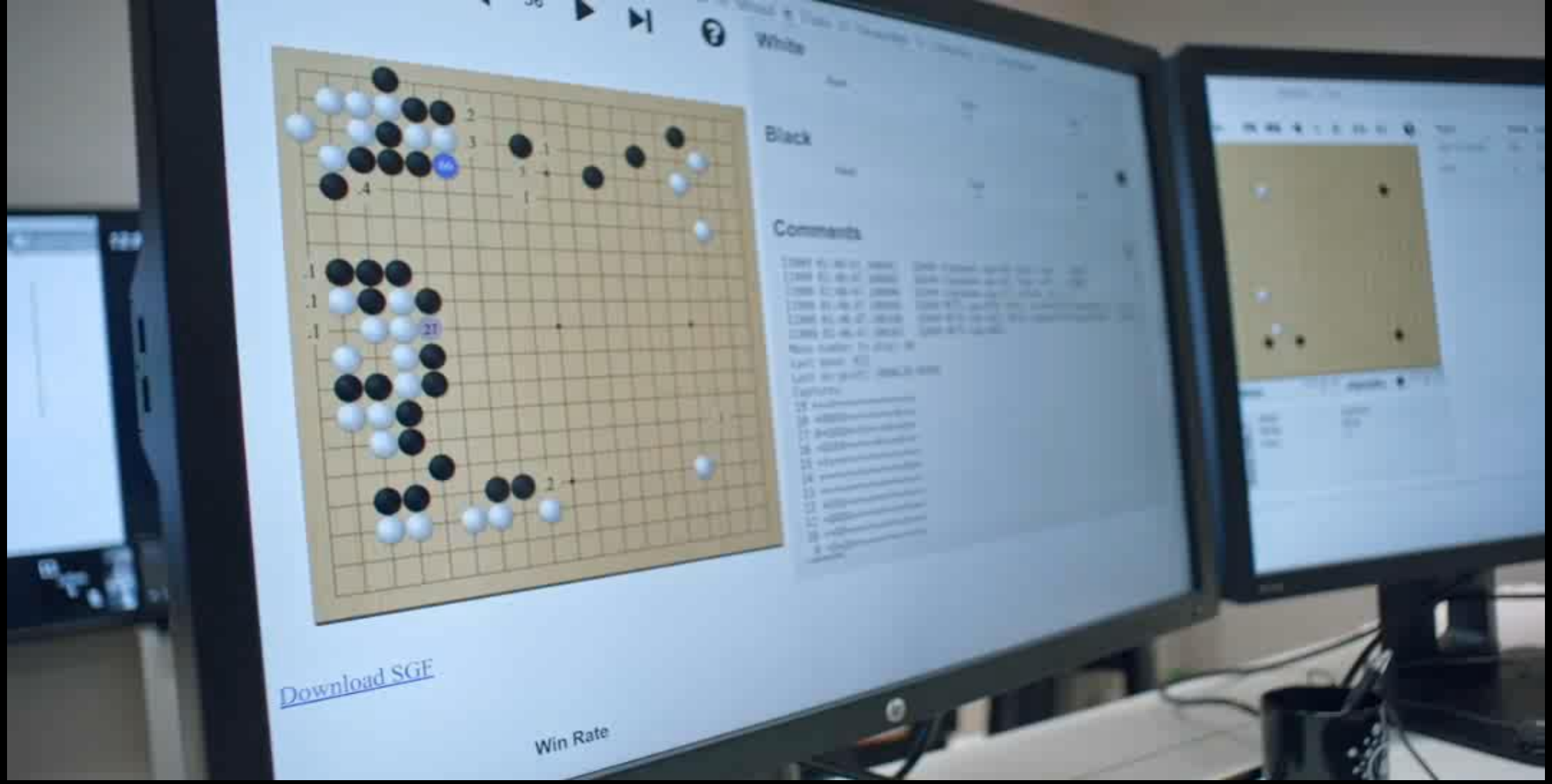
Tufts
UNIVERSITY

LEARNING BY EXAMPLE

ARTIFICIAL INTELLIGENCE | COMP 131

TODAY ON AI

- Machine Learning paradigms
- Data, discriminate functions, and performance
- Supervised learning
- Decision trees
- Questions?

