INTRODUCTION TO MACHINE LEARNING

SHAHABEDIN NABAVI

S_nabavi@sbu.ac.ir

Faculty of Computer Science and Engineering
Shahid Beheshti University

WHAT IS LEARNING?

"The activity or process of gaining knowledge or skill by studying, practicing, being taught, or experiencing something."

Merriam Webster dictionary

"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."

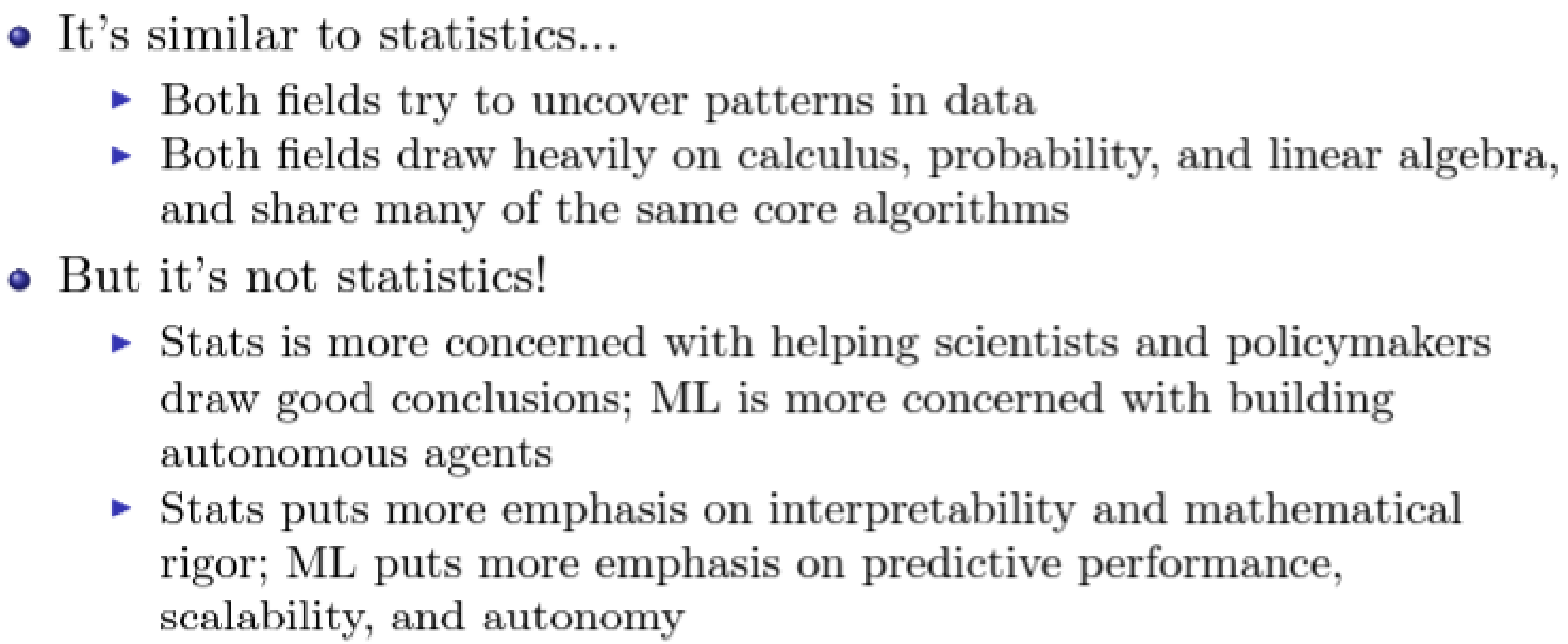
Tom Mitchell

WHAT IS MACHINE LEARNING?

- Machine learning is a subfield of artificial intelligence, which is broadly defined as **the capability of a machine to imitate intelligent human behavior**.

- Artificial intelligence systems are used to perform complex tasks in a way that is similar to how humans solve problems.

MACHINE LEARNING VS STATISTICS



- It's similar to statistics...
  - Both fields try to uncover patterns in data
  - Both fields draw heavily on calculus, probability, and linear algebra, and share many of the same core algorithms
- But it's not statistics!
  - Stats is more concerned with helping scientists and policymakers draw good conclusions; ML is more concerned with building autonomous agents
  - Stats puts more emphasis on interpretability and mathematical rigor; ML puts more emphasis on predictive performance, scalability, and autonomy

RELATIONS TO HUMAN LEARNING

- Human learning is:
  - Very data efficient
  - An entire multitasking system (vision, language, motor control, etc.)
  - Takes at least a few years :)

- For serving specific purposes, machine learning doesn't have to look like human learning in the end.

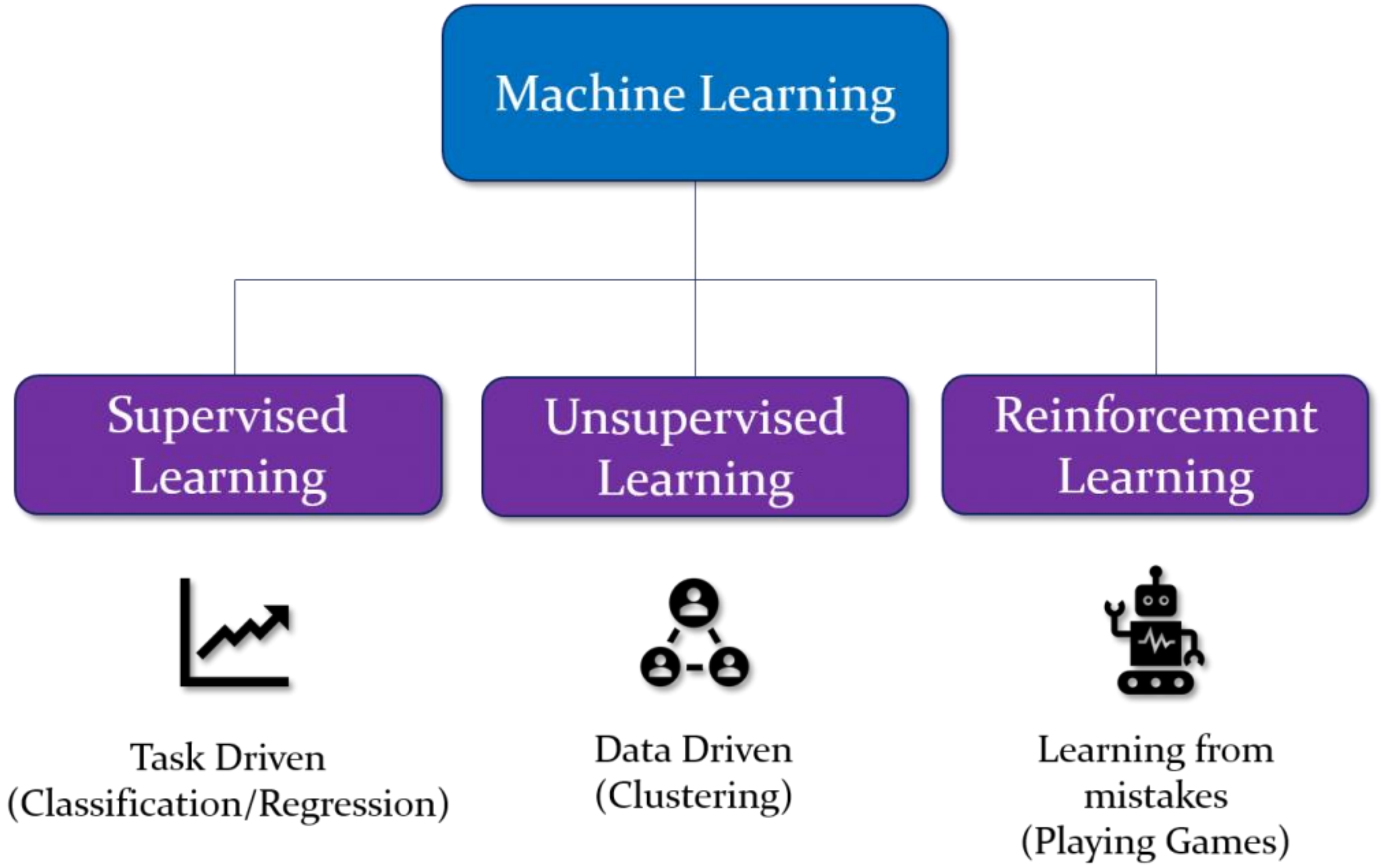
- It may borrow ideas from biological systems, e.g., neural networks.

- It may perform better or worse than humans.

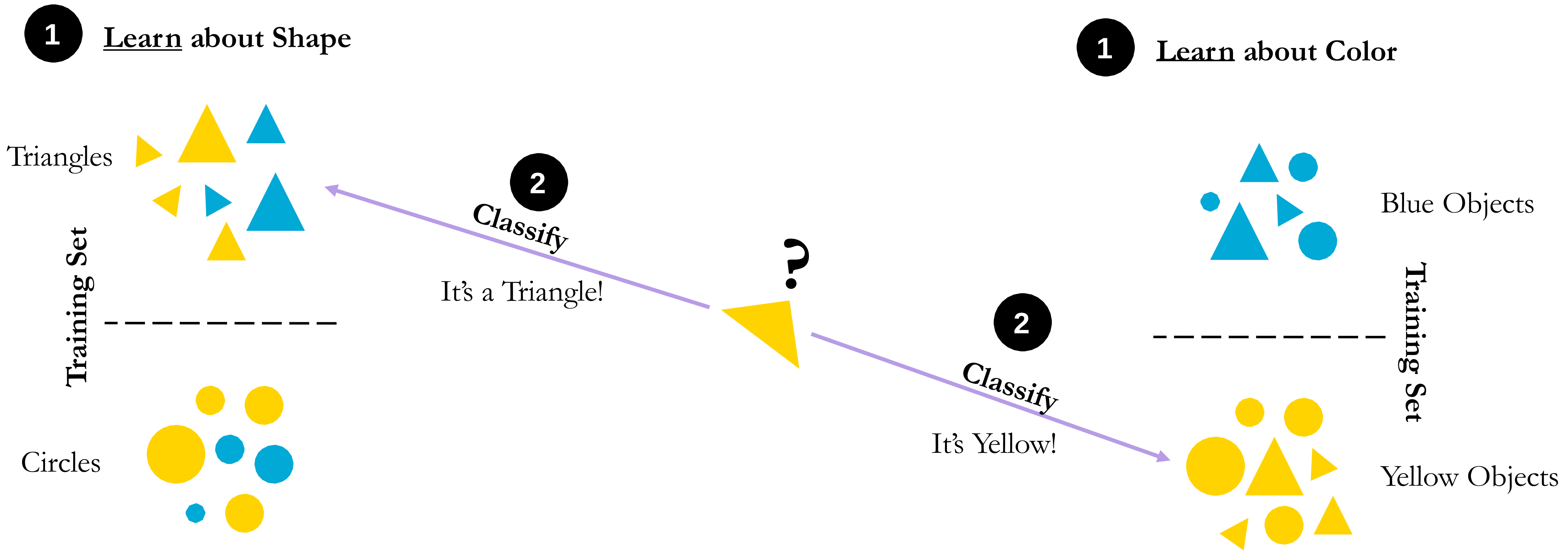
TYPES OF ML

- Types of machine learning
  - **Supervised learning:** have labeled examples of the correct behavior
  - **Reinforcement learning:** learning system (agent) interacts with the world and learns to maximize a scalar reward signal
  - **Unsupervised learning:** no labeled examples – instead, looking for "interesting" patterns in the data

