POLITECNICO
MILANO 1863

IMAGE AND SOUND
ISPG
PROCESSING GROUP

CREATIVE PROGRAMMING AND COMPUTING

Cognitive agents

AGENTS ARCHITECTURE - REVIEW

- **Cognitive agents**: maintain internal symbolic representation of the world

- **Reactive agents**: no explicit representation of the world and focus behavioural rules
  - **Reactive with no memory (Reflex)**: no internal memory
  - **Reactive with memory**: with internal states to keep track of the memory, but not cognitive

- **Hybrid**: mixing reactive and behavioural components to balance reactiveness and deliberativeness

COGNITIVE AGENTS - THEORY

- There are several architectures presented for Cognitive agents
- One of the most effective -> **Belief–desire–intention (DBI) model**
- Based on Michael Bratman's theory of human practical reasoning
- It is based on the idea of reasoning:
  - **Deliberation**: decide what to do
  - **Means-end reasoning**: decide how to do it
- The output of deliberation are **intensions**

- The model is based on
  - **Knowledge**
  - **Behaviour**

COGNITIVE AGENTS - KNOWLEDGE

- Composed by:
- **Beliefs (what it thinks)**
  - *Example: it believes there is a fire*
  - Beliefs is the internal knowledge the agent has about the world
  - The belief base can be updated during the simulation (interactive or online learning)
  - It can be updated by
    - **Perception**: external stimuli update the knowledge of the world
    - **Reasoning:** the internal elaboration of data and stimuli
  - It maintains a certain coherence *(I cannot believe it is raining and it is sunny and the same time)*

COGNITIVE AGENTS - KNOWLEDGE

- Composed by:
- **Desires (what it wants)**
  - *Example: it desires the fire to be extinct*
  - Objectives that the agent would like to accomplish
  - The desirers can be updated during the simulation (interactive or online learning)
  - They are usually given (hard coded)
  - They can be not coherent (it desires *the fire to be extinct and to run away)*

COGNITIVE AGENTS - KNOWLEDGE

- Composed by:
- **Intentions (what it is doing)**
  - *Example: put out the fire*
  - What the agent has chosen to do
  - Are the result of deliberation
  - Should not conflict and are persistent
  - They should be possible to be achieved

….to complete the process:

Deliberation is the process of choosing which desires should be pursued according to the current beliefs. Deliberation produces Intentions.

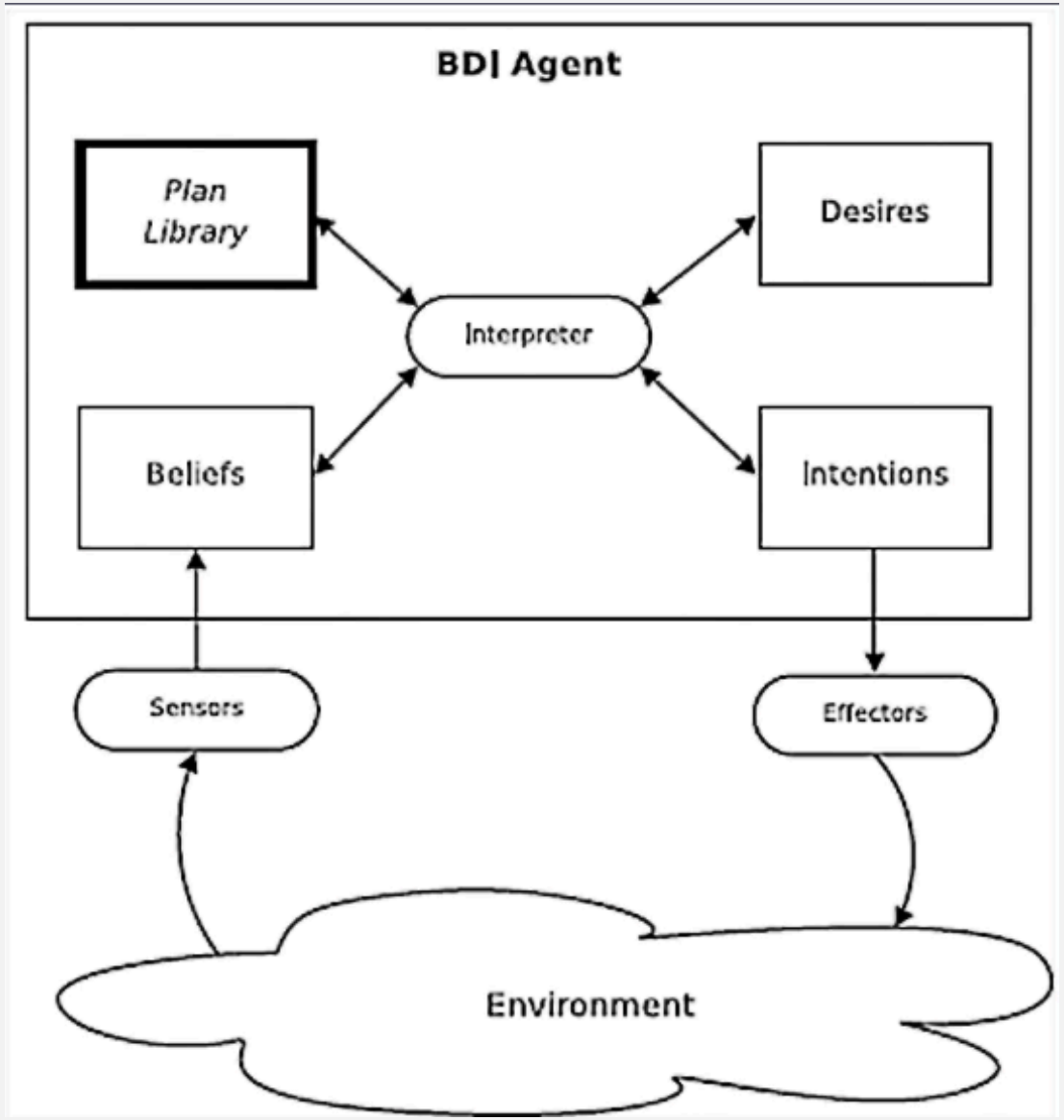
COGNITIVE AGENTS - BEHAVIOURS

- Two elements**: perception and plans**
- **Plans**
  - It is a sequence of actions
  - They are behaviours defined to reach specific desires/intensions
  - Means-end reasoning generates a plan
  - They can be deterministic or not deterministic

- **Perception**

COGNITIVE AGENTS – THINKING

- The interpreter does two operations:
- **Deliberation**: select which desire to pursue as an intention based on the current beliefs
- **Mean-end reasoning**: generates or select a plan to be executed as an attempt to achieve the intention



COGNITIVE AGENTS – THINKING

- BDI-interpreter
  - (B,D,I):= Initialize-state();
  - While true do
    - Update(B,D, I); // according to internal and external perceptions
    - Options:= option-generator(B,D,I);
    - Selected-options:=deliberate(B,D,I);
    - Update-intentions(Selected-options,I);
    - Plan:=Planning(I,B);
    - Execute(Plan);
    - Get-new-perceptions();
  - End While

MACHINE LEARNING

WHAT'S MACHINE LEARNING

Machine learning provides systems the ability to **automatically learn** and **improve from experience** without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it learn from themselves

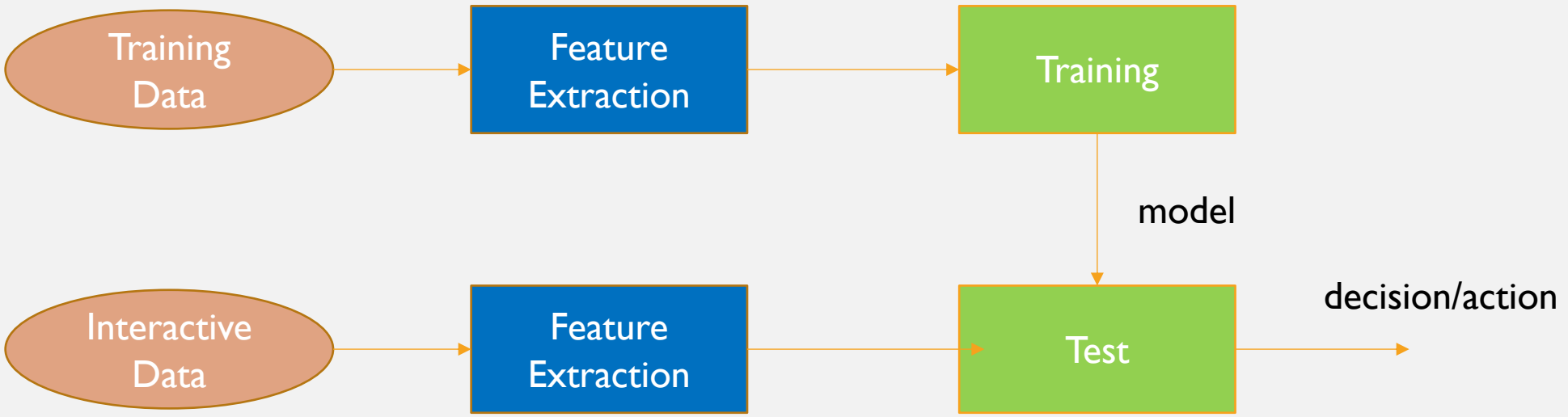
WHY MACHINE LEARNING

- The are contexts where
  - The process of building a symbolic representation of the world is not clear or risk to not be exhaustive
  - It is not possible to define a set of exhaustive actions to achieve a specific results

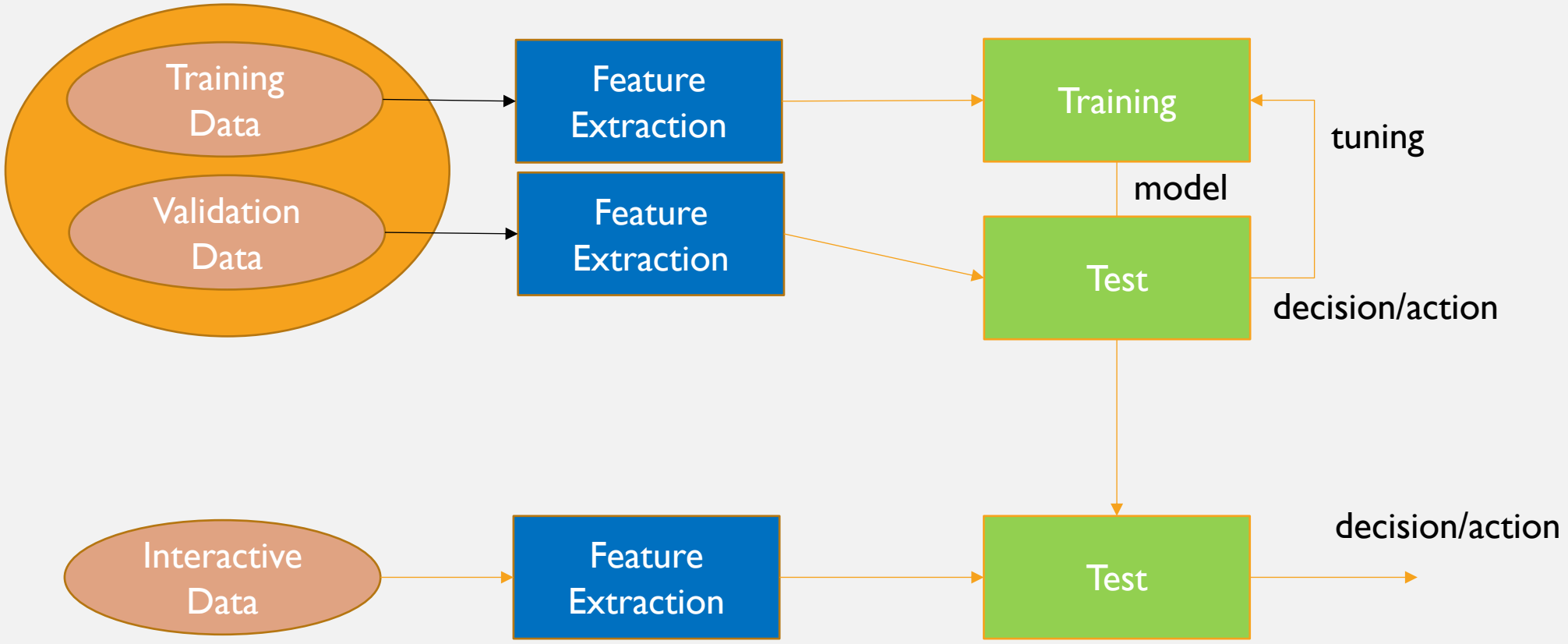
- Instead of trying to model the set of rules (link in rule-based systems), let's train a machine to automatically generate the rules by the examples in a corpus

Data-driven learning

THE MACHINE LEARNING SCHEMA



THE MACHINE LEARNING SCHEMA



FEATURE EXTRACTION

- It is the process of describing the world through descriptors able to capture very specific facets
- Audio features: MFCC, Spectral Centroid, Spectral Flatness, …
- Image Descriptors: RGB pixel values, Blob positions, …
- Video descriptors: Optical Flow, …

FEATURE NORMALIZATION

- Most of the times, datasets contain features highly varying in magnitudes, units and range
- But since, most of the machine learning algorithms use distances between two data points in their computations, this is a problem
- The features with high magnitudes will weight a lot more in the distance calculations than features with low magnitudes
- Features in different ranges are not comparable

FEATURE NORMALIZATION

- To supress this effect, we need to bring all features to the same level of magnitudes
- 3 methods

  - **Standardization** (or Z-score normalization): the features will be rescaled so that they'll have the properties of a standard normal distribution with μ=0 and σ=1 where μ is the mean (average) and σ is the standard deviation from the mean

$$z = \frac{x - \mu}{\sigma}$$

FEATURE NORMALIZATION

- **Min-Max scaling:** In this approach, the data is scaled to a fixed range - usually 0 to 1
  - The cost of having this bounded range - in contrast to standardization - is that we will end up with smaller standard deviations, which can suppress the effect of outliers.

