

Big data science

Day 2

F. Legger - INFN Torino

<https://github.com/Course-bigDataAndML/MLCourse-2425>

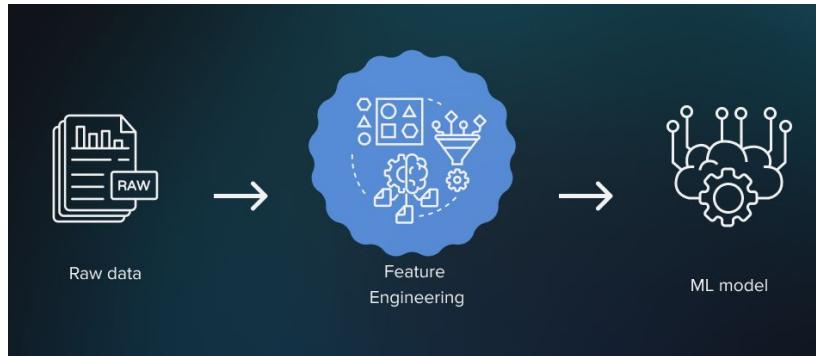
- So long, and thanks for all the images!
 - Taken freely from the web
 - Credits go to original creators



Yesterday

- Big data
- Analytics
- Distributed computing
- Heterogeneous computing

Today



- **Machine learning**
 - ML models
 - Model training
 - Feature engineering

A PROPOSAL FOR THE
DARTMOUTH SUMMER RESEARCH PROJECT
ON ARTIFICIAL INTELLIGENCE

J. McCarthy, Dartmouth College
M. L. Minsky, Harvard University
N. Rochester, I.B.M. Corporation
C. E. Shannon, Bell Telephone Laboratories

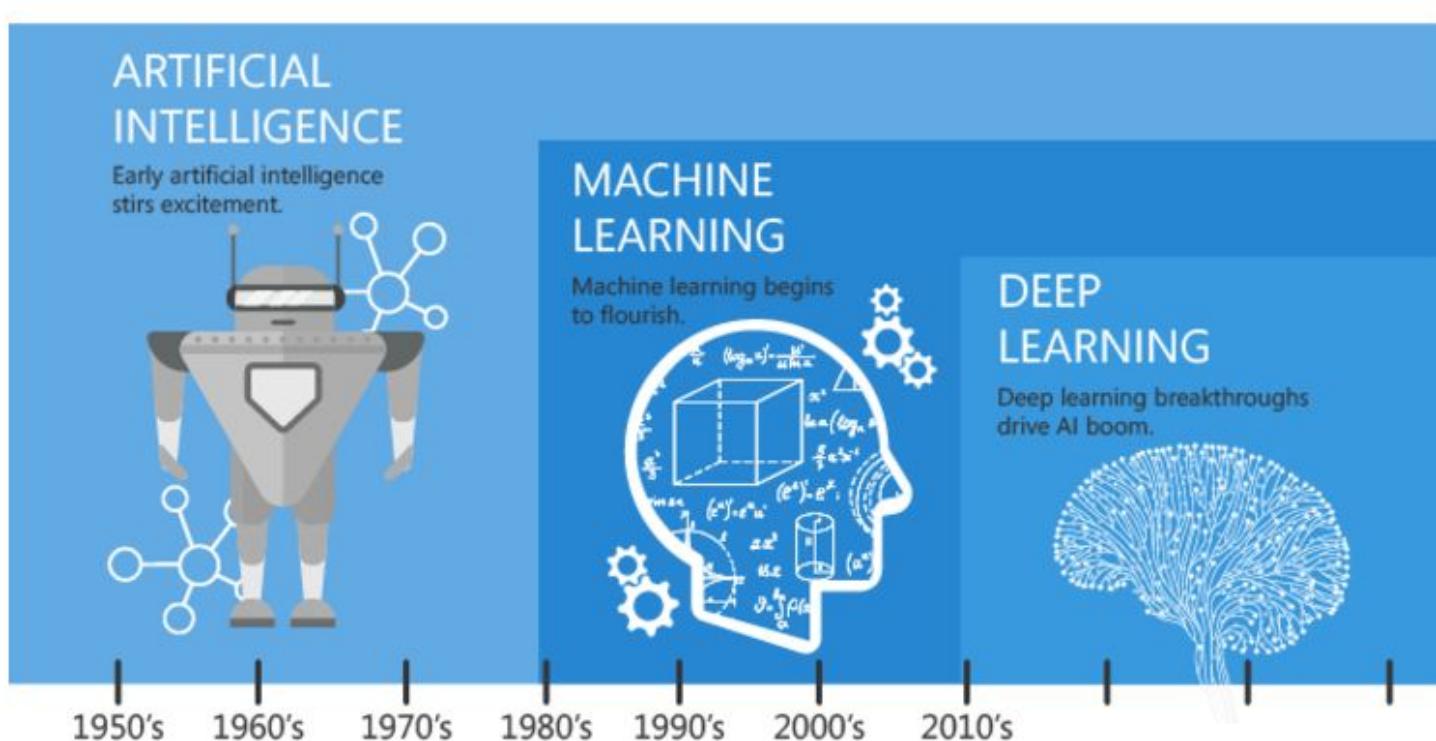
August 31, 1955

Our ultimate objective is to make programs that learn from their experience as effectively as humans do

[John McCarthy, 1958]