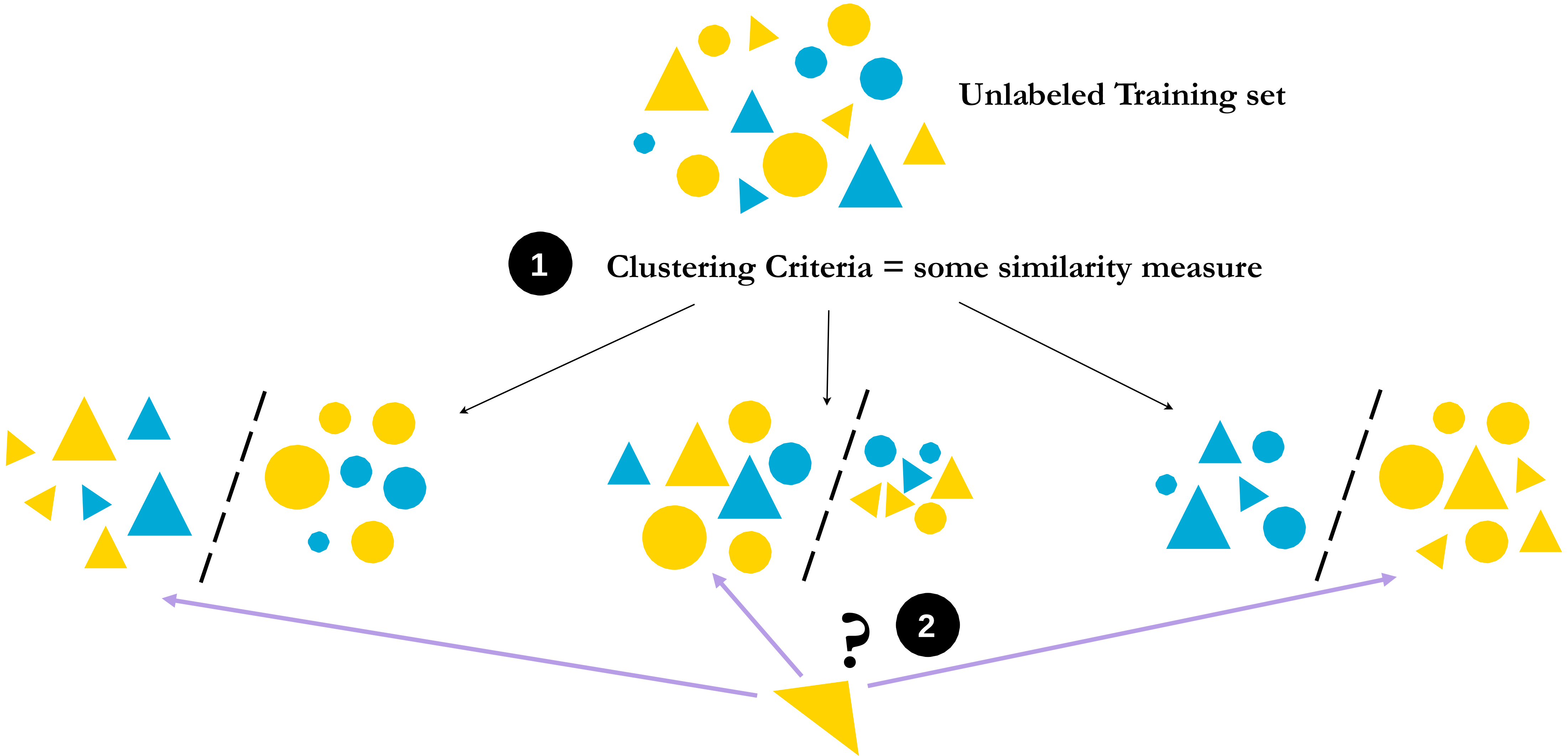
TYPES OF ML


Machine Learning
Supervised Learning
Unsupervised Learning
Reinforcement Learning
Task Driven (Classification/Regression)
Data Driven (Clustering)
Learning from mistakes (Playing Games)

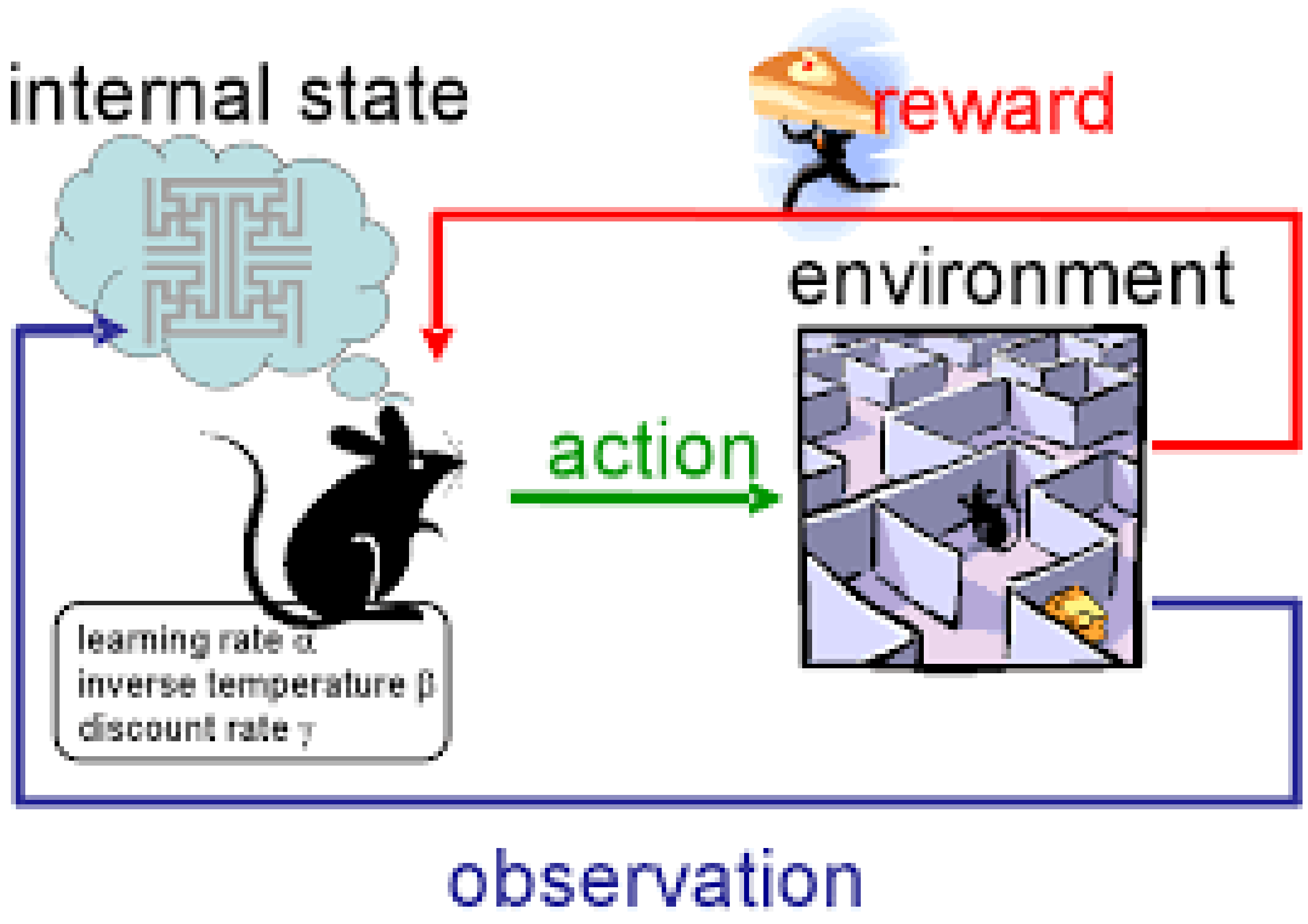
SUPERVISED LEARNING



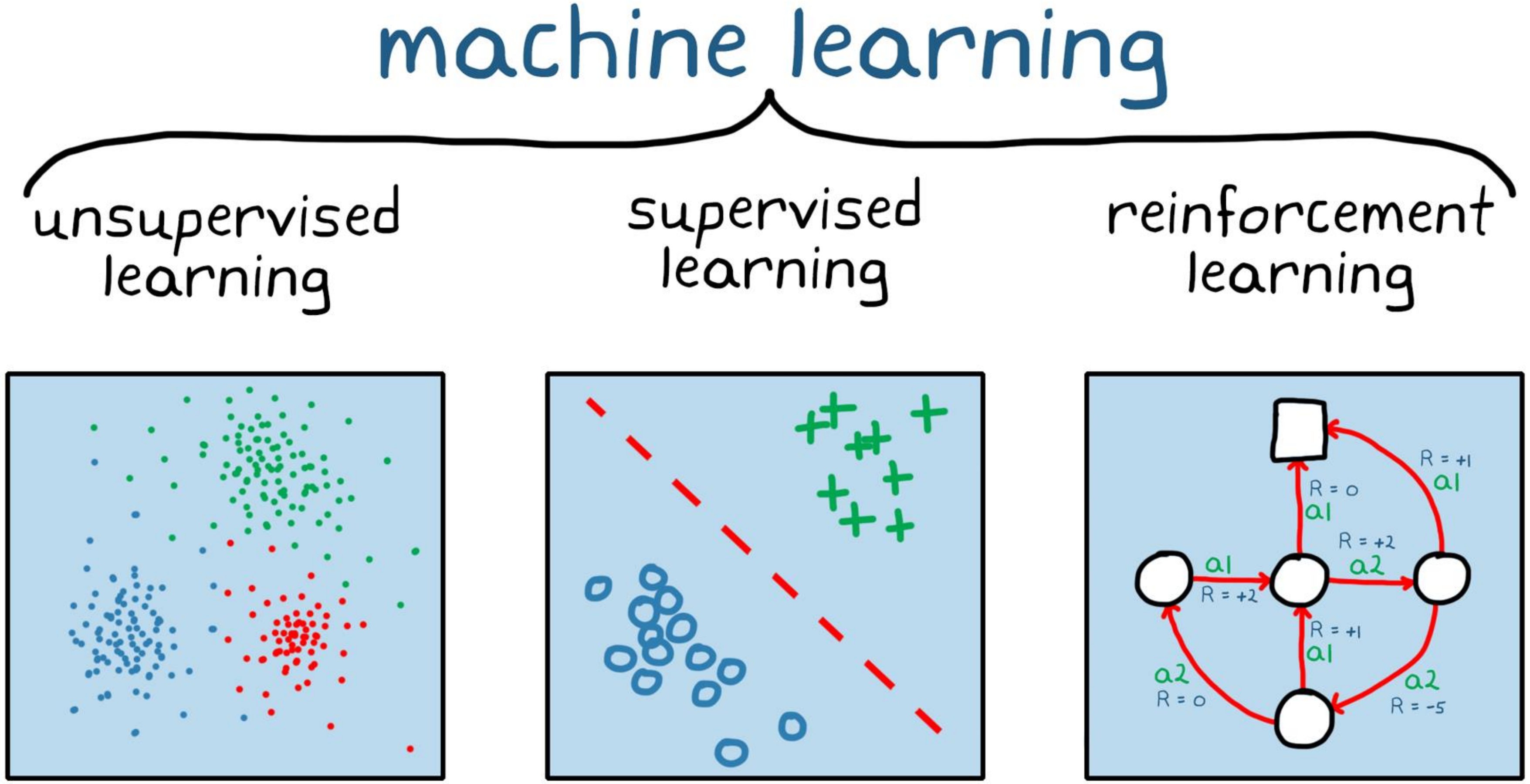
UNSUPERVISED LEARNING



REINFORCEMENT LEARNING



SO…


machine learning
unsupervised learning
supervised learning
reinforcement learning
R = +1
a1
R = 0
a1
R = +2
a2
a1
R = +2
R = +1
a1
a2
R = 0
a2
R = -5