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

FEATURE NORMALIZATION

- **Mean Normalisation:** This distribution will have values between -1 and 1with μ=o

$$x' = \frac{x - \text{mean}(x)}{\max(x) - \min(x)}$$

- Which feature normalization algorithm to use ?
- Depends on the distribution of data (Gaussian ?) and on the presence of outliers

FEATURE SPACE

- Normalized features for an element can be seen as its coordinated in a **multidimensional space** where the axis are the features

- In the space a distance can be defined

FEATURE SELECTION AND REDUCTION

- Needs to select very descriptive and discriminant features
- It is difficult to manually select the features that are useful for the final goal to reach
- Feature not useful can ”confuse” the learning -> may decrease the impact of other features

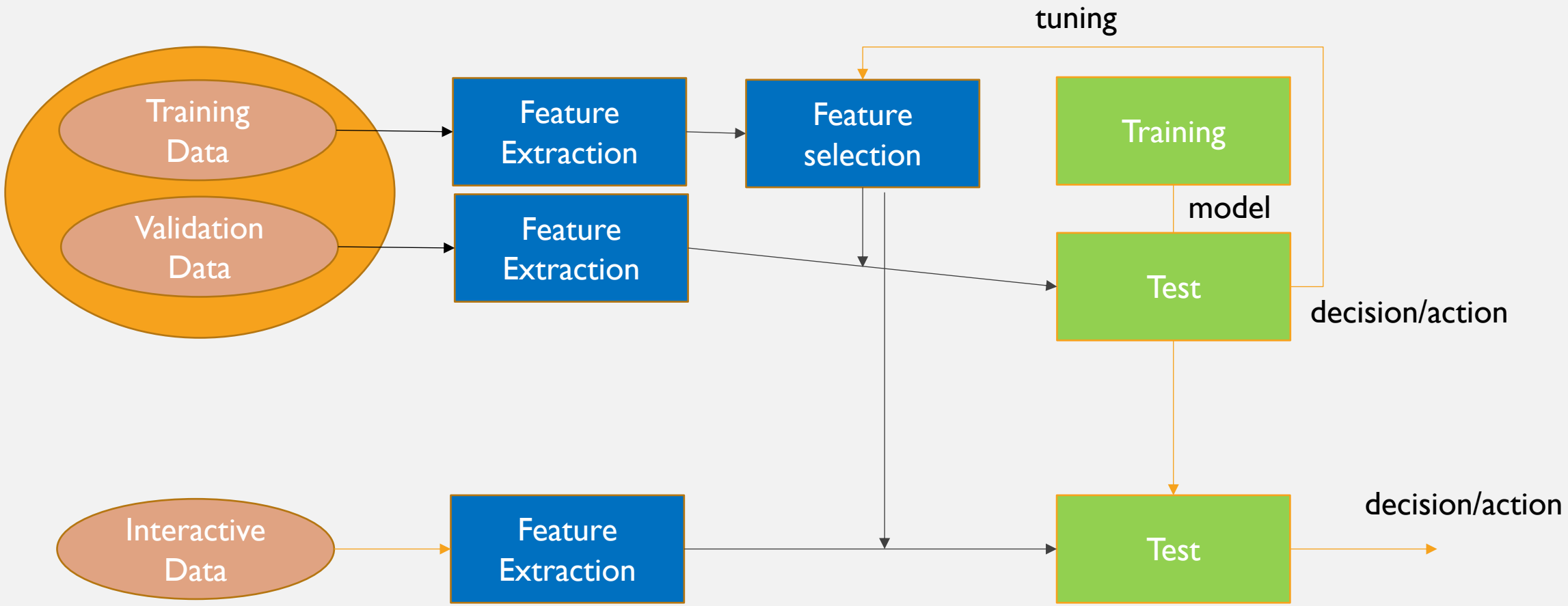
- Needs of automatic methods for feature selection:

- Two families of methods
  - **Feature Selection:** reduce the feature space by selecting a subset of features
  - **Feature Reduction**: reduce the feature space by generating new synthetic features

FEATURE SELECTION

- Reasons for performing FS may include:
  - removing irrelevant and redundant data.
  - increasing predictive accuracy of learned models.
  - reducing the cost of the data.
  - improving learning efficiency, such as reducing storage requirements and computational cost.
  - reducing the complexity of the resulting model description, improving the understanding of the data and the model.

FEATURE SELECTION



- The selection is guided by the tuning process

SEQUENTIAL FORWARD GENERATION (SFG)

- It starts with an empty set of features S.
- As the search starts, features are added into S according to some criterion that distinguish the best feature from the others.
- S grows until it reaches a full set of original features.
- The stopping criteria can be a threshold for the number of relevant features m or simply the generation of all possible subsets in brute force mode.

- The criterion can be the accuracy of the final prediction

- Sequential Backward Generation (SBG) is the reverse: starts with the full set of features

SEQUENTIAL FORWARD GENERATION (SFG)

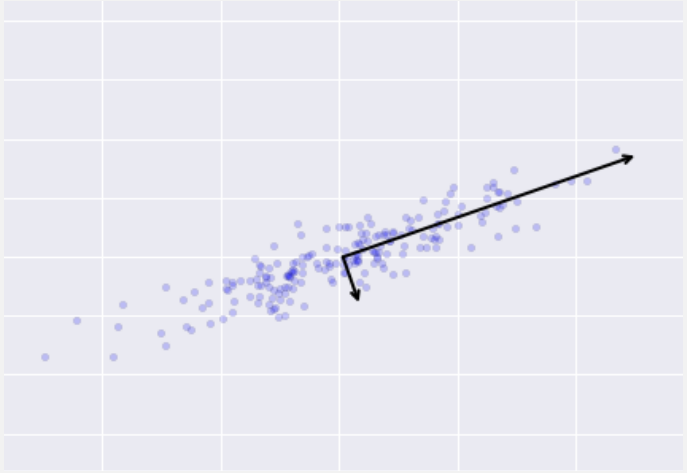
- **Bidirectional Generation (BG):**
  - Begins the search in both directions, performing SFG and SBG concurrently.
    - They stop in two cases:
      - when one search finds the best subset comprised of m features before it reaches the exact middle
      - both searches achieve the middle of the search space. It takes advantage of both SFG and SBG.

- **Random Generation (RG):**
- It starts the search in a random direction.
- The choice of adding or removing a features is a random decision.
- RG tries to avoid the stagnation into a local optima by not following a fixed way for subset generation.

FEATURE REDUCTION - PCA

PRINCIPAL COMPONENT ANALYSYS (PCA)

- Dimensionality reduction technique
- Orthogonal transformation
- Converts observations of possibly correlated variables into linearly uncorrelated ones



- Rough explanation:
  - First component has largest possible variance
  - Each succeeding component has the maximum possible variance but is constrained to be orthogonal to the preceding one.

FEATURE REDUCTION - PCA

- Mathematical explanation:
  - Orthogonal linear transformation that converts the data to a new coordinate system, such that the biggest variance by some scalar projection of the data lies on the first coordinate (first principal component), the second biggest on the second coordinate, and so on…

  - Consider a matrix **X,** with zero sample-mean related to the columns, each row represents a different repetition of the experiment
  - The transformation is defined by a set of p-dimensional vectors of coefficients $\mathbf{w}_{(k)} = (w_1,\ldots,w_p)_{(k)}$ that map each row vector x(i) of X to a new vector of principal components $\mathbf{t}_{(i)} = (t_{(1)},\ldots,t_{(l)})$, given by

$$\mathbf{t}_{k(i)} = \mathbf{t}_{(1)} * \mathbf{w}_{(1)} \text{ for } i = 1,\ldots,n \qquad k = 1,\ldots,l$$

LEARNING

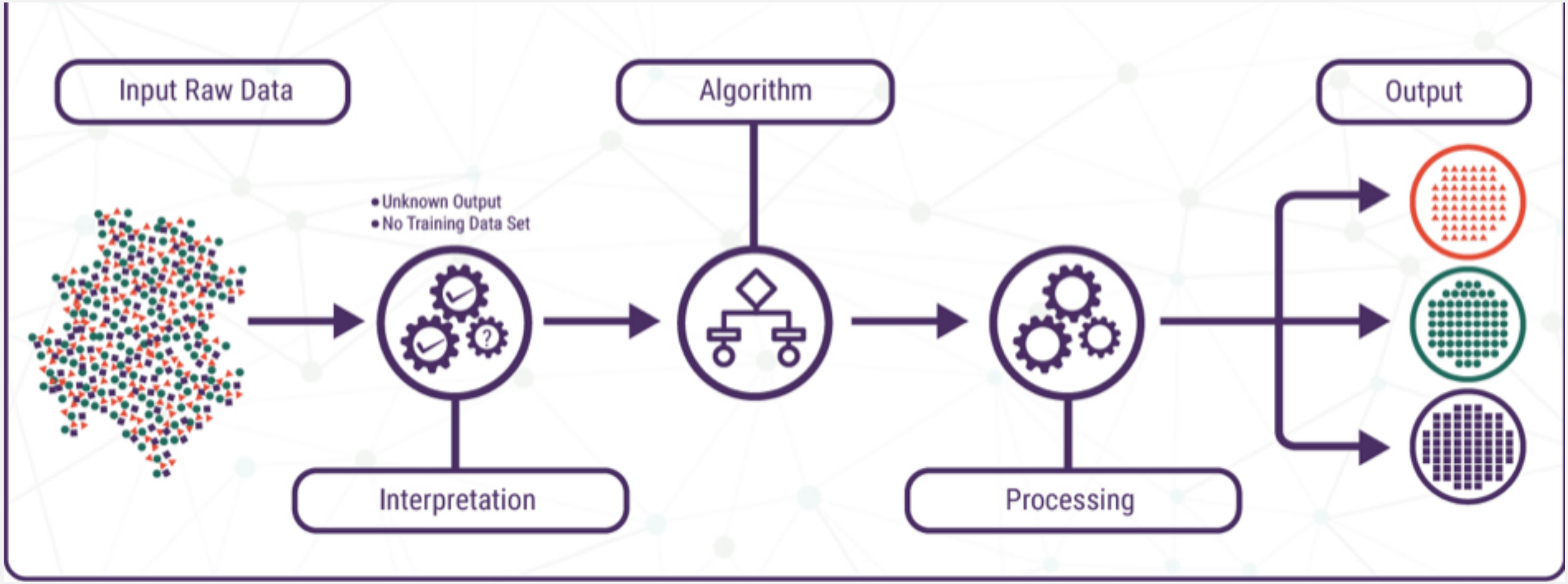
- **Going back to cognitive agents**
- Feature extraction (a multi-dimensional representation of the world) and Training are the process of **learning a symbolic representation** of the world
- Final decision –> **Reasoning**

- Two kinds of learning:
  - **Unsupervised Learning:**
  - **Supervised Learning:**

UNSUPERVISED LEARNING

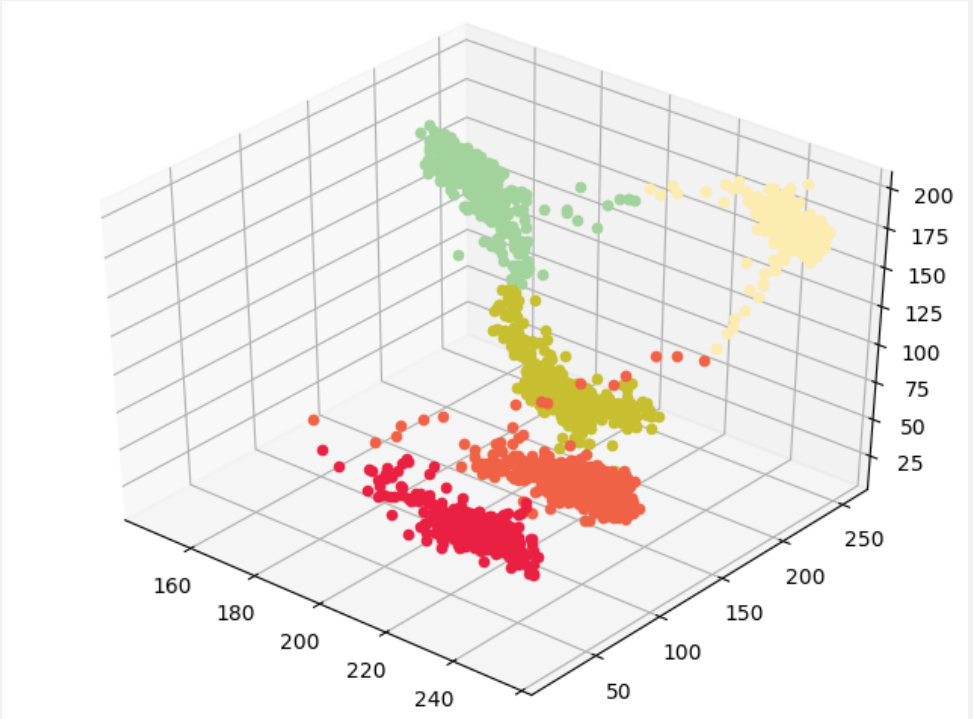
UNSUPERVISED LEARNING

- Unsupervised learning allow the model to work on its own to discover information and build the model. It mainly deals with the unlabelled data



CLUSTERING

- **Clustering** is part of unsupervised learning algorithm types
- Is the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar (in some sense) to each other than to those in other groups (clusters)
- The similarity is computed in the feature space



CLUSTERING

- A good clustering method will produce high quality clusters with
- high intra-class similarity
- low inter-class similarity

- The quality of a clustering method is also measured by its ability to discover some or all of the hidden patterns
- The quality of a clustering result depends on both the similarity measure

CLUSTERING

- **Distance measure** will determine how the similarity of two elements is calculated and it will influence the shape of the clusters. They include:

- 1. The Euclidean distance (also called 2-norm distance)

$$d(x, y) = \sqrt[2]{\sum_{i=1}^{p} | x_i - y_i |^2}$$

- 2. The Manhattan distance (also called taxicab norm or 1-norm)

$$d(x, y) = \sum_{i=1}^{p} | x_i - y_i |$$

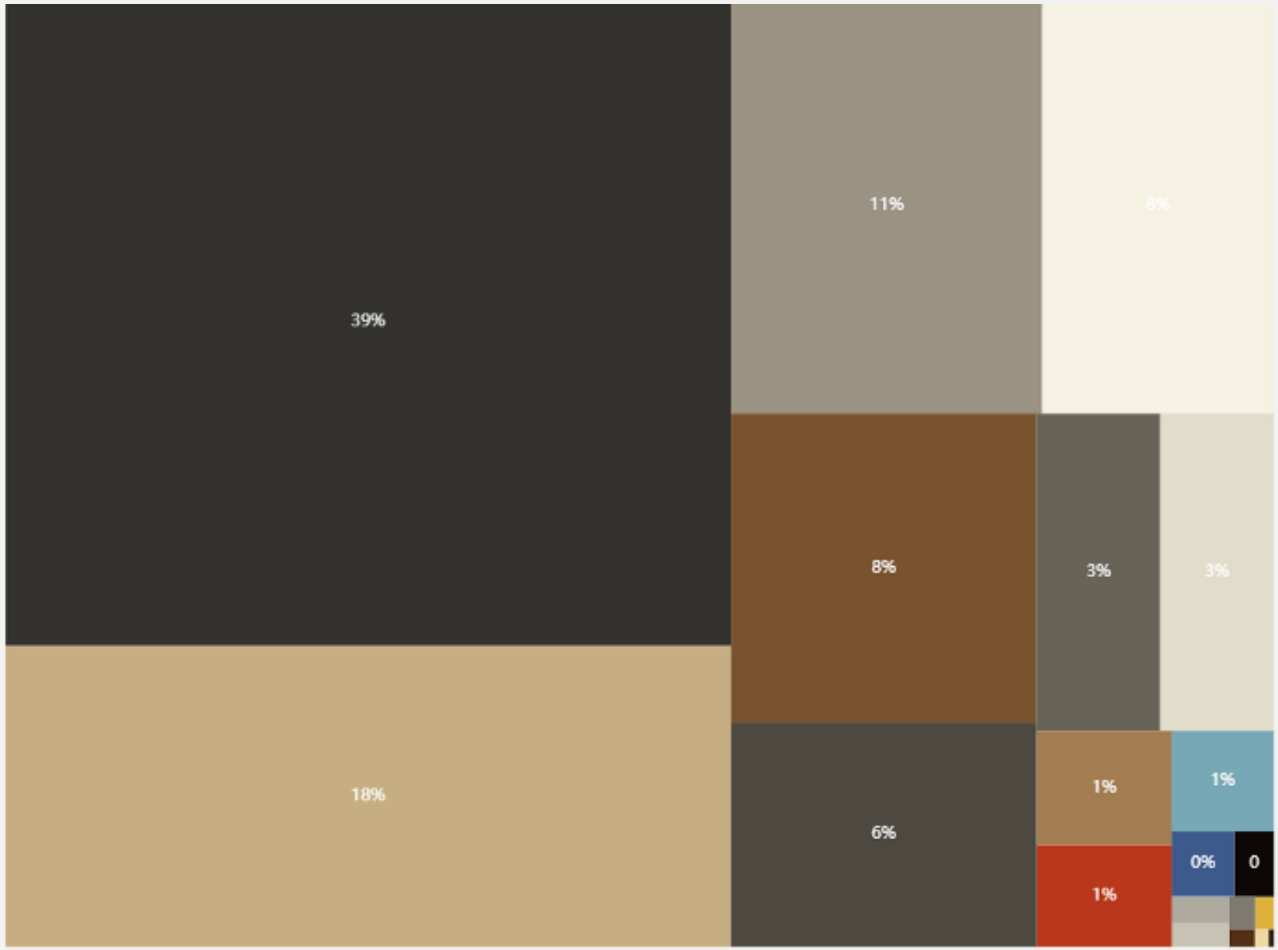
K-MEAN CLUSTERING

- The k-means algorithm is an algorithm to cluster n objects based on attributes into k partitions, where k < n

- Given the cluster number K, the K-means algorithm is carried out in three steps
- after initialization:
- Initialisation: set seed points (randomly)
- 1) Assign each object to the cluster of the nearest seed point measured with a
- specific distance metric
- 2) Compute new seed points as the centroids of the clusters of the current
- partition (the centroid is the centre, i.e., mean point, of the cluster)
- 3) Go back to Step 1), stop when no more new assignment (i.e., membership in
- each cluster no longer changes)

http://shabal.in/visuals/kmeans/2.html

K-MEANS - APPLICATION



T-SNE

- t-Distributed Stochastic Neighbour Embedding (t-SNE) is a dimensionality reduction technique used to represent high-dimensional dataset in a low-dimensional space of two or three dimensions so that we can visualize it

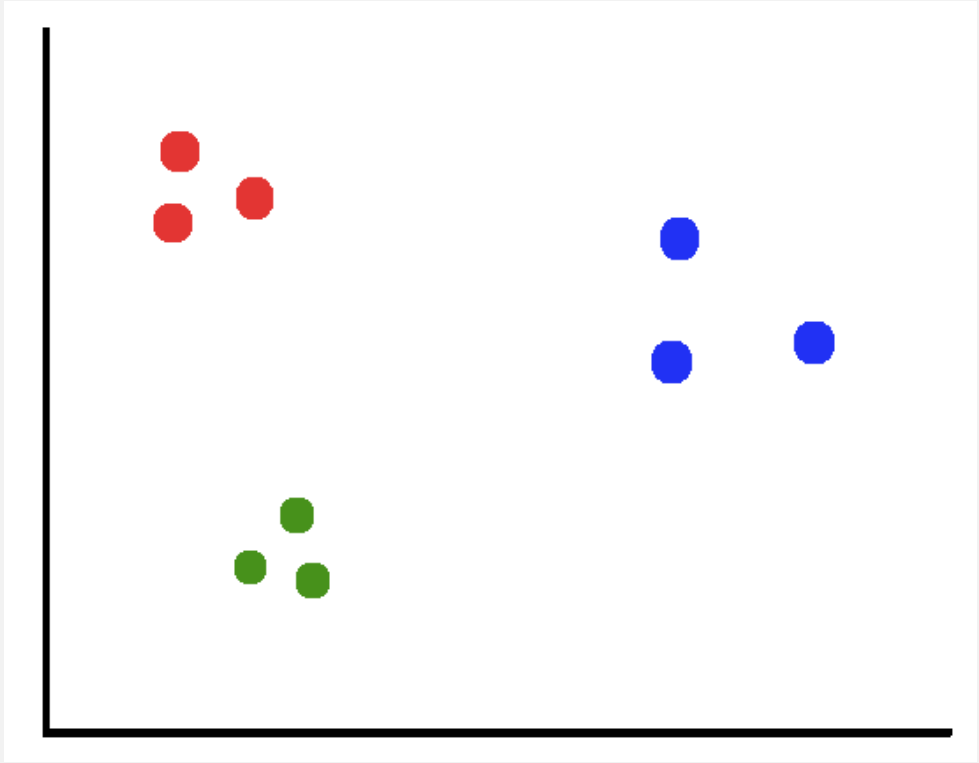
- **t-SNE creates a reduced feature space where similar samples are modelled by nearby points and dissimilar samples are modelled by distant points with high probability**

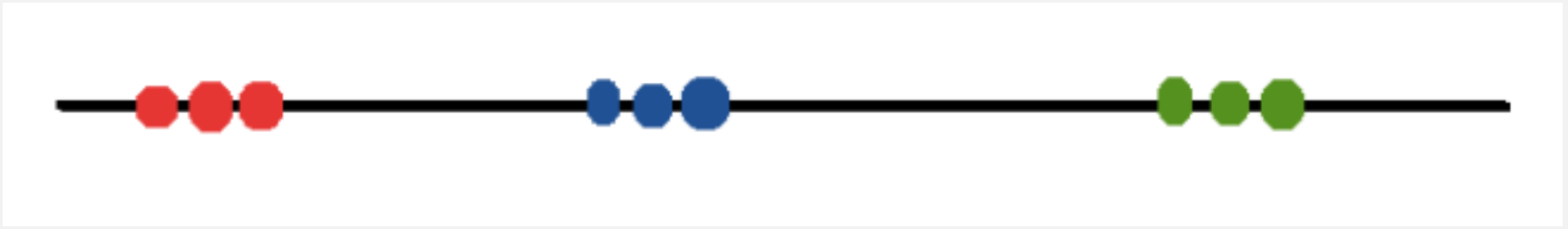
- It based on the idea that give two data, if they are similar they are more likely to be picket

T-SNE

- Constructs a probability distribution for the high-dimensional samples in such a way that similar samples have a high likelihood of being picked while dissimilar points have an extremely small likelihood of being picked. Then, t-SNE defines a similar distribution for the points in the low-dimensional embedding.
- Finally, t-SNE minimizes the Kullback–Leibler divergence between the two distributions with respect to the locations of the points.

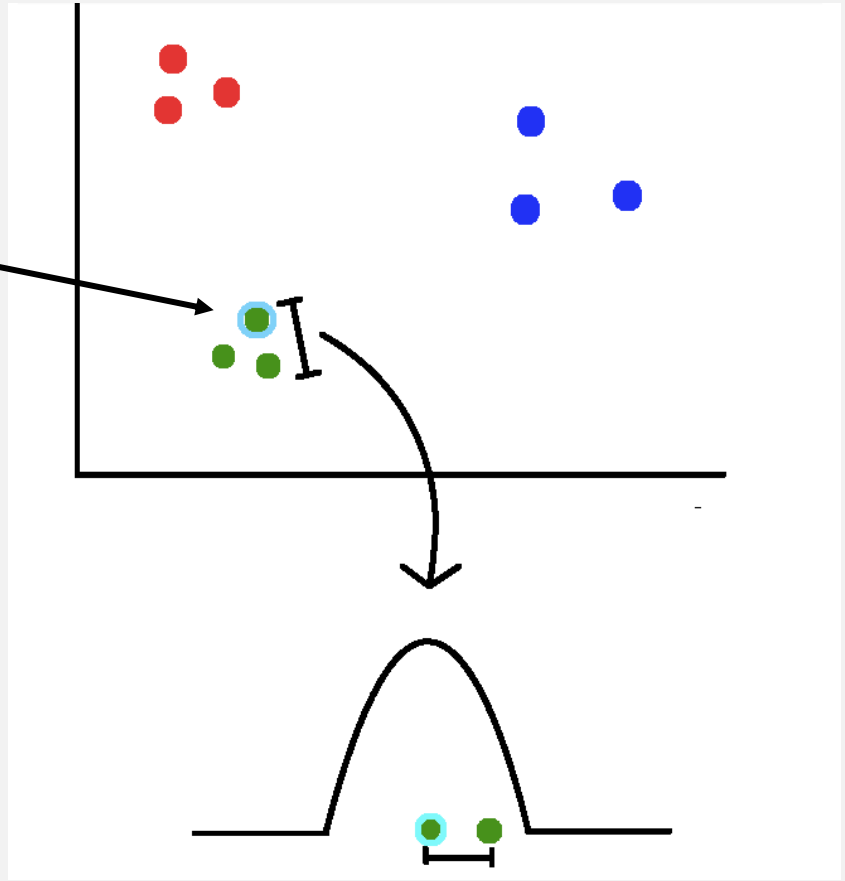
T-SNE





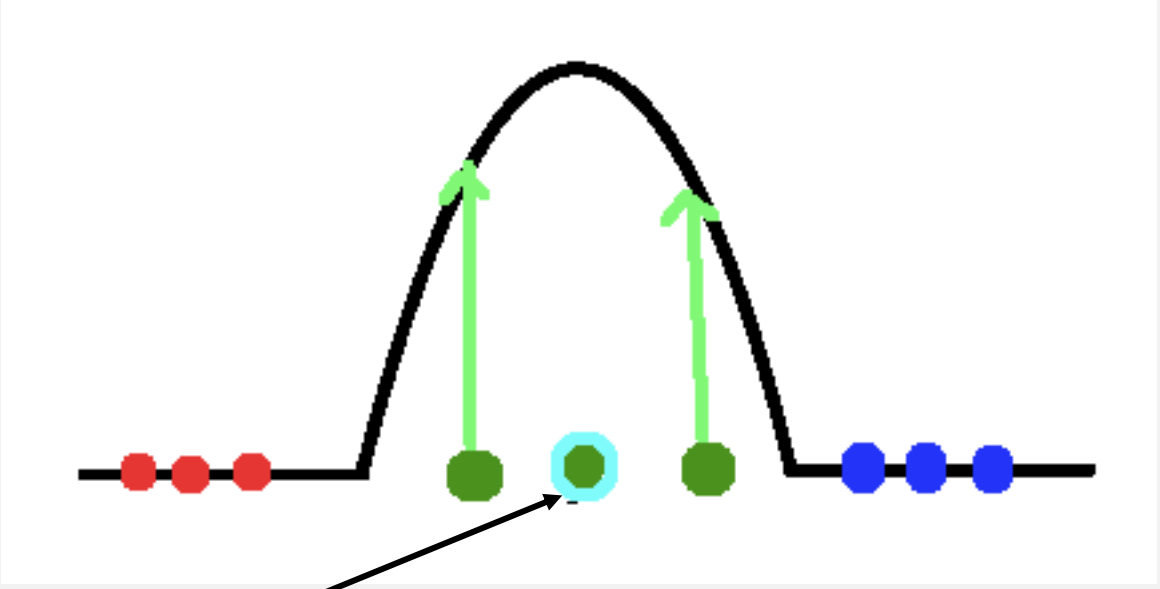
T-SNE

- The first step in the t-SNE algorithm involves measuring the distance from one point with respect to every other point. Instead of working with the distances directly, we map them to a probability distribution.



T-SNE

- In the distribution, the points with the smallest distance with respect to the current point have a hight likelihood, whereas the points far away from the current point have very low likelihoods.