AI >> ML >> DL



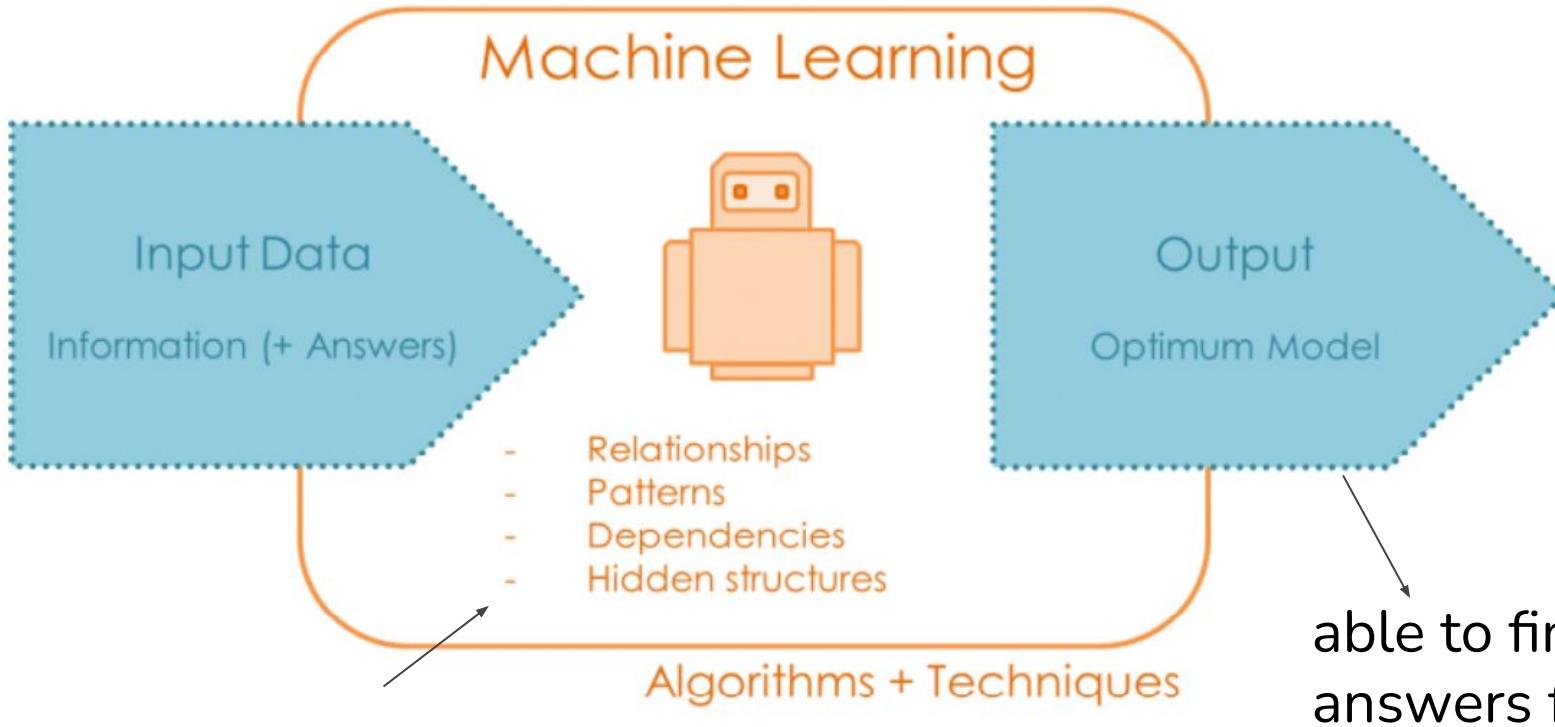
Machine Learning

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 task in T , as measured by P , improves with experience E

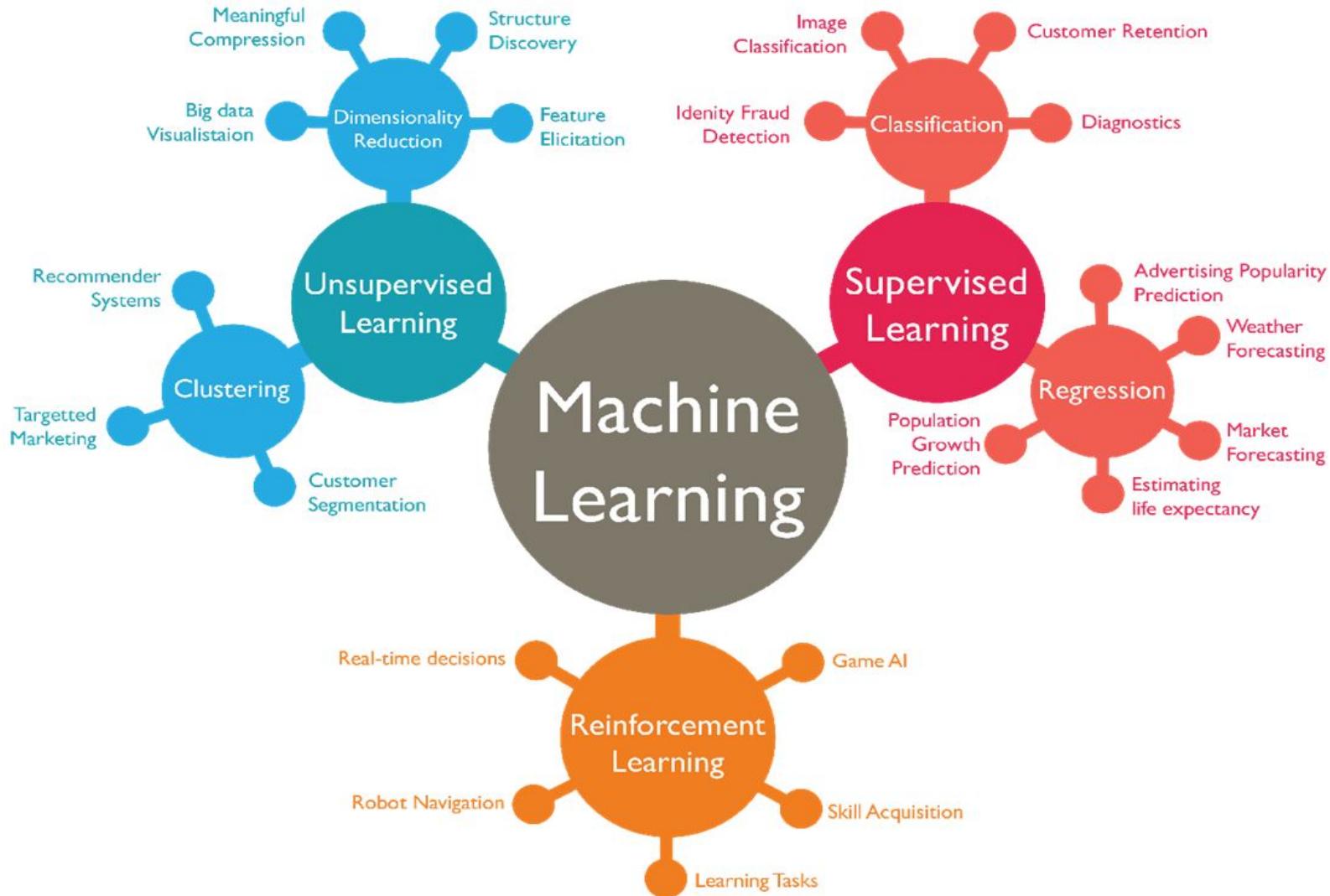
[Tom Mitchell, 1997]

Machine Learning is the science of getting computers to act without being explicitly programmed