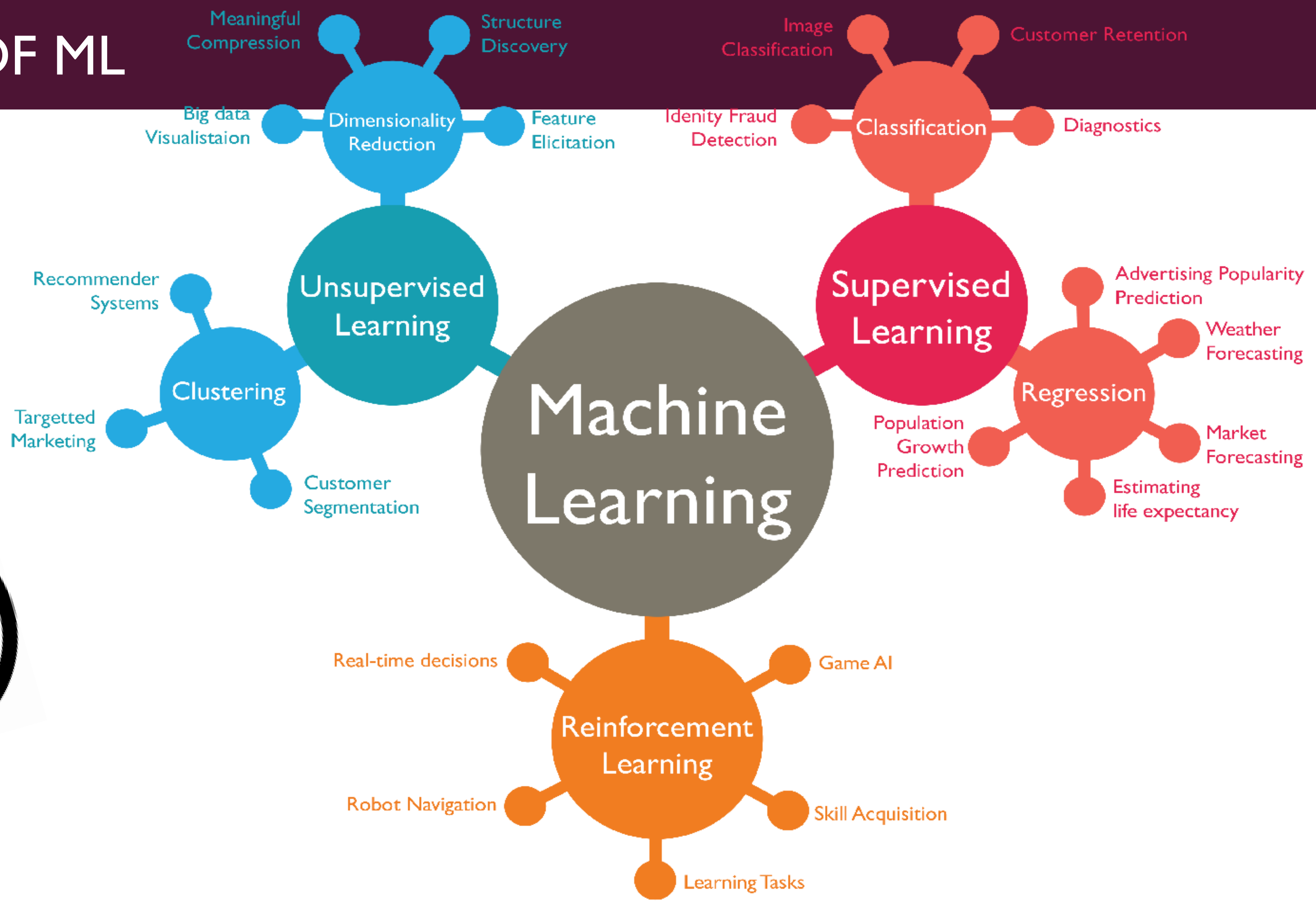
HISTORY OF ML

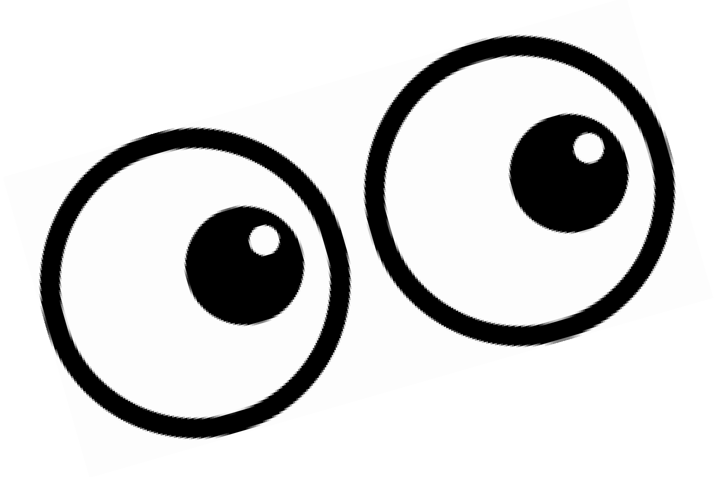
- 1957 — Perceptron algorithm (implemented as a circuit!)
- 1959 — Arthur Samuel wrote a learning-based checkers program that could defeat him
- 1969 — Minsky and Papert's book *Perceptrons* (limitations of linear models)
- 1980s — Some foundational ideas
  - Connectionist psychologists explored neural models of cognition
  - 1984 — Leslie Valiant formalized the problem of learning as PAC learning
  - 1988 — Backpropagation (re-)discovered by Geoffrey Hinton and colleagues
  - 1988 — Judea Pearl's book *Probabilistic Reasoning in Intelligent Systems* introduced Bayesian networks

HISTORY OF ML

- 1990s — the “AI Winter”, a time of pessimism and low funding
- But looking back, the ’90s were also sort of a golden age for ML research
  - Markov chain Monte Carlo
  - variational inference
  - kernels and support vector machines
  - boosting
  - convolutional networks
  - reinforcement learning
- 2000s — applied AI fields (vision, NLP, etc.) adopted ML
- 2010s — deep learning
  - 2010–2012 — neural nets smashed previous records in speech-to-text and object recognition
  - increasing adoption by the tech industry
  - 2016 — AlphaGo defeated the human Go champion
  - 2018-now — generating photorealistic images and videos
  - 2020 — GPT3 language model
- now — increasing attention to ethical and societal implications

APPLICATIONS OF ML





APPLICATIONS IN COMPUTER VISION

Computer vision: Object detection, semantic segmentation, pose estimation, and almost every other task is done with ML.

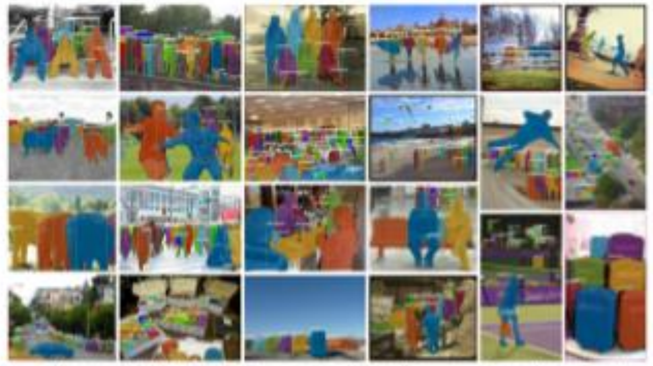
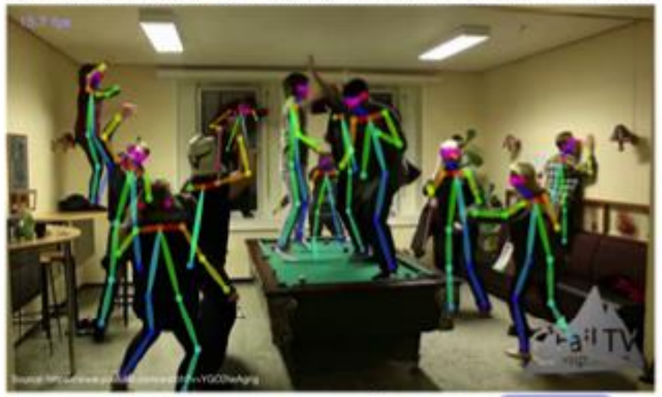
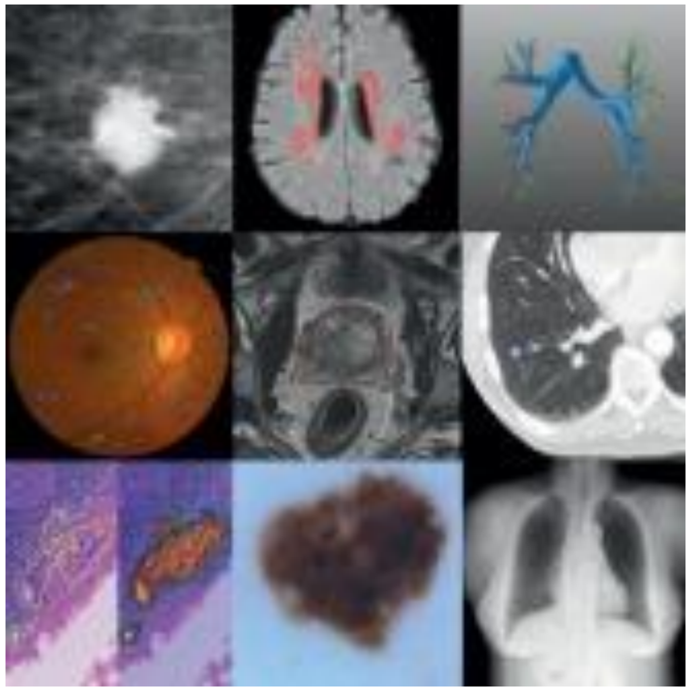


Figure 4. More results of **Mask R-CNN** on COCO test images, using ResNet-101-FPN and running at 5 fps, with 35.7 mask AP (Table 1).



DAQUAR 1553
What is there in front of the sofa?
Ground truth: table
IMG+BOW: table (0.74)
2-VIS+BLSTM: table (0.88)
LSTM: chair (0.47)

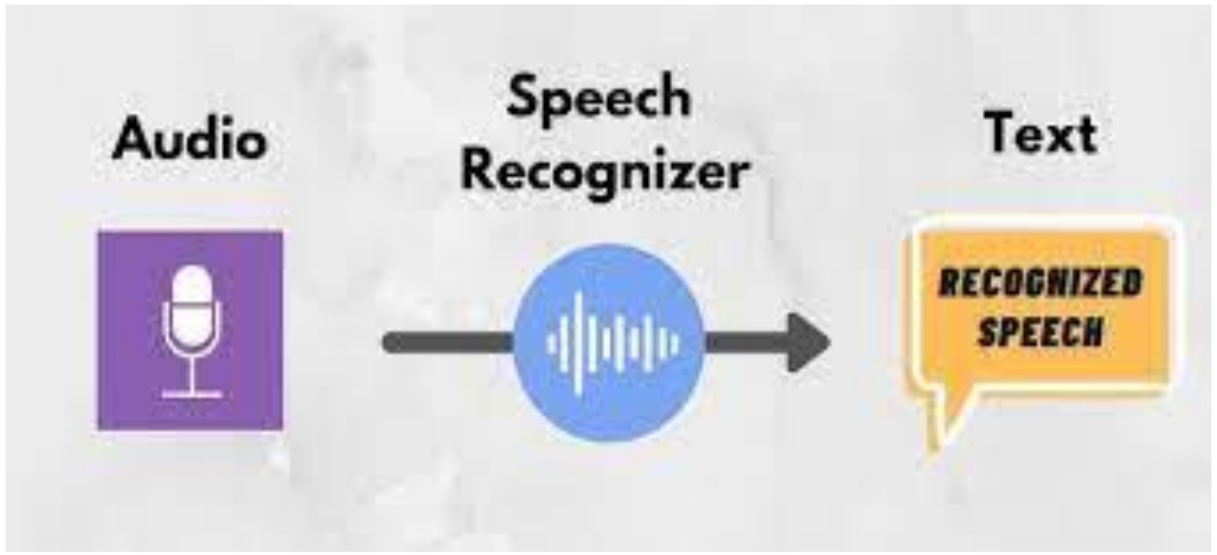
COCOQA 5078
How many leftover donuts is the red bicycle holding?
Ground truth: three
IMG+BOW: two (0.51)
2-VIS+BLSTM: three (0.27)
BOW: one (0.29)



