UNIVERSITÀ DEGLI STUDI DI SALERNO
UNIVERSITÀ DEGLI STUDI DI SALERNO
DIPARTIMENTO DI INFORMATICA
DIPARTIMENTO DI ECCELLENZA

Carmen Bisogni, PhD Student

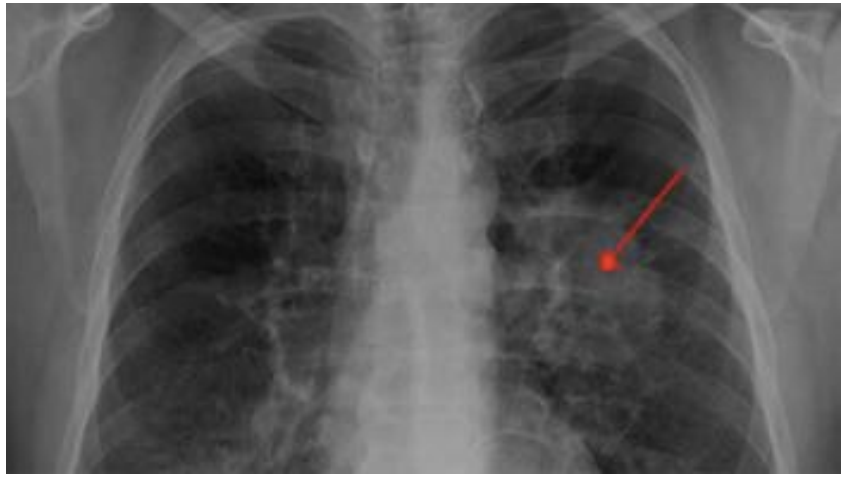
C.A.S.A. Course

CONTEXT AWARE SECURITY ANALYTICS IN COMPUTER VISION
Lesson 2

Classification: Why?

Let's say you are trying to write a machine learning program that will be able to detect cancerous tumors in lungs.

- Input: images of lung x-rays .
- Output: "yes" if there is a tumor; "no" if there isn't tumors.
- Training: several images of x-rays, half of the images contain tumors and are labelled "yes" and the other half do not contain tumors and are labelled "no."

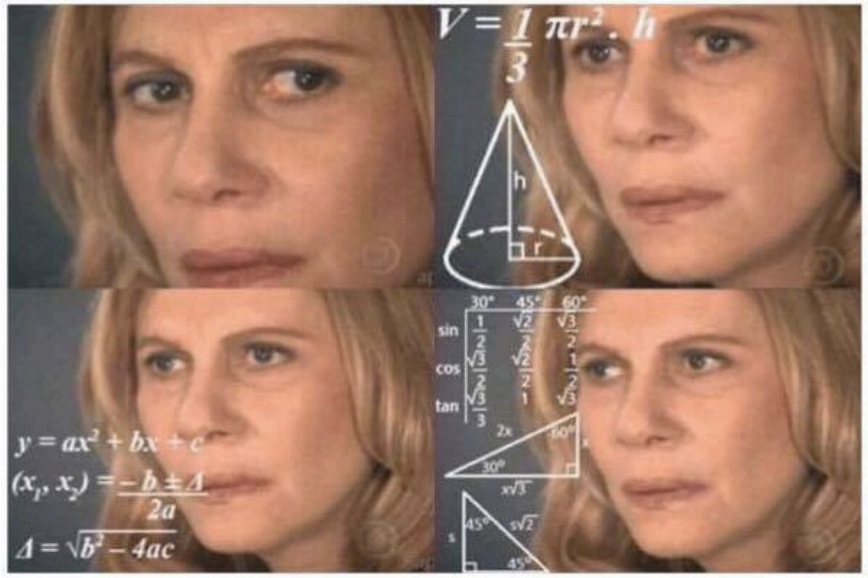


If the algorithm learns how to identify tumors with high accuracy, it could save doctors time by analyzing x-ray images quickly.

Classification: Example

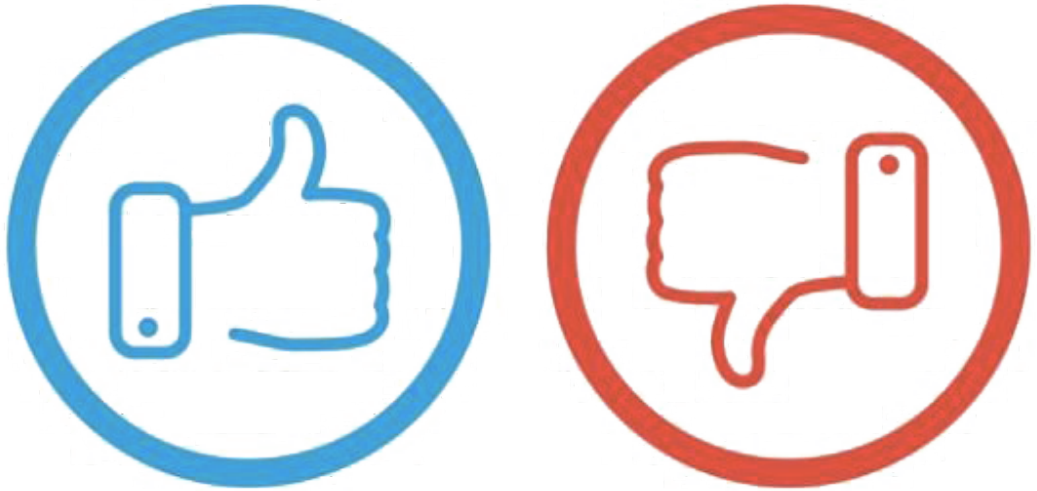
Let's say you are writing a program that recommends new music to a user based on their music preferences. The user selects from a list of songs the songs that he or she likes and the program should show him or her new songs that they are likely to enjoy.

In this case, what is the input training data?
What are the labels?



Classification: Example

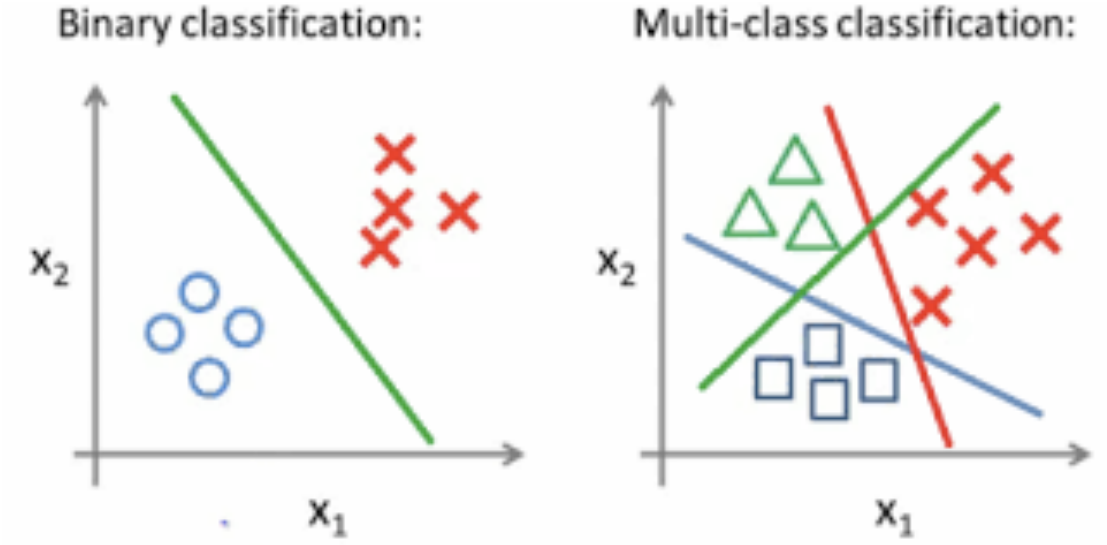
The input data would be the songs that the user selected that they liked, this could be labelled as “like” and the songs that the user did not select from the list are labelled as “dislike.”