[Andrew Ng]



able to find answers from new data



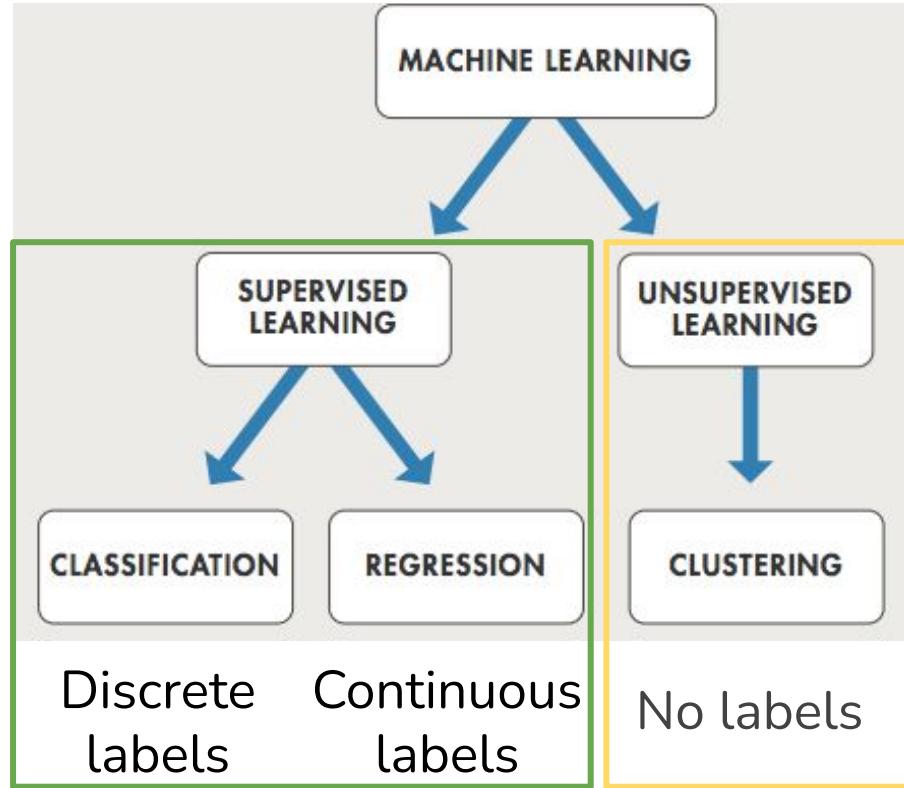
Supervised learning

Are input data labelled?

Additional annotations
about the data

Features								Label
date	lat	long	temp	humidity	cloud_coverage	wind_direction	atmp_pressure	rainfall
2021-09-09	49.71N	82.16W	74	20	3	N	18.6	.01
2021-09-09	32.71N	117.16W	82	42	6	SW	29.94	.23