Instance segmentation - ▸ Link

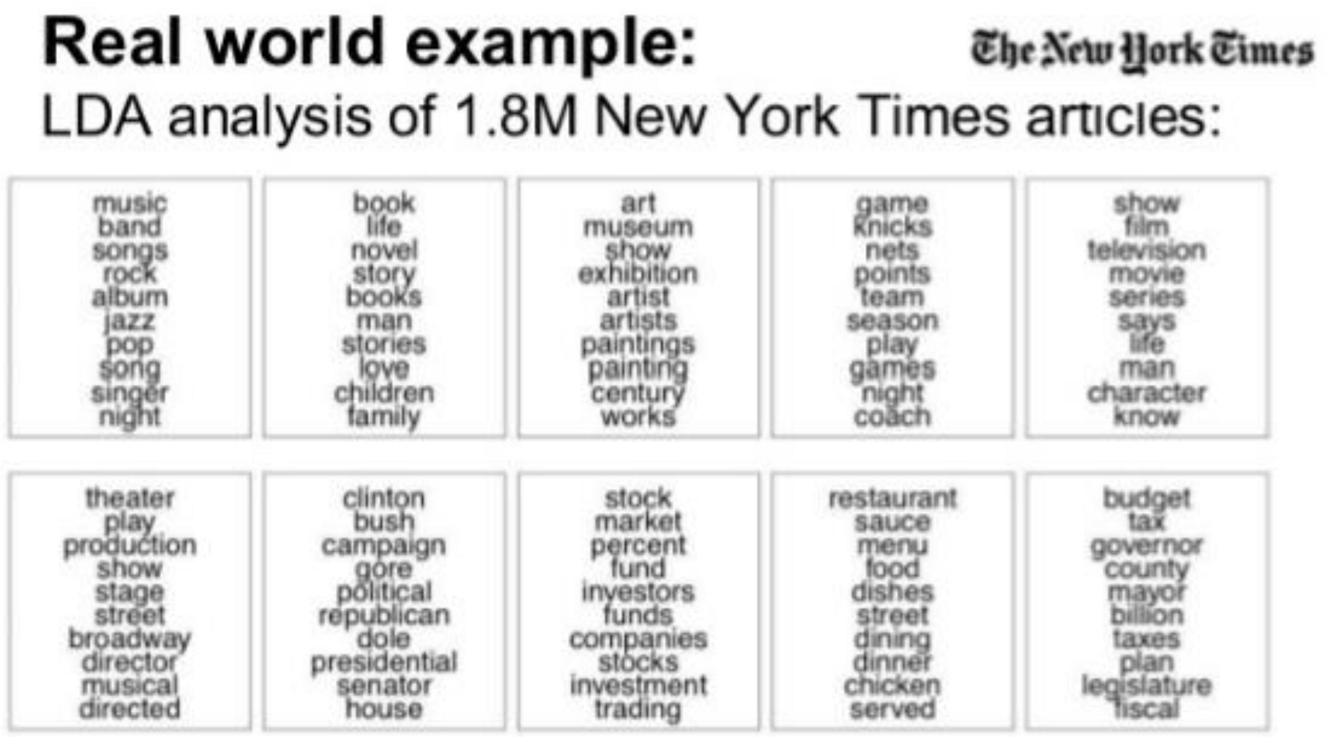
APPLICATIONS IN SPEECH RECOGNITION

Speech: Speech to text, personal assistants, speaker identification...



APPLICATIONS IN NATURAL LANGUAGE PROCESSING (NLP)

NLP: Machine translation, sentiment analysis, topic modeling, spam filtering.

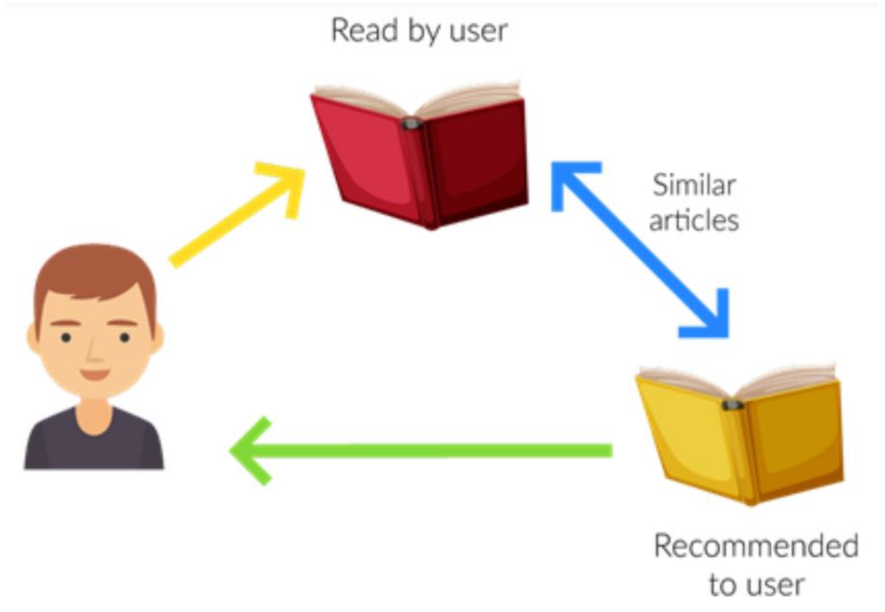


Real world example: The New York Times

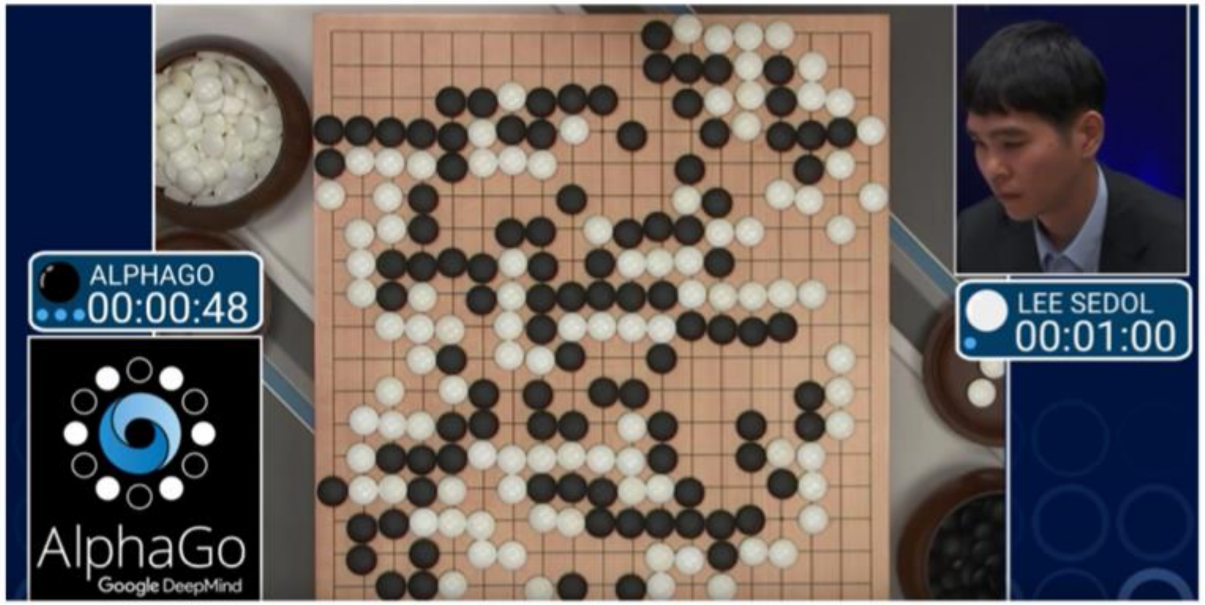
LDA analysis of 1.8M New York Times articles:

music	book	art	game	show
band	life	museum	knicks	film
songs	novel	show	nets	television
rock	story	exhibition	points	movie
album	books	artist	team	series
jazz	man	artists	season	says
pop	stories	paintings	play	life
song	love	painting	games	man
singer	children	century	night	character
night	family	works	coach	know

theater	clinton	stock	restaurant	budget
play	bush	market	sauce	tax
production	campaign	percent	menu	governor
show	gore	fund	food	county
stage	political	investors	dishes	mayor
street	republican	funds	street	billion
broadway	dole	companies	dining	taxes
director	presidential	stocks	dinner	plan
musical	senator	investment	chicken	legislature
directed	house	trading	served	fiscal



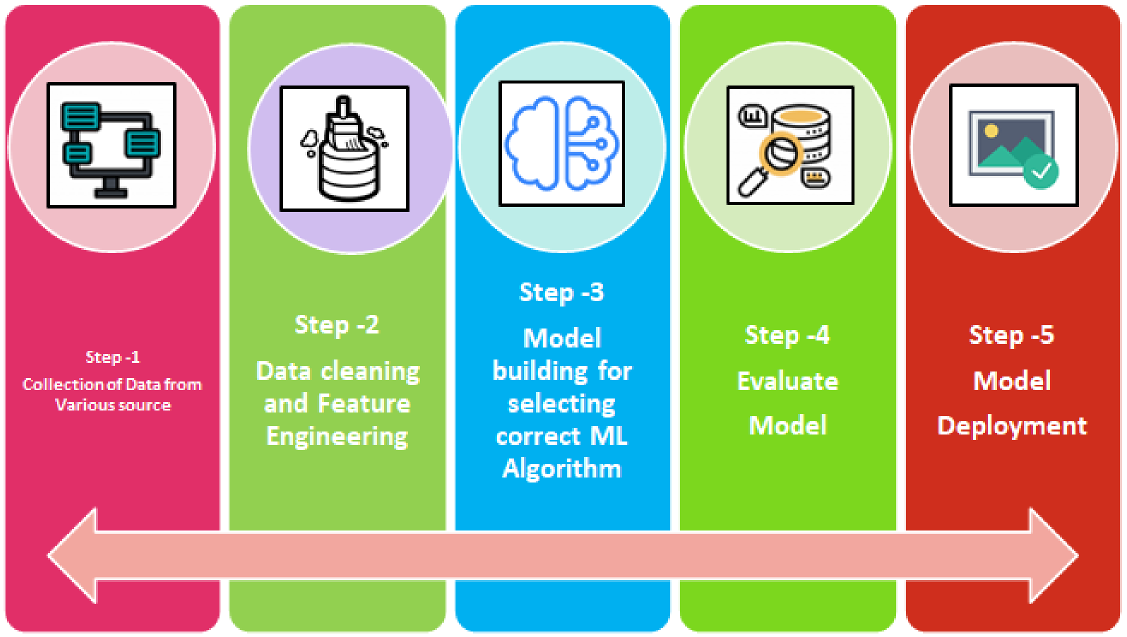
PLAYING GAMES



ML WORKFLOW

ML workflow sketch:

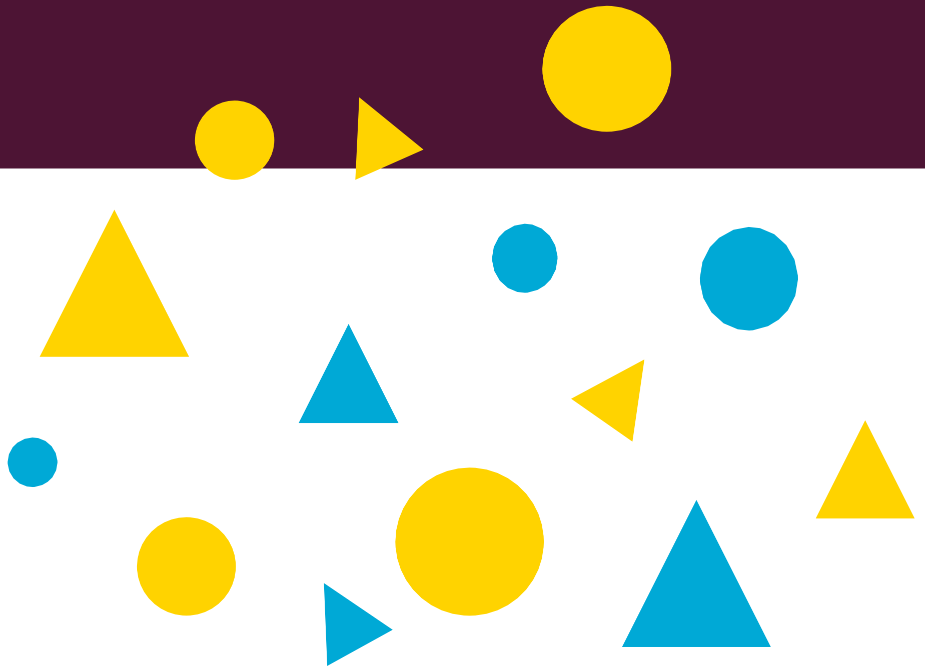
1. Should I use ML on this problem?
   - Is there a pattern to detect?
   - Can I solve it analytically?
   - Do I have data?
2. Gather and organize data.
   - Preprocessing, cleaning, visualizing.
3. Establishing a baseline.
4. Choosing a model, loss, regularization, ...
5. Optimization (could be simple, could be a Phd...).
6. Hyperparameter search.
7. Analyze performance & mistakes, and iterate back to step 4 (or 2).