Classification

There are two main types of classification problems:

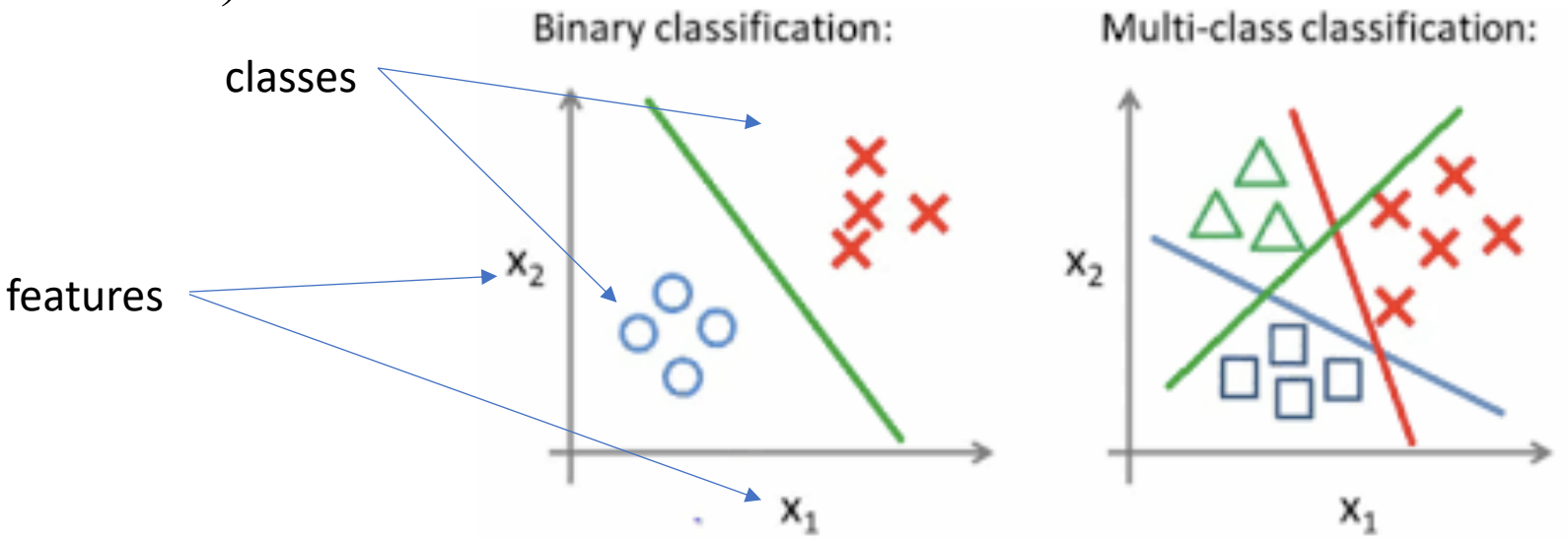
- Binary classification: Like the previous example.
- Multi-class classification: Like handwritten character recognition (where classes go from 0 to 9).



Classification

There are two main types of classification problems:

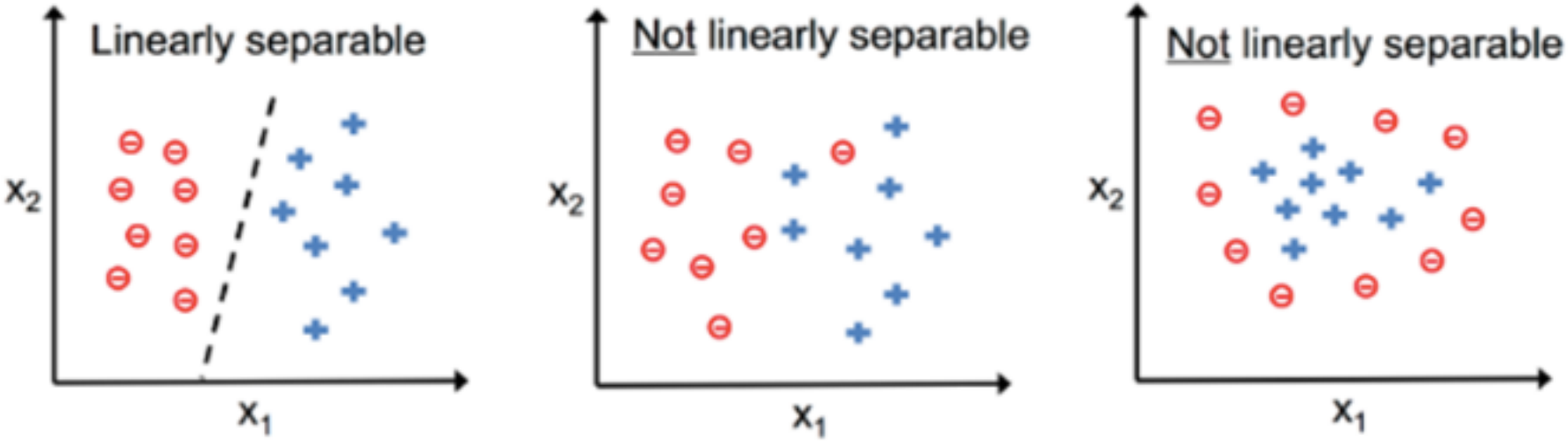
- Binary classification: Like the previous example.
- Multi-class classification: Like handwritten character recognition (where classes go from 0 to 9).



Classification

Every classification models will be useful to separate properly different classes from a dataset.

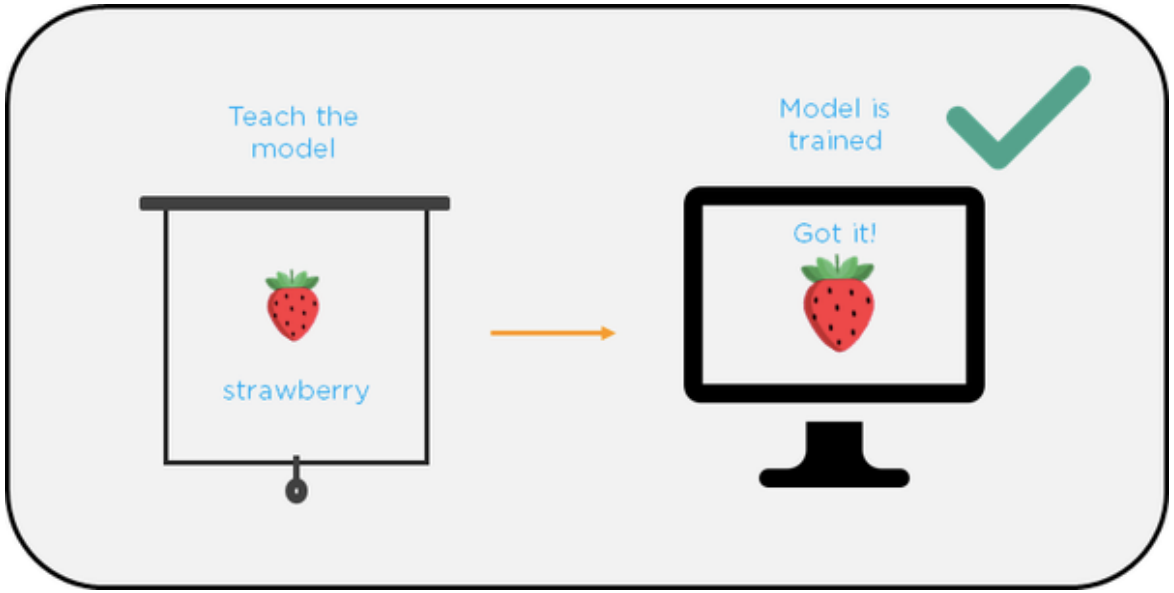
Some of the most typical casses are represented in the following picture:



Classification: what do we need?

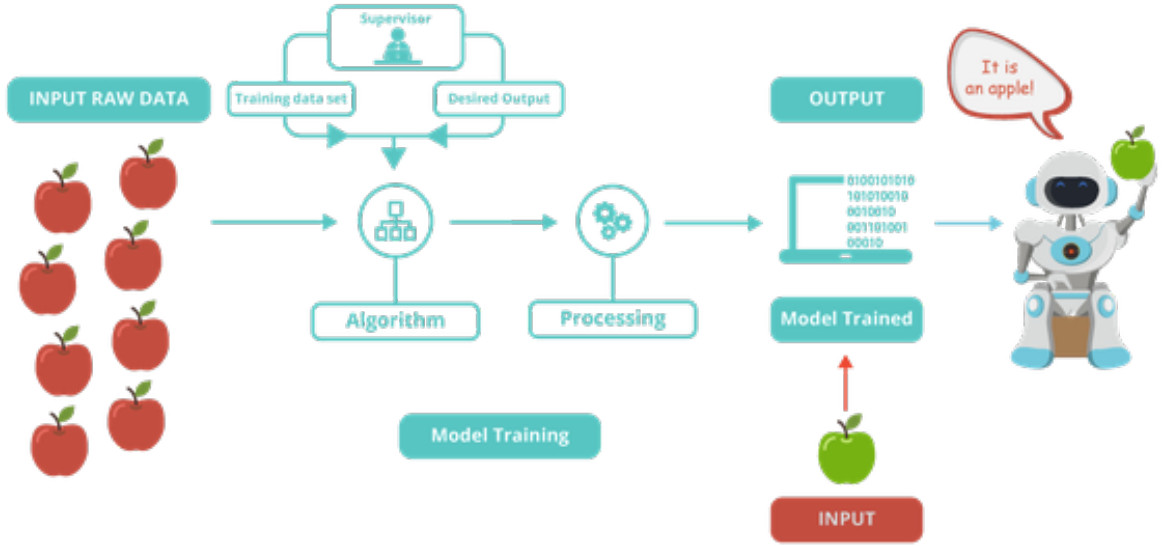
In order to perform classification we need labeled data.