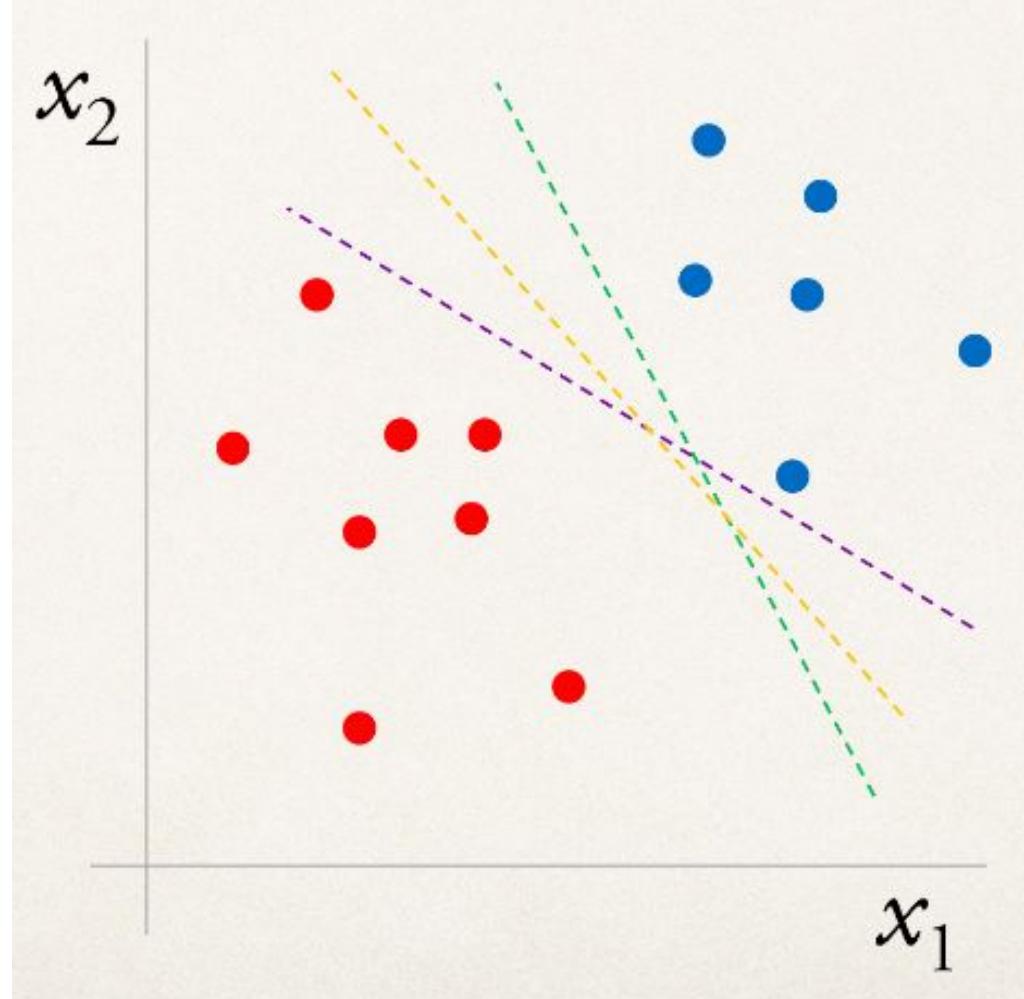
Example



Classification

Supervised, discrete labels

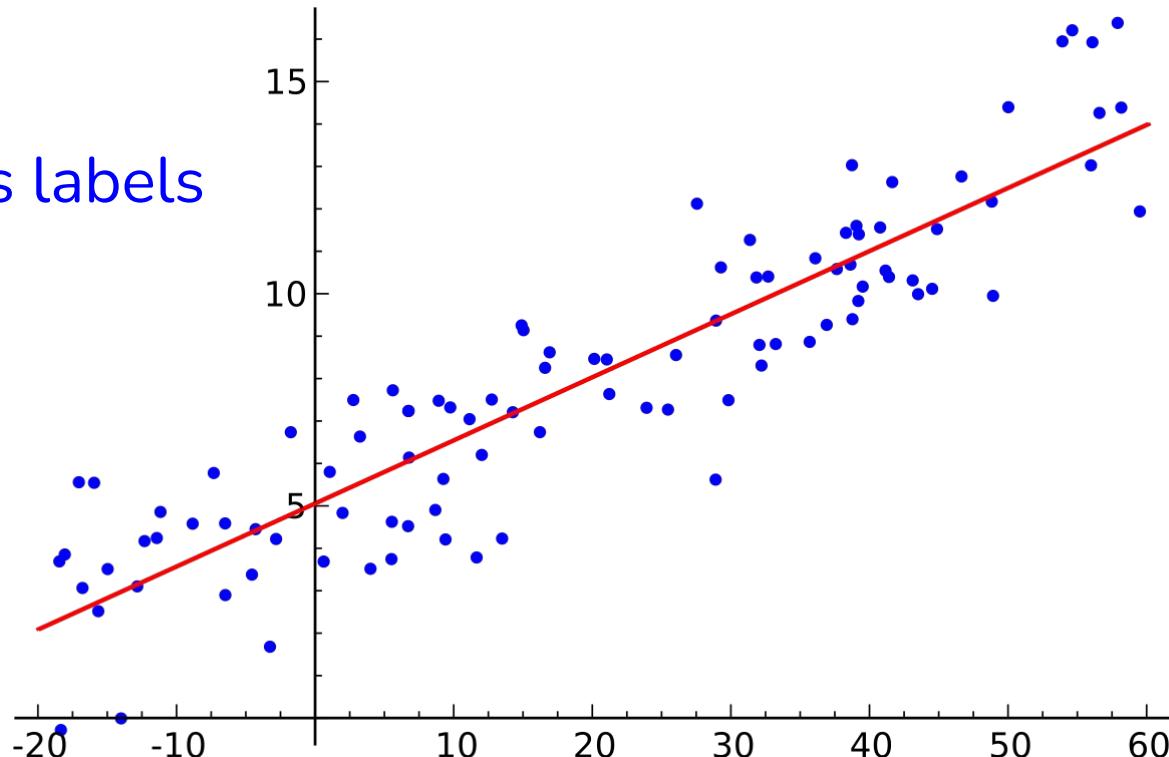
- Predict **one or more** output class
 - Businesses who target customers: good vs bad, stay or leave
 - **Signal vs background**



Regression

Supervised, continuous labels

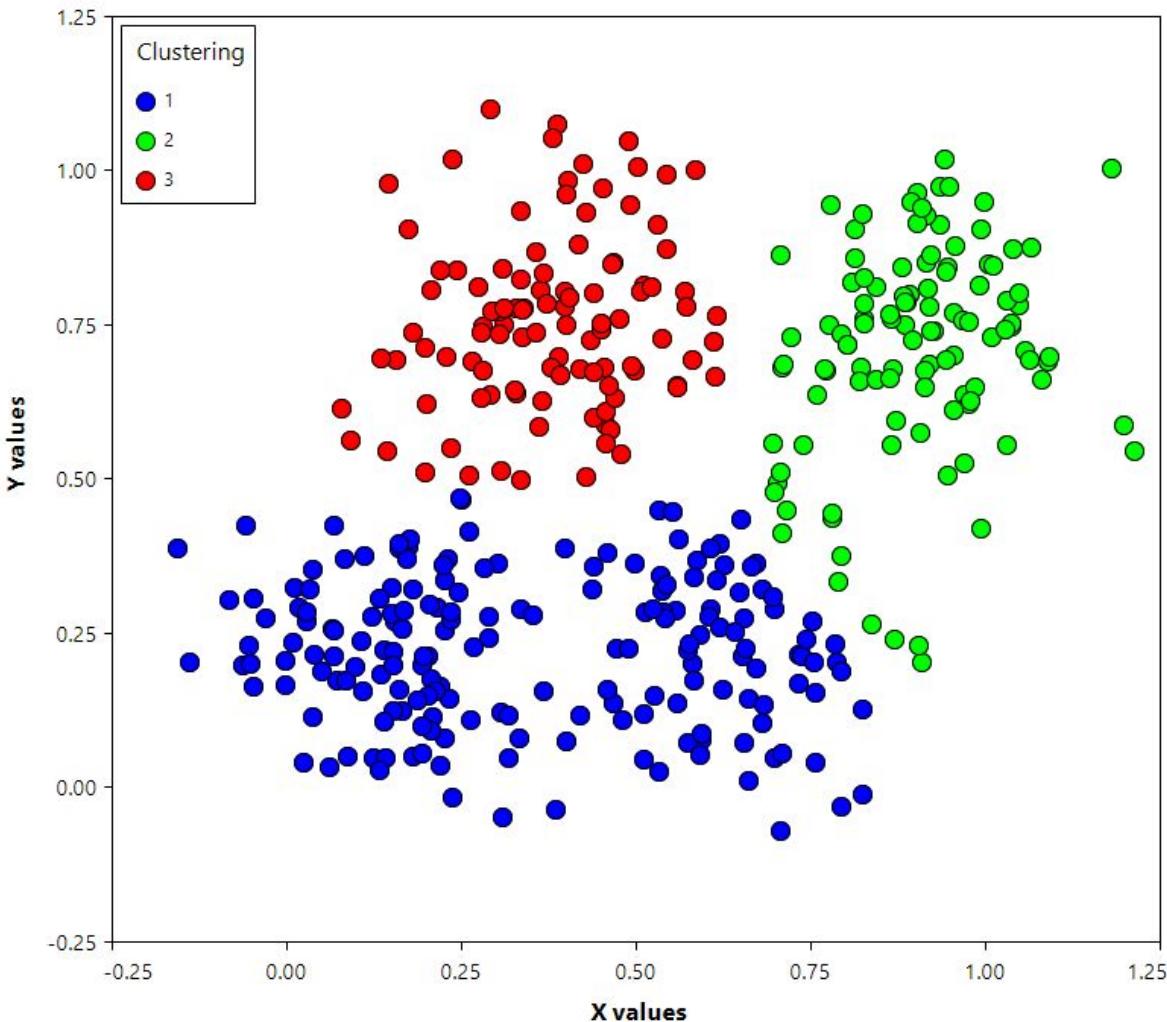
- Predict a trend:
 - Businesses who predict customer behavior: e.g. house prices, ...