Step -1
Collection of Data from Various source
Step -2
Data cleaning and Feature Engineering
Step -3
Model building for selecting correct ML Algorithm
Step -4
Evaluate Model
Step -5
Model Deployment

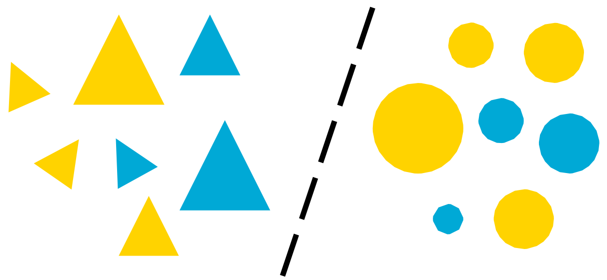
FEATURES

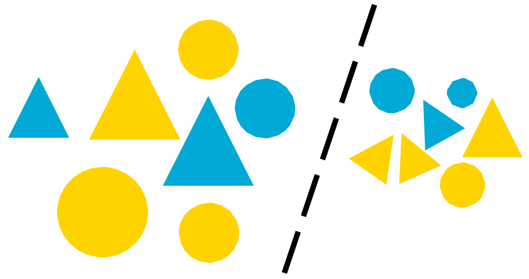
- Features are properties of an object:
  - Ideally representative of a specific type (i.e. class) of objects
  - Compact (memory efficient)
  - Computationally simple (CPU efficient)
  - Perceptual relevant (if trying to implement a human inspired classifier)
- **Should pave the way to a good discrimination of different classes of objects!**

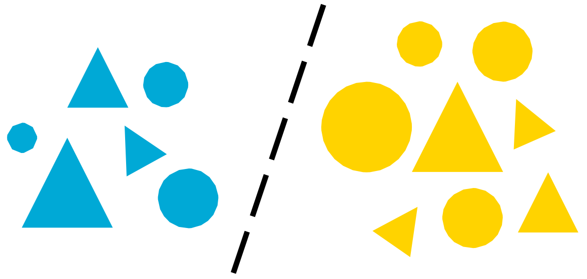
FEATURES



- Take a group of graphical objects
  - Possible features:
    - Shape
    - Color
    - Size
    - …
  - Allows to group them into different classes:

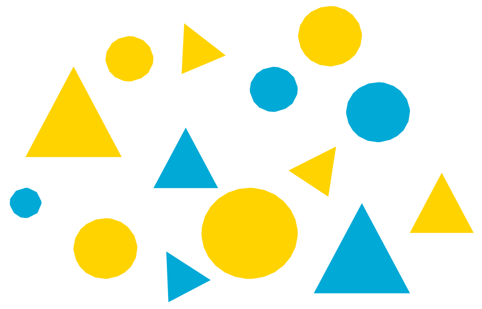






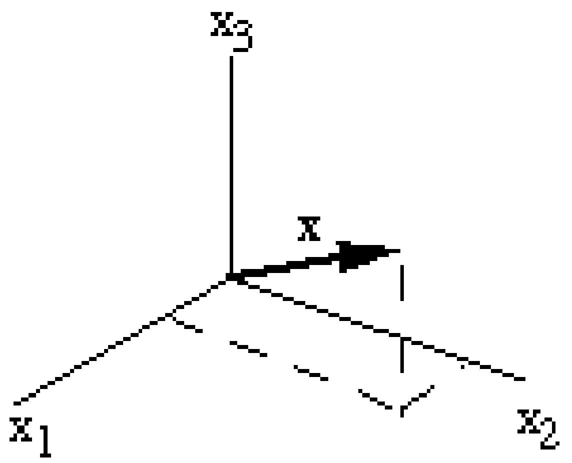
FEATURE VECTORS

- Usually a single object can be represented using several features, e.g.
  - **x1** = shape (e.g. nr of sides)
  - **x2** = size (e.g. some numeric value)
  - **x3** = color (e.g. rgb values)
  - ...
  - **xd** = some other (numeric) feature.



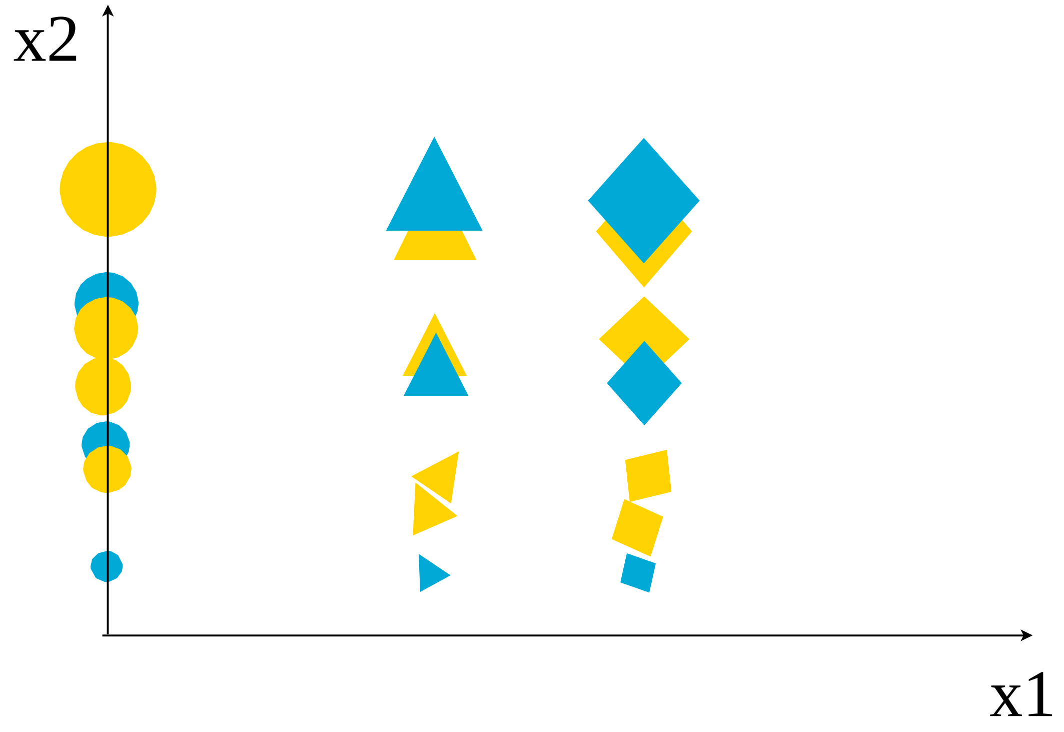
- **X** becomes a feature vector
  - x is a point in a d-dimensional **feature space.**

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ x_d \end{bmatrix}$$



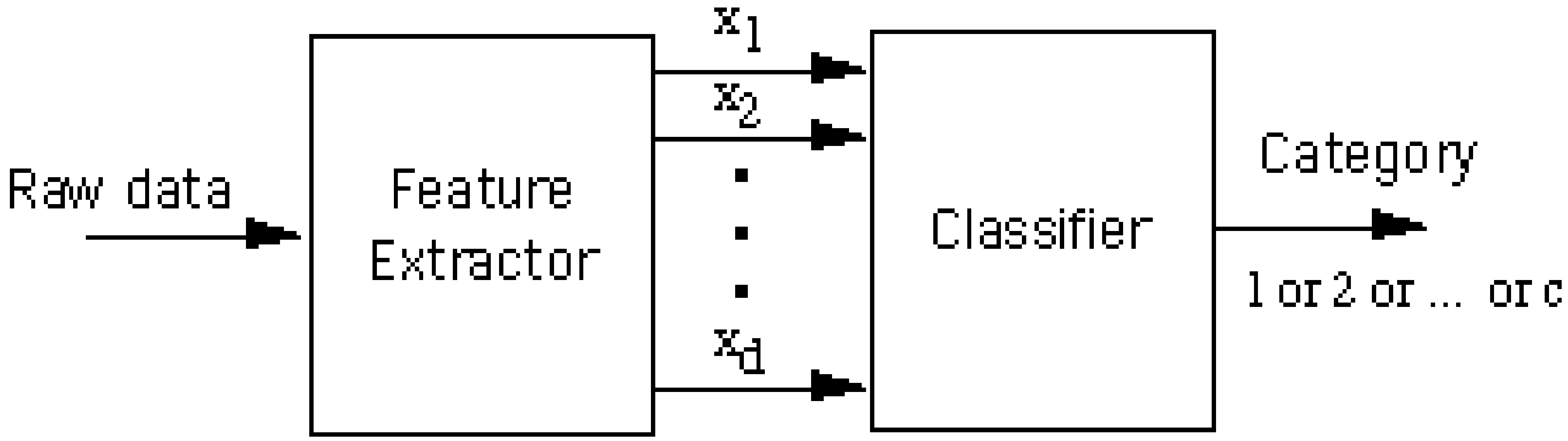
FEATURE VECTORS

- Example of a 2D Feature Space
  - **x1** = shape (e.g. nr of sides)
  - **x2** = size (e.g. some numeric value)



AN ML CLASSICAL MODEL

1. A **Feature Extractor** extracts features from raw data (e.g. audio, image, weather data, etc)
2. A **Classifier** receives X and assigns it to one of c categories, Class 1, Class 2, ..., Class c (i.e. labels the raw data).



AN ML CLASSICAL MODEL

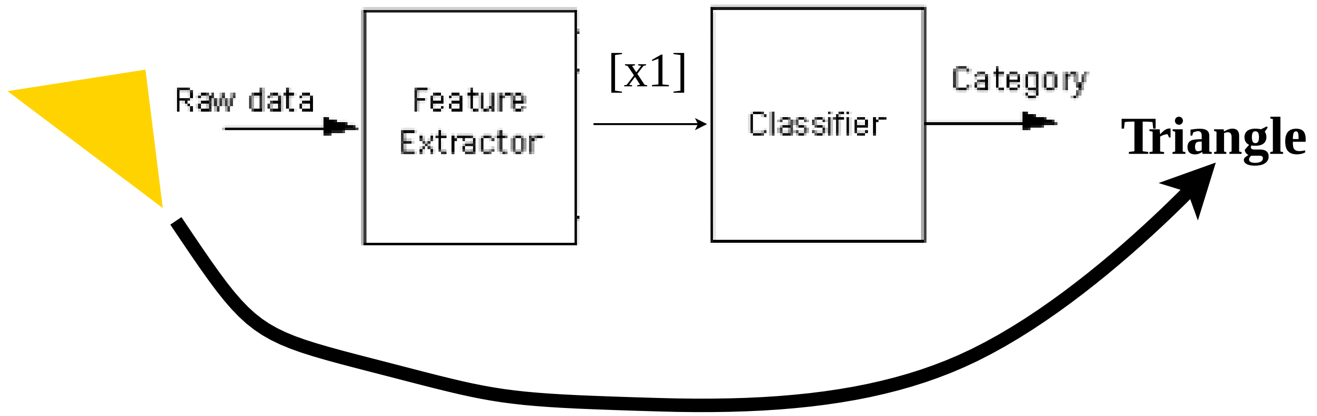
- Example: classify graphic objects according to their shape

– Feature extracted:

  - # of sides (x1)

– Classifier:

  - 0 sides => circle
  - 3 sides => triangle
  - 4 sides => rectangle



– How does the classifier know that a circle has no sides and that a triangle has 3 sides?!