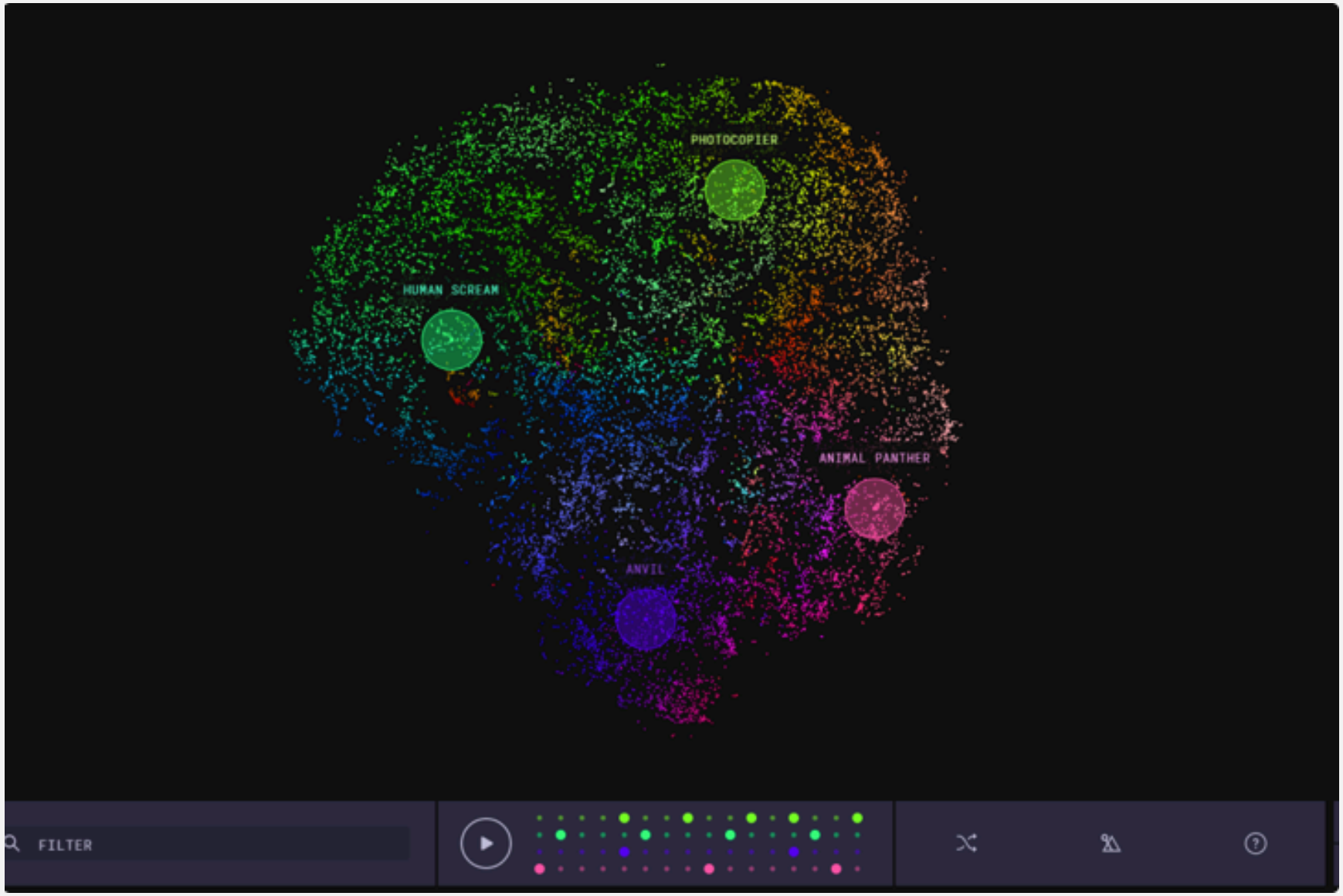


T-SNE

- Give the probability distribution P for each point, t-SNE create a low-dimensional feature space
- The feature space are built in the way to preserve the probability distribution of a point with respect the all the other: that is, low-dimensional embedding, generate a similar distribution Q for the points.

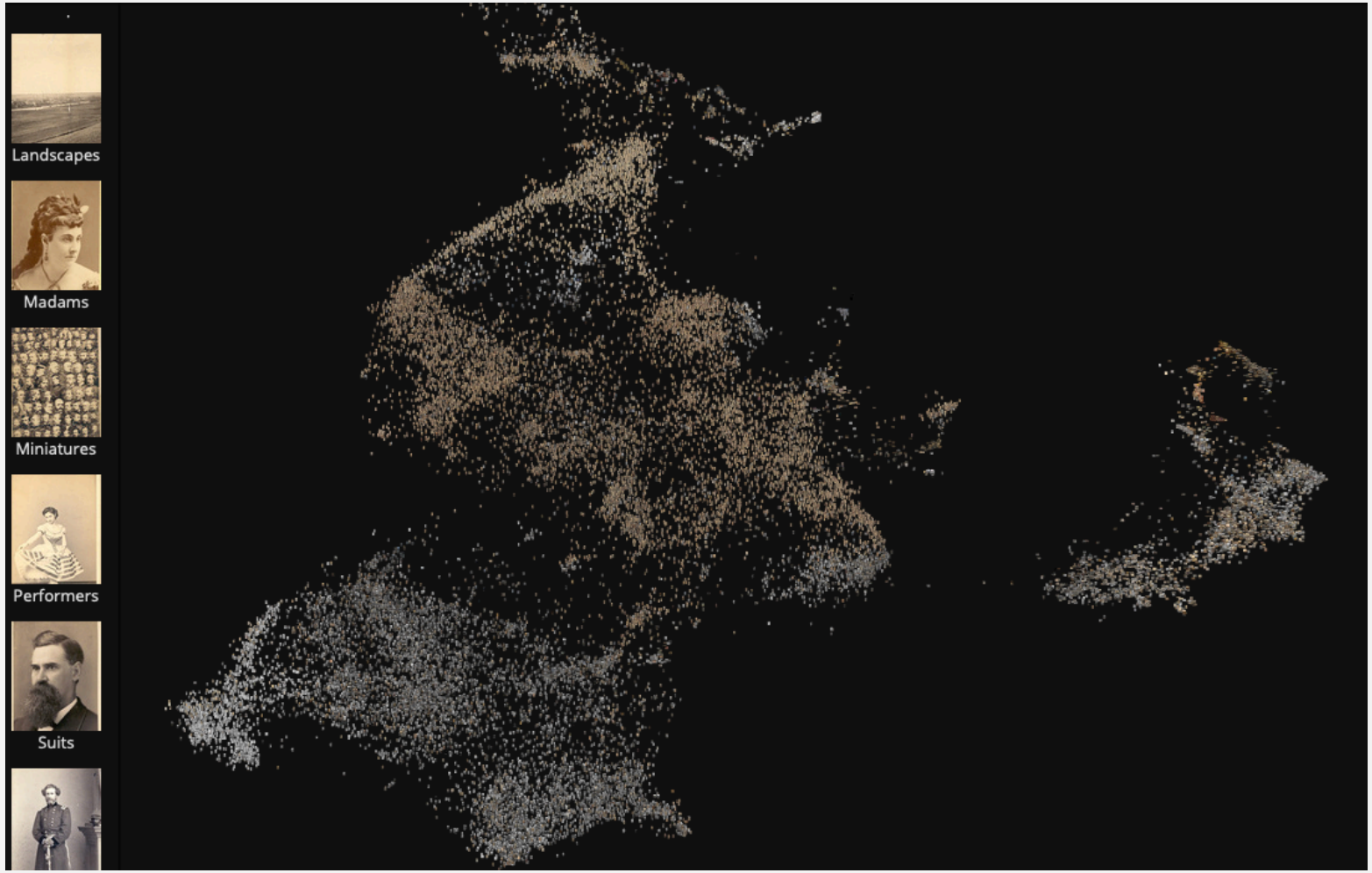
- The final space is created by minimizing the Kullback–Leibler divergence between P and Q with respect to the locations of the points

EXAMPLE: THE INFINITE DRUM



https://experiments.withgoogle.com/ai/drum-machine/view/

EXAMPLE: IMAGE BROWSING



Convolutional Neural Network is used to extract a high dimensional feature vector for each image: about 200 features

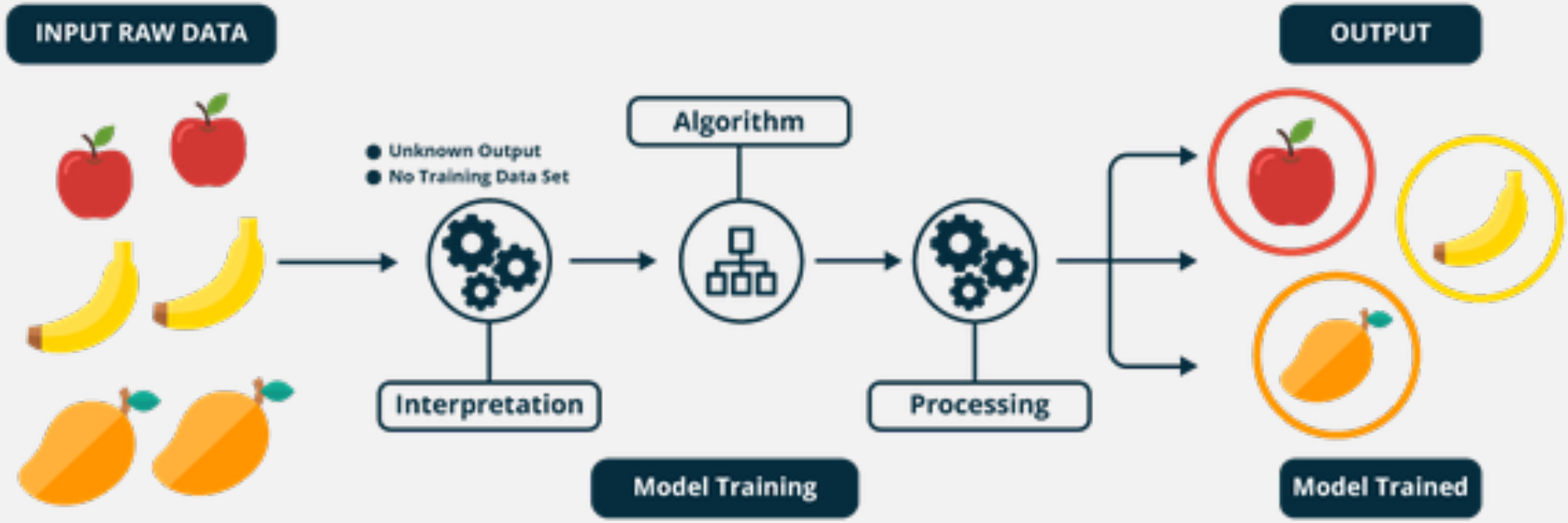
T-SNE reduces the 200D feature space into 3D feature space

https://s3-us-west-2.amazonaws.com/lab-apps/pix-plot/index.html

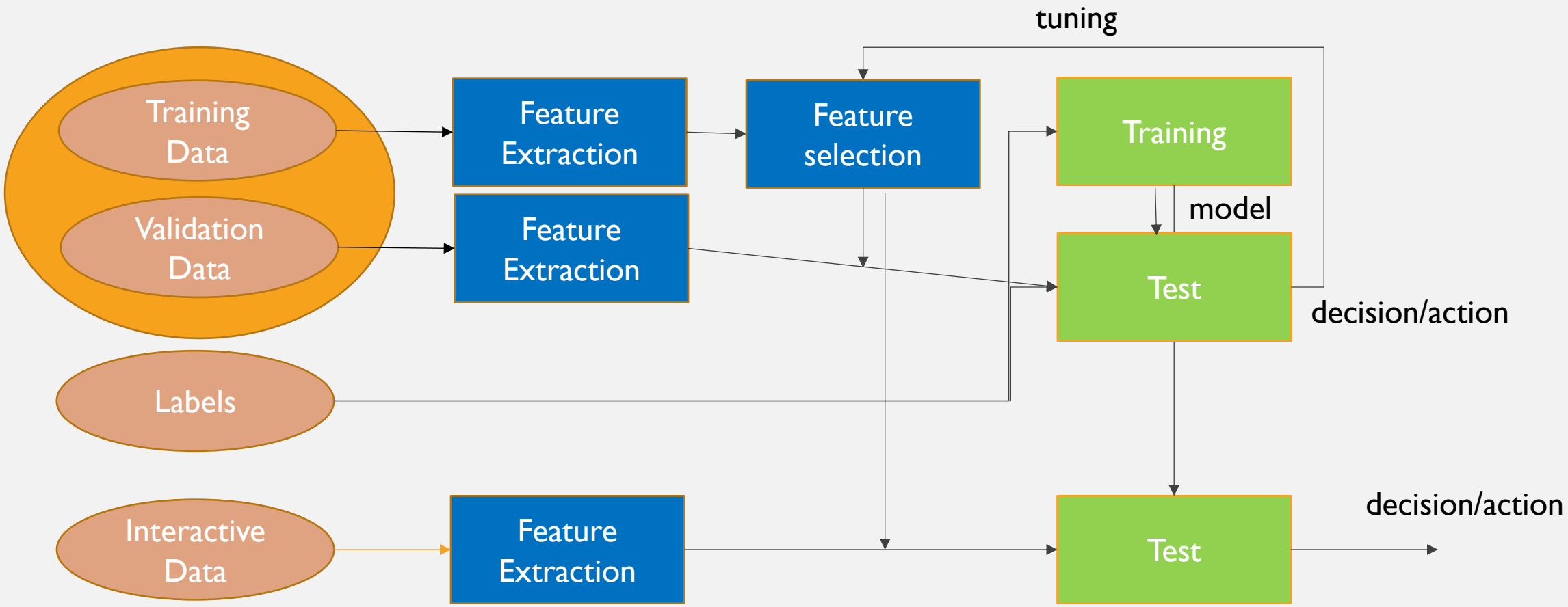
SUPERVISED LEARNING

SUPERVISED LEARNING

- Supervised learning: train the machine using data which is well "**labelled**."
- It means some data is already tagged with the correct answer. It can be compared to learning which takes place in the presence of a supervisor or a teacher.
- A supervised learning algorithm learns from labelled training data, helps you to predict outcomes for unforeseen data.