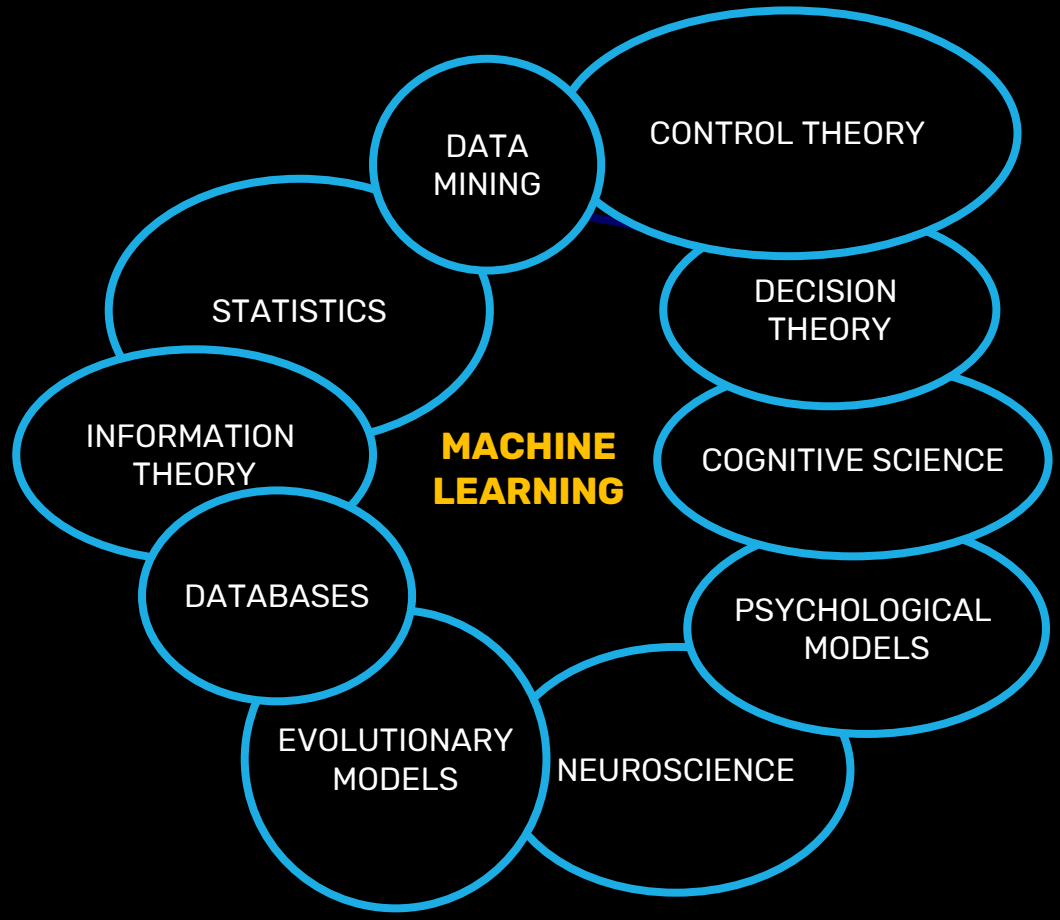


03

ARTIFICIAL INTELLIGENCE

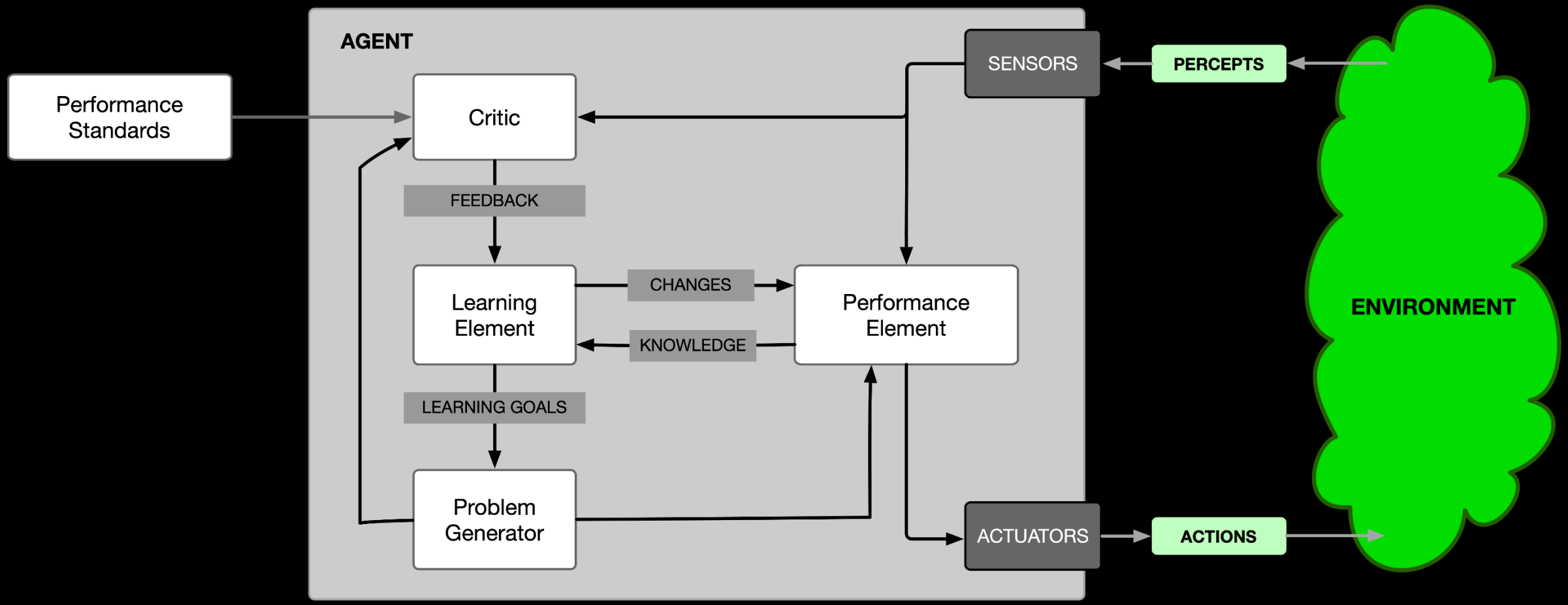
from AlphaGo (2017).

- No human experts
  - Industrial and manufacturing control
  - Mass spectrometer analysis, drug design, astronomic discovery
- Black-box human expertise
  - Face, handwriting, and speech recognition
  - Driving a car, flying a plane
- Rapidly changing phenomena
  - Credit scoring and financial modeling
  - Diagnosis, fraud detection
- Need for customization/personalization
  - Personalized news reader
  - Movie/book recommendation



- There are many types of learning: **rote learning**, **associative learning**, **learning by being told**, **learning from examples**, **learning by analogy**
- Humans tend to learn from past experiences
- A computer system **does not have** life experiences
- A computer system learns from **data** as a representation of past **experiences** of an application domain

The ultimate goal of **Machine learning** is to learn a target function that can be used to predict the values of another unknown or complex function.


AGENT
Performance Standards
Critic
FEEDBACK
Learning Element
CHANGES
KNOWLEDGE
Performance Element
LEARNING GOALS
Problem Generator
SENSORS
PERCEPTS
ENVIRONMENT
ACTUATORS
ACTIONS

SECTION 01

ML paradigms

- Type of feedback:
  - Labeled examples: **supervised**
  - None or unlabeled examples: **unsupervised**
  - Reward: **reinforcement**

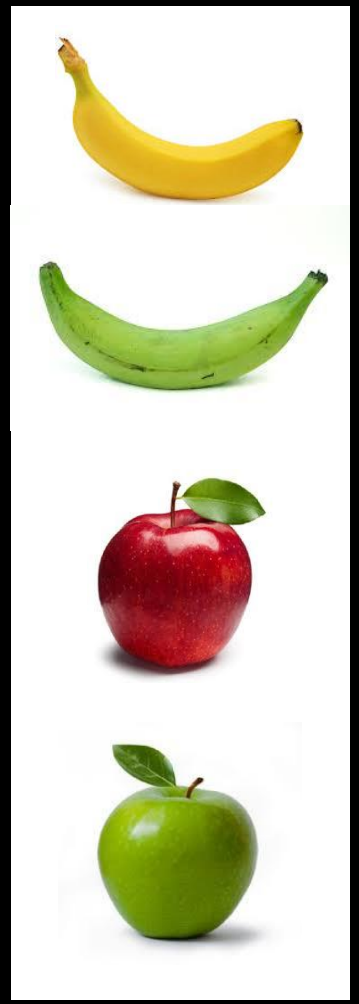
- Data representation:
  - Feature vector: **attribute-based**
  - First-order logic or graph: **relational**

- Origin of the knowledge:
  - Knowledge-free: **empirical**
  - Knowledge-guided: **analytical**

The design of a ML system is affected by:

- **Performance element used:** utility-based agent, reactive agent, logical agent
- **Functional component to be learned:** classifier, evaluation function, perception-action function
- **Representation of functional component**: weighted linear function, logical theory, HMM
- **Feedback available**: correct action, reward, relative preferences

Supervised learning finds a prediction function given **features** and **associated labels**. The prediction function is used to **predict** the label of features **never seen:**