Clustering

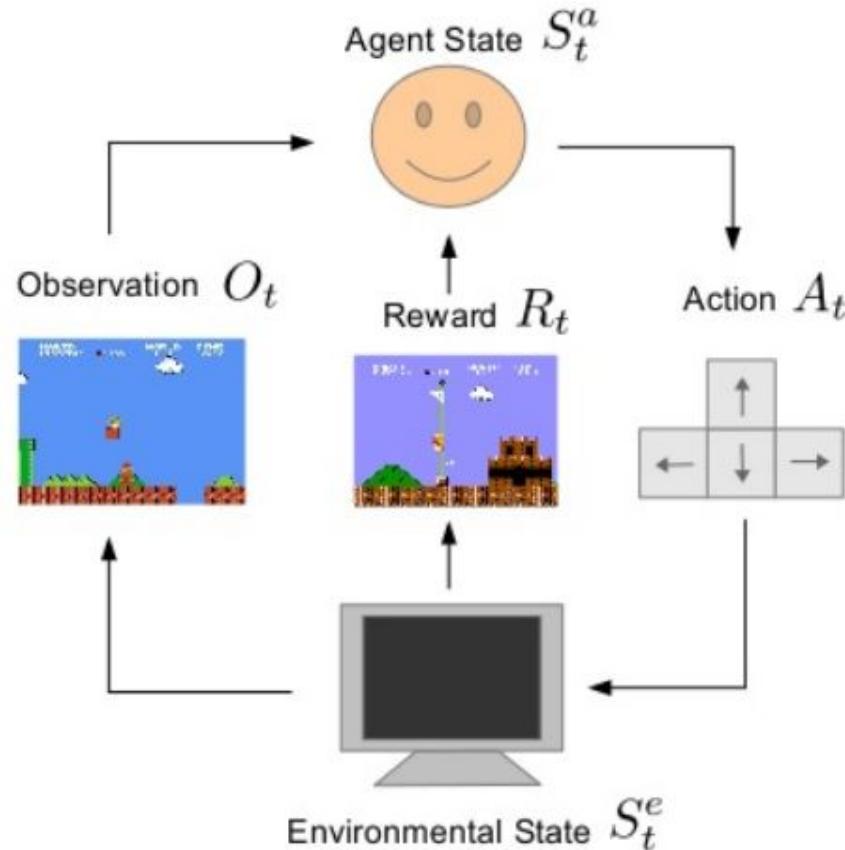
Unsupervised

- Businesses who identify customer categories
- Light vs heavy flavour jets



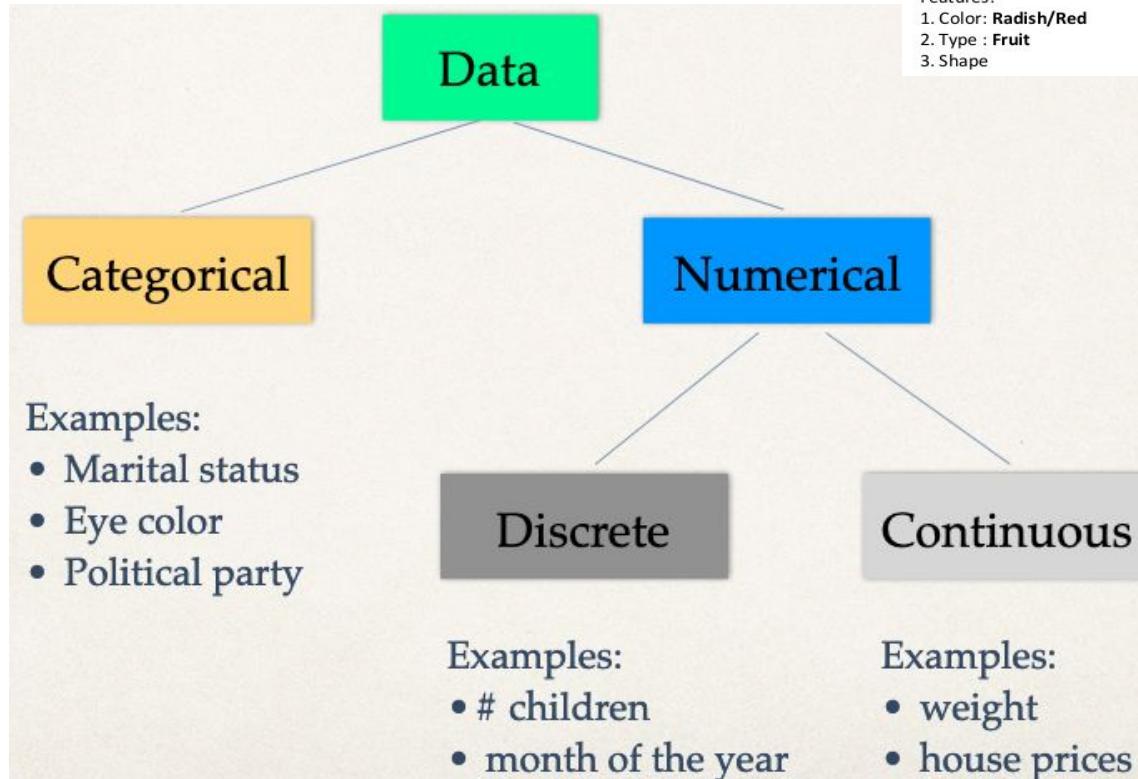
Reinforcement learning

- getting an agent to act in the world so as to maximize its rewards
- sparse and time delayed labels (rewards)



