Tufts
UNIVERSITY

LEARNING FROM EXAMPLES

Fabrizio Santini | COMP 131A

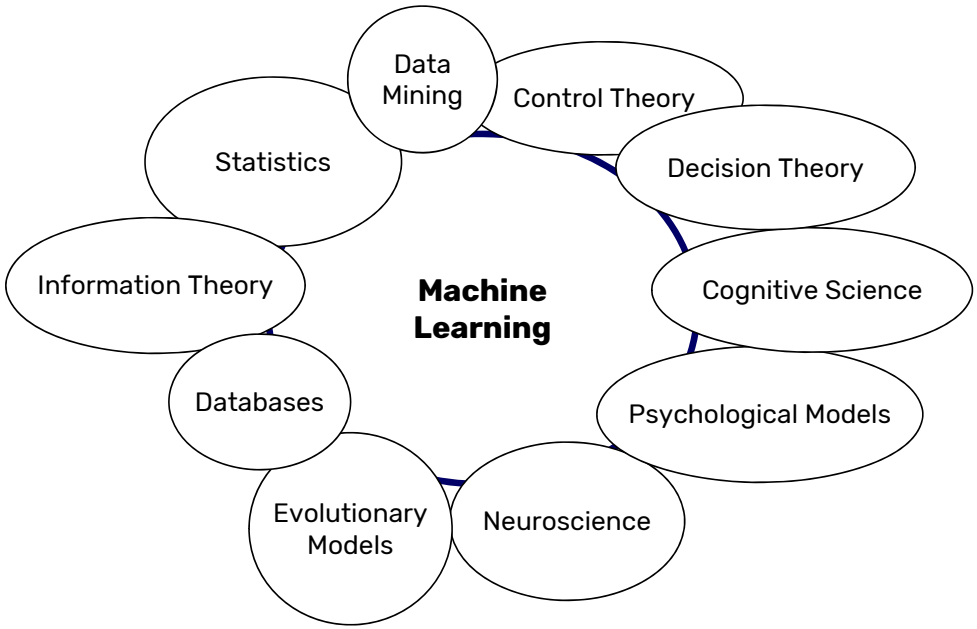
VERSION 1.1

TODAY ON AI

- ML paradigms
- Data & discriminate functions
- Supervised learning
- Decision trees
- Questions?

Why Machine Learning?

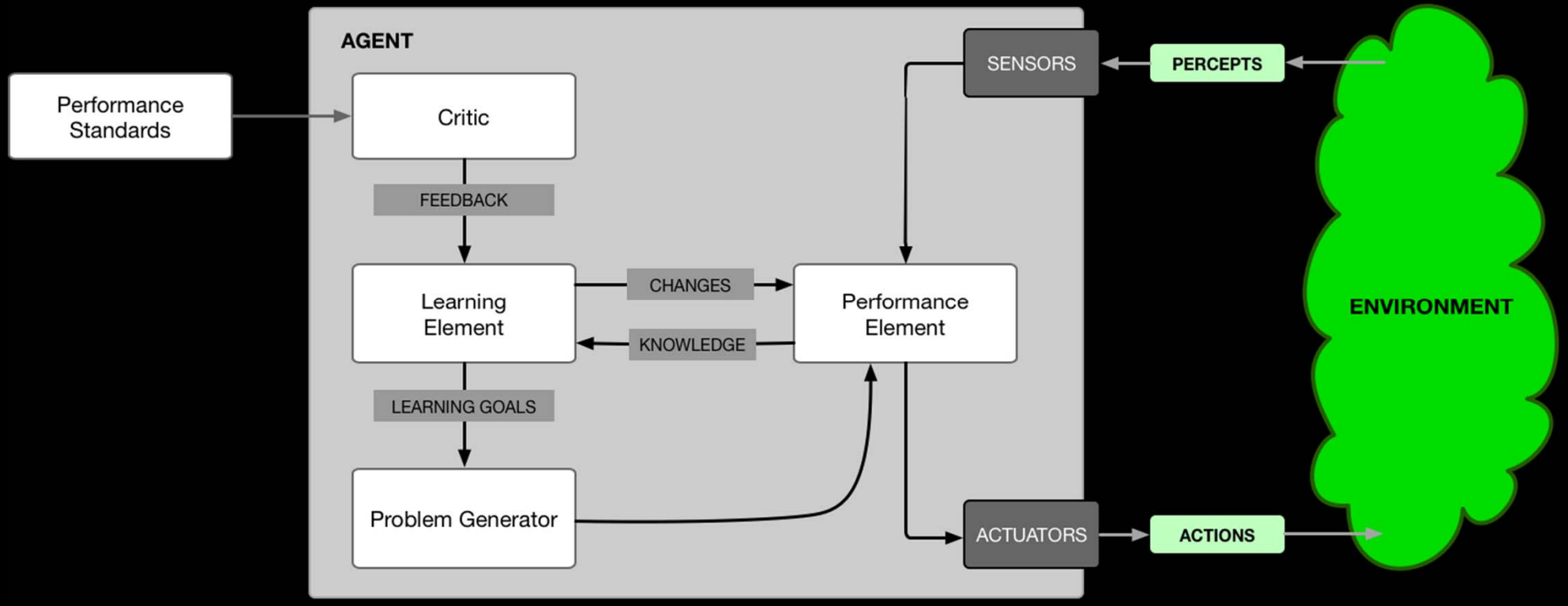
- No human experts
  - Industrial and manufacturing control
  - Mass spectrometer analysis, drug design, astronomic discovery
- Black-box human expertise
  - Face, handwriting, and speech recognition
  - Driving a car, flying a plane
- Rapidly changing phenomena
  - Credit scoring and financial modeling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