label 1

label 2

label 3

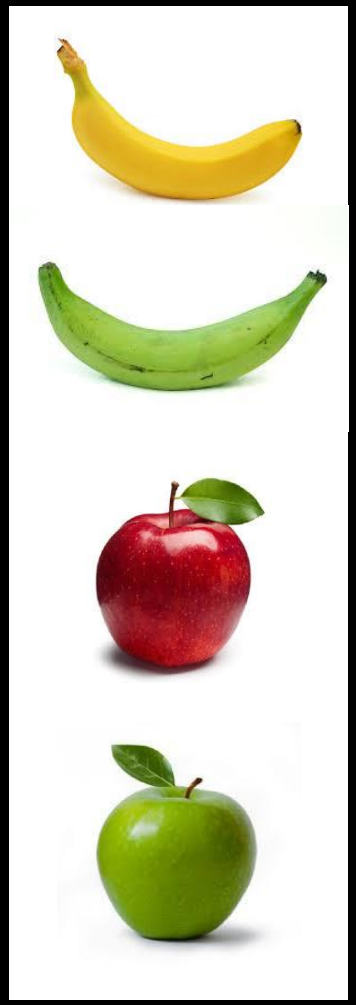
label 4

$f(x) = y$

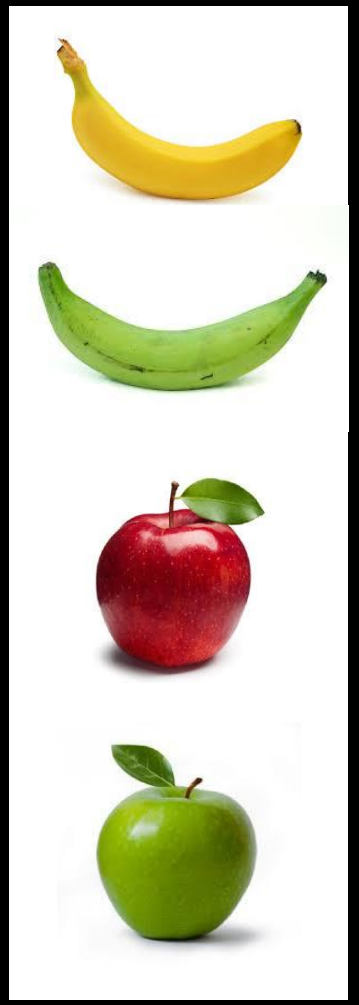
$f($  $) =$ **label 4**

PREDICTED LABEL

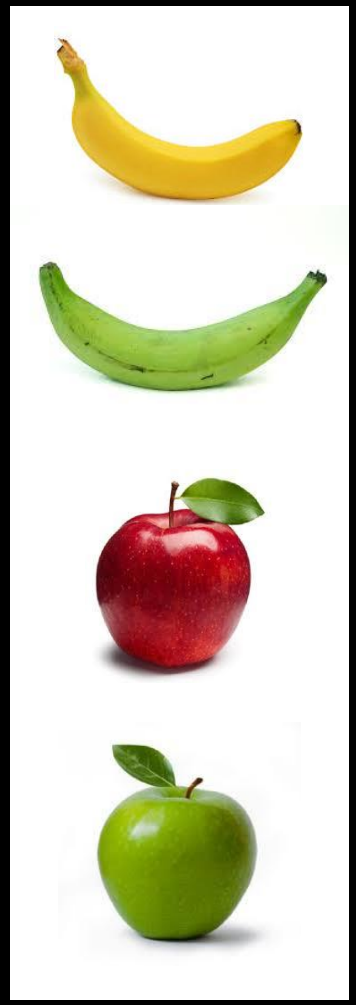
Classification: a finite set of labels



Ranking: Labels are a ranking number



Regression: Labels are real-valued numbers



Classification

- Face recognition
- Low-risk and high-risk loans
- Character recognition
- Spam detection
- Medical diagnosis: From symptoms to illnesses
- Biometrics: Recognition and authentication using physical and/or behavioral characteristics: Face, iris, signature, etc

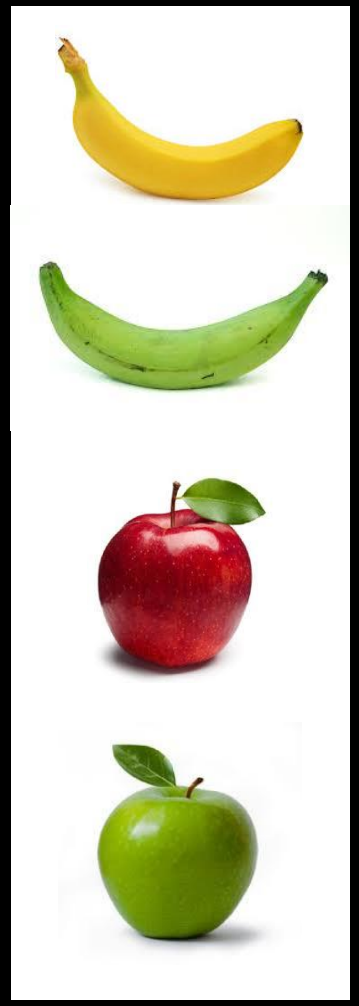
Regression

- Economics/Finance: predict the value of a stock
- Epidemiology
- Car/plane navigation: angle of the steering wheel, acceleration, …
- Temporal trends: weather over time
- Price of a used car

Ranking

- Netflix movie queue ranking
- iTunes
- Flight search
- Web pages ranking

Unsupervised learning finds a prediction function given **features** without **associated labels**. The prediction function is used to **cluster or group** the label of features **never seen:**



? ? ? ?

$f(x) = y$ $f($  $) =$ **cluster 2**

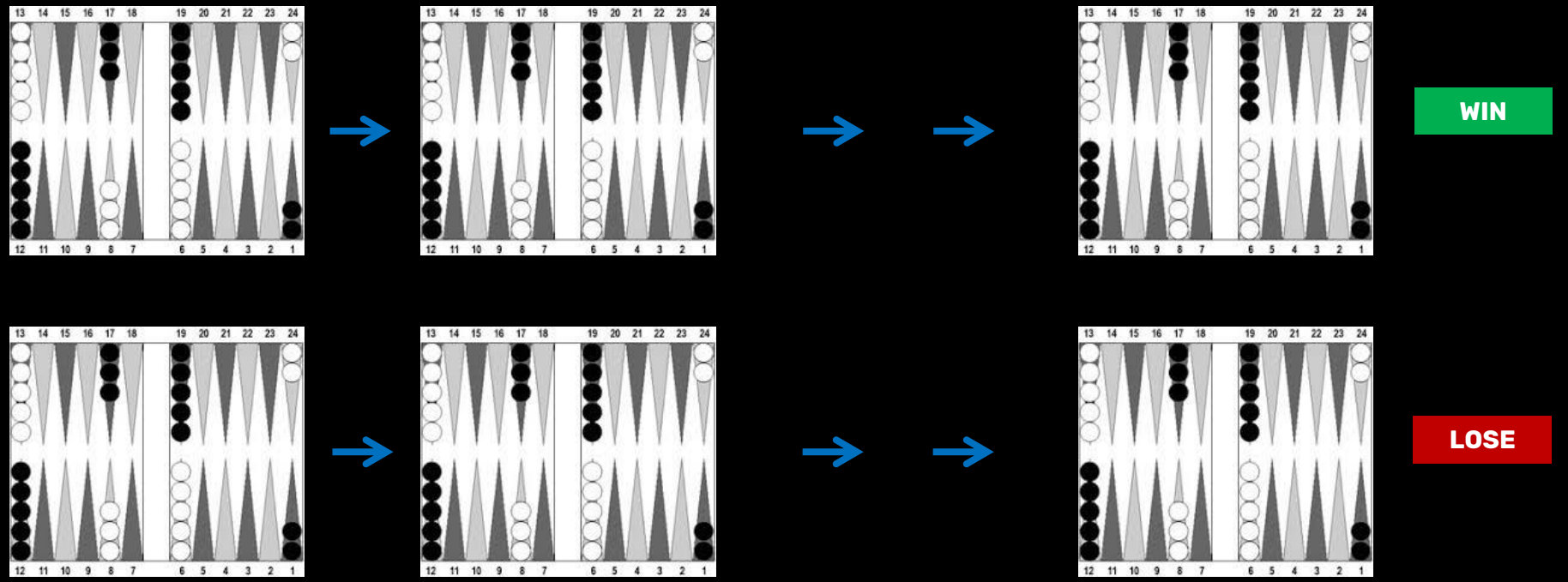
PREDICTED CLUSTER

Applications of Unsupervised learning are:

- Customer segmentation/grouping
- Image compression
- Protein structure motifs
- Data dimensionality reduction

Reinforcement learning learns what action to take in a **sequence** of **examples or states** to maximize a **reward** at the end of a sequence of actions.