Data types

Typically
strings

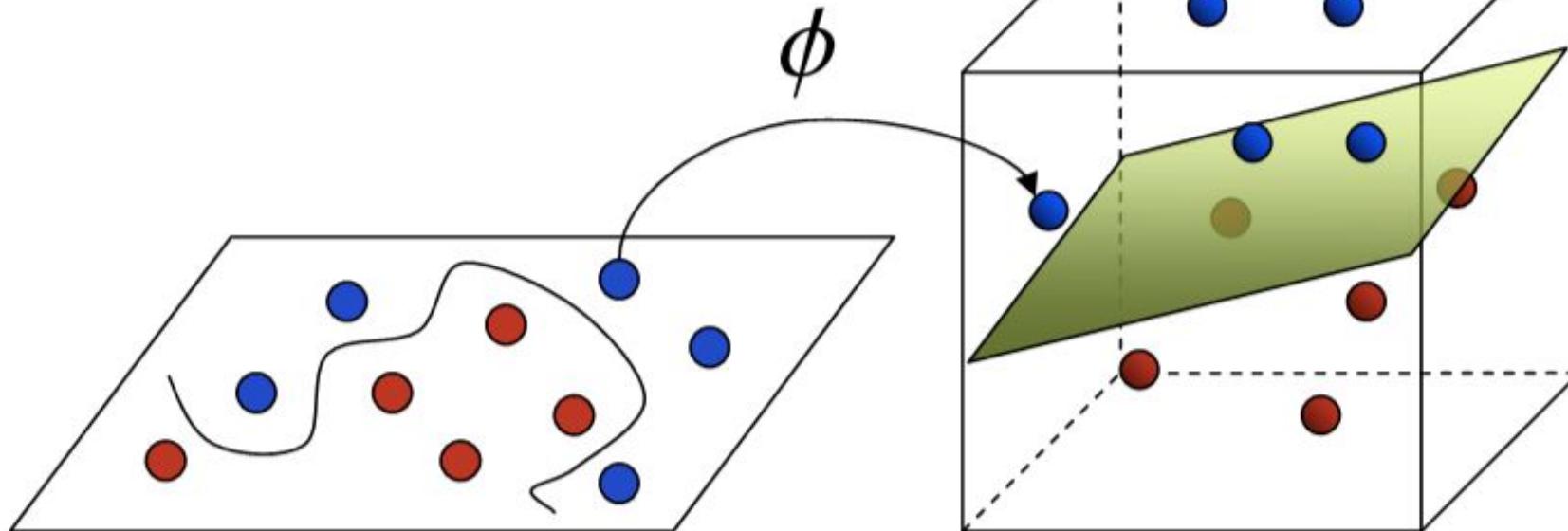


Features:
1. Color: Radish/Red
2. Type : Fruit
3. Shape

Features:
1. Sky Blue
2. Logo
3. Shape

Features:
1. Yellow
2. Fruit
3. Shape

First step



Input Space

Raw data

Feature Space

Preprocessing

Feature engineering

Raw Data

```
0 : {  
    house_info : {  
        num_rooms: 6  
        num_bedrooms: 3  
        street_name: "Shorebird Way"  
        num_basement_rooms: -1  
    }  
    ...  
}
```

Feature Engineering

Raw data doesn't come to us as feature vectors.

Feature Vector

```
[  
    6.0,  
    1.0,  
    0.0,  
    0.0,  
    0.0,  
    9.321,  
    -2.20,  
    1.01,  
    0.0,  
    ...  
]
```

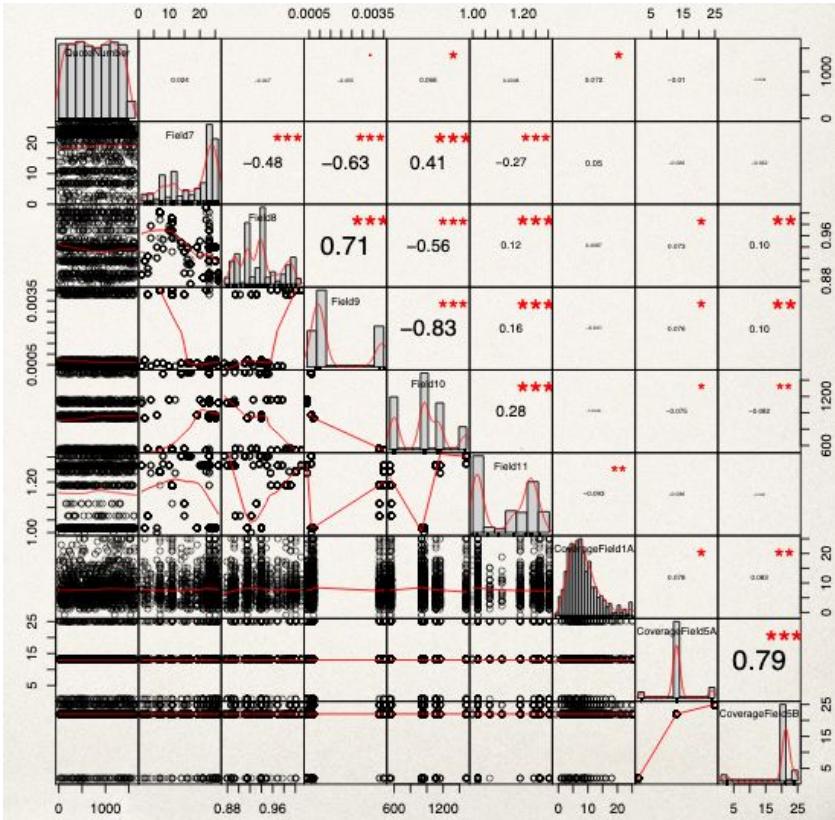
Process of creating features from raw data is **feature engineering**.

Data preprocessing

- Most of the time will be spent in this step
- Data clean-up, data transformation, feature engineering
 - **data transformation**
 - scaling and normalization
 - encoding, aggregation features, log-transformation
 - remove outliers
 - **data visualization, exploration**
 - **data augmentation, bucketing, binning, ...**
 - **dimensionality reduction**
- Your programming skills will be required here: **R, Python, ...**

Data visualization

- **Graphical representation** may reveal important features of the data
 - find correlations, identify range, etc.
- **Identify features** which may require transformations
 - outliers or skewness (asymmetry in probability distribution) in data
- It helps to identify a strategy how to deal with different features



Data transformation

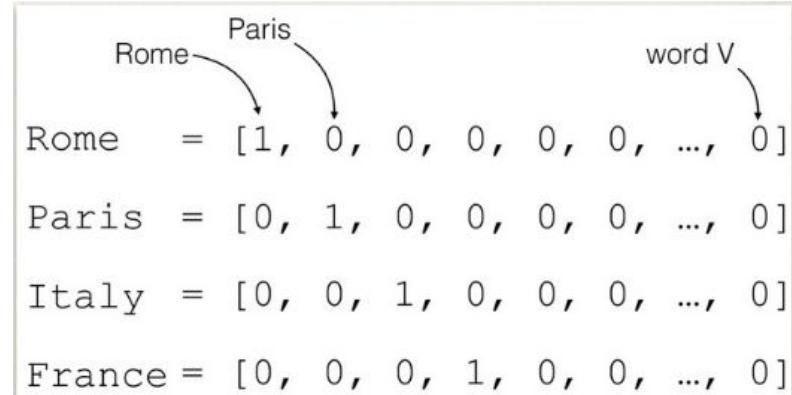
- **Data transformation and aggregation:** log, sum of values, average
- **Scaling:** scale data to a given range [0,1] or any other range
- **Normalization/Standardization:** a technique to scale data to mean with zero and unit-variance
- **Augmentation:** a technique to create additional data based on input sample which slightly differ from it, e.g. image rotation, flip, scale, crop, etc.
- **Bucketing/Binning:** a technique to place similar values into buckets/bins

$$x' = \frac{x - \bar{x}}{\sigma}$$

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

One-hot encoding

- Technique to handle **categorical** data
- “One-Hot” refers to a state in electrical engineering where all of the bits in a circuit are 0, except a single bit with a value of 1 (“hot”)
- It represents a **categorical column as a vector of words**
- You need to define the word vector for the full set of data (train + test datasets)
 - Issues with NULL or missing data
 - delete rows with missing data
 - input data for missing values
 - Problematic with high cardinality