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

Machine learning is to learn a target function that can be used to predict the values of a discrete class attribute, e.g., approve or not-approved, and high-risk or low risk.

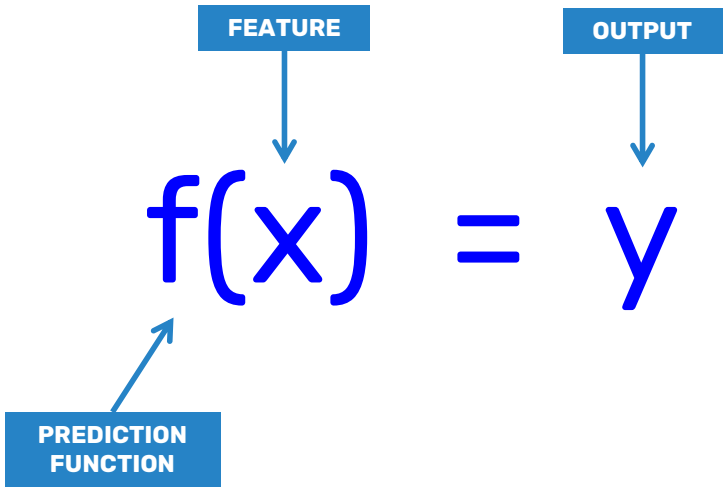
Machine learning is about predicting the future based on the past.



ML paradigms

SECTION 01

The concept is to apply a **prediction function** to a feature (or set of features) to predict a class of belonging:



f() = **banana**

f() = **tomato**

f() = **apple**

- Type of feedback:
  - **supervised** (labeled examples)
  - **unsupervised** (unlabeled examples)
  - **reinforcement** (reward)
- Representation:
  - **attribute-based** (feature vector)
  - **relational** (first-order logic)
- Use of knowledge
  - **empirical** (knowledge-free)
  - **analytical** (knowledge-guided)