BACKGAMMON



SECTION 02

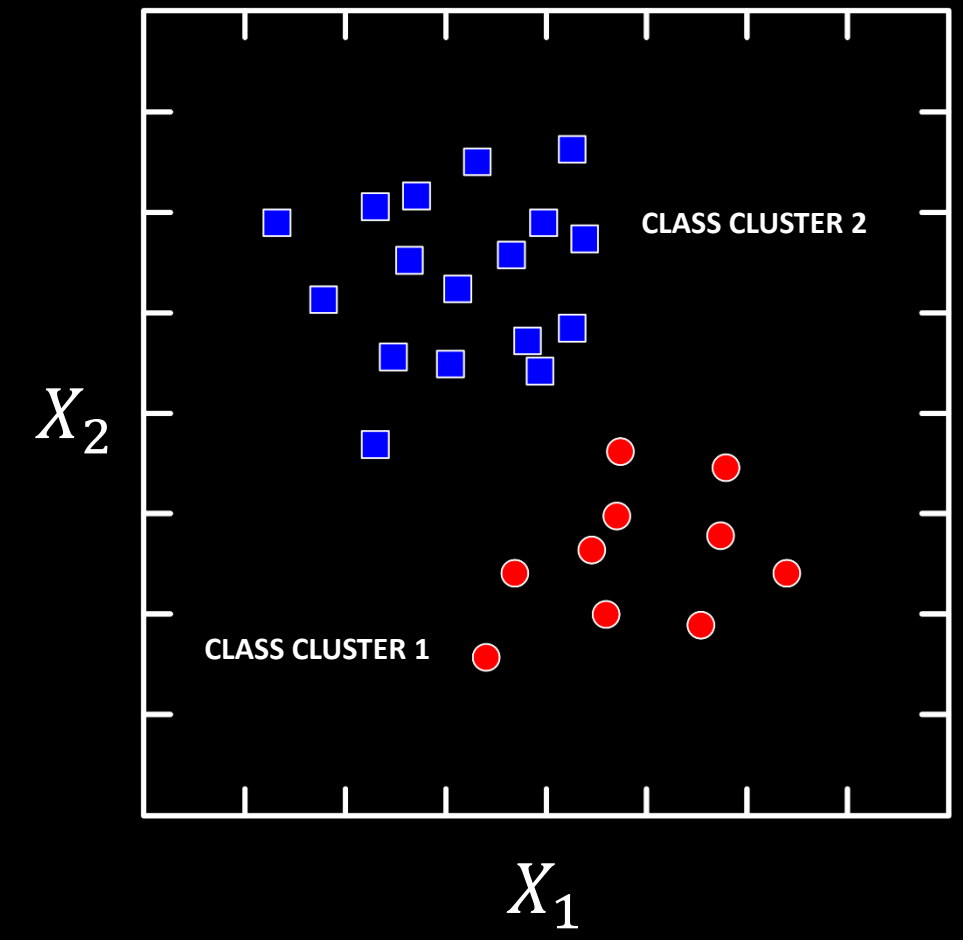
DATA, DISCRIMINATE FUNCTIONS AND PERFORMANCE

Data: A set of data records (also called **examples**, **instances**, or **cases**) described by:

- k **attributes**: A k-dimensional array representing $X_1, X_2, \ldots X_k$
- a **class**: A label for the attributes array

ATTRIBUTE SPACE

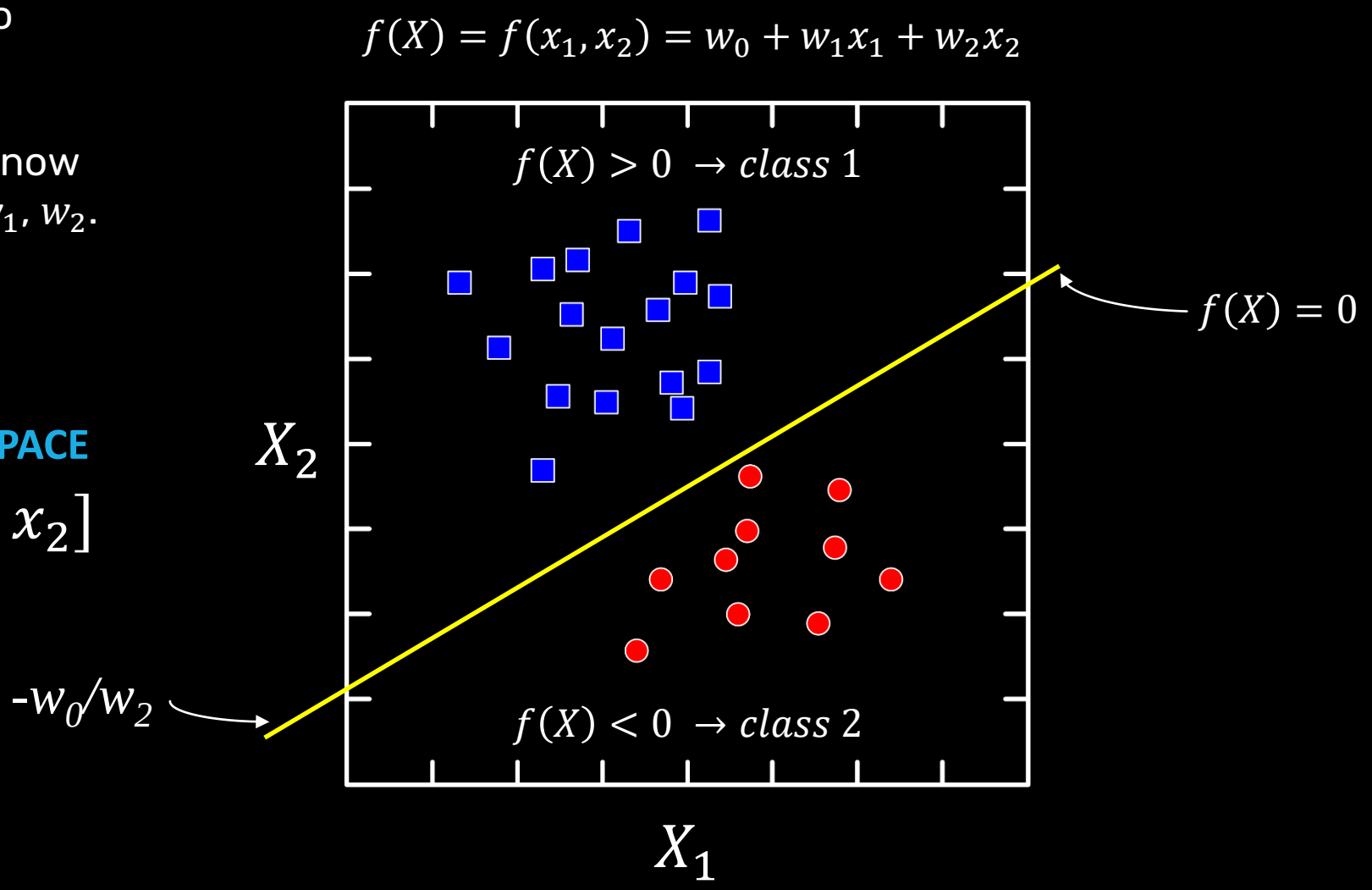
$$X = [x_1, x_2]$$



A **discriminate function** divides the space in areas that belong to different classes.