Leave-one-out encoding

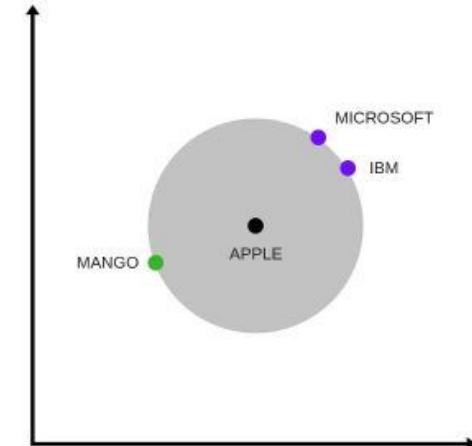
- Effective by **high cardinality**
- Y is what we want to predict (label)
- Encode UserID:
 - **Train dataset:**
 - Take mean of Y's for all rows with same UserID except the one you want to encode
 - multiply random noise
 - **Test dataset:**
 - If there is no Y (label), just use frequency of UserID

Split	UserID	Y	mean_y	random	newID
Train	A1	0	0.667	1.05	0.70035
Train	A1	1	0.333	0.97	0.32301
Train	A1	1	0.333	0.98	0.32634
Train	A1	0	0.667	1.02	0.68034
Test	A1	-	0.5	1	0.5
Test	A1	-	0.5	1	0.5
Train	A2	0			

Mean of
[1,1,0]
random
mean_y*

Word embedding

- A way to capture multi-dimensional relationships between categories by using semantics
- Use neural networks or other ML algorithms to train the model to find the best representation of embedded variables
- **Word2vec** based on neural networks
 - **Continuous Bag of words (CBOW)**: predicts the probability of a word given a context
 - **Skip-Gram model**: predicts the context given a word



Frequency-based word embedding

- Count Vector

- Corpus C of D documents $\{d_1, d_2, \dots, d_D\}$ and N unique tokens (words) in C
- The N tokens will form our dictionary and the size of the Count Vector matrix M will be given by $D \times N$. Each row in the matrix M contains the frequency of tokens in $D(i)$

- TF-IDF Vector

- Similar to Count vector, but frequency is calculated with respect to all documents

- Co-Occurrence Vector

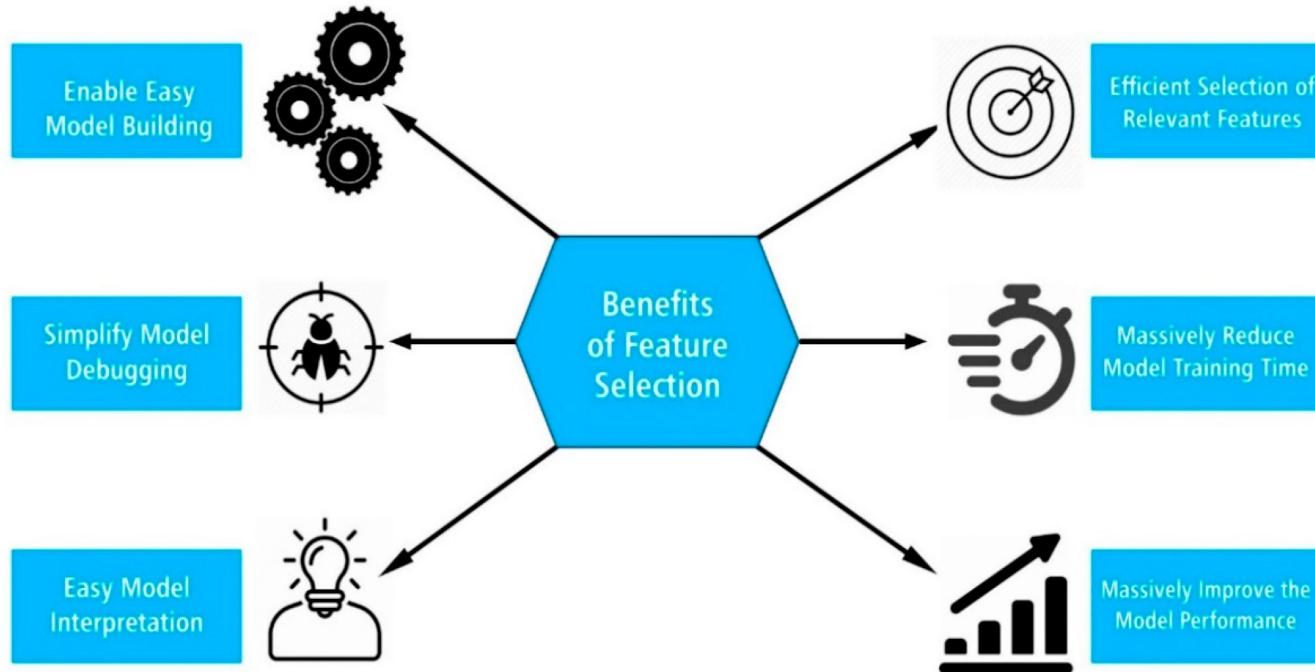
- Based on frequency of words appearing together (for example, it is)

	Document 1	Document 2	Document 3	Document 4	Document 5	Document 6	Document 7	Document 8
Term(s) 1	10	0	1	0	0	0	0	2
Term(s) 2	0	2	0	0	0	18	0	2
Term(s) 3	0	0	0	0	0	0	0	2
Term(s) 4	6	0	0	4	6	0	0	0
Term(s) 5	0	0	0	0	0	0	0	2
Term(s) 6	0	0	1	0	0	1	0	0
Term(s) 7	0	1	8	0	0	0	0	0
Term(s) 8	0	0	0	0	0	3	0	0