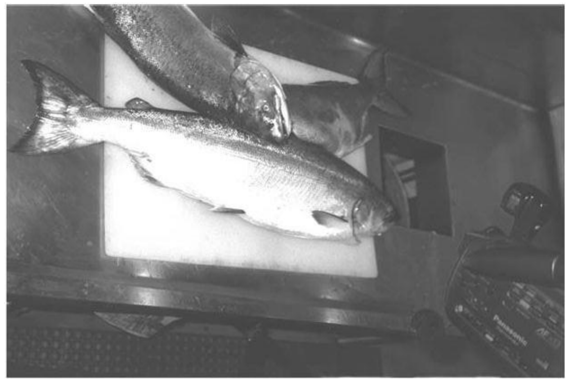
A CASE STUDY: FISH CLASSIFICATION

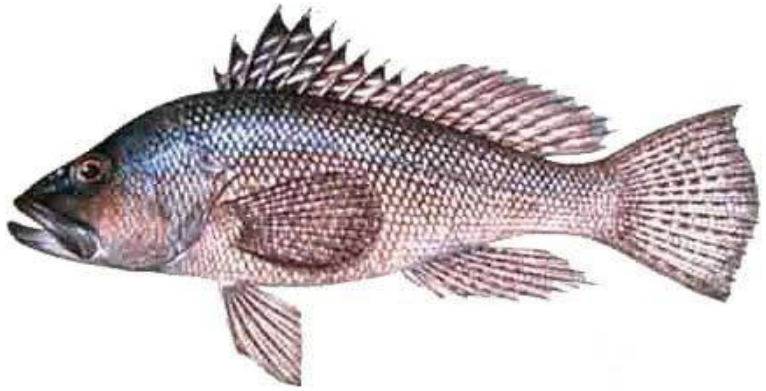
- Problem:

– sort incoming fish on a conveyor belt according to species

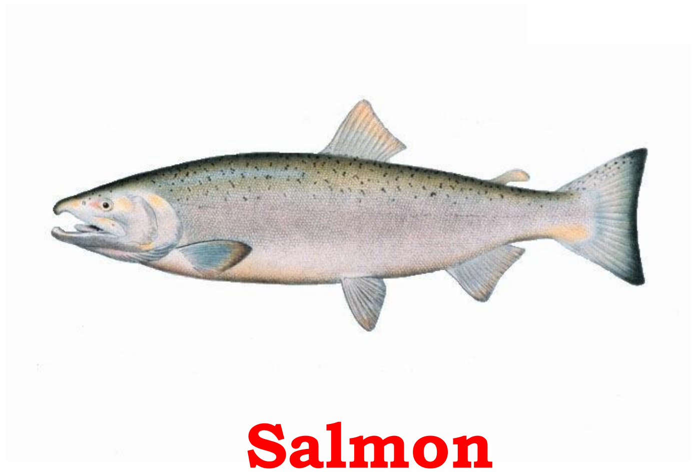
– Assume only two classes exist:

  - Sea Bass and Salmon





Sea-bass

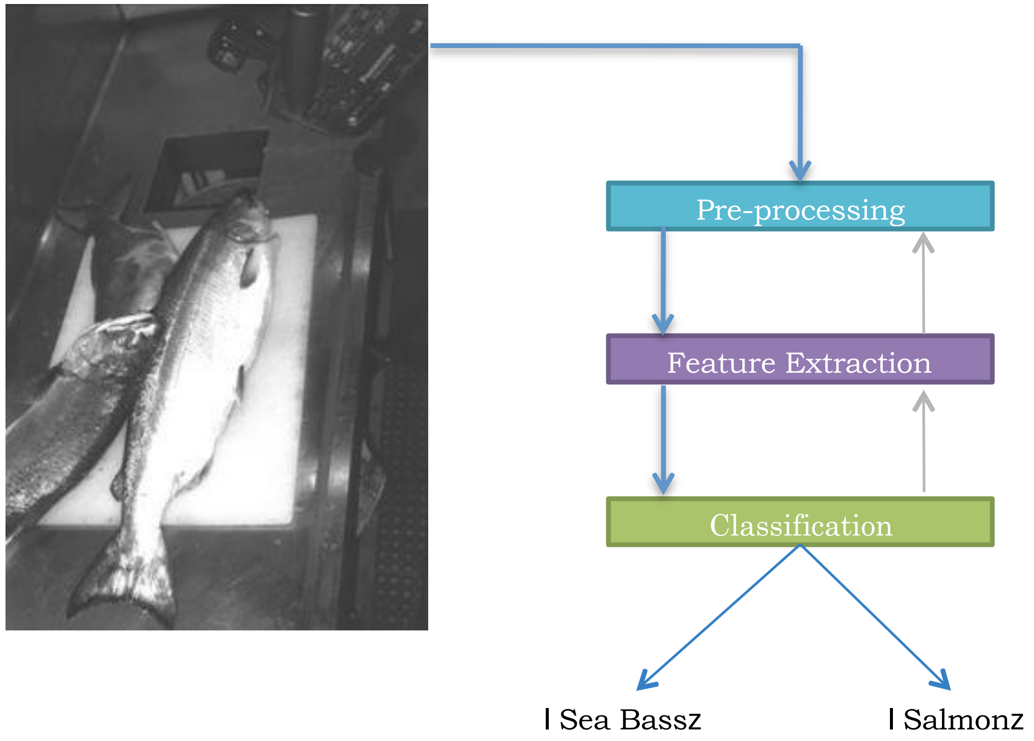


Salmon

A CASE STUDY: FISH CLASSIFICATION

- What kind of information can distinguish one species from the other?
  - length, width, weight, number and shape of fins, tail shape, etc.
- What can cause problems during sensing?
  - lighting conditions, position of fish on the conveyor belt, camera noise, etc.
- What are the steps in the process?

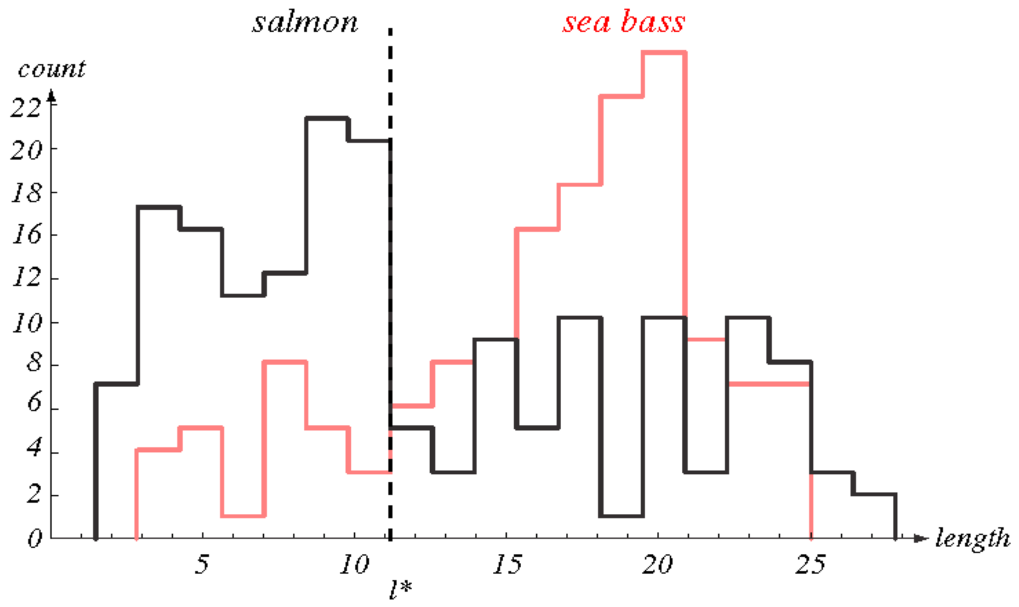
  1. Capture image.
  2. Isolate fish
  3. Take measurements
  4. Make decision



A CASE STUDY: FISH CLASSIFICATION

- Selecting Features
  - Assume a fisherman told us that a sea bass is generally longer than a salmon.
  - We can use length as a feature and decide between sea bass and salmon according to a threshold on length.
  - How can we choose this threshold?

Histograms of the length feature for two types of fish in training samples. How can we choose the threshold to make a reliable decision?

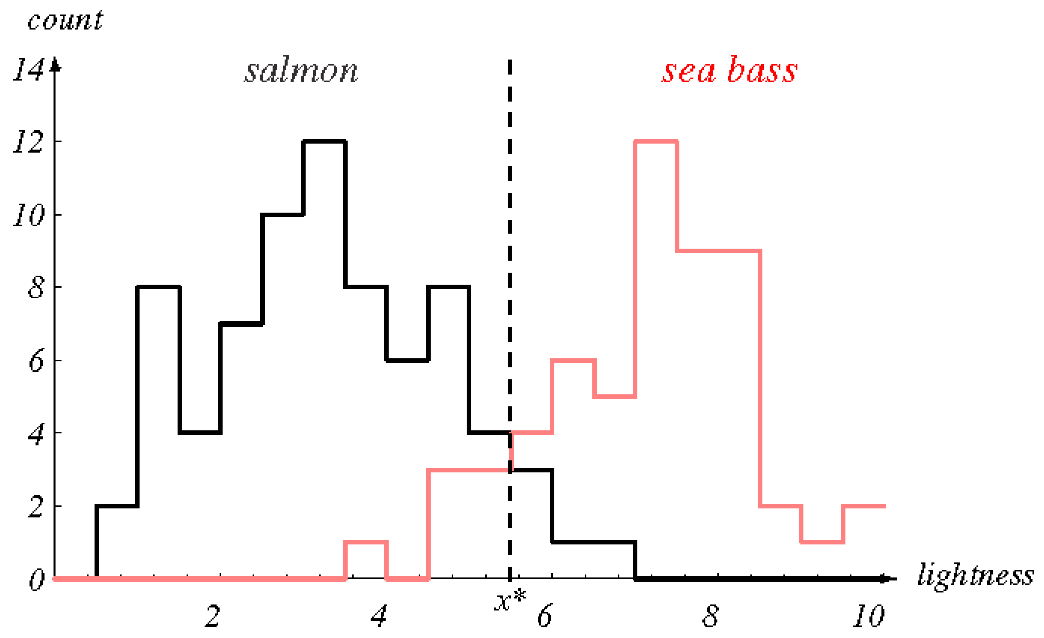


Even though "sea bass" is longer than "salmon" on the average, **there are many examples of fish where this observation does not hold...**

A CASE STUDY: FISH CLASSIFICATION

- Selecting Features
  - Let's try another feature and see if we get better discrimination
    - -Average Lightness of the fish scales

Histograms of the lightness feature for two types of fish in training samples.



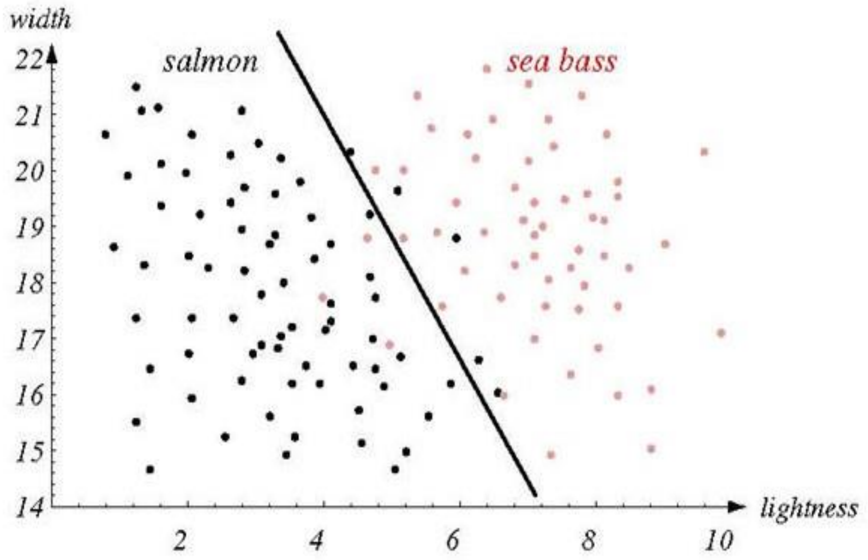
It looks easier to choose the threshold but we still cannot make a perfect decision.