$f(X) = WX = 0$ will allow you to discriminate class A from B.

The problem is that we do not know the appropriate values for w_0, w_1, w_2.

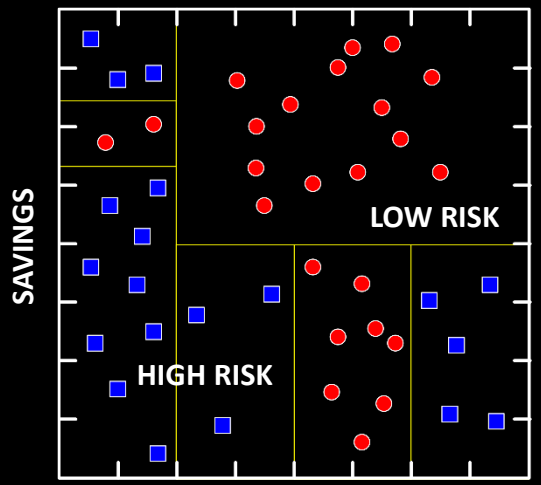


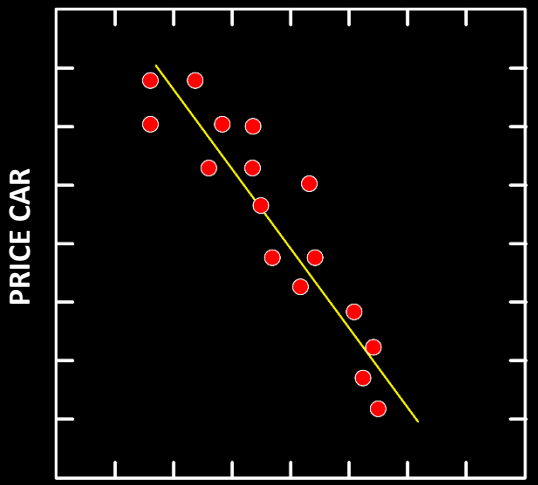
ATTRIBUTE SPACE

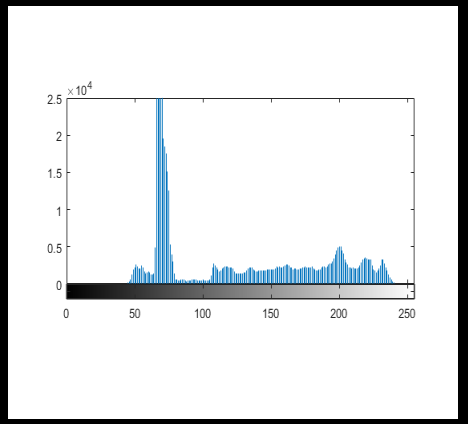
$$X = [x_1, x_2]$$

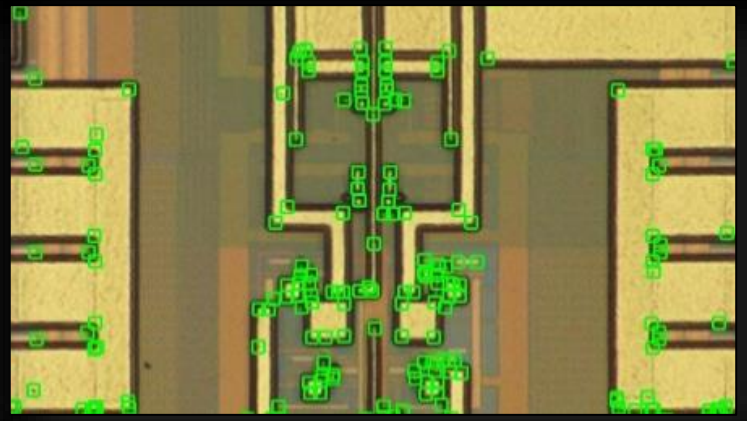
Features used during learning can be anything:

- Raw pixels
- Income/ Level of savings
- Histograms
- Price/Mileage
- Image descriptors









A **learned model** helps the system to perform T **better** as compared to no learning. Measuring the **performance** of the model is done via several metrics:

- **Accuracy**: $$Accuracy = \frac{Number\ of\ correct\ classifications}{Total\ number\ of\ test\ cases}$$

- **Confusion matrix**:

PREDICTED OUTPUT \ ACTUAL OUTPUT	CLASS 1	CLASS 2	CLASS 3
CLASS 1	95%	0	0
CLASS 2	4%	97%	0
CLASS 3	1%	3%	100%

- **RMSE**: $$RMSE = \left[\frac{1}{N}\sum_{c=1}^{N}(o_p - o_a)^2\right]^{\frac{1}{2}}$$