- 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


banana


banana


apple


apple

Regression: label is real-valued


6.0


2.5


7.1


4.2

Ranking: label is a ranking


2


1


4


3

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

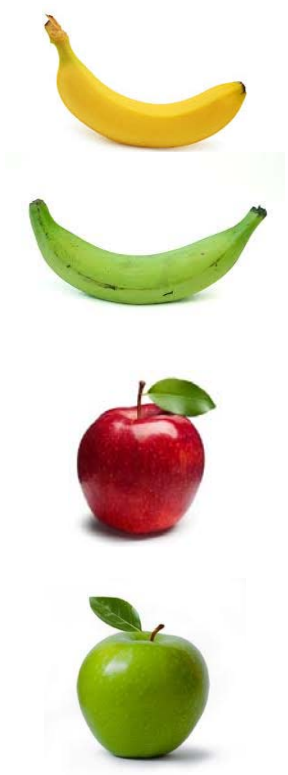
Regression

- 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

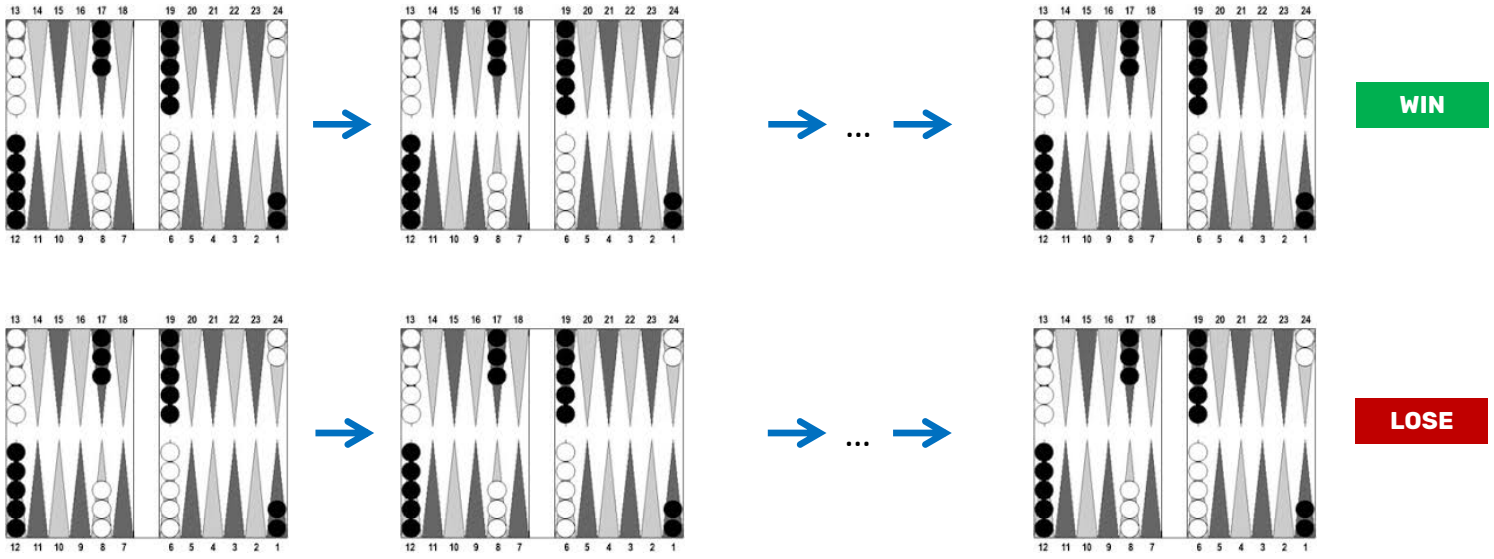
f() = **cluster 2**

PREDICTED CLUSTER

- Customer segmentation/grouping
- Image compression
- Bioinformatics: learn motifs
- Dimensionality reduction

Reinforcement learning learns what action to take in a **sequence** of **examples or states** to maximize a **reward** after completing that sequence.