↑
Document Vector

Feature selection

- Low vs high level variables

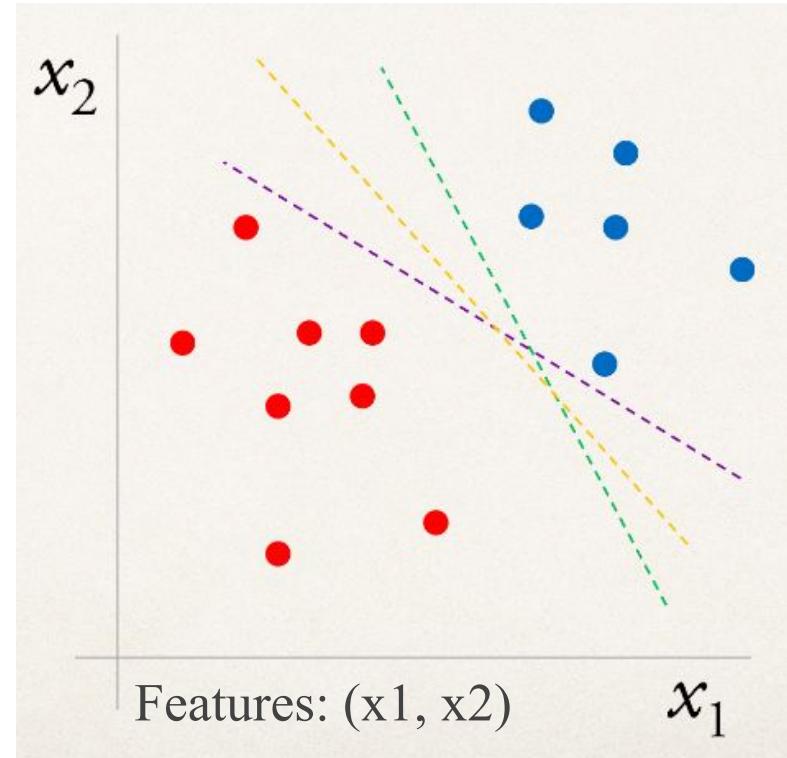


Model training

Example: supervised classification

Ingredients

- **Inputs:** X , is a matrix of size:
 - n (number of data points) \times m (number of features)
- **Features:** X , transformed inputs, matrix $n \times m$
- **Labels/Targets:** y , vector size n



Recipe: supervised classification

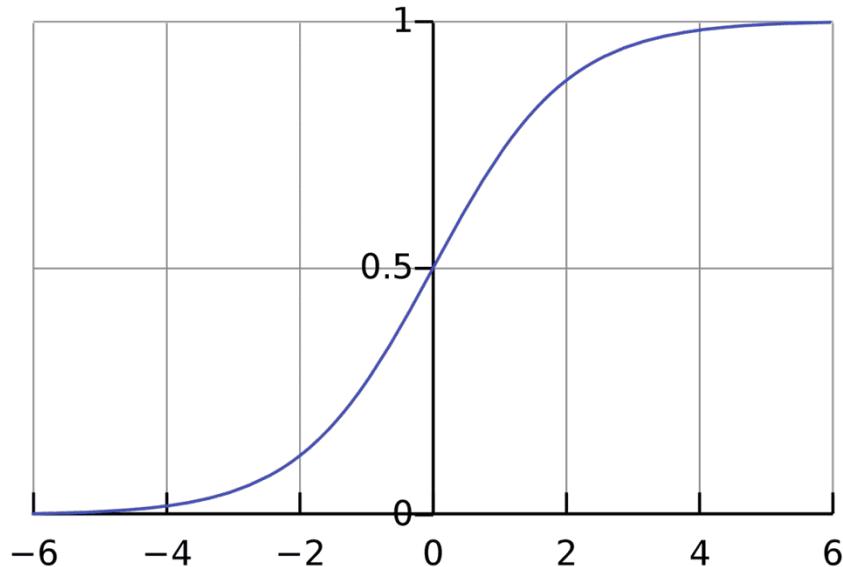
- For each input vector X_i predict z_i
 - z_i yields (0,1)
 - $z_i = \varphi(W^T X_i)$, $i=1\dots n$
 - **Weights:** W (matrix) contains the model parameters
 - **Activation function:** φ
 - step function, sigmoid



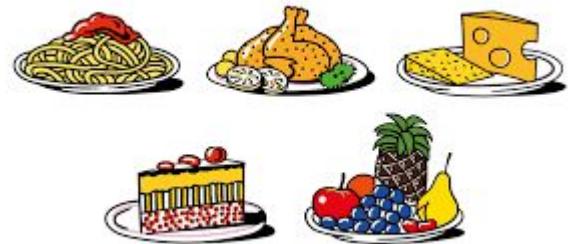
Activation function

- Turns unbounded output into a known range/shape
- For example, **sigmoid** function outputs numbers in the range (0, 1)
 - big negative numbers become ~0
 - big positive numbers become ~1.

$$S(x) = \frac{1}{1 + e^{-x}} = \frac{e^x}{e^x + 1}.$$



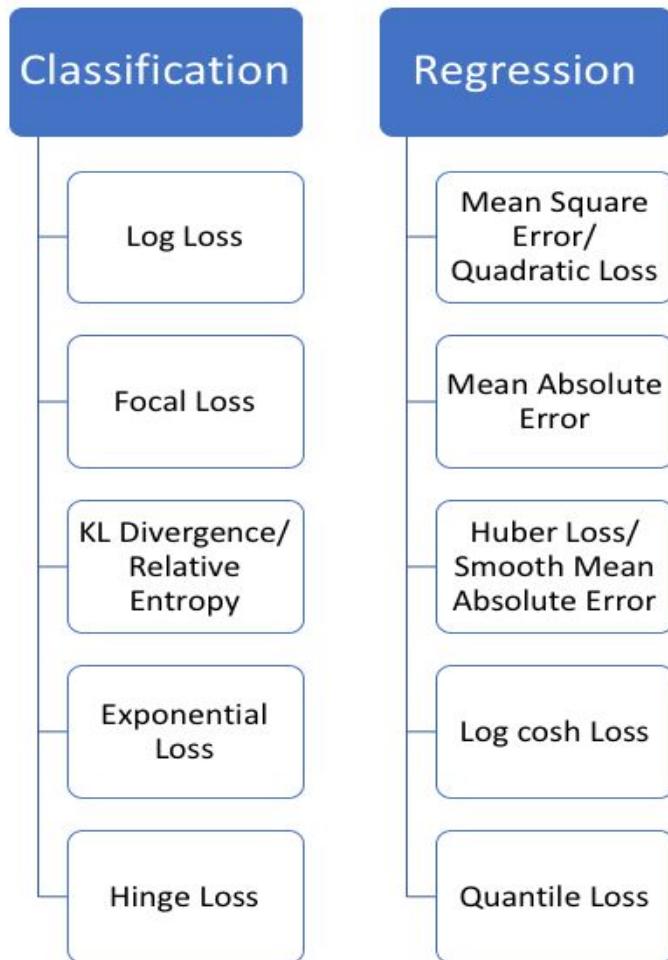
Model training: the cookpot



- This is the actual learning phase
 - Define the **loss function L** == **cost function** == **prediction error**, function of the model parameters **W**
 - The **loss L** quantifies how well the model learns by measuring the difference between the predicted values z_i and the actual values y_i
- **Aim of the training:** find the weights **W** that minimize the loss **L**

Loss functions

- <https://heartbeat.fritz.ai/5-regression-loss-functions-machine-learners-should-know-4fb140e9d4b0>
- https://www.wikiwand.com/en/Loss_functions_for_classification



Example: linear regression

- Inputs (features): x_i
- Labels: y_i
- Model: $y = a + bx$
- Weight+bias: a, b
 - parameters to be found
- Cost function: **Mean Square Error (MSE)** $MSE = \frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2$.
- No activation function:
problem is linear