A CASE STUDY: FISH CLASSIFICATION

- Multiple Features
  - Single features might not yield the best performance.
  - To improve recognition, we might have to use more than one feature at a time.
  - Combinations of features might yield better performance.
  - Assume we also observed that sea bass are typically wider than salmon.

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad \begin{matrix} x_1 : lightness \\ x_2 : width \end{matrix}$$

Each fish image is now represented by a point in this 2D feature space



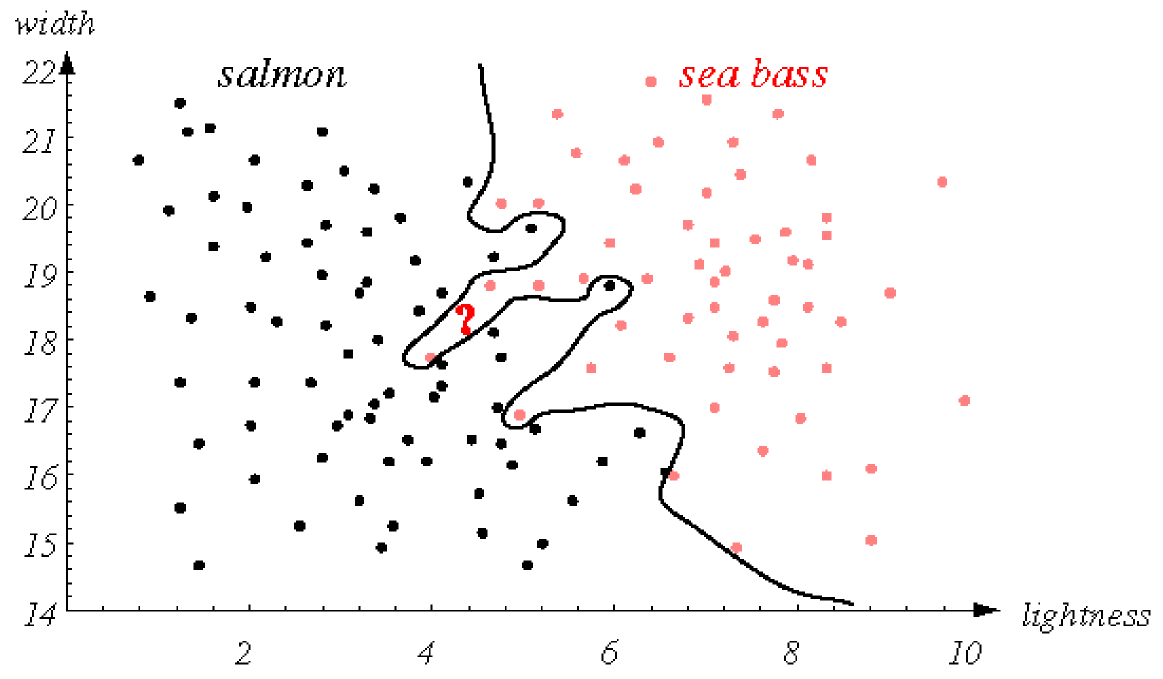
Scatter plot of lightness and width features for training samples. **We can draw a decision boundary to divide the feature space into two regions.**
Does it look better than using only lightness?

A CASE STUDY: FISH CLASSIFICATION

- Designing a **Classifier**
  - Can we do better with another decision rule?
  - More complex models result in more complex boundaries.

DANGER OF
OVER FITTING!!

We may distinguish training samples perfectly but **how can we predict how well we can generalize to unknown samples?**

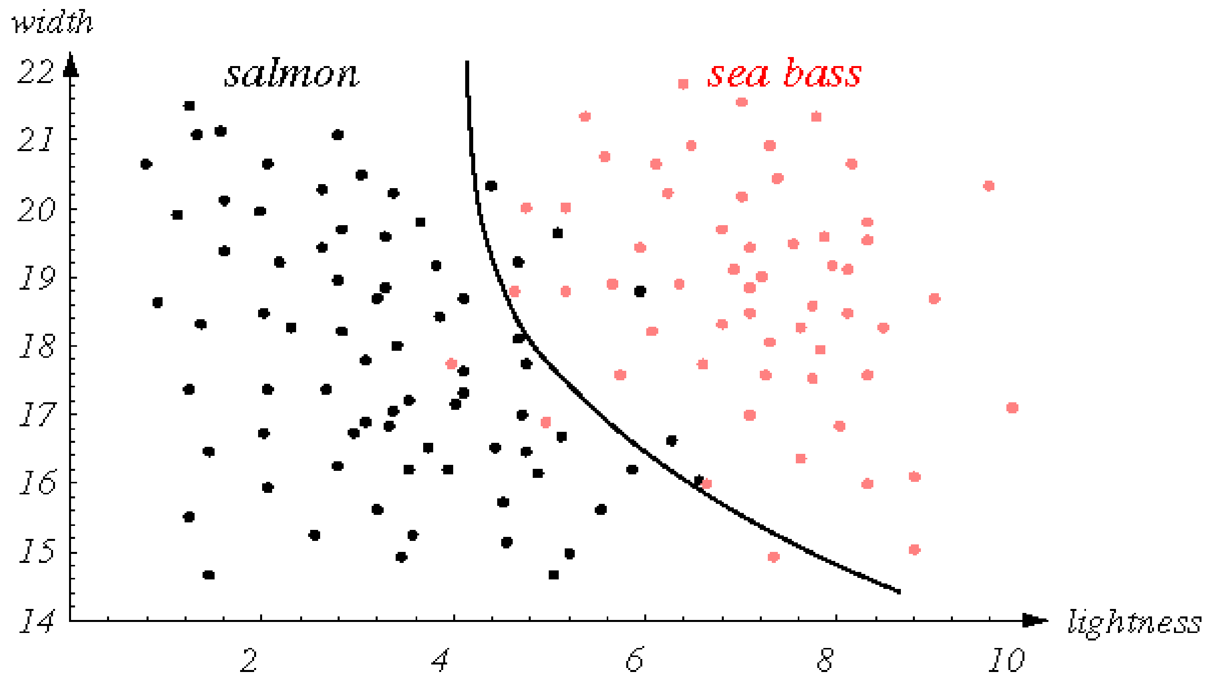


CLASSIFIER WILL FAIL TO GENERALIZE TO NEW DATA...

A CASE STUDY: FISH CLASSIFICATION

- Designing a **Classifier**
    - How can we manage the tradeoff between complexity of decision rules and their performance to unknown samples?

Different criteria lead to different decision boundaries



FEATURE EXTRACTION

- Designing a **Feature Extractor**
  - Its design is **problem specific** (e.g. features to extract from graphic objects may be quite different from sound events...)
  - The ideal feature extractor would produce the same feature vector X for all patterns in the same class, and different feature vectors for patterns in different classes.
  - In practice, different inputs to the feature extractor will always produce different feature vectors, but we hope that the within-class variability is small relative to the **between-class variability**.

- **Designing a good set of features is sometimes "more of an art than a science"...**

FEATURE EXTRACTION

- Multiple Features
  - Does adding more features always improve the results?
    - No!! So we must:
      - Avoid unreliable features.
      - Be careful about correlations with existing features.
      - Be careful about measurement costs.
      - Be careful about noise in the measurements.
  - **Is there some curse for working in very high dimensions?**
    - YES THERE IS! ==> **CURSE OF DIMENSIONALITY**

→ thumb rule: $n >= d(d-1)/2$

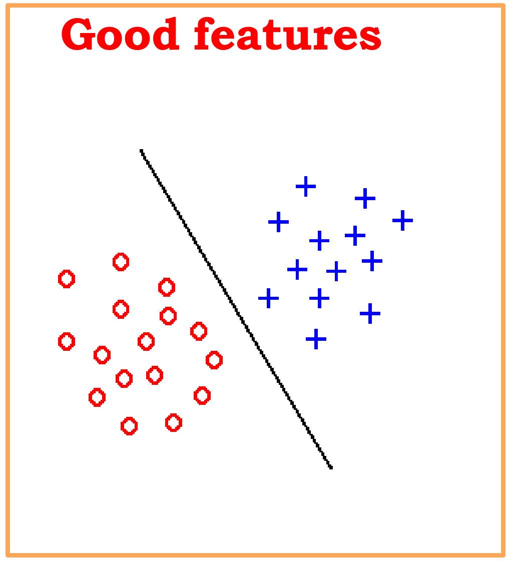
n = nr of examples in training dataset
d = nr of features

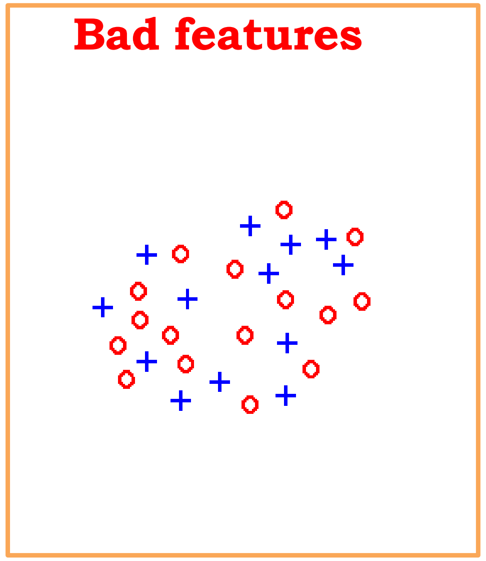
FEATURE EXTRACTION

- Problem: Inadequate Features

– features simply do not contain the information needed to separate the classes, it doesn't matter how much effort you put into designing the classifier.

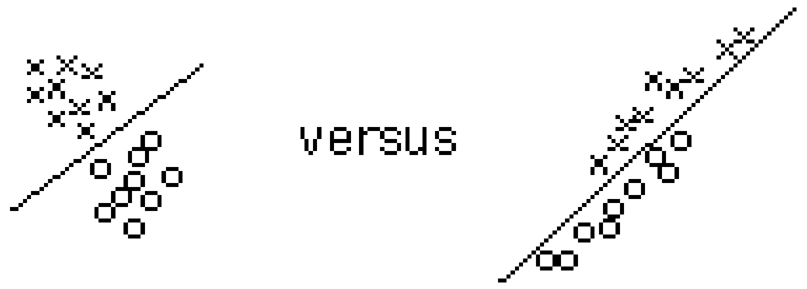
– **Solution**: go back and design better features.





FEATURE EXTRACTION

- Problem: Correlated Features
  - Often happens that two features that were meant to measure different characteristics are influenced by some common mechanism and tend to vary together.
    - E.g. the perimeter and the maximum width of a figure will both vary with scale; larger figures will have both larger perimeters and larger maximum widths.
  - This degrades the performance of a classifier based on Euclidean distance to a template.
    - A pattern at the extreme of one class can be closer to the template for another class than to its own template. A similar problem occurs if features are badly scaled, for example, by measuring one feature in microns and another in kilometers.
- **Solution**: (Use other metrics, e.g. Mahalanobis...) or extract features known to be uncorrelated!