BACKGAMMON

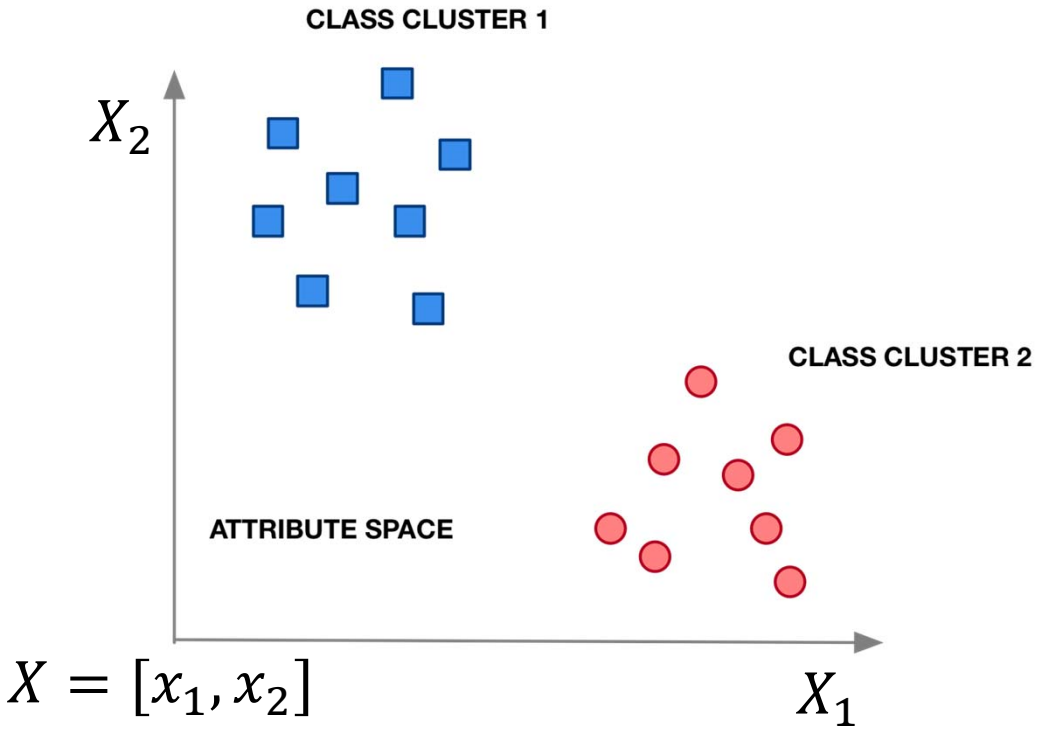


DATA & DISCRIMINATE FUNCTIONS

SECTION 02

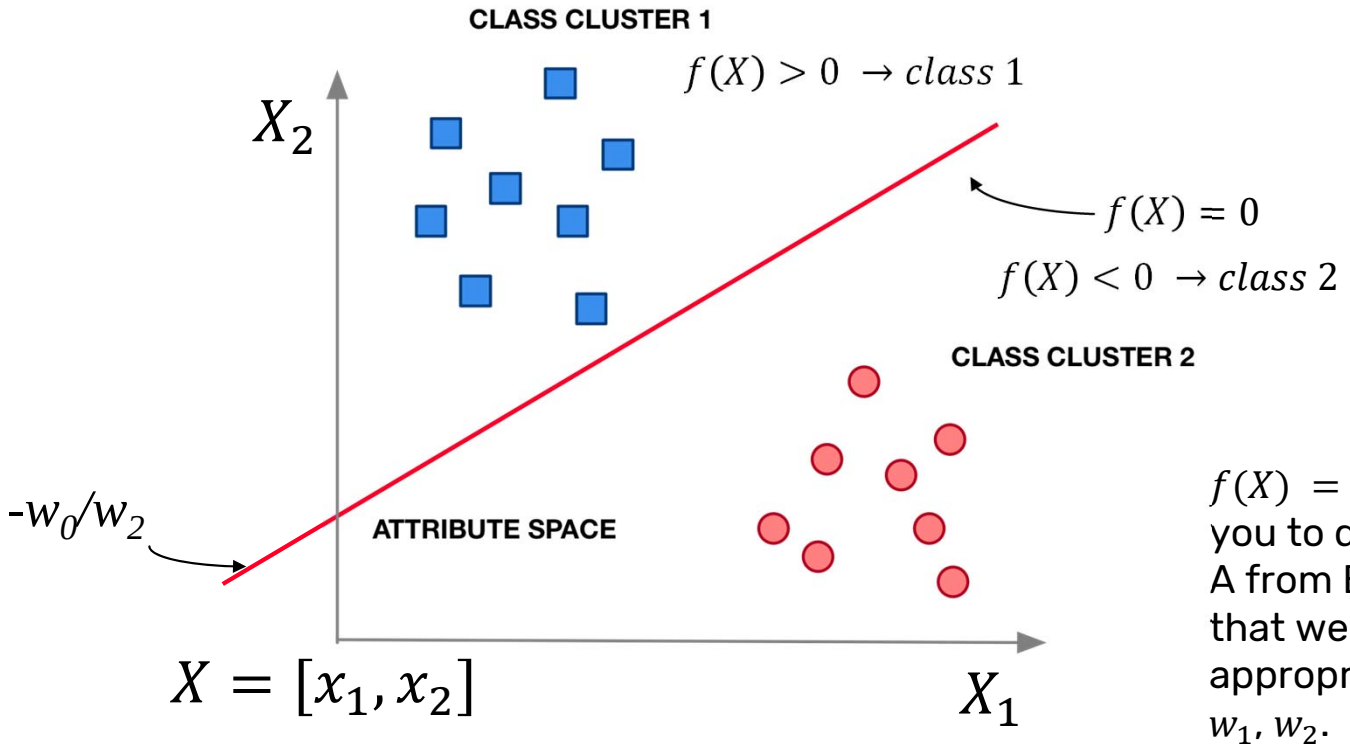
Data: A set of data records (also called examples, instances or cases) described by:

- k attributes: $A_1, A_2, \ldots Ak$
- a $class$: each example is labelled with a pre-defined class.



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

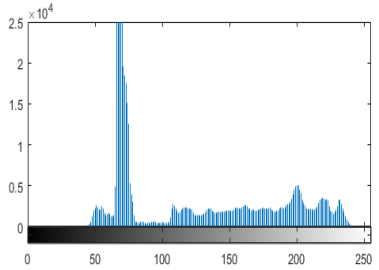
$$f(X) = f(x_1, x_2) = w_0 + w_1 x_1 + w_2 x_2$$

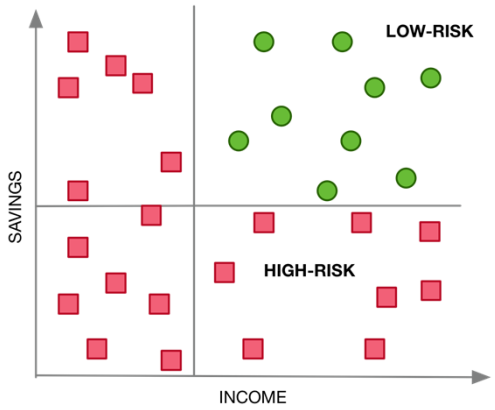


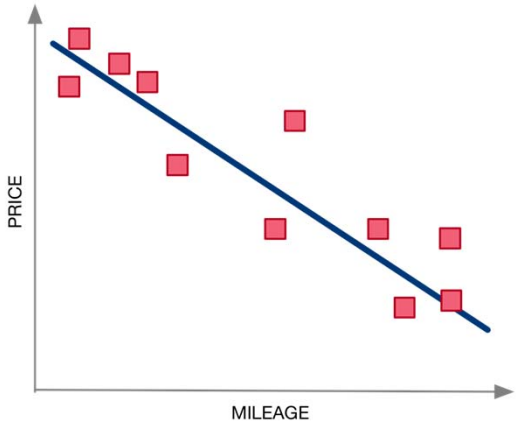
$f(X) = WX = 0$ will allow you to discriminate class A from B. The problem is that we do not know the appropriate values for w_0, w_1, w_2.