Supervised learning



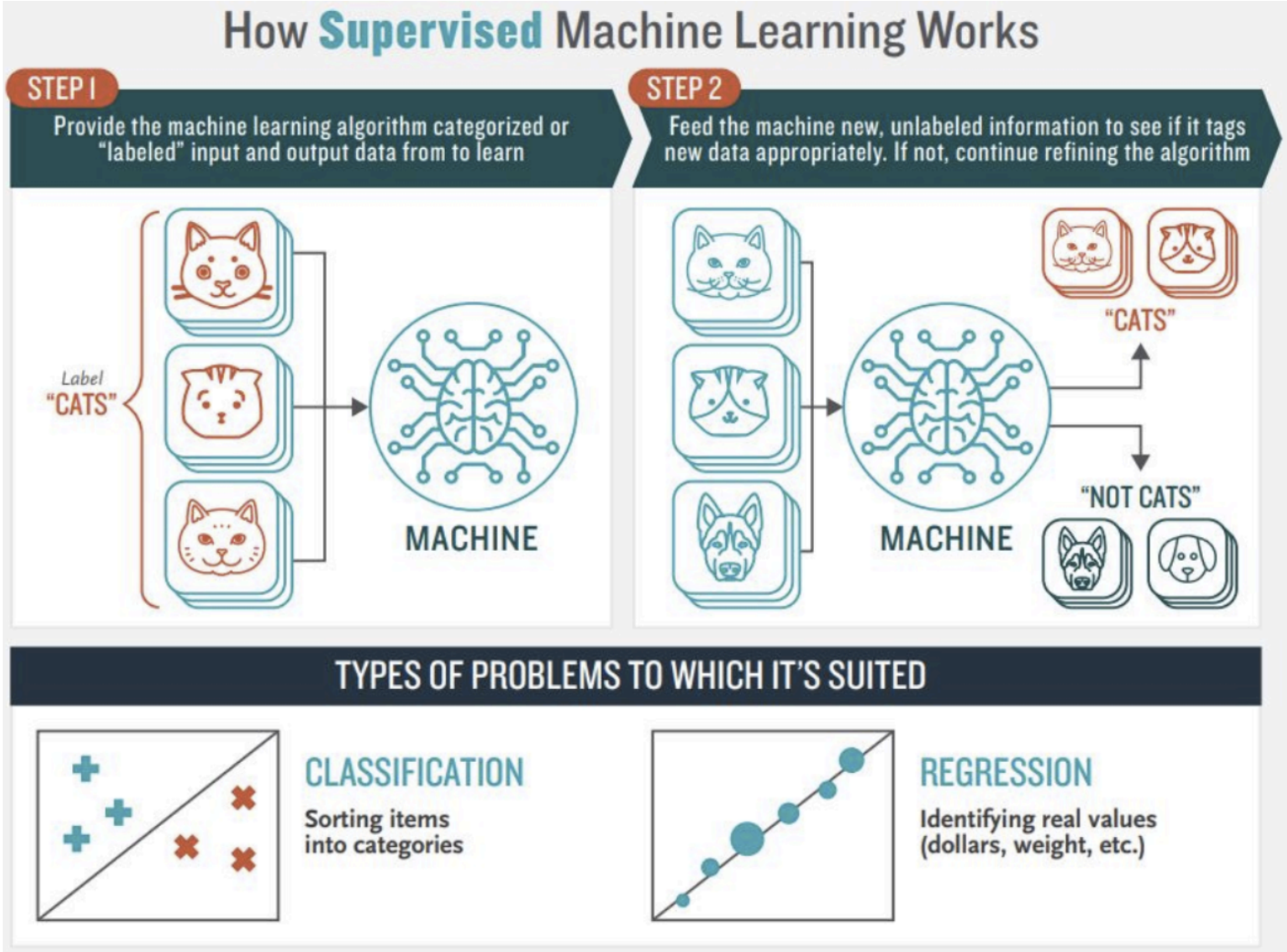
Supervised Learning

If you're learning a task under supervision, someone is present judging whether you're getting the right answer. Similarly, in supervised learning, that means having a full set of labeled data while training an algorithm.



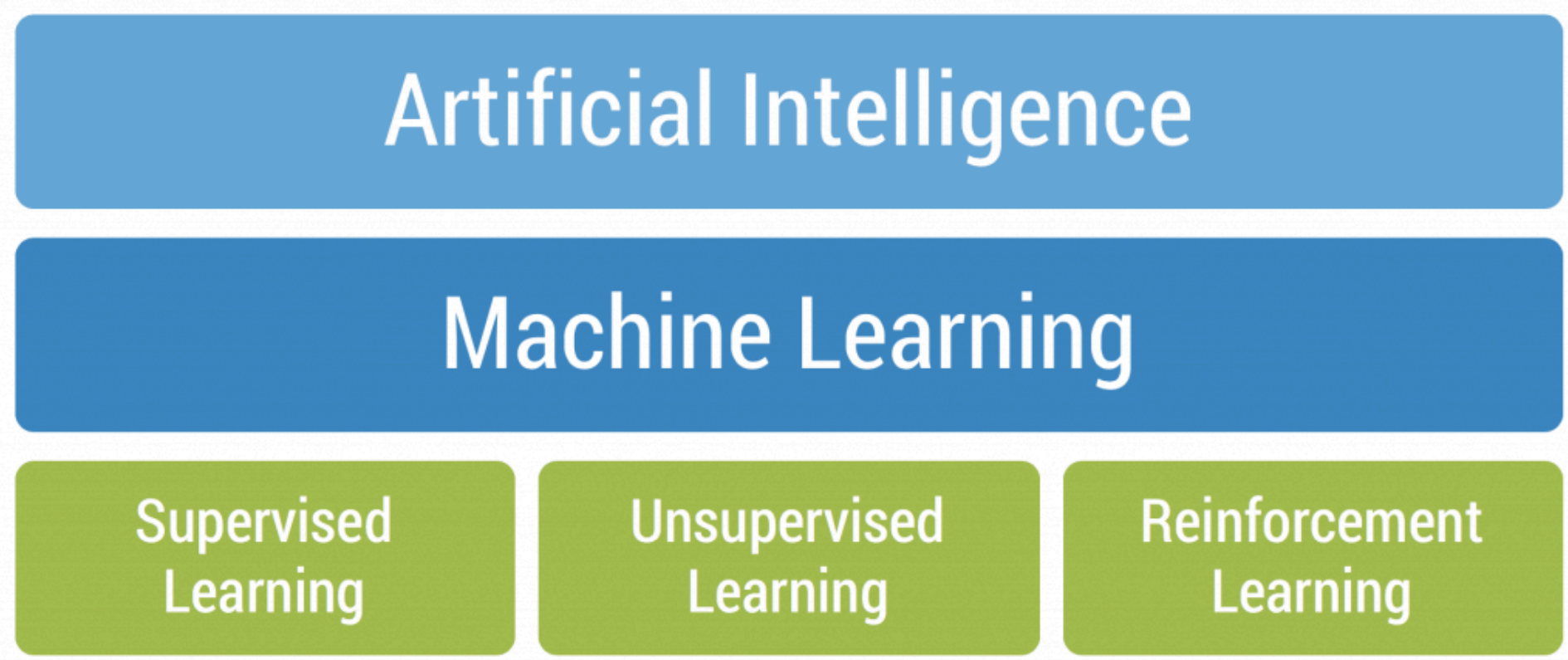
Supervised learning is, thus, best suited to problems where there is a set of available reference points or a ground truth with which to train the algorithm. But those aren't always available.

