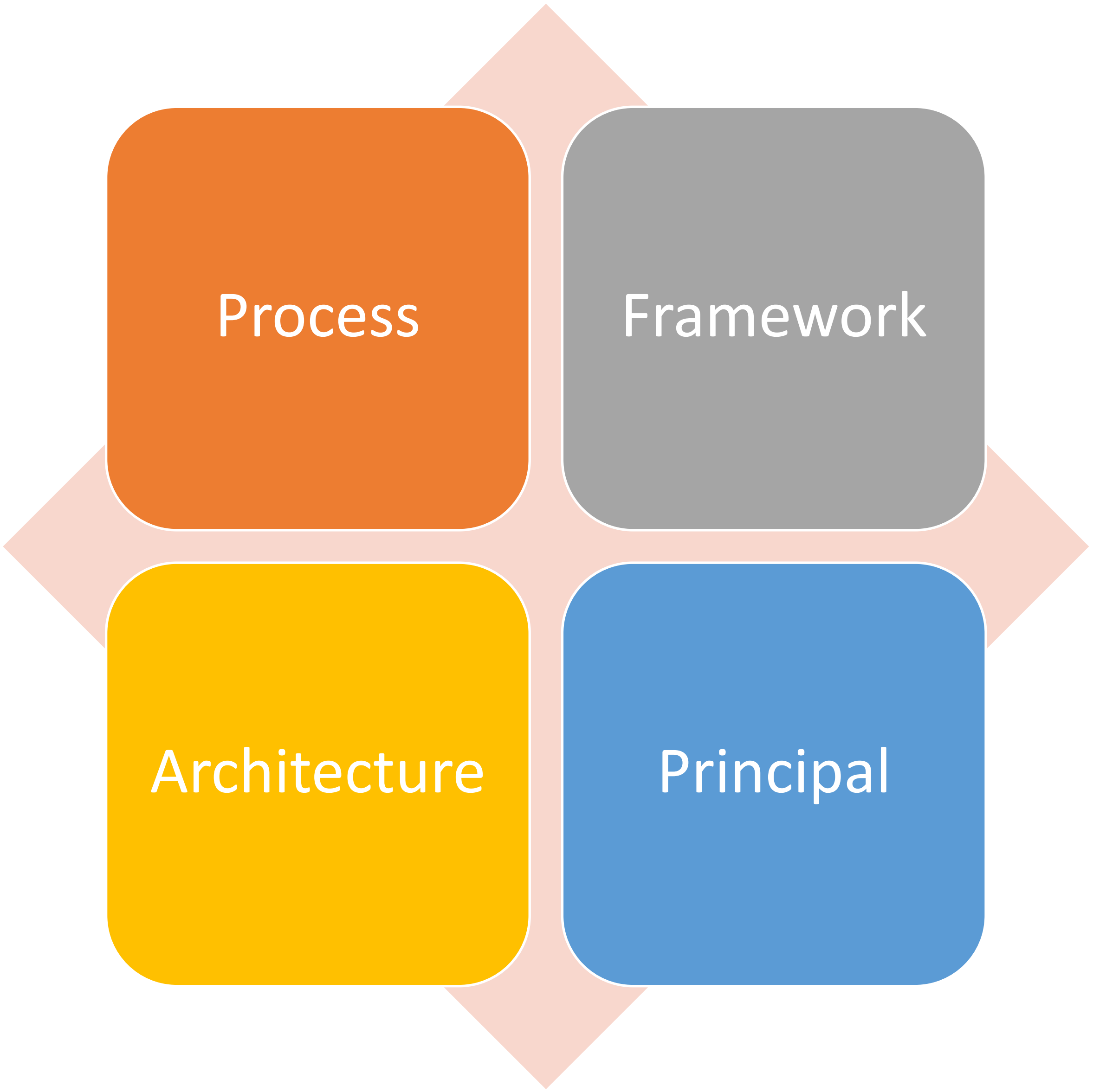
“AI can help me to do homework and the teacher will not find out I didn’t do that?”