Features can be anything:

- Raw pixels
- Level of savings
- Histograms
- Income
- Price
- Descriptors







Supervised learning

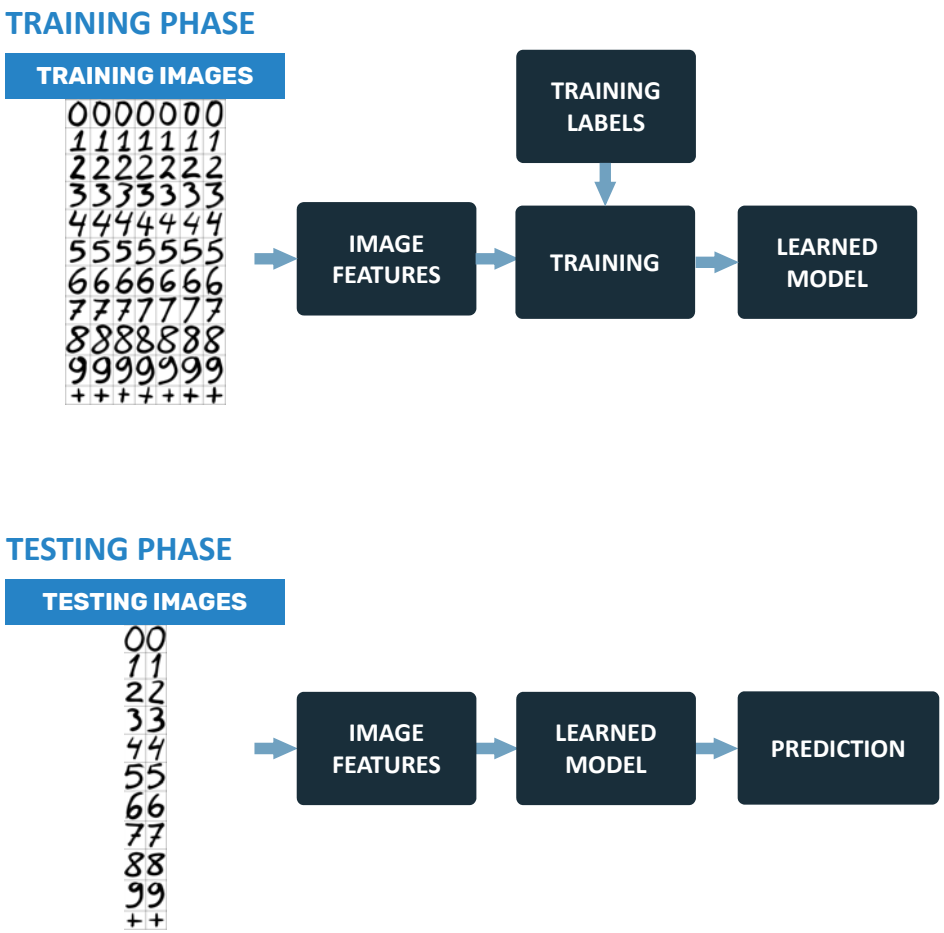
SECTION 03

- **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.



- **Training set**: It is a set of labeled examples $\{(\boldsymbol{x}1, y1), \cdots, (\boldsymbol{x}N, yN)\}$, used in the training phase

- **Testing set**: It is a set of labeled examples $\{(\boldsymbol{x}1, y1), \cdots, (\boldsymbol{x}N, yN)\}$, never used in the training phase

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

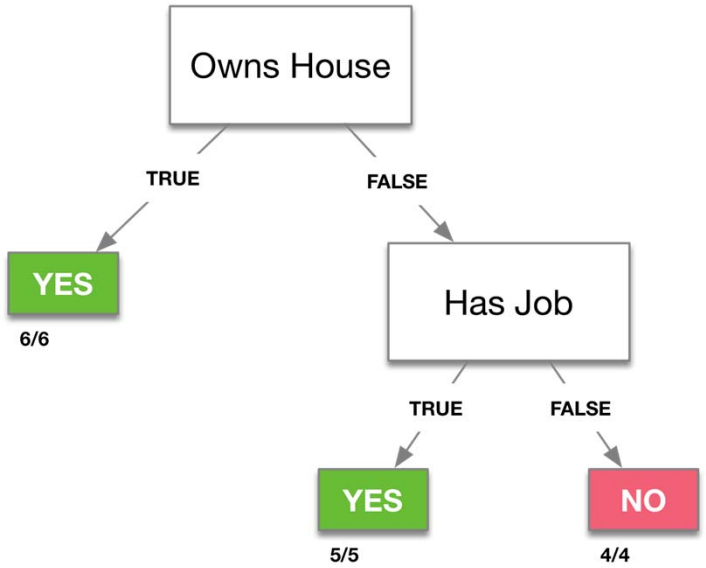
In other words, the **learned model** helps the system to perform T **better** as compared to no learning.