Supervised Learning



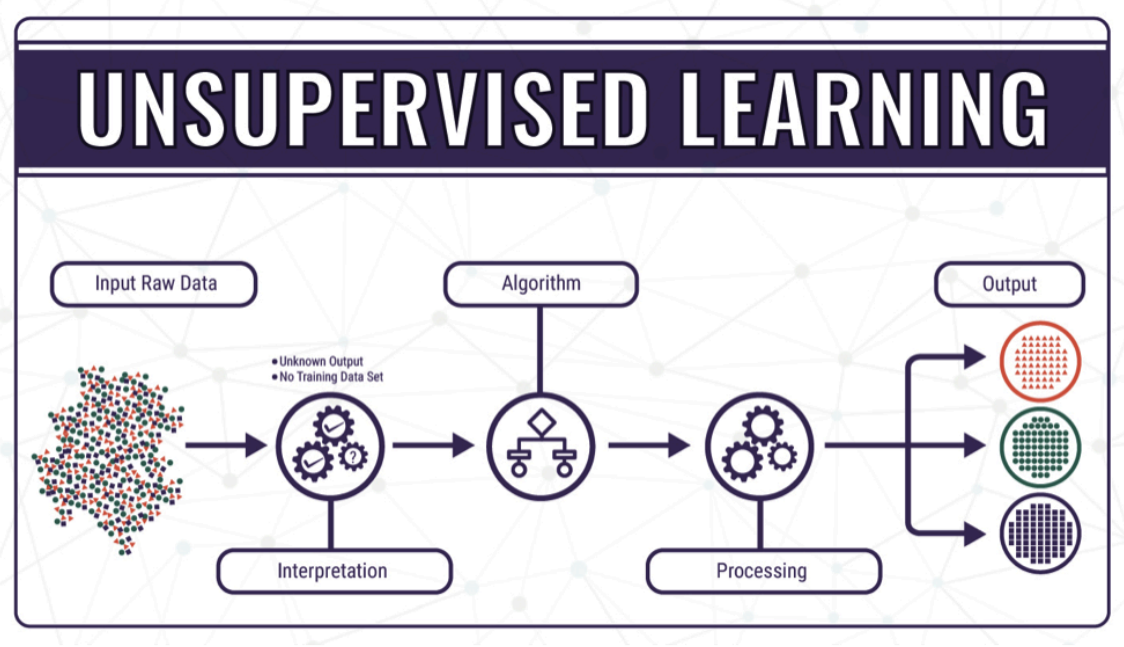
ML Models

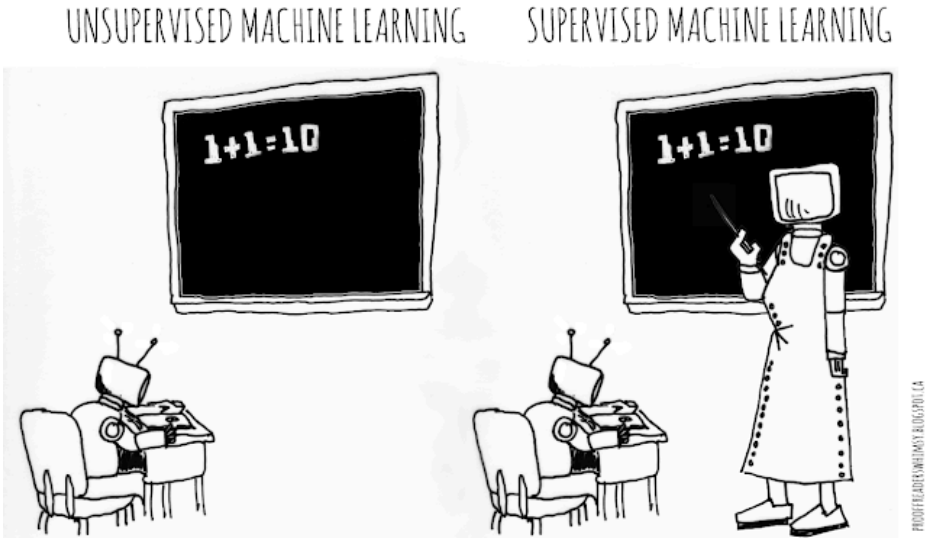
But we do not always have available labeled data…



Unsupervised Learning

In unsupervised learning, a deep learning model is handed a dataset without explicit instructions on what to do with it. The training dataset is a collection of examples without a specific desired outcome or correct answer. The neural network then attempts to automatically find structure in the data by extracting useful features and analyzing its structure.





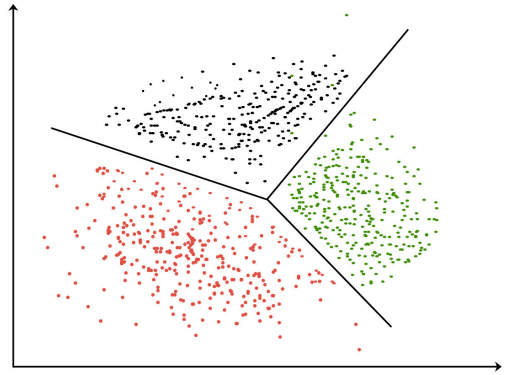
Unsupervised Learning: data

The unsupervised learning model can organize the data in different ways.

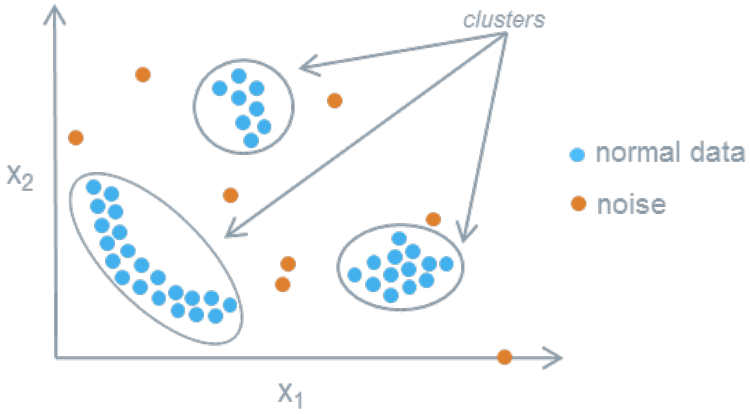
- **Clustering:** the deep learning model looks for training data that are similar to each other and groups them together.
- **Anomaly detection:** unsupervised learning can be used to flag outliers in a dataset.
- **Association:** By looking at a couple key attributes of a data point, an unsupervised learning model can predict the other attributes with which they're commonly associated.
- **Autoencoders:** Autoencoders take input data, compress it into a code, then try to recreate the input data from that summarized code.

Unsupervised Learning: examples

- **Clustering: l**ook at a collection of bird photos and separate them roughly by species, relying on cues like feather color, size or beak shape.

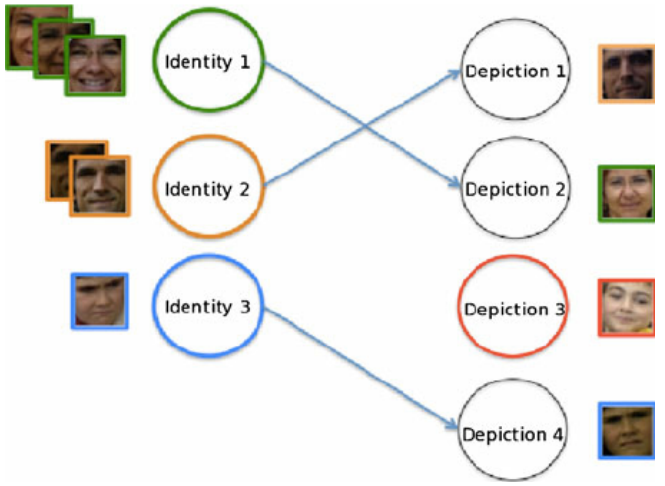


- **Anomaly detection:** if the same credit card is used in California and Denmark within the same day, that's cause for suspicion.

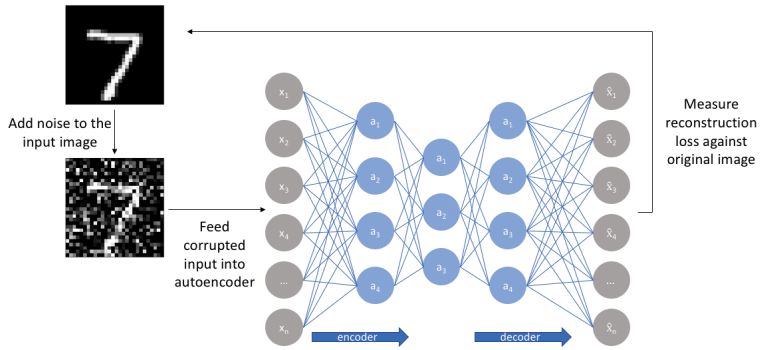


Unsupervised Learning: examples

- **Association:** Fill an online shopping cart with diapers, applesauce and sippy cups and the site just may recommend that you add a bib and a baby monitor to your order.