SUPERVISED LEARNING



SUPERVISED LEARNING

Functions $\mathcal{F}$

$f : \mathcal{X} \to \mathcal{Y}$

Training data

$\{(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}\}$

LEARNING

find $\hat{f} \in \mathcal{F}$
s.t. $y_i \approx \hat{f}(x_i)$

Learning machine

PREDICTION

$y = \hat{f}(x)$

New data

x

SUPERVISED LEARNING

- Tow kind of problems:
- **classification** algorithms attempt to estimate the mapping function (f) from the input variables (x) to discrete or categorical output variables (y)

- **regression** algorithms attempt to estimate the mapping function (f) from the input variables (x) to numerical or continuous output variables (y)
  - regression basically estimates a mapping from multi-dimensional space to a function in a1-dimensional space

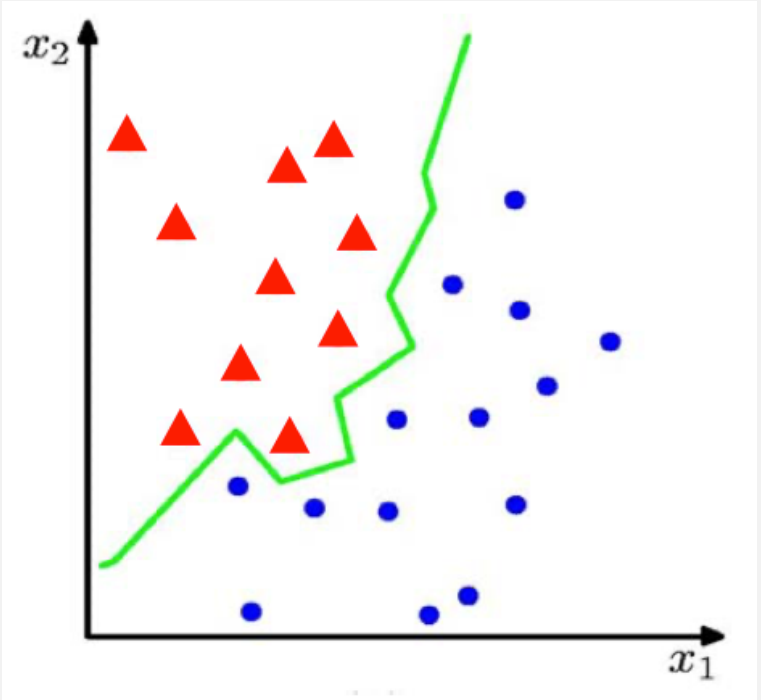
CLASSIFICATION

- Suppose we are given a training set of N observations

$$(x_1, \ldots, x_N) \text{ and } (y_1, \ldots, y_N), x_i \in \mathbb{R}^d, y_i \in \{-1, 1\}$$

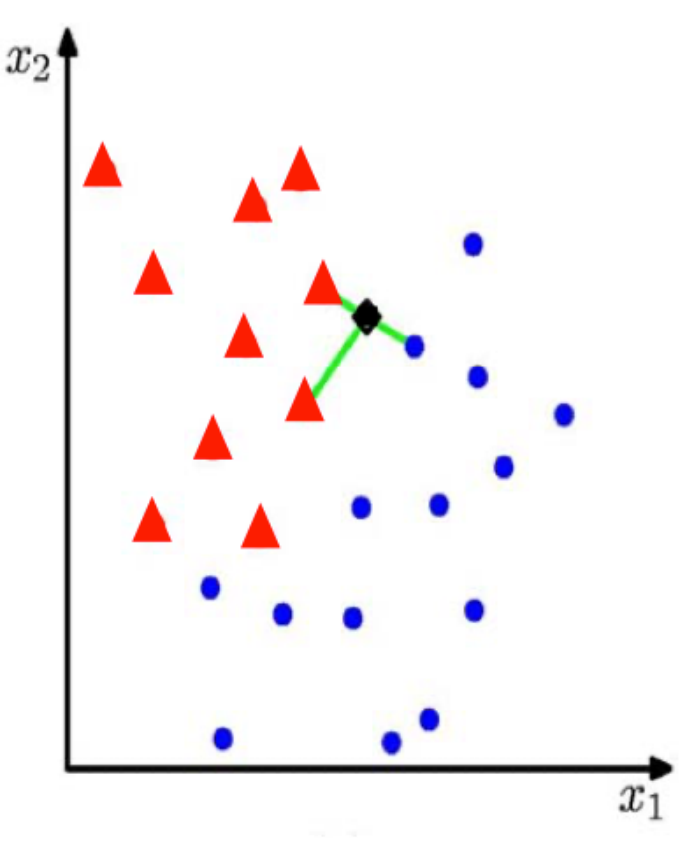
- Classification problem is to estimate f(x) from this data such that

$$f(x_i) = y_i$$



KNN

- For each test point, x, to be classified, find the K nearest samples in the training data
- Classify the point, x, according to the majority vote of their class labels

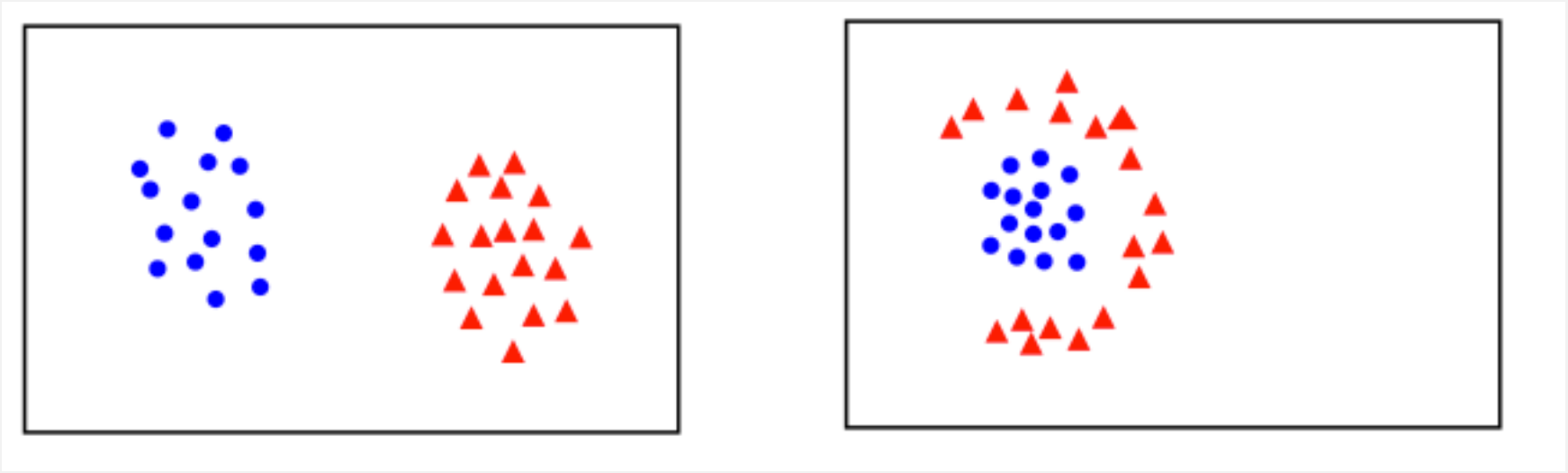


K=3

SVM

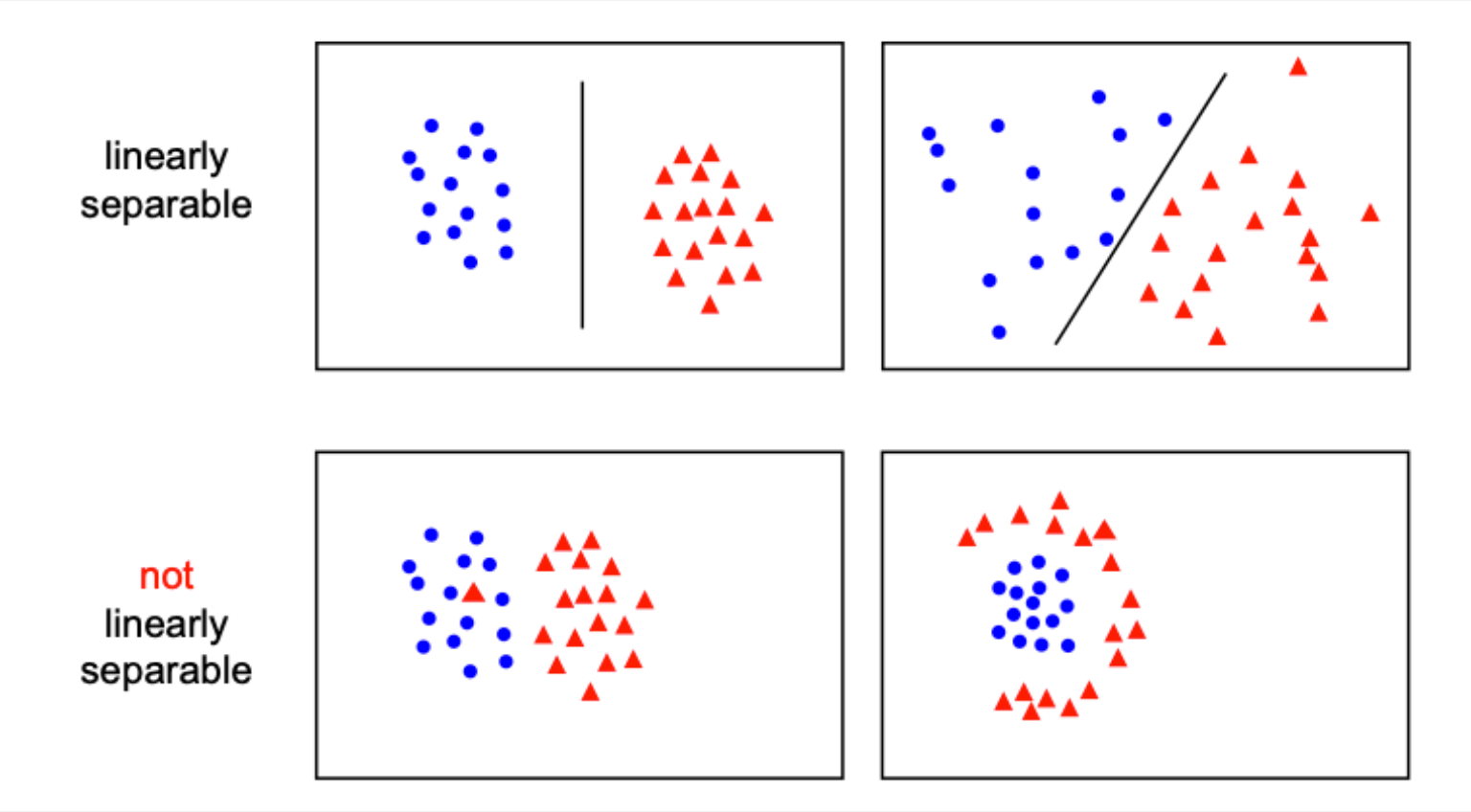
- Binary Classification
- Given training data (xi,yi) for i = 1...N, and yi ∈ {−1, 1}, learn a classifier f (x) such that

$$f(\mathbf{x}_i) \begin{cases} \geq 0 & y_i = +1 \\ < 0 & y_i = -1 \end{cases}$$