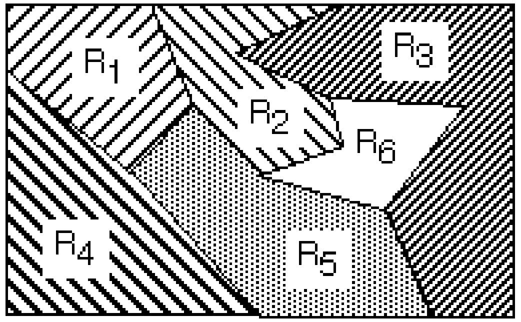


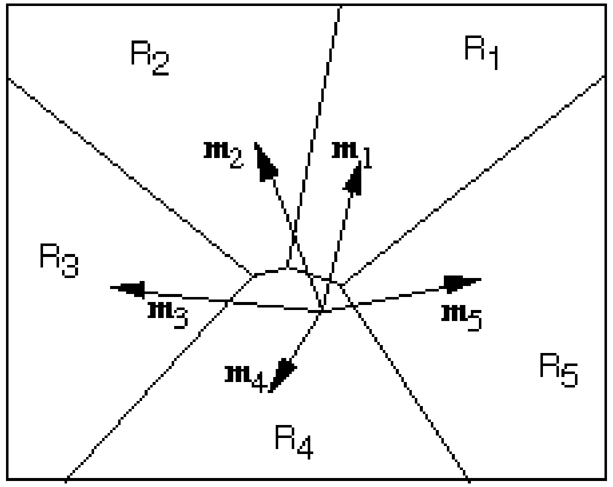
DESIGNING A CLASSIFIER

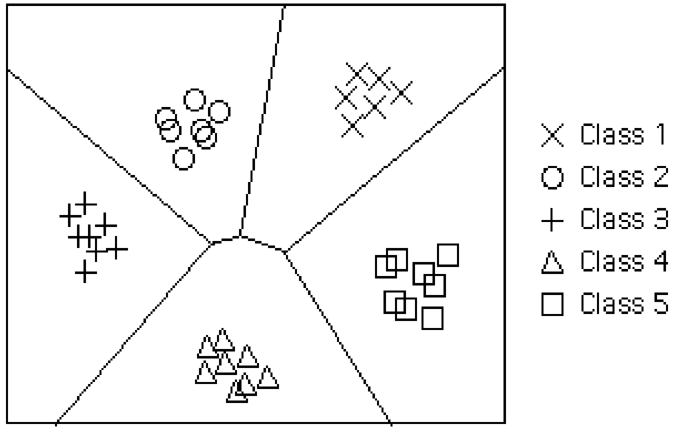
- Model selection:
  - Domain dependence and prior information.
  - Definition of design criteria.
  - Parametric vs. non-parametric models.
  - Handling of missing features.
  - Computational complexity.
  - Types of models: templates, decision-theoretic or statistical, syntactic or structural, neural, and hybrid.
  - How can we know how close we are to the true model underlying the patterns?

DESIGNING A CLASSIFIER

- Designing a **Classifier**
  - How can we manage the tradeoff between complexity of decision rules and their performance to unknown samples?







Different criteria lead to different decision boundaries

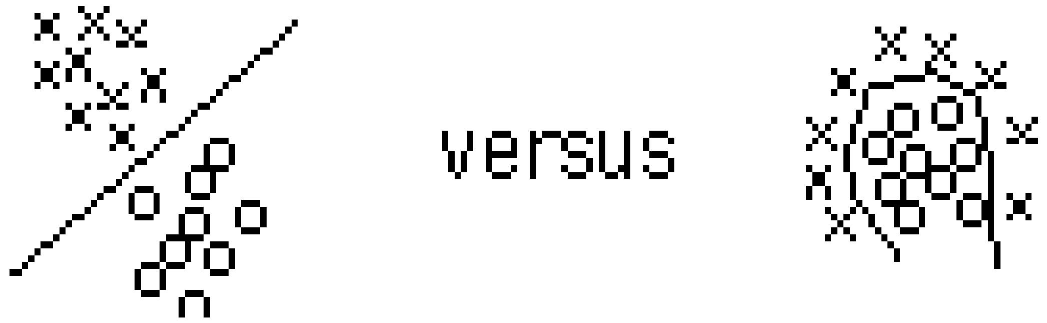
DESIGNING A CLASSIFIER

- Problem: Curved Boundaries
  - linear boundaries produced by a minimum-Euclidean-distance classifier may not be flexible enough.
    - For example, if x1 is the perimeter and x2 is the area of a figure, x1 will grow linearly with scale, while x2 will grow quadratically. This will "warp" the feature space and prevent a linear discriminant function from performing well.

– **Solutions:**

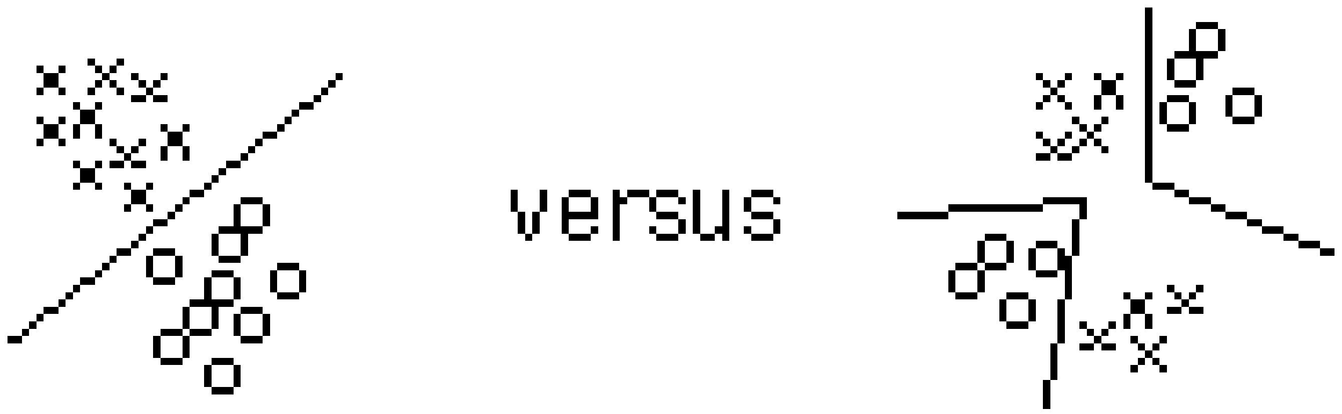
- Redesign the feature set (e.g., let x2 be the square root of the area)

- *Try using a neural network*



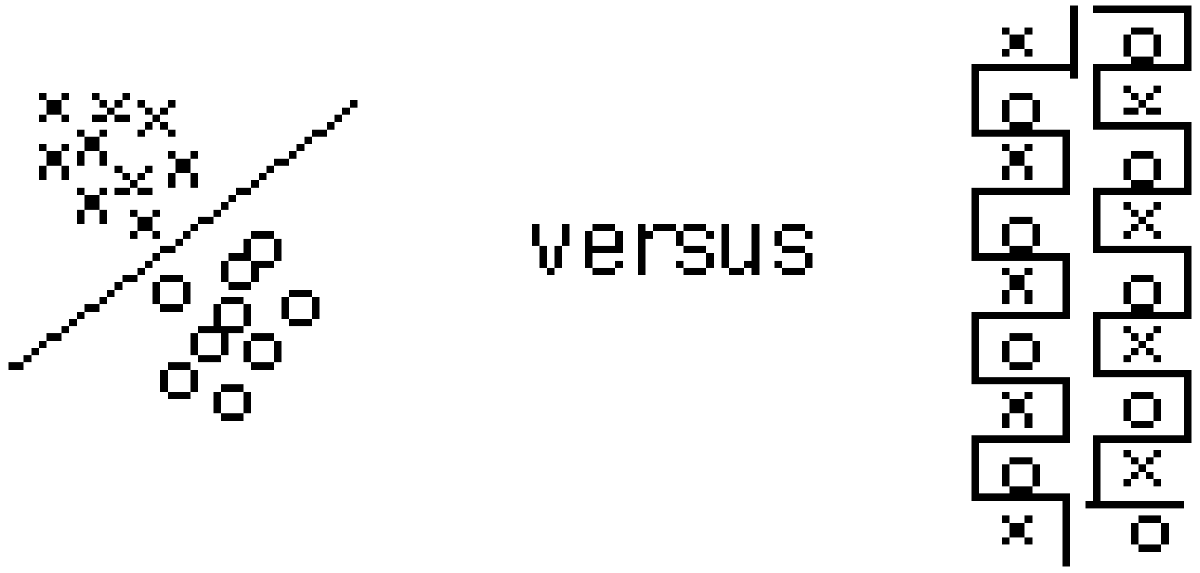
DESIGNING A CLASSIFIER

- Problem: Subclasses in the dataset
  - frequently happens that the classes defined by the end user are not the "natural" classes...
  - **Solution**: use CLUSTERING.

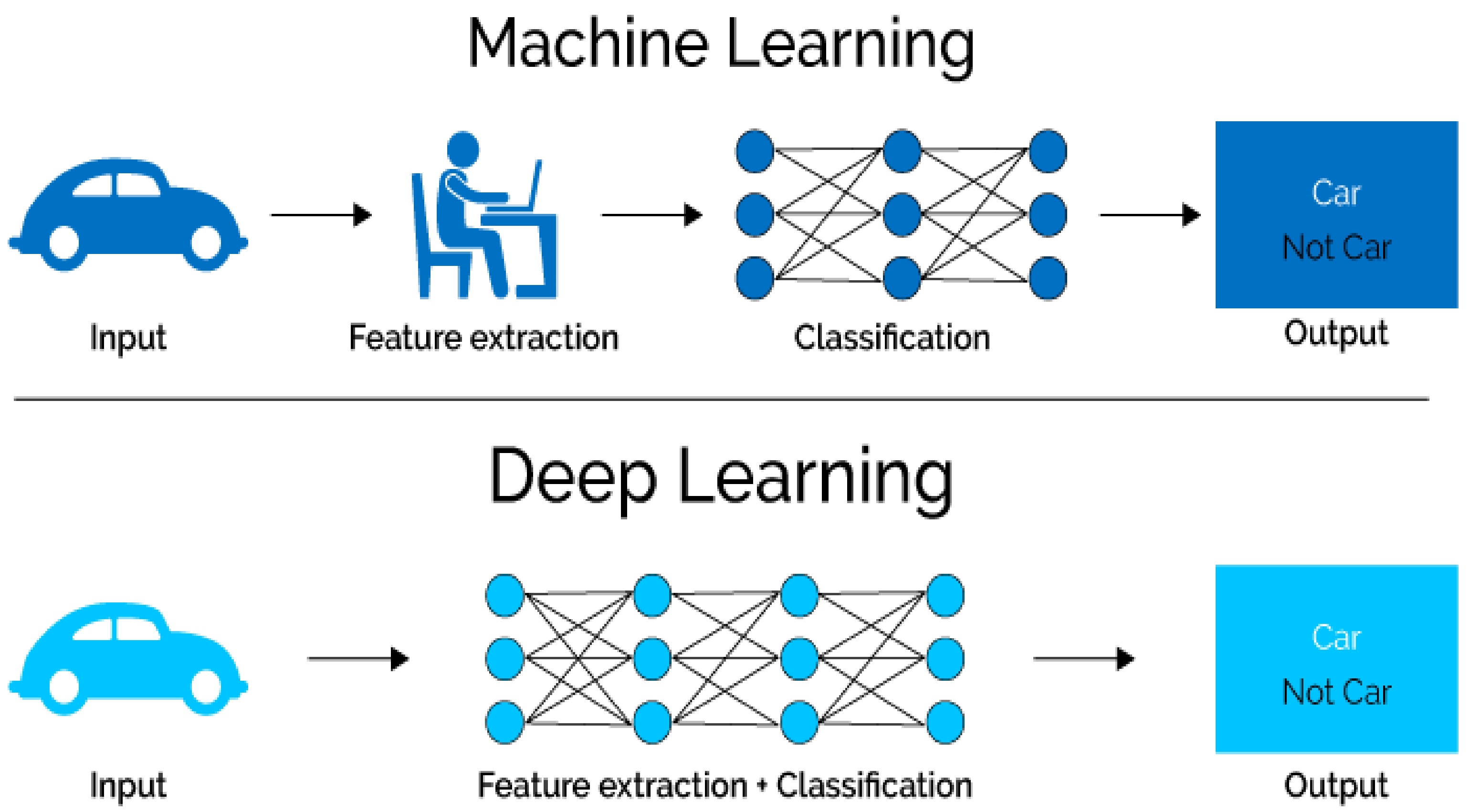


DESIGNING A CLASSIFIER

- Problem: Complex Feature Space
  - – **Solution**: use different type of Classifier...



MACHINE LEARNING VS DEEP LEARNING


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