- **Autoencoders:** It's like starting with Moby Dick, creating a SparkNotes version and then trying to rewrite the original story using only SparkNotes for reference.



Learning:
Supervised vs Unsupervised

	SUPERVISED LEARNING	UNSUPERVISED LEARNING
Input Data	Uses Known and Labeled Data as input	Uses Unknown Data as input
Computational Complexity	Very Complex	Less Computational Complexity
Real Time	Uses off-line analysis	Uses Real Time Analysis of Data
Number of Classes	Number of Classes are known	Number of Classes are not known
Accuracy of Results	Accurate and Reliable Results	Moderate Accurate and Reliable Results

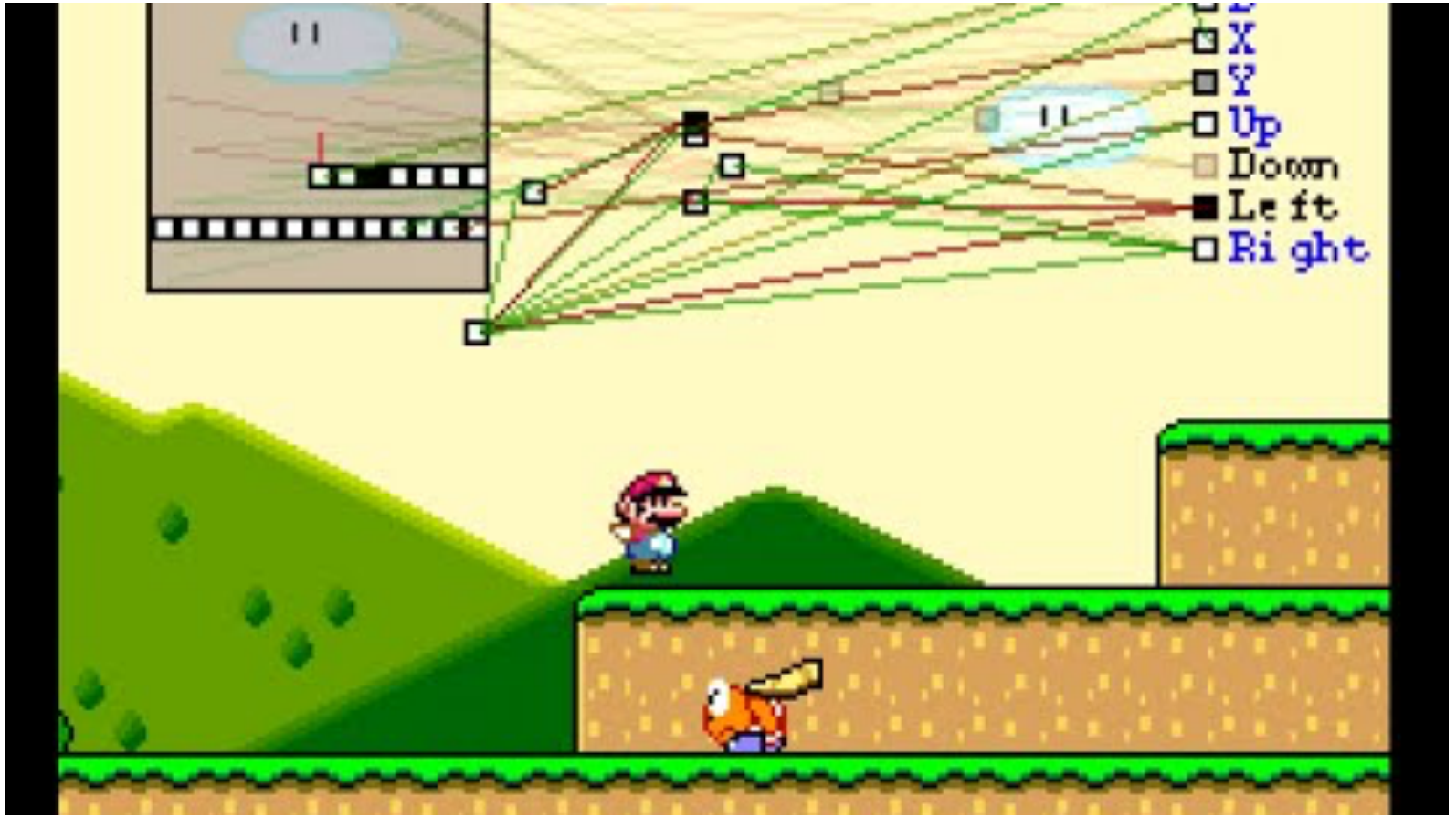
Reinforcement Learning

Video games are full of reinforcement cues. Complete a level and earn a badge. Defeat the bad guy in a certain number of moves and earn a bonus. Step into a trap — game over.

Without this feedback, they would just take random actions around a game environment in the hopes of advancing to the next level.

In this kind of machine learning, AI agents are attempting to find the optimal way to accomplish a particular goal, or improve performance on a specific task. As the agent takes action that goes toward the goal, it receives a reward. The overall aim: predict the best next step to take to earn the biggest final reward.

Reinforcement Learning: Example



https://www.youtube.com/watch?v=qv6UVOQ0F44&t=8s

What happend if we have both labeled and unlabeled data?

Semi-supervised Learning

This method is particularly useful when extracting relevant features from the data is difficult, and labeling examples is a time-intensive task for experts.

Common situations for this kind of learning are medical images like CT scans or MRIs.
A trained radiologist can go through and label a small subset of scans for tumors or diseases.

The deep learning network can still benefit from the small proportion of labeled data and improve its accuracy compared to a fully unsupervised model.

Semi-supervised Learning: Example

A popular training method that starts with a fairly small set of labeled data is using generative adversarial networks, or GANs.

Imagine two machine learning flow (e.g. deep learning networks) in competition, each trying to outsmart the other. That's a GAN. One of the networks, called the generator, tries to create new data points that mimic the training data. The other network, the discriminator, pulls in these newly generated data and evaluates whether they are part of the training data or fakes.

https://thispersondoesnotexist.com/

https://thiscatdoesnotexist.com/

Machine Learning: Let’s see in details…

Supervised Learning: Details

Given a set of N training examples of the form $\{(x_1, y_1), \ldots, (x_N, y_N)\}$ such that x_i is the feature vector of the i-th example and y_i is its label (i.e., class), a learning algorithm seeks a function $g : X \to Y$, g is defined as returning the y value that gives the highest score $g(x) = \arg\max_y f(x, y)$

What is f?
f is a scoring function $f : X \times Y \to \mathbb{R}$.
f and g can be any space of functions. Example: in probabilistic models g is a conditional probability ($g(x) = P(y|x)$) and f is a joint probability ($f(x, y) = P(x, y)$).

Supervised Learning: Details

Regardless of the model chosen for risk minimization, it is assumed that the training set consists of a sample of independent and identically distributed pairs,(x_i,y_i) In order to measure how well a function fits the training data, a loss function is defined.

$$L : Y \times Y \to \mathbb{R}^{\geq 0}$$

For training example (x_i,y_i) the loss of predicting the value $\hat{y}$ is $L(y_i, \hat{y})$.

Supervised Learning: Details

Common loss functions in machine learning are:

Mean Square Error/Quadratic Loss/L2 Loss

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}$$

Mean Absolute Error/L1 Loss

$$MAE = \frac{\sum_{i=1}^{n} \mid y_i - \hat{y}_i \mid}{n}$$

Mean Bias Error

$$MBE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)}{n}$$

Supervised Learning: Details

Hinge Loss/Multi class SVM Loss

$$SVMLoss = \sum_{j \neq y_i} max(0, s_j - s_{y_i} + 1)$$

The score of correct category should be greater than sum of scores of all incorrect categories by some safety margin. And hence hinge loss is used for maximum-margin classification, most notably for support vector machines.