SVM

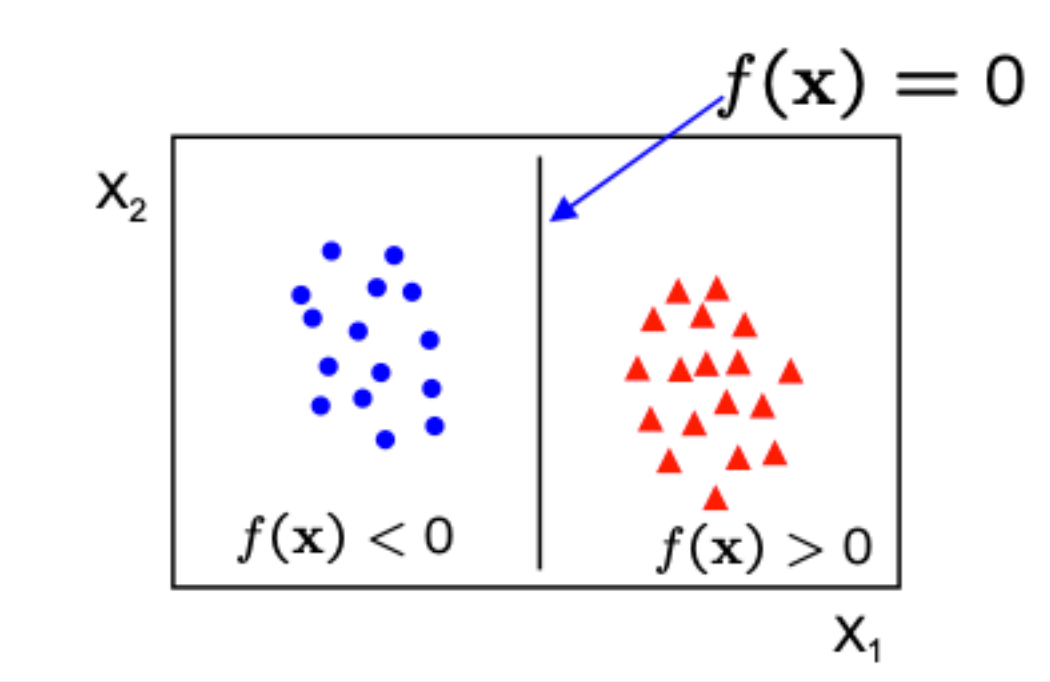
- Linear separability



SVM

- A linear classifier has the form

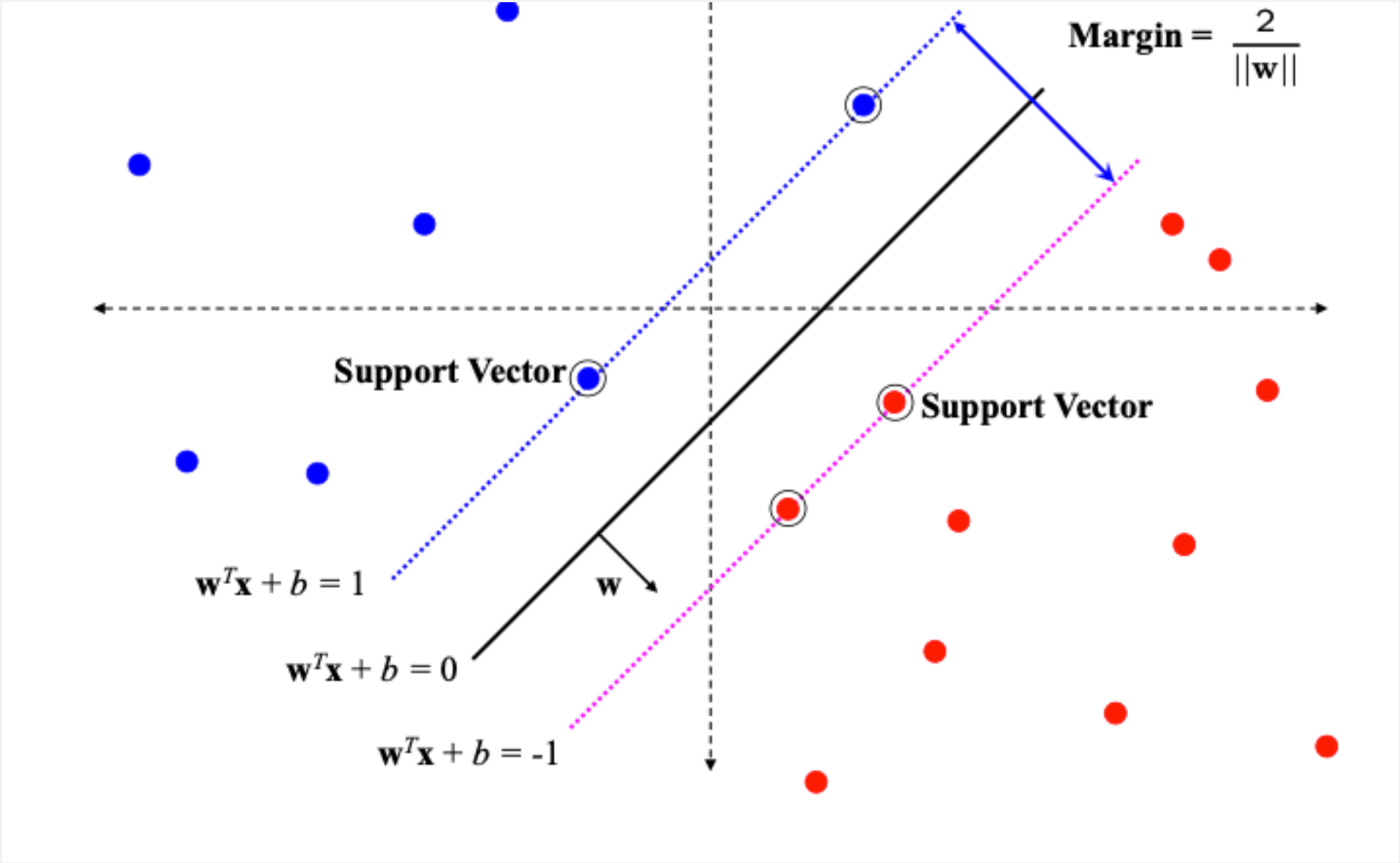
$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$$



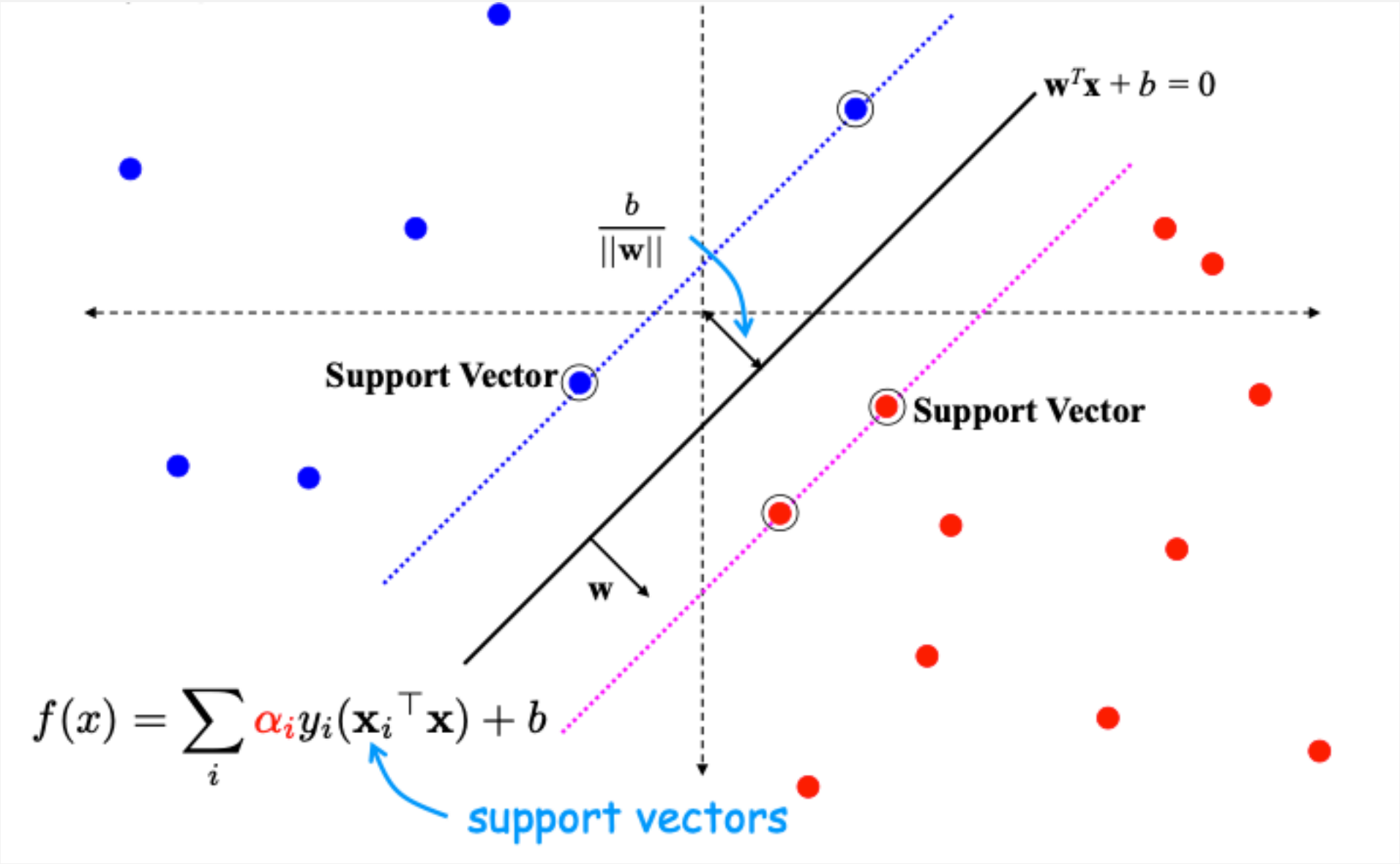
- **w** is the normal to the line, and **b** the bias
- **w** is known as the weight vector

- in 2D the discriminant is a line
- in 3D the discriminant is a plane, and in nD it is a hyperplane

SVM

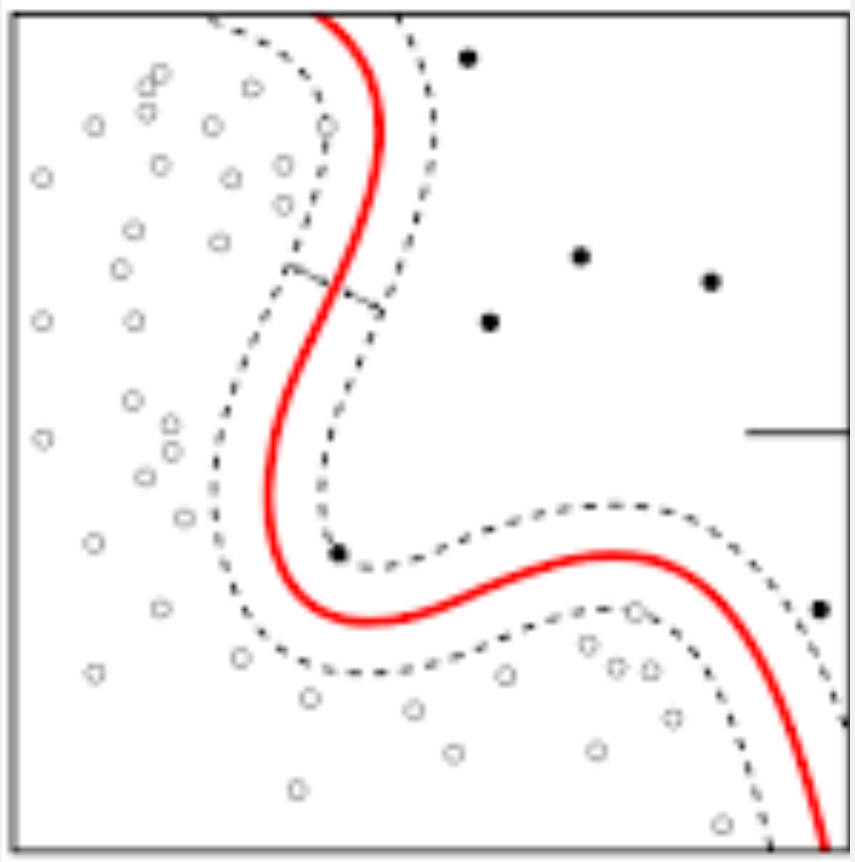

Margin = $\frac{2}{||\mathbf{w}||}$
Support Vector
Support Vector
$\mathbf{w}^T\mathbf{x} + b = 1$
$\mathbf{w}$
$\mathbf{w}^T\mathbf{x} + b = 0$
$\mathbf{w}^T\mathbf{x} + b = -1$

SVM

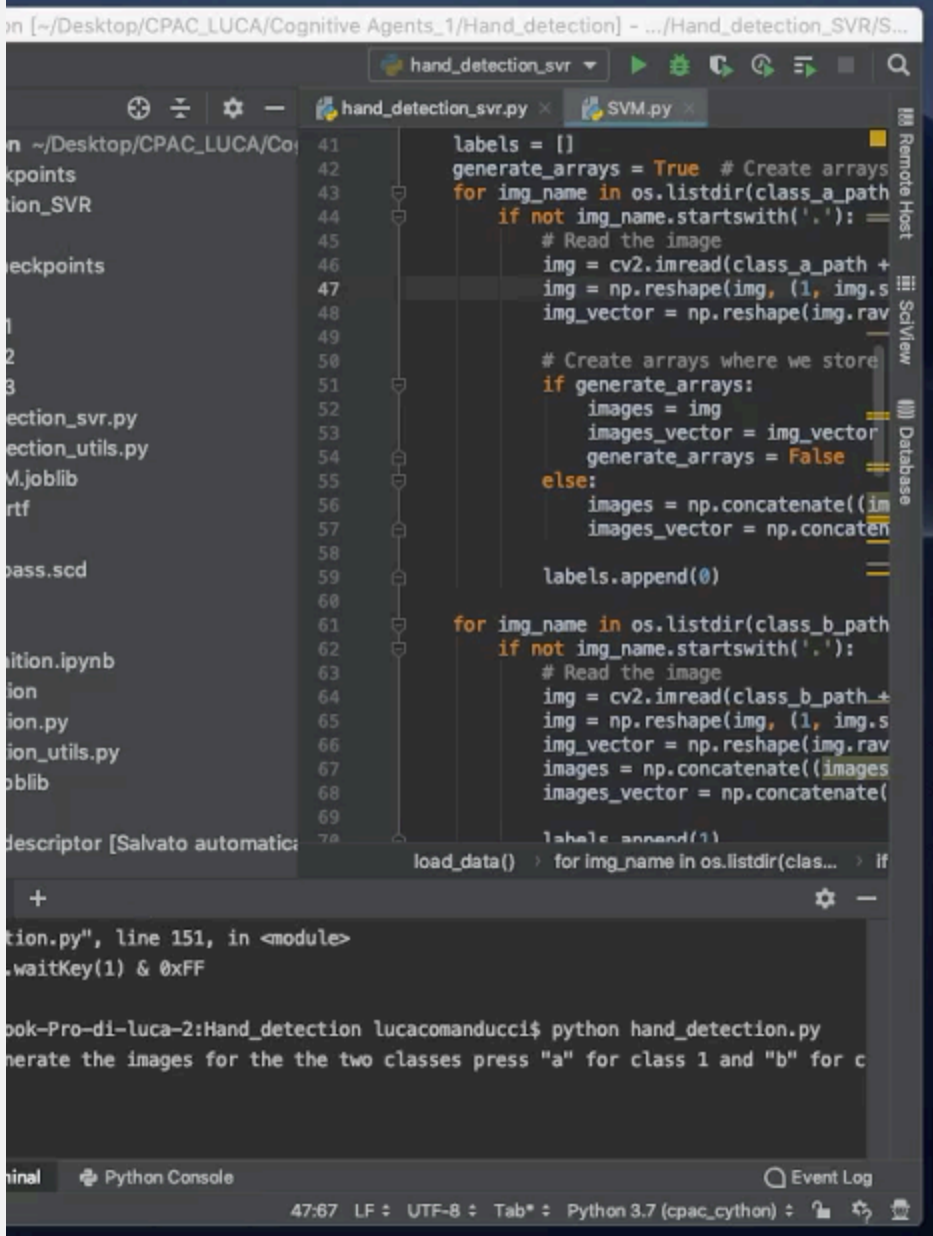

$\mathbf{w}^T\mathbf{x} + b = 0$
$\frac{b}{||\mathbf{w}||}$
Support Vector
Support Vector
$\mathbf{w}$
$f(x) = \sum_i \alpha_i y_i (\mathbf{x}_i{}^\top \mathbf{x}) + b$
support vectors