Decision trees

SECTION 04

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**



Decision tree learning is one of the most widely used techniques for classification. The **classification model** is a **tree**, called **decision tree**:

- The classification accuracy is competitive with other methods
- It is very efficient

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

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

A greedy divide-and-conquer algorithm:

- At start, all the training examples are at the root
- Examples are partitioned recursively based on selected attributes
- Attributes are selected on the basis of an information gain

Conditions for stopping:

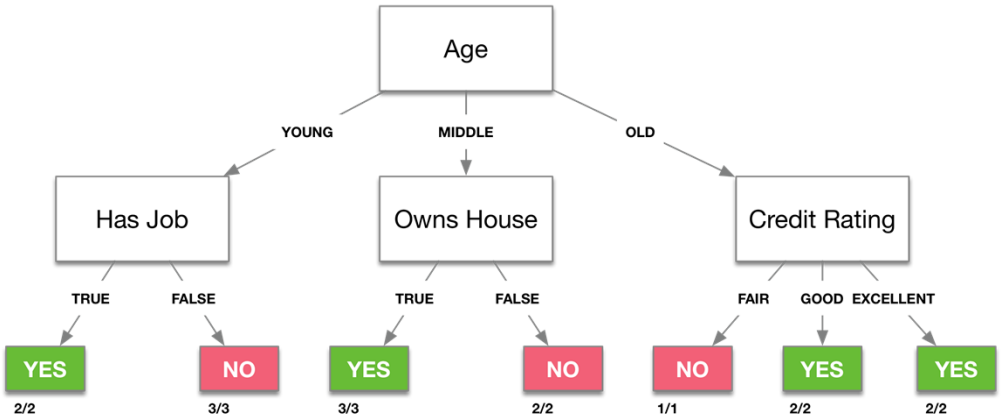
- All examples for a given node belong to the same class
- There are no remaining attributes for further partitioning – majority class is the leaf
- There are no examples left

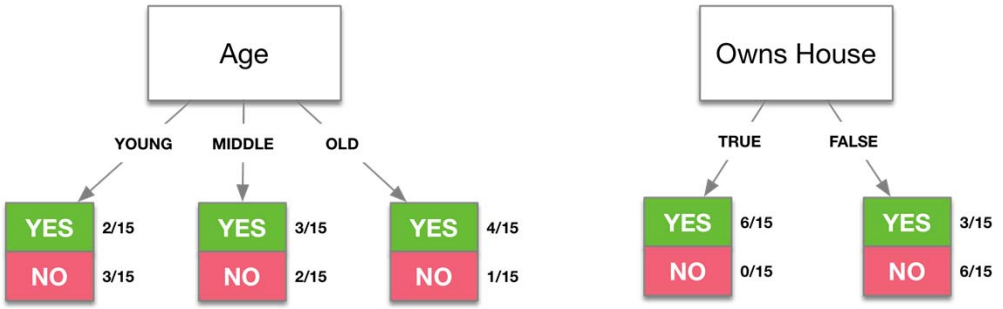
```
1  function DT-learning(EXAMPLES, ATTRIBUTES, P-EXAMPLES)
2    return TREE
3    if EXAMPLES is empty then
4      return PLURALITY-VALUE(P-EXAMPLES)
5    else if all EXAMPLES have the same classification then
6      return the classification
7    else if ATTRIBUTES is empty then
8      return PLURALITY-VALUE(EXAMPLES)
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
```

In order to branch the tree, we choose the attribute according to the principal of **information gain**:

- **Entropy**: impurity of a set of examples ($E = 0$ for a perfectly homogeneous)
- **Information gain**: expected reduction in entropy caused by partitioning

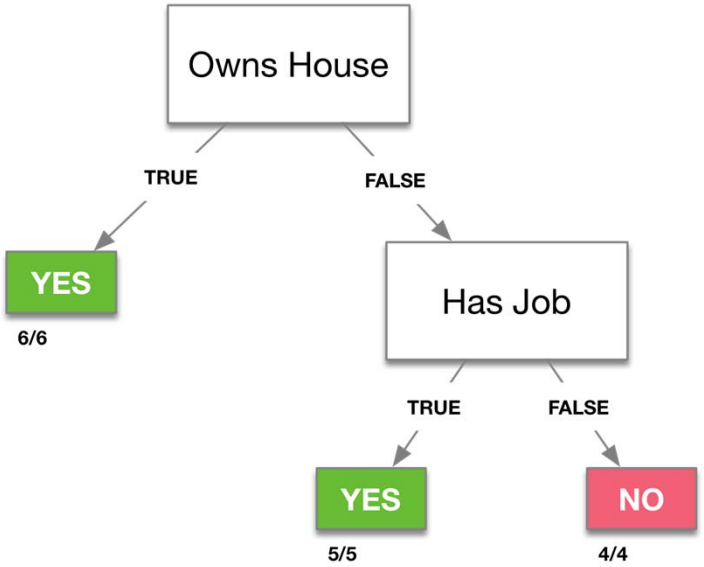
Intuitively a **good attribute** splits the examples into subsets that are (ideally) all positive or all negative.





- A decision tree can be **converted** to a set of 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**



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
- One bit of information is enough to answer a yes/no question about which one has no idea

- 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 ($P(negative) = 0.8$):

$$E(D) = -0.2 \times \log_2 0.2 - 0.8 \times \log_2 0.8 = 0.722$$

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

$$E(D) = -1.0 \times \log_2 1.0 - 0 \times \log_2 0 = 1$$

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

- 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**

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

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

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

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

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

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

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

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

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

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

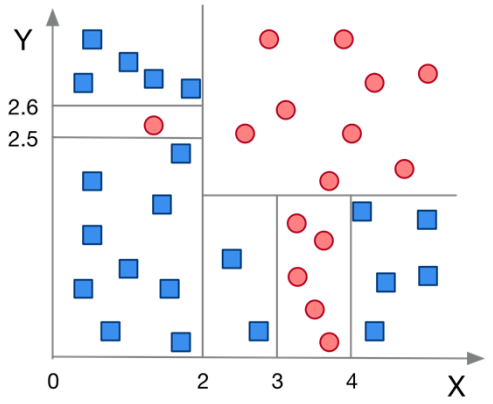
$$gain(D, Age) = 0.971 - 0.888 = 0.083$$

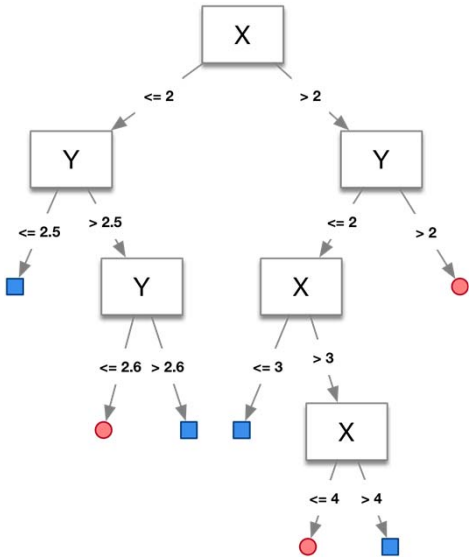
$$\mathbf{gain(D, Owns\_house) = 0.420}$$

$$gain(D, Has\_job) = 0.324$$

$$gain(D, Credit\_rating) = 0.363$$

- 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 an 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)





- **Overfitting**: A tree may overfit the training data
  - 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

QUESTIONS ?