Cross Entropy Loss/Negative Log Likelihood

$$CrossEntropyLoss = -(y_i log(\hat{y}_i) + (1 - y_i) log(1 - \hat{y}_i))$$

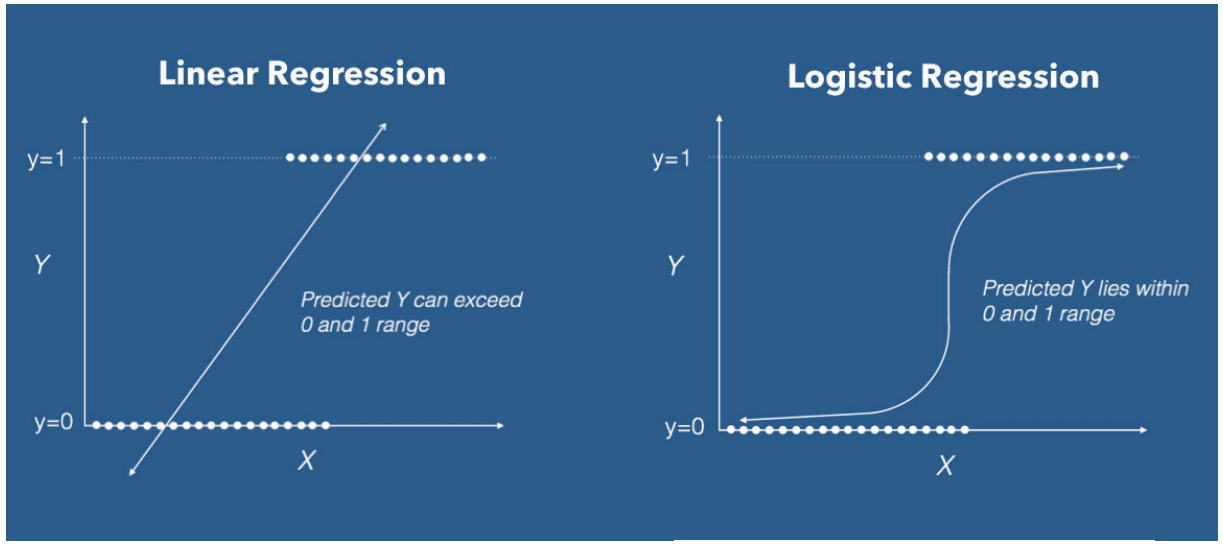
This is the most common setting for classification problems. Cross-entropy loss increases as the predicted probability diverges from the actual label. Notice that when actual label is 1 (y(i) = 1), second half of function disappears whereas in case actual label is 0 (y(i) = 0) first half is dropped off. In short, we are just multiplying the log of the actual predicted probability for the ground truth class.

Supervised Learning Algorithms: Logistic Regression

When the classes aren't perfectly linear separable, the Logistic Regression is one of the most used algorithms. It is a predictive analysis algorithm and based on the concept of probability.

Logistic Regression uses a more complex cost function, this cost function can be defined as the Sigmoid function:

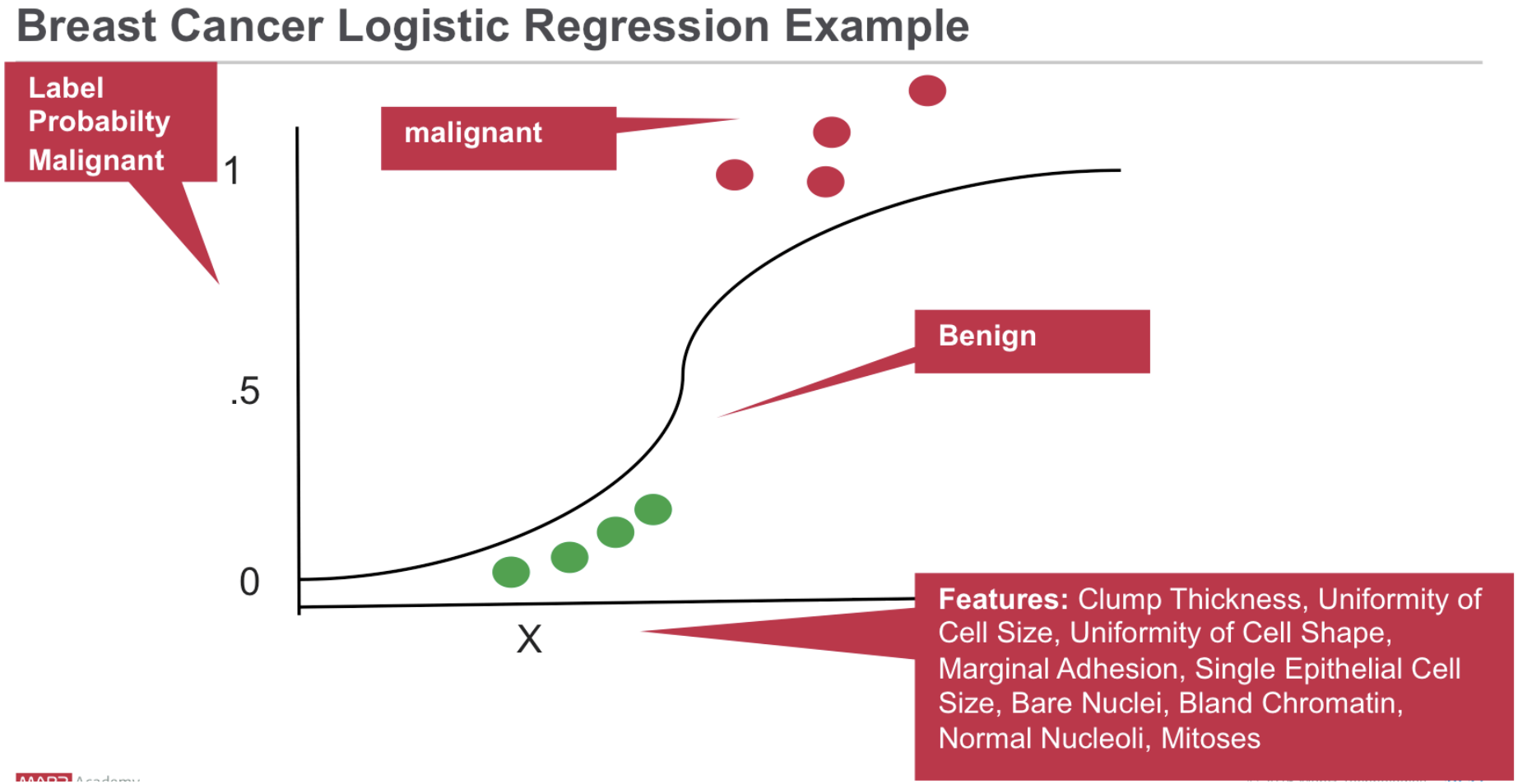
$$f(x) = \frac{1}{1 + e^{-(x)}}$$



The hypothesis of logistic regression tends it to limit the cost function between 0 and 1. $0 \leq h_\theta(x) \leq 1$

For Example, We have 2 classes, let's take them like cats and dogs(1 dog , 0 cats). We basically decide with a threshold value above which we classify values into Class 1 and of the value goes below the threshold then we classify it in Class 2.