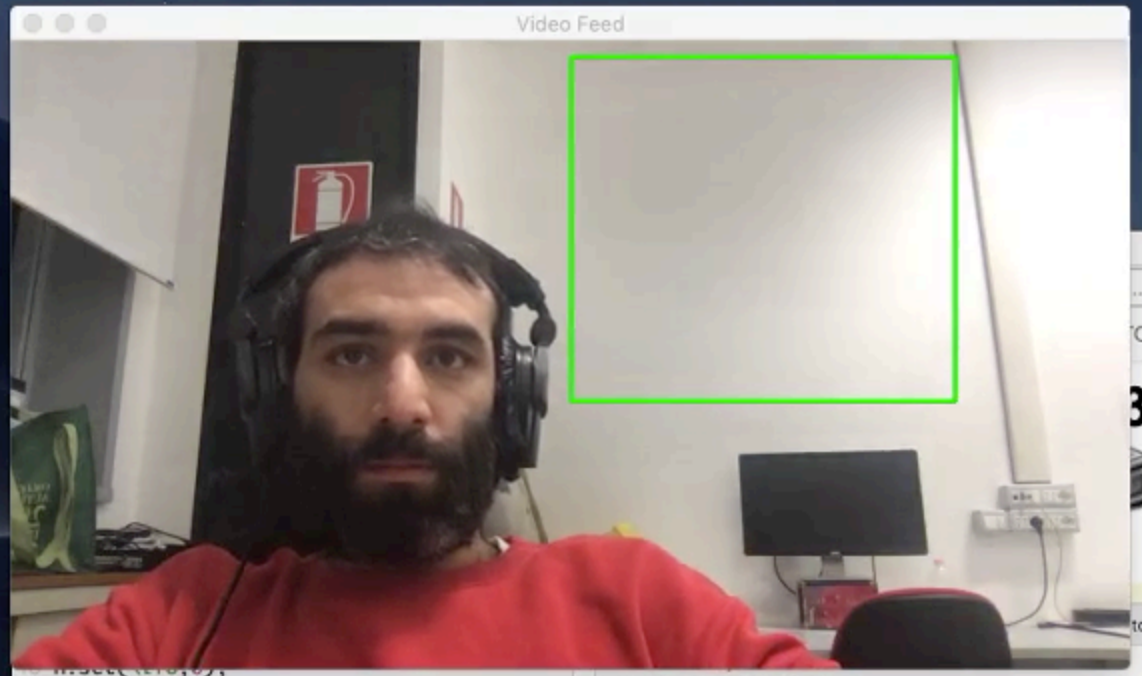
SVM

- Higher-order kernel



Wekinator - classification





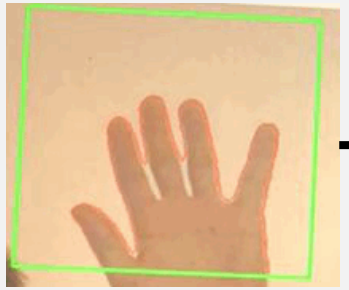

```
// start communication with hand_detector client


(
x = OSCFunc( { | msg, time, addr, port |
    var pyFreq,pyAmp,pyDetune,pyLfo;
    //( msg[0]).postln;
    if (msg[1] =='stop'){
        h.free
    }
    {
        // handle class A message (freq and amplitude)
        if (msg[1]=='a'){
            // Parse message
            pyFreq = msg[2].asFloat;
```

```
UGen(LinExp): 2280.82
UGen(LinExp): 1869.99
UGen(LinExp): 1189.25
UGen(LinExp): 586.669
UGen(LinExp): 480.996
UGen(LinExp): 305.897
UGen(LinExp): 237.28
UGen(LinExp): 123.721
UGen(LinExp): 78.6825
UGen(LinExp): 64.5099
UGen(LinExp): 41.0261
UGen(LinExp): 3977.92
UGen(LinExp): 2074.14
UGen(LinExp): 1608.89
UGen(LinExp): 1023.2
UGen(LinExp): 838.895
```

Interpreter: Active Server: 0.30% 0.61% 25u 1s 2g 107d 0.0dB

IDEA



Background removal & segmentation

PCA + SVM

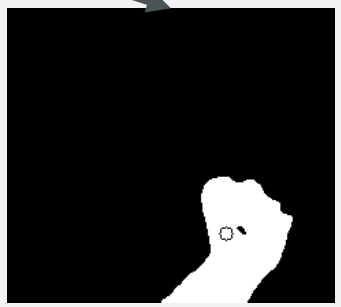
Palm Center Estimation

OSC

Supercollider

Training set Generation

Class A

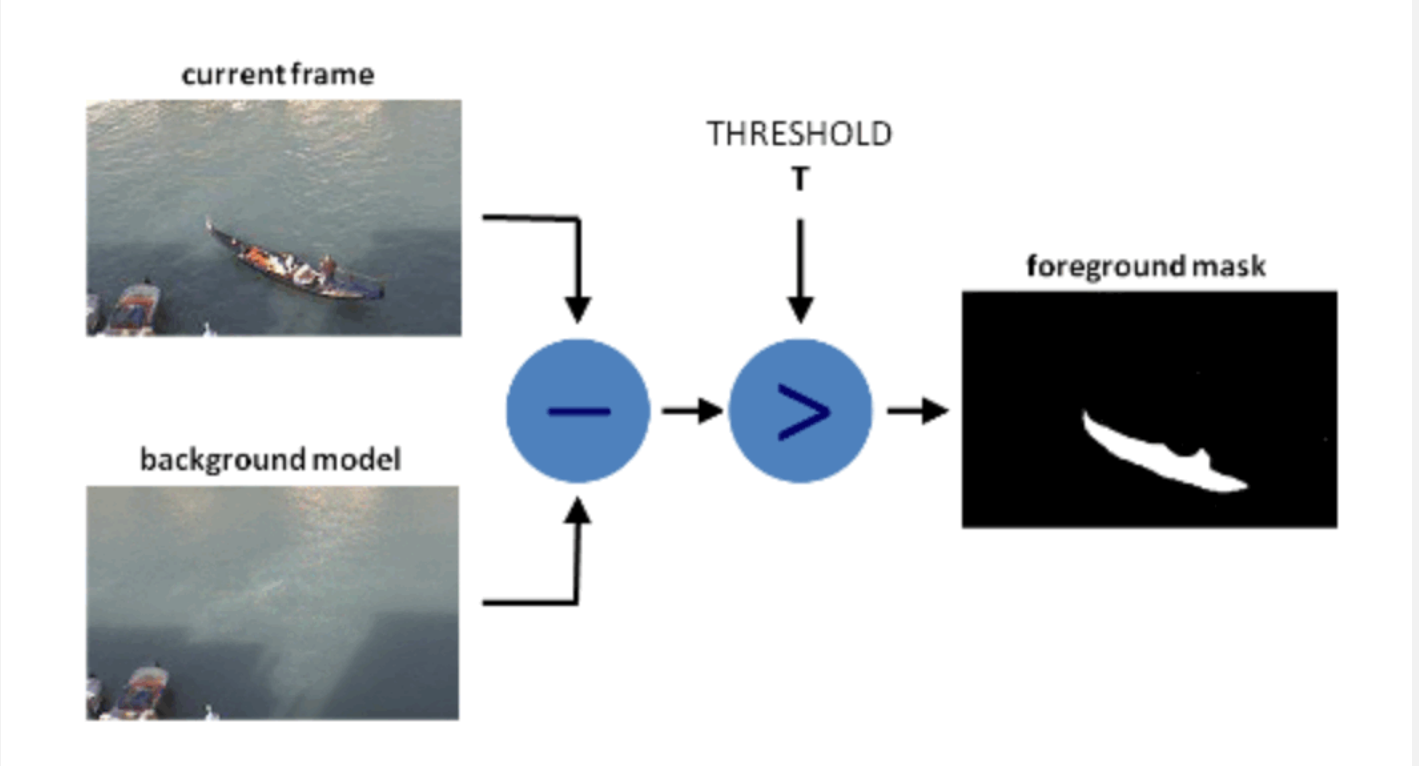


Class B

- Mapping
  - Class A:
    - x-> f0
    - y-> amplitude
  - Classe B:
    - x-> LFO
    - y->detuning

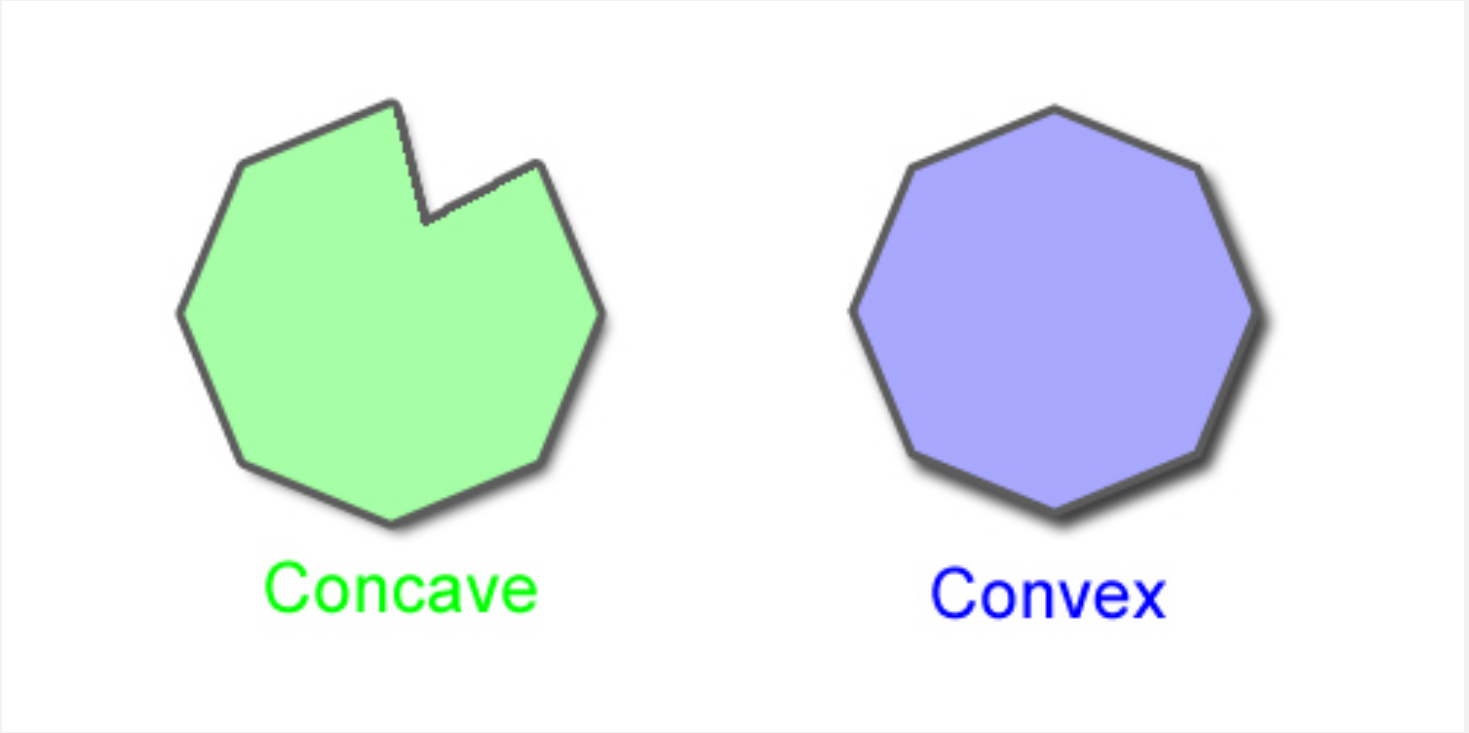
IDEA - BACKGROUND REMOVAL

- Running average over frames = background model
- Subtraction of background model from current frame



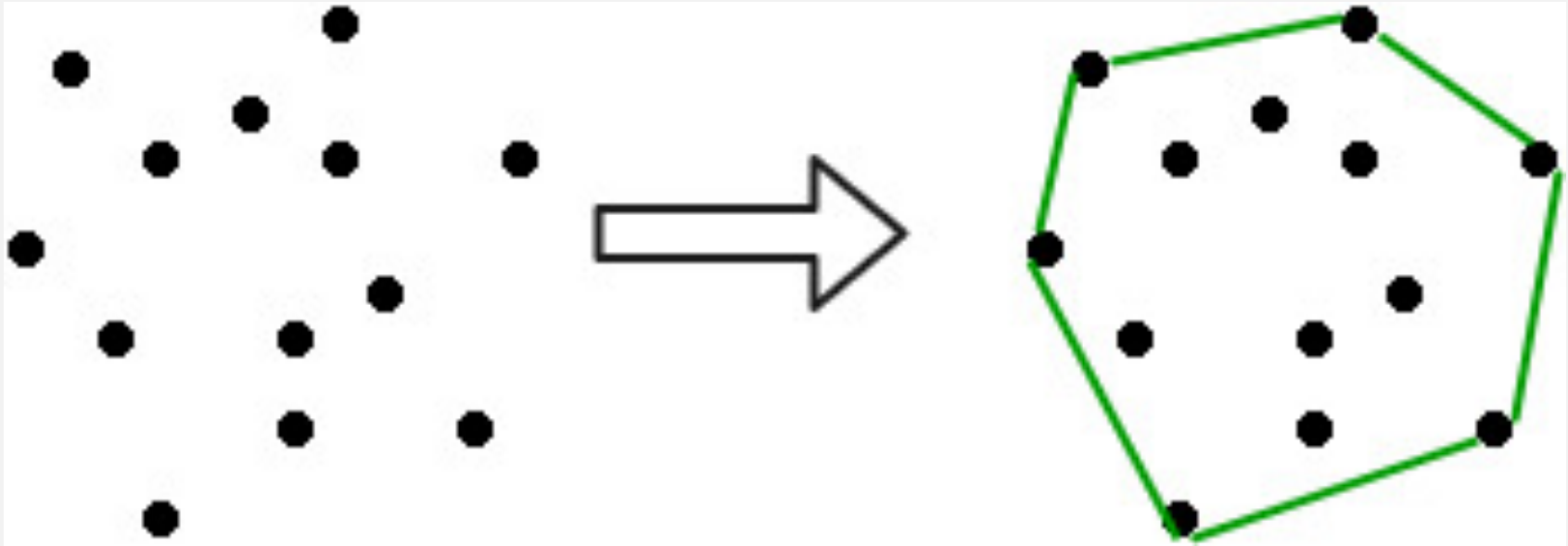
IDEA - PALM CENTER ESTIMATION

- Convex Polygon
  - A convex polygon is a simple polygon (not self-intersecting) in which no line segment between two points on the boundary ever goes outside the polygon



IDEA - PALM CENTER ESTIMATION

- Convex Hull
  - Given a set of point, the Convex hull is the Convex Polygon that is able to contain all the points