There is no Right or Wrong, only Right or Left. But no matter which direction you pick, be persistent and you will cross the finish line of success via either route

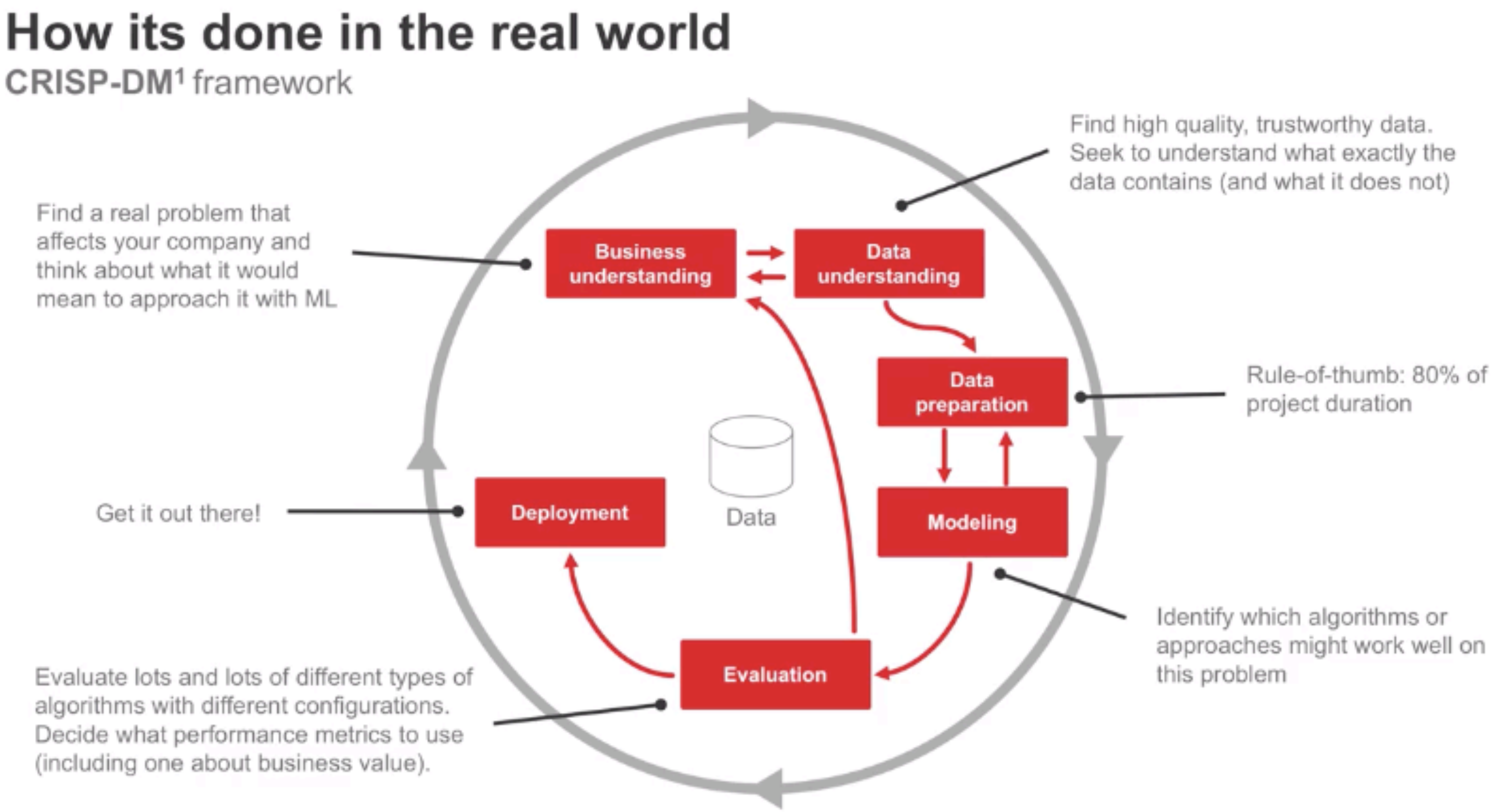
BI is dead

Unless you incorporate AI and other approaches into your BI Strategy

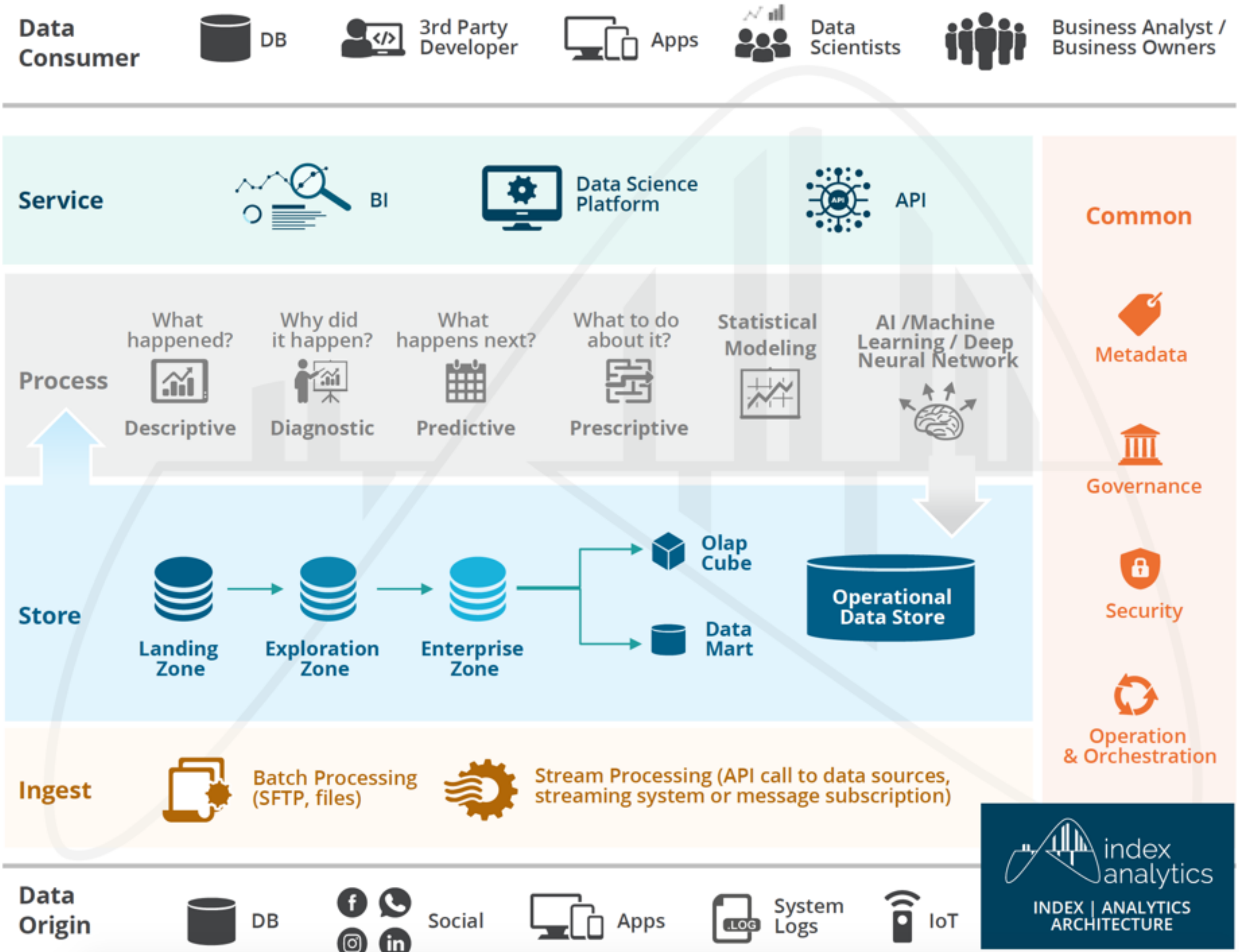
Strategies to resurrect BI



Process: Cross-industry standard process for data mining



https://en.wikipedia.org/wiki/Cross-industry_standard_process_for_data_mining


Data Consumer
DB
3rd Party Developer
Apps
Data Scientists
Business Analyst / Business Owners
Service
BI
Data Science Platform
API
Common
Process
What happened?
Descriptive
Why did it happen?
Diagnostic
What happens next?
Predictive
What to do about it?
Prescriptive
Statistical Modeling
AI /Machine Learning / Deep Neural Network
Metadata
Governance
Store
Landing Zone
Exploration Zone
Enterprise Zone
Olap Cube
Data Mart
Operational Data Store
Security
Operation & Orchestration
Ingest
Batch Processing (SFTP, files)
Stream Processing (API call to data sources, streaming system or message subscription)
index analytics
INDEX | ANALYTICS ARCHITECTURE
Data Origin
DB
Social
Apps
System Logs
LOG
IoT

Principal

Single comprehensive platform to centralize business data used for machine learning

One version of the truth helps protect against data scientists training models on the wrong data

Enterprise grade self-service that empowers business users and data scientists to work together

Open to the ecosystem of machine learning technology and services

APIs to expose your governed business data to cutting edge open-source and vendor ML products used by your data scientists today

Enable the workforce to take action on predictive workflows wherever they are

Framework: S.O.W.E

YouTube

Very Good Story for a presentation Cupid
But I don’t believe you really taught your kid AI as you said

Cupid Chan

- CTO of Index Analytics
- Board of Directors and TSC, Linux Foundation ODPi
- Organizer of a Big Data In Action Meetup in Washington/Baltimore area with 1600 members

www.linkedin.com/in/cupidchan/

@cupidckchan