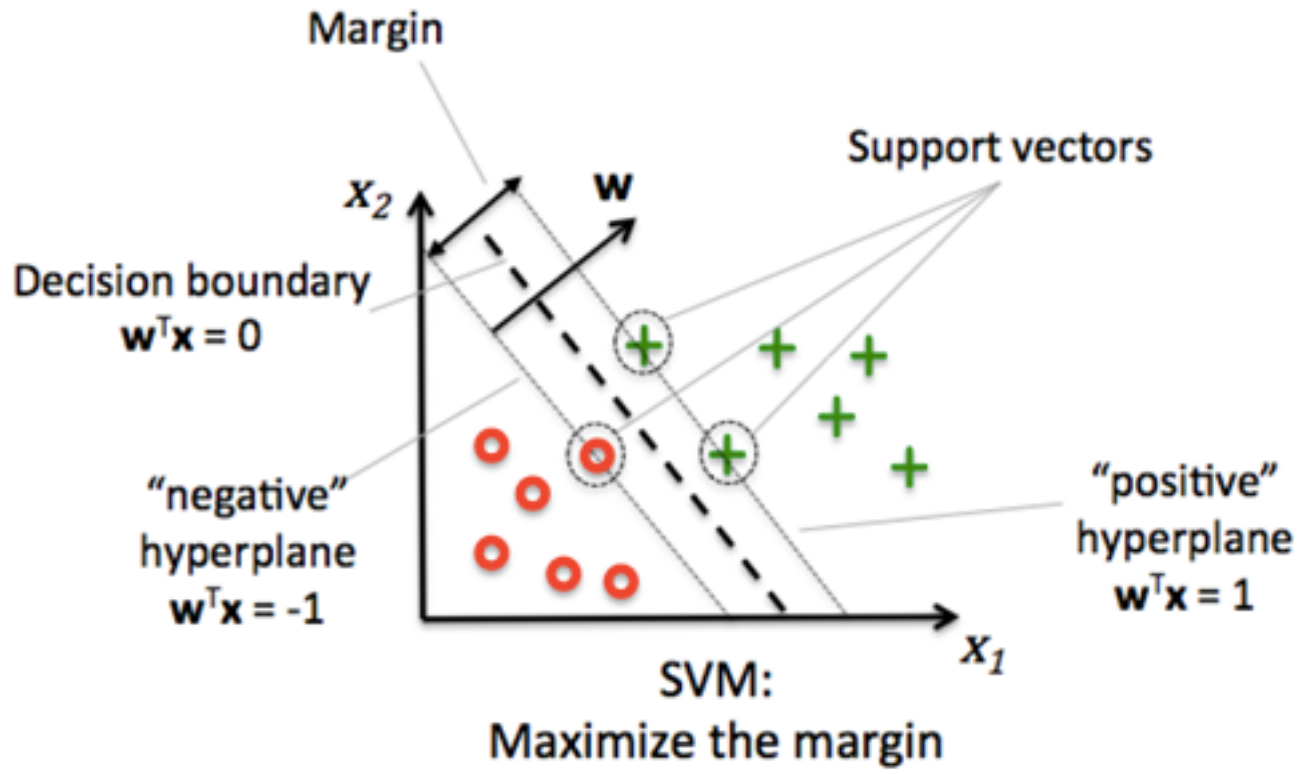
Supervised Learning Algorithms: Logistic Regression Example



Supervised Learning Algorithms: Support Vector Machines

In SVM, The optimization objective is to **set a decision line** that separates the classes by maximizing the marging between this line and the sample points that are closest to this hyperplane. This points are called support vectors.

In other words we find the points closest to the line from both the classes.These points are called support vectors. Now, we compute the distance between the line and the support vectors. This distance is called the margin. Our goal is to find the line that maximize the margin. The hyperplane for which the margin is maximum is the optimal hyperplane.



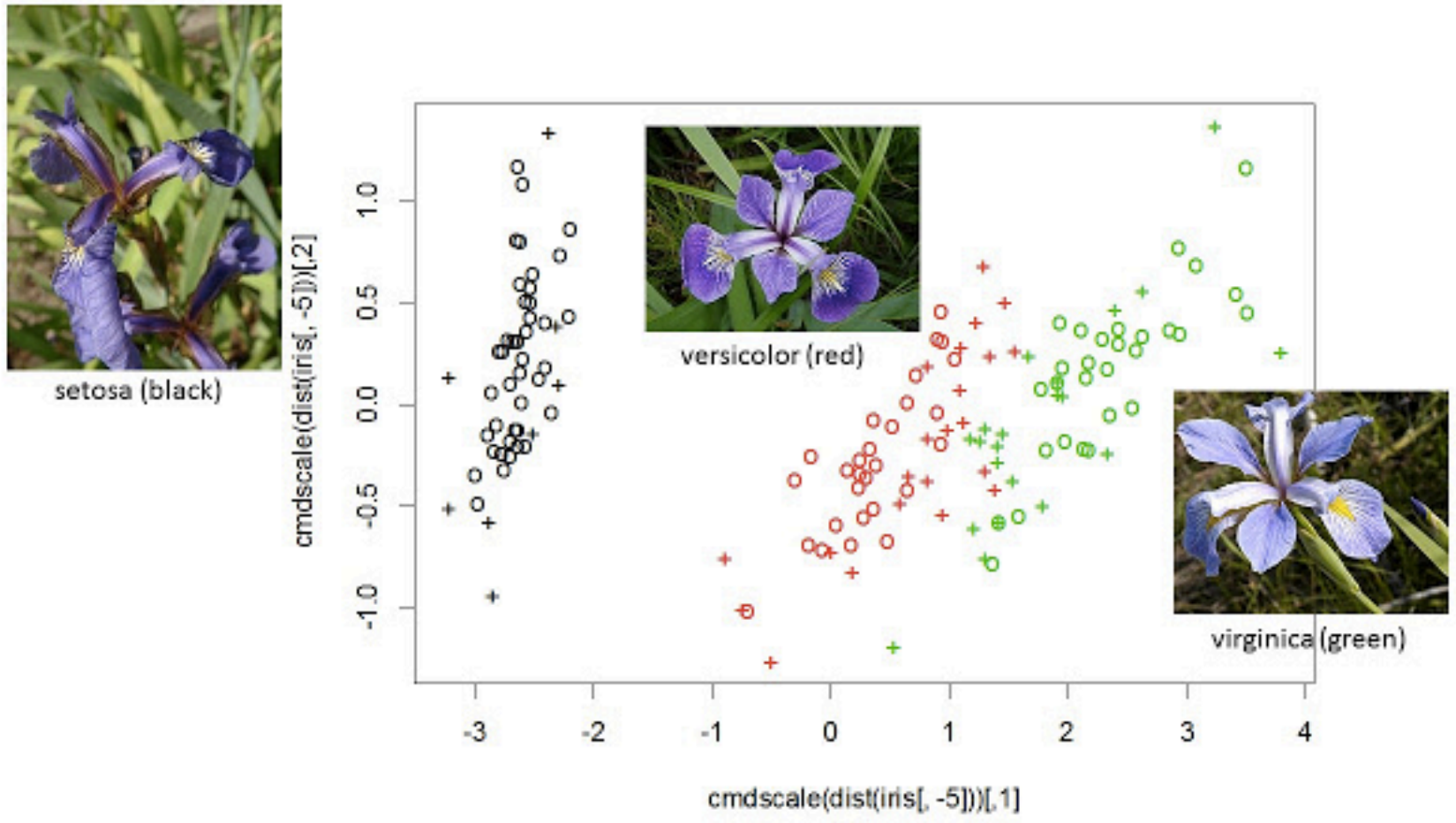
Supervised Learning Algorithms: SVM Example

Features:

- Sepal lenght
- Petal lenght

Cass:

- Iris Virginica
- Iris Versicolor
- Iris Setosa

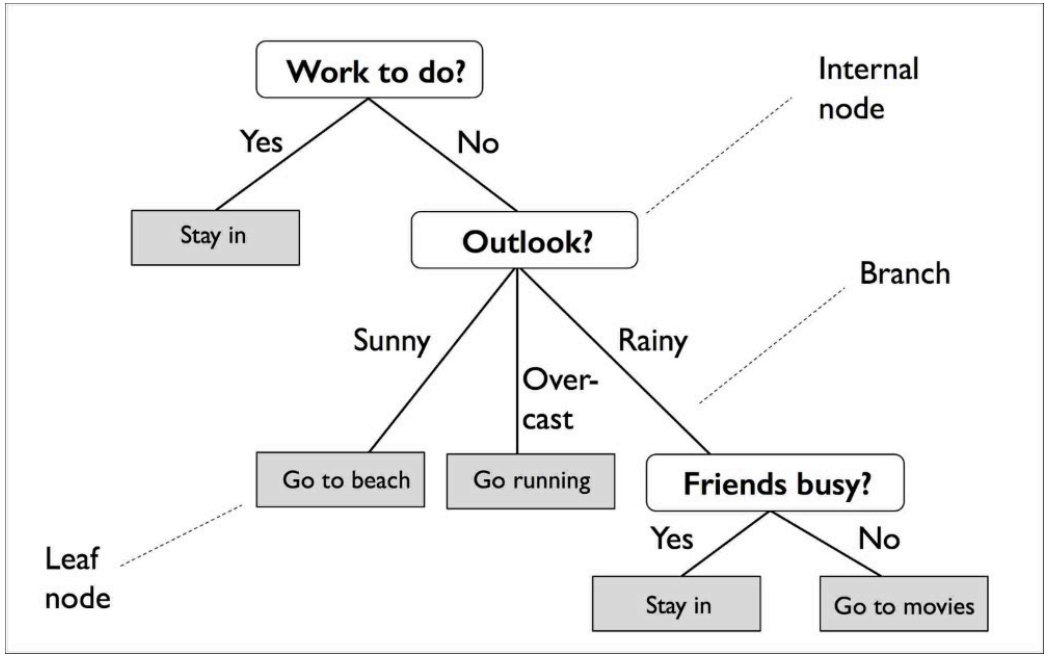


Supervised Learning Algorithms: Decision Trees

Decision trees algorithms break down the dataset by making questions until they have narrowed data enough to make a prediction.