SECTION 03

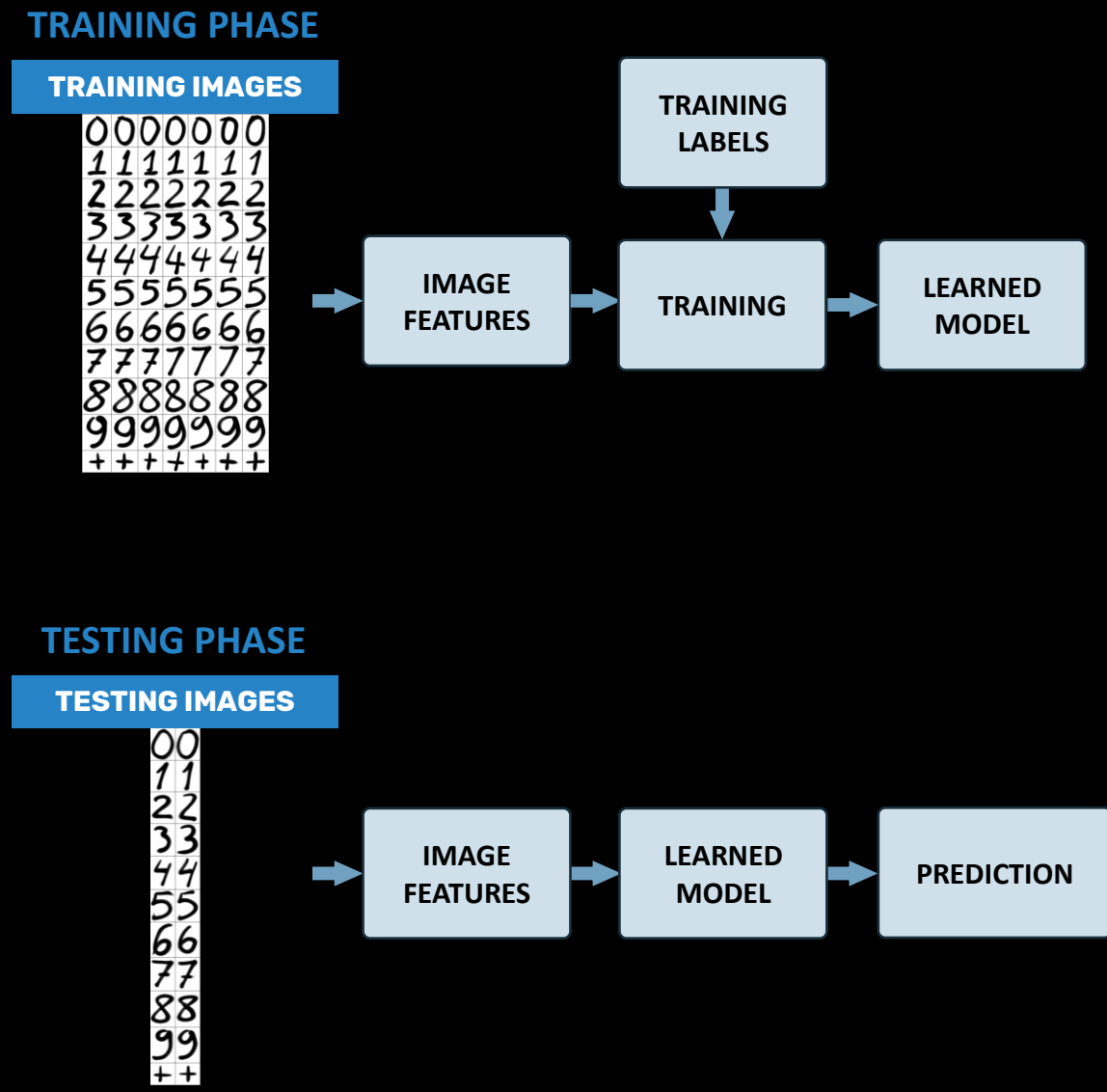
Supervised learning

- **Training** (or **learning**): Learn a model using the training data to **minimize** the prediction error of the discriminate function f
- **Testing**: Using unseen test data, assess the model accuracy applying f to never before testing data

Given:

- a **data set** D
- a **task** T
- a **performance measure** M

A computer system is said to **learn** from D to perform the task T if after learning the system's performance on T improves as measured by M.



The data must be split in three mutually exclusive sets:

- **Training set**: It is a set of labeled examples $\{(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_N, y_N)\}$, used for the training of the model.
- **Validation set**: It is a set of labeled examples $\{(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_N, y_N)\}$, used in the training phase to detect over-fitting.
- **Testing set**: It is a set of labeled examples $\{(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_N, y_N)\}$, used after the training phase to determine the actual performance of the model.

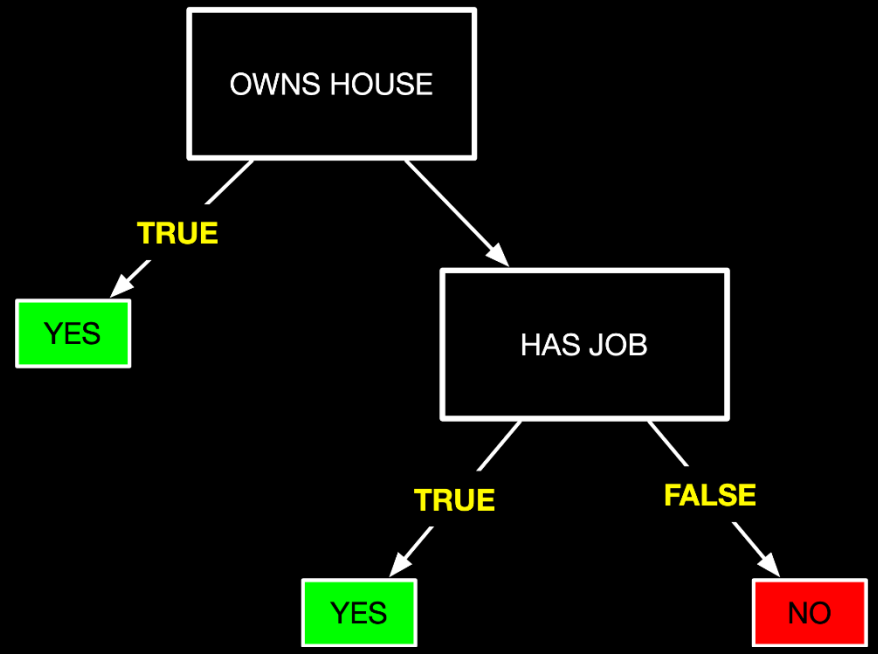
Considerable care must be put into creating the sets.

SECTION 04

Decision trees

Should a loan to be issues to a person?

ID	AGE	HAS JOB	OWNS HOUSE	CREDIT RATING	LOAN?
1	Young	False	False	Fair	**No**
2	Young	False	False	Excellent	**No**
3	Young	True	False	Good	**Yes**
4	Young	True	True	Good	**Yes**
5	Young	False	False	Fair	**No**
6	Middle	False	False	Fair	**No**
7	Middle	False	False	Good	**No**
8	Middle	True	True	Good	**Yes**
9	Middle	False	True	Excellent	**Yes**
10	Middle	False	True	Excellent	**Yes**
11	Old	False	True	Excellent	**Yes**
12	Old	False	True	Good	**Yes**
13	Old	True	False	Good	**Yes**
14	Old	True	False	Excellent	**Yes**
15	Old	False	False	Fair	**No**



Should a loan to be issued?

Decision tree learning is a widely used techniques for classification. The **classification model** is a **tree**, called **decision tree**.

The basic idea is a **greedy divide-and-conquer recursively partition** of the training set according to the **most important** attribute (category or continuous values) of the examples.

Conditions for stopping:

- All examples for a given node belong to the same class
- There are no remaining attributes for further partitioning – majority class is the leaf
- There are no examples left

```
1  function DT-learning(EXAMPLES, ATTRIBUTES, P-EXAMPLES)
2    return TREE
3    if EXAMPLES is empty then
4      return PLURALITY-VALUE(P-EXAMPLES)
5    else if all EXAMPLES have the same classification then
6      return the classification
7    else if ATTRIBUTES is empty then
8      return PLURALITY-VALUE(EXAMPLES)
9    else
10     A = argmax_{a ∈ attributes} Importance(a, EXAMPLES)
11     tree = a new decision tree with root test A
12     for each value v_k of A do
13       exs = {e : e ∈ examples ∧ e.A = v_k}
14       subtree = DT-learning(exs, ATTRIBUTES − A, EXAMPLES)
15       add subtree to tree with label (A = v_k)
16     return TREE
```

Information theory provides a mathematical basis for measuring the information content.

- Information theory measures the added value of information:
  - With a **fair coin**, there is no information about the result of the tossing, so any prior information to the event is extremely valuable
  - With a rigged coin, the information is much less valuable

- Information theory uses **bits** to measure information content

Entropy is defined as:

- **$P(c_i)$ is the probability of class c_i in data set D**
- We use entropy as a measure of impurity or disorder of data set D. (Or, a measure of information in a tree)

$$E(D) = -\sum_{i=1}^{|C|} P(c_i) \log_2 P(c_i)$$

$$\sum_{i=1}^{|X|} P(c_i) = 1$$

Example:
A data set D has **50%** positive examples ($P(positive) = 0.5$) and **50%** negative examples ($P(negative) = 0.5$):

$$E(D) = -0.5 \times \log_2 0.5 - 0.5 \times \log_2 0.5 = 1$$

A data set D has **20%** positive examples ($P(positive) = 0.2$) and **80%** negative examples ($P(negative) = 0.8$):

$$E(D) = -0.2 \times \log_2 0.2 - 0.8 \times \log_2 0.8 = 0.7219$$

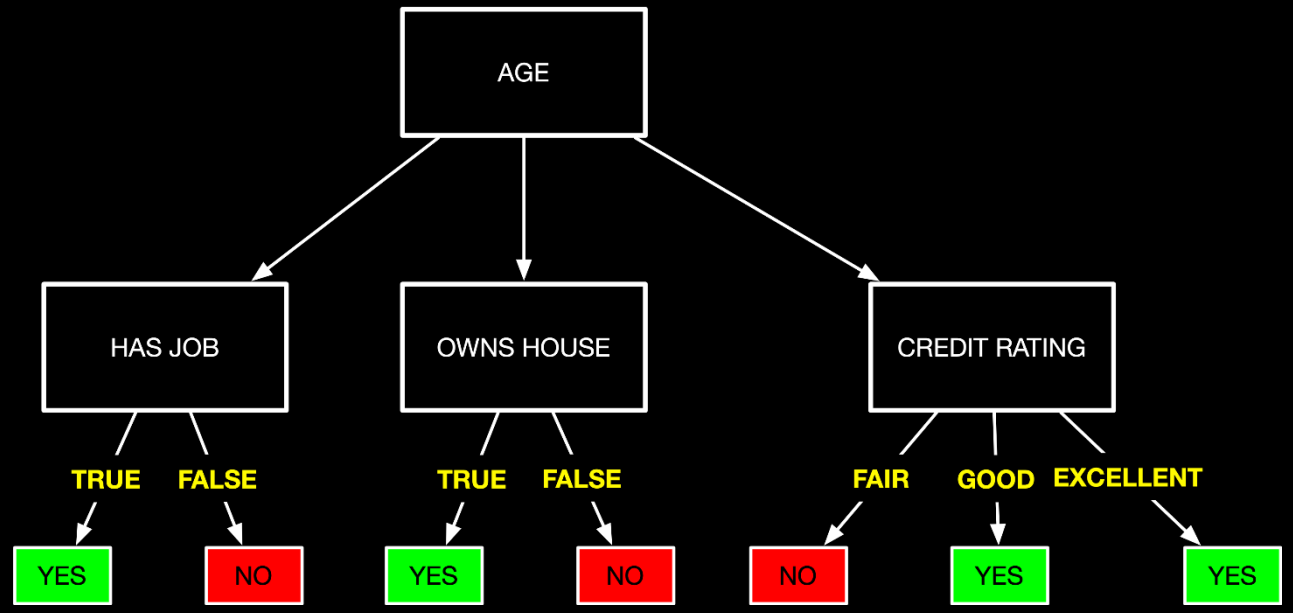
A data set D has **100%** positive examples ($P(positive) = 1.0$) and **no** negative examples ($P(negative) = 0$):

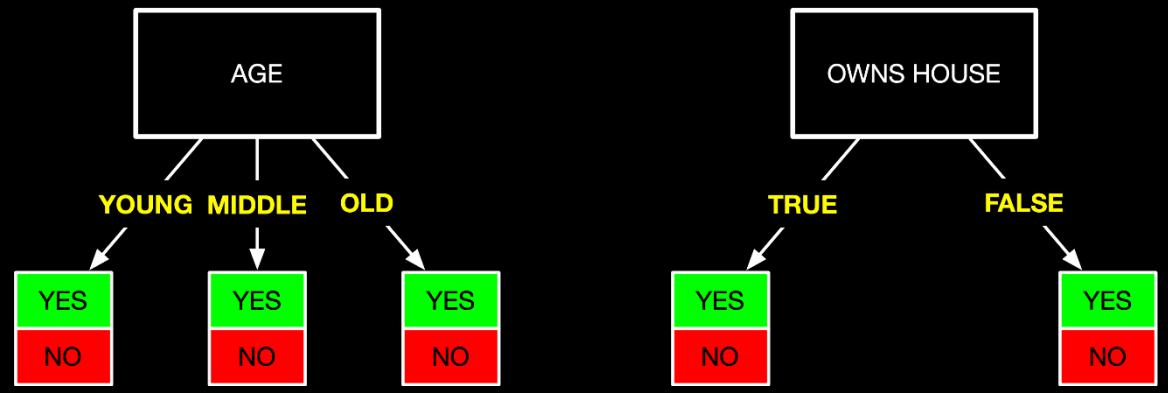
$$E(D) = -1.0 \times \log_2 1.0 - 0 \times \log_2 0 = 0$$

As the data includes **more examples of the same sign** (it becomes purer), the entropy value **becomes smaller**.

In order to branch the tree, we choose the attribute according to the principal of **information gain**, that is the expected reduction of entropy caused by the partitioning.

Intuitively a **good attribute** splits the examples into subsets that are (ideally) homogeneous.





Given a set of examples D:

- We compute its entropy: $E(D) = -\sum_{i=1}^{|C|} P(c_i) \log_2 P(c_i)$

- Selecting the attribute A_i will partition D into subsets D_1, D_2,…,D_v. For A_i, the estimated entropy is:

$$E_{A_i}(D) = \sum_{i=1}^{v} \frac{|D_i|}{|D|} E(D_i)$$

- The **information gained** by selecting A_i to partition the data is:

$$gain(D, A_i) = E(D) - E_{A_i}(D)$$

- **Choose** the attribute A_i with **maximum information gain**

Should a loan to be issues to a person?

ID	AGE	HAS JOB	OWNS HOUSE	CREDIT RATING	LOAN?
1	Young	False	False	Fair	**No**
2	Young	False	False	Excellent	**No**
3	Young	True	False	Good	**Yes**
4	Young	True	True	Good	**Yes**
5	Young	False	False	Fair	**No**
6	Middle	False	False	Fair	**No**
7	Middle	False	False	Good	**No**
8	Middle	True	True	Good	**Yes**
9	Middle	False	True	Excellent	**Yes**
10	Middle	False	True	Excellent	**Yes**
11	Old	False	True	Excellent	**Yes**
12	Old	False	True	Good	**Yes**
13	Old	True	False	Good	**Yes**
14	Old	True	False	Excellent	**Yes**
15	Old	False	False	Fair	**No**

$$E(D) = -\frac{6}{15} \times \log_2 \frac{6}{15} - \frac{9}{15} \times \log_2 \frac{9}{15} = 0.971$$

Should a loan be issued?

$$E_{Owns\_house}(D) = \frac{6}{15} E(D_1) + \frac{9}{15} E(D_2) = 0.551$$

$Owns = T \quad D_{1\,NO} = \{\} \; D_{1\,YES} = \{4, 8, 9, 10, 11, 12\} \; E(D_1) = 0$

$Owns = F \quad D_{2\,NO} = \{1, 2, 5, 6, 7, 15\} \; D_{2\,YES} = \{3, 13, 14\} \; E(D_2) = 0.918$

$$E_{Age}(D) = \frac{5}{15} E(D_3) + \frac{5}{15} E(D_4) + \frac{5}{15} E(D_5) = 0.888$$

$Age = \text{Young} \quad D_{3\,NO} = \{1, 2, 5\} \; D_{3\,YES} = \{3, 4\} \; E(D_3) = 0.971$

$Age = \text{Middle} \quad D_{4\,NO} = \{6, 7\} \; D_{4\,YES} = \{8, 9, 10\} \; E(D_4) = 0.971$