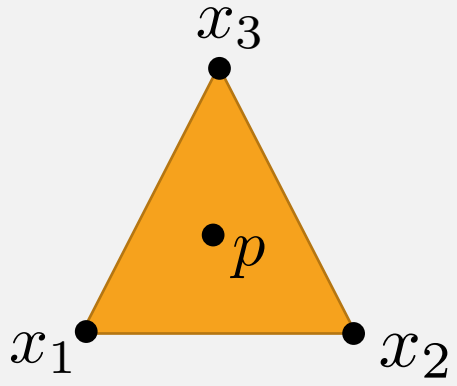


IDEA - PALM CENTER ESTIMATION

- Convex combination
  - linear combination of points where all coefficients are non-negative and sum up to 1

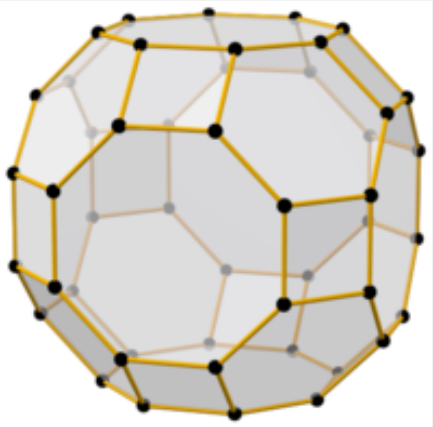
    i.e. given a finite number of points $x_1, x_2, \ldots, x_n$ in a real vector space, a convex combination p of points, is a point of the form

    $p = a_1x_1 + a_2x_2 + \ldots + a_nx_n$ where $a_i > 0$ and $a_1 + a_2 + \ldots + a_n = 1$



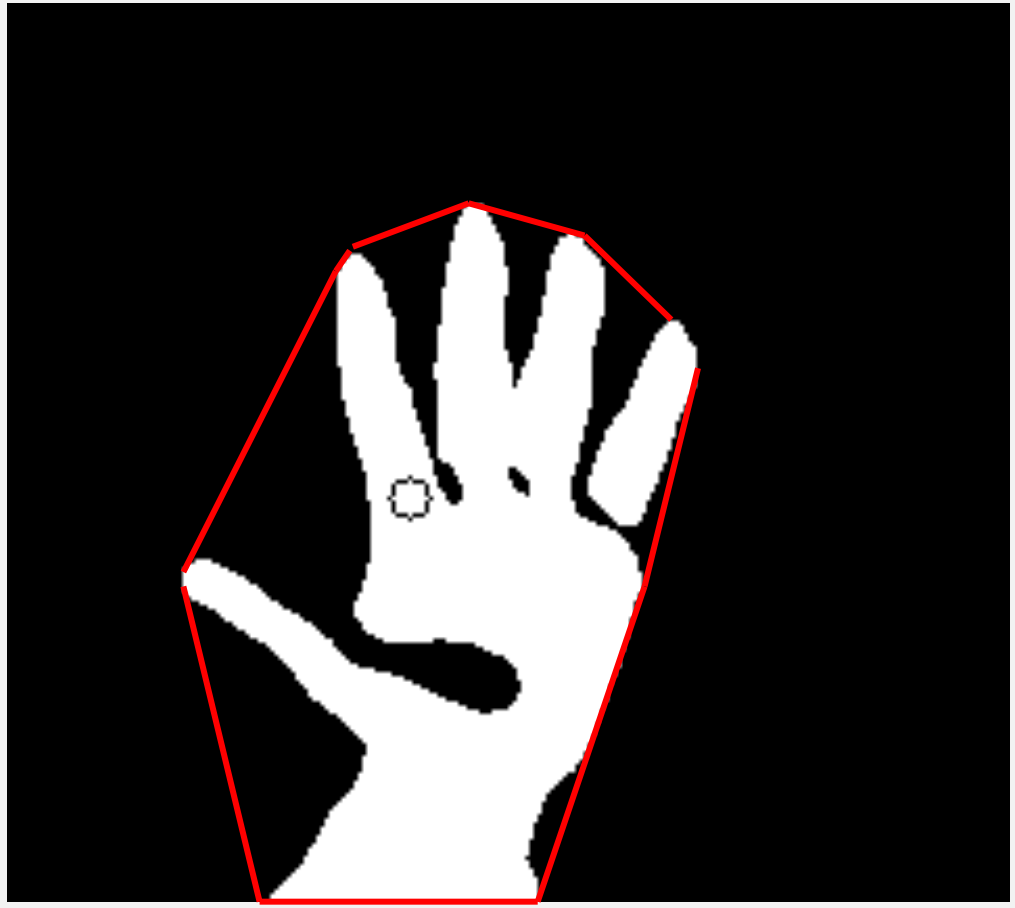
- Convex Hull
  - Convex hull of a finite point set S is the set of all convex combinations of its points

$$\mathrm{Conv}(S) = \left\{ \sum_{i=1}^{|S|} \alpha_i x_i \;\middle|\; (\forall i : \alpha_i \geq 0) \wedge \sum_{i=1}^{|S|} \alpha_i = 1 \right\}.$$



IDEA - PALM CENTER ESTIMATION

- Let's consider the white pixels as a set points
- We can apply Convex Hull to find the outline of the hand
- The center of the Convex Hull is than the estimated center of the palm

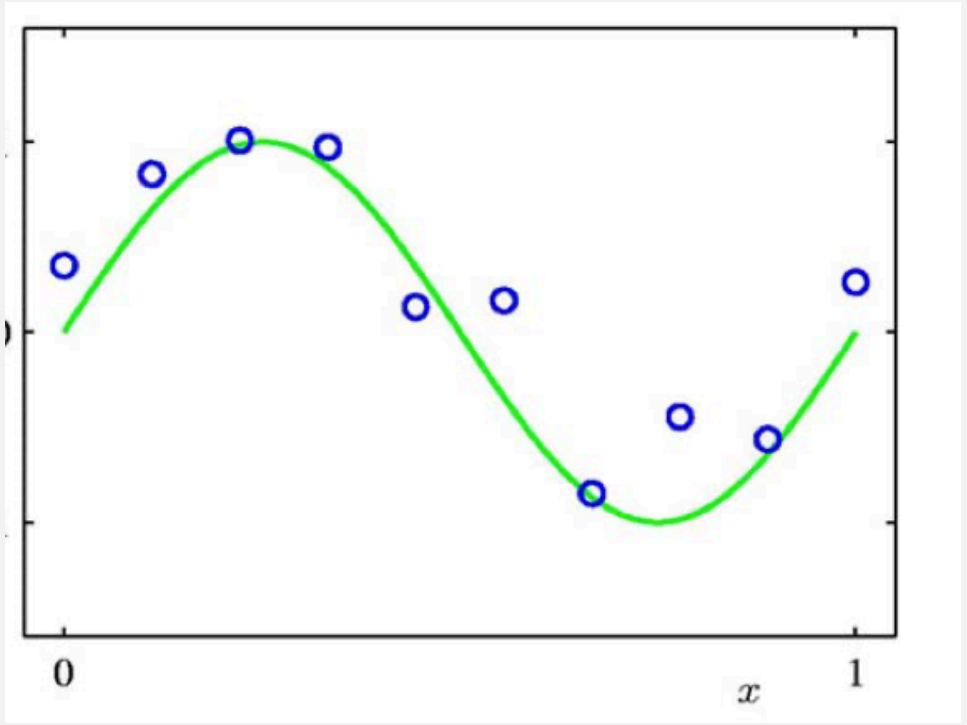


REGRESSION

- Suppose we are given a training set of N observations

(x1,...,xN) and (y1,...,yN)

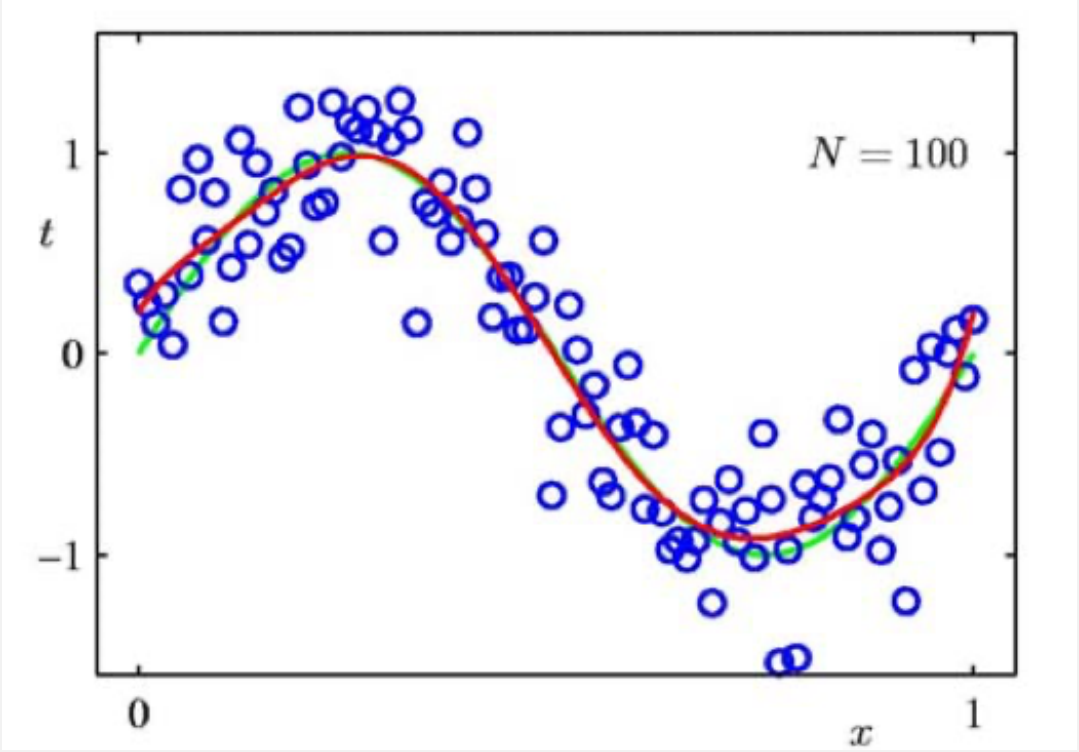
Regression problem is to estimate y(x) from this data



KNN REGRESSION

- For each test point, x, find the K nearest sample xi in the training data and their values yi
- The output will be

$$f(\mathbf{x}) = \frac{1}{K} \sum_{i=1}^{K} y_i$$



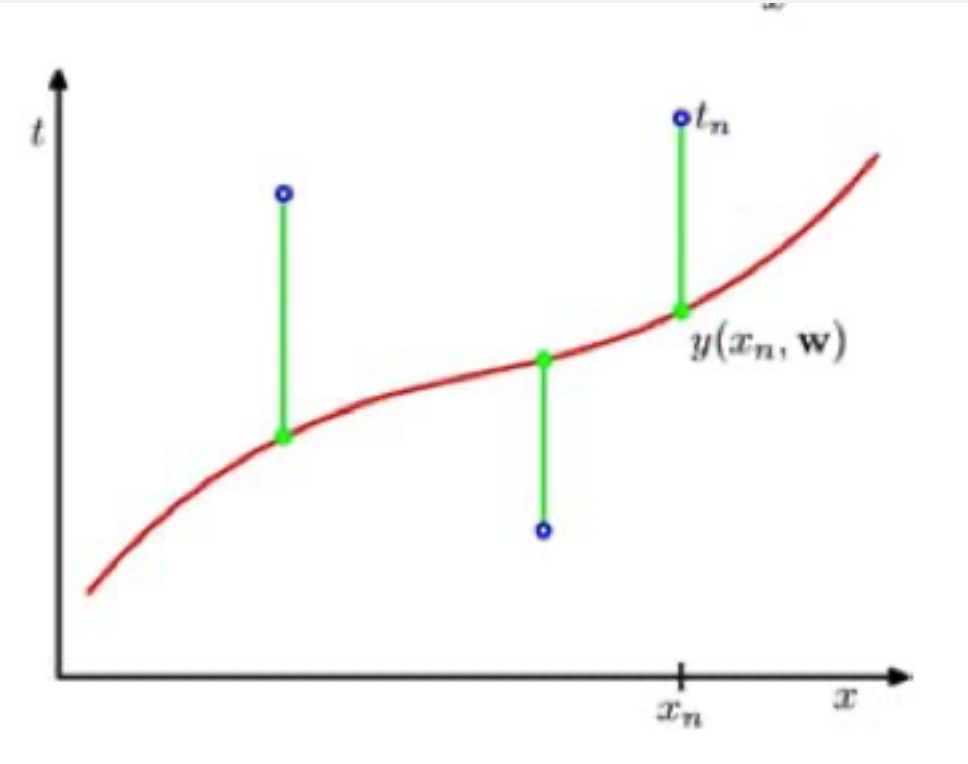
POLYNOMIAL CURVE FITTING

- Use of a loss function that measures the squared error in the prediction of y(x) from x. The loss for the red polynomial is the sum of the squared vertical errors

$$E(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}\{y(x_i, \mathbf{w}) - t_i\}^2$$

target value

$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$



Wekinator - classification

MATERIALS

- **References**
  - A. Rao, and M. George. BDI agents: from theory to practice, *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95) - 1995*
  - Roweis, Sam; Hinton, Geoffrey, *Stochastic neighbor embedding*, Neural Information Processing Systems (2002)
  - F. Camastra, A. Vinciarelli, Machine Learning for Audio, Image and Video Analysis, Springer, 2015 – Chapter 4, 6, 9, 11
  - S.A.A. Shah, H.M. Shabbir, S.U. Rhman, M, Waqas, A Comparative Study of Feature Selection Approaches: 2016-2020, International Journal of Scientific & Engineering Research Volume 11, Issue 2, February-2020
  - https://www.analyticsvidhya.com/blog/2016/12/introduction-to-feature-selection-methods-with-an-example-or-how-to-select-the-right-variables/

- **Further readings**
  - Russell - Artificial Intelligence a modern approach, 2012 – Chapter 7, 12
  - Van der Maaten, L.J.P.; Hinton, G.E. (Nov 2008). "Visualizing Data Using t-SNE", *Journal of Machine Learning Research*. **9**