Based on the features of the training set, the decision tree learns a series of questions to infer the class labels of the samples.

The starting node is called the tree root, and the algorithm will split the dataset on the feature that contains the maximum Information Gain iteratively, until the leaves (the final nodes) are pure.



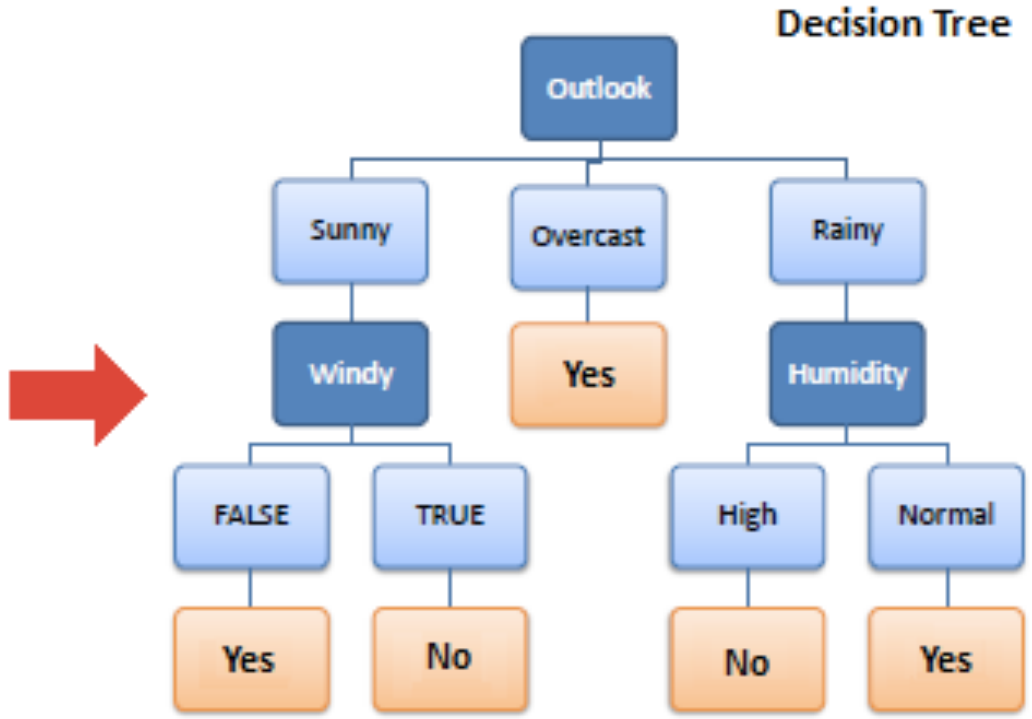
Supervised Learning Algorithms: Decision Trees

What we have to set to perform a decision tree algorithm?

- Maximum Depth: the largest length from the root to the leaf.
- Maximum number of samples for each leaf: to avoid of having 99 samples inone of the splits and 1 in the other one.
- Minimum Number of Samples: analog to the previous one but for minumum.
- Maximum Number of Features: limit the number of features that one looks for in each split. If this number is high enough, we are likely to find a good feature among the ones we look for.

Supervised Learning Algorithms: Decision Tree Example

Predictors				Target
Outlook	Temp.	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No

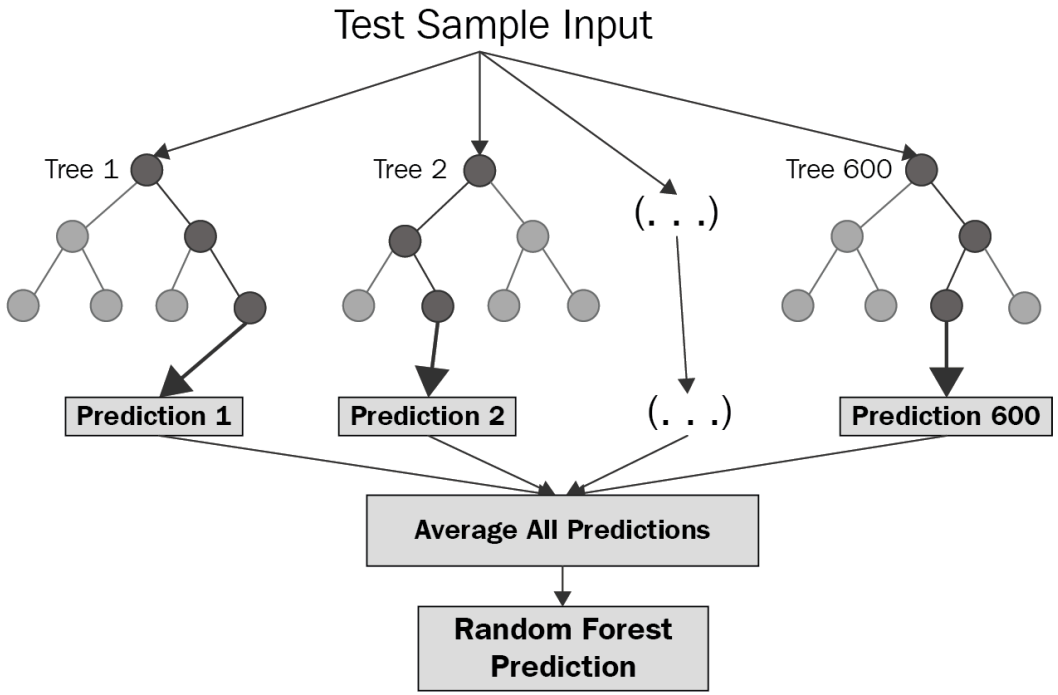


Supervised Learning Algorithms: Random Forests

If we have a dataset with a lot of features (columns), the decision tree Algorithm usually tend to overcomplicate the model and the learning process.
We can solve this issue by selecting each colum randomly and making decision trees for each batch of columns.

So the idea is to develop an ensemble learning algorithm which will combine a number of weaker models to build a more robust one.

The main advantage of this method is that we usually won't need to prune the random forest but it is less interpretable than decision trees.
The only parameter to set is the number of trees.



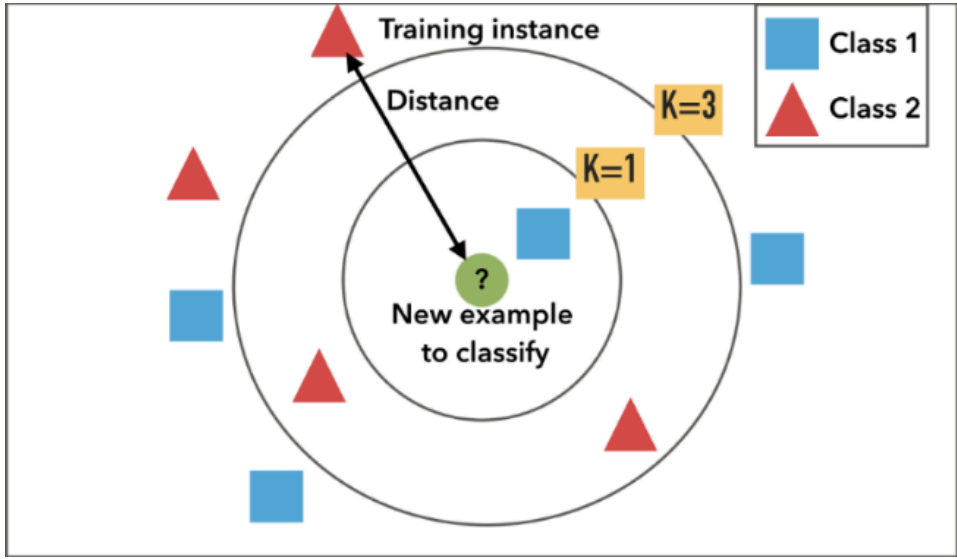
Supervised Learning Algorithms: K-Nearest Neighbor

They receive this name because they do not learn how to discriminate the dataset with an optimized function, but memorize the dataset instead.

It is frequently called ‘lazy algorithm’ beacuse it is characterized by memorizing the training dataset, and lazy learning is a specific case of these algorithms, asociated with zero computational cost during the learning.

Steps:

1. Choosing the number of k and the distance metric.
2. Finding the k nearest neighbor of the sample to classify.
3. Assigning the class label by majority vote.