$Age = \text{Old} \quad D_{5\,NO} = \{15\} \; D_{5\,YES} = \{11, 12, 13, 14\} \; E(D_5) = 0.722$

$$E_{Has\_job}(D) = 0.647$$

$$E_{Credit\_rating}(D) = 0.608$$

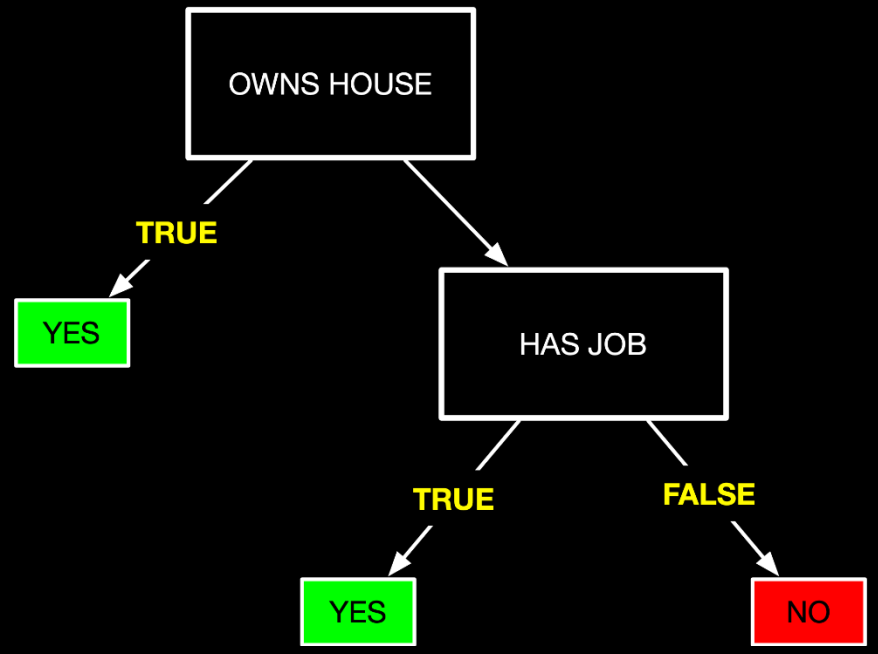
$$gain(D, Owns\_house) = 0.971 - 0.551 = 0.420$$
$$gain(D, Age) = 0.971 - 0.888 = 0.083$$
$$gain(D, Has\_job) = 0.971 - 0.647 = 0.324$$
$$gain(D, Credit\_rating) = 0.971 - 0.608 = 0.363$$

The best first attribute is:

$$\boldsymbol{gain(D, Owns\_house) = 0.420}$$

An important aspect of decision trees is that they can be **converted** to a set of human-readable **rules**:

Each path from the root to a leaf is a rule:

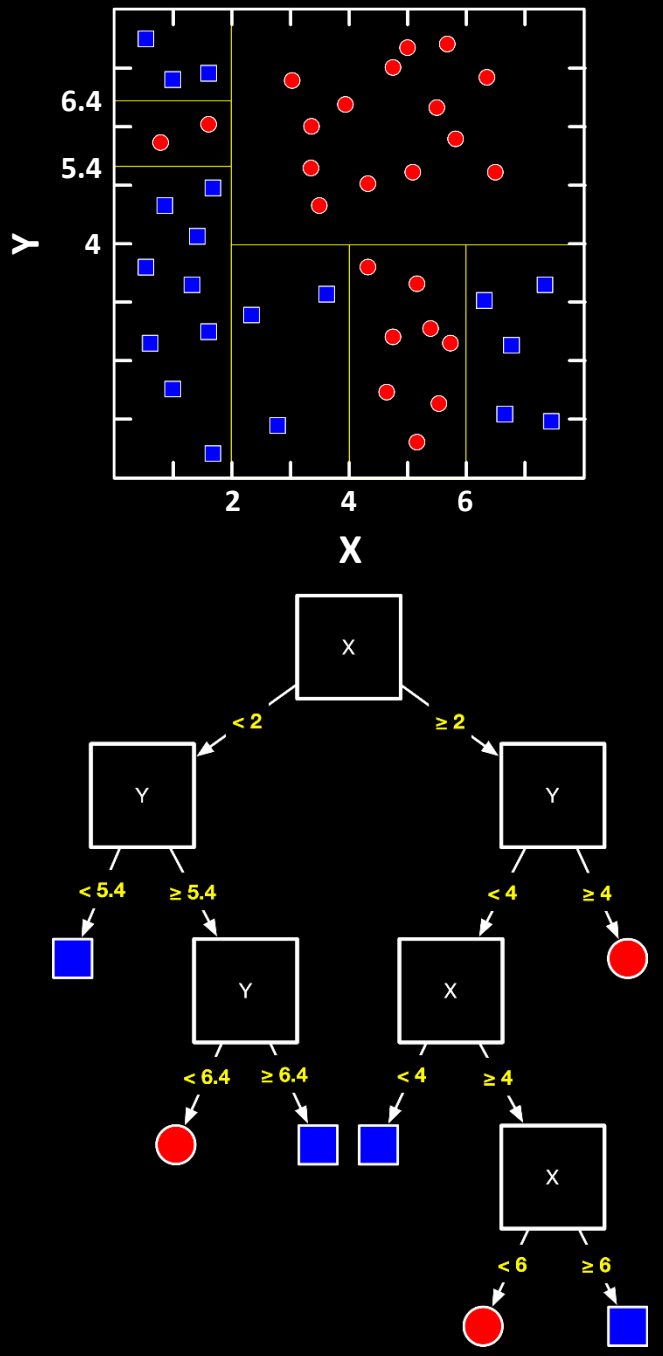


Own_house = **TRUE** → CLASS = **YES**
Own_house = **FALSE** ∧ Has_job = **TRUE** → CLASS = **YES**
Own_house = **FALSE** ∧ Has_job = **FALSE** → CLASS = **NO**

One can handle continuous values attributes by splitting into two (or more) intervals at each node:

How to find the best threshold to divide?

- Use information gain or gain ratio again
- Sort all the values of a continuous attribute in increasing order $\{v_1, v_2, \ldots, v_r\}$
- Select one threshold between two adjacent values v_i and v_{i+1}. Try all possible thresholds and find the one that maximizes the gain (or gain ratio)



A tree may **overfit** the training data when:

- The accuracy is very good on training data but poor on test data
- The tree too deep and too many branches, some may reflect anomalies due to noise or outliers

Two approaches to avoid overfitting:

- **Pre-pruning**: Halt tree construction early. It is difficult to decide because we do not know what may happen subsequently if we keep growing the tree

- **Post-pruning**: Remove branches or sub-trees from a fully formed tree. Statistical methods can be used to estimates the errors at each node for pruning

Random forest classification is a machine learning technique that:

- Randomly divides the examples in groups
- Generates a tree for each group of examples
- Classifies incoming data seeking an answer from all the trees
- Selects the class of belonging of the example via a majority vote

GOOD

- It is one of the most widely used techniques for classification
- The classification accuracy is competitive with other methods
- It is very efficient

BAD

- Decision trees are not unique
- Finding the best tree is NP-hard
- All current tree building algorithms are heuristic algorithm

READINGS

Chapters 18.1 through 18.7 and 19

QUESTIONS ?

Tufts
UNIVERSITY

ARTIFICIAL INTELLIGENCE
COMP 131

FABRIZIO SANTINI

VERSION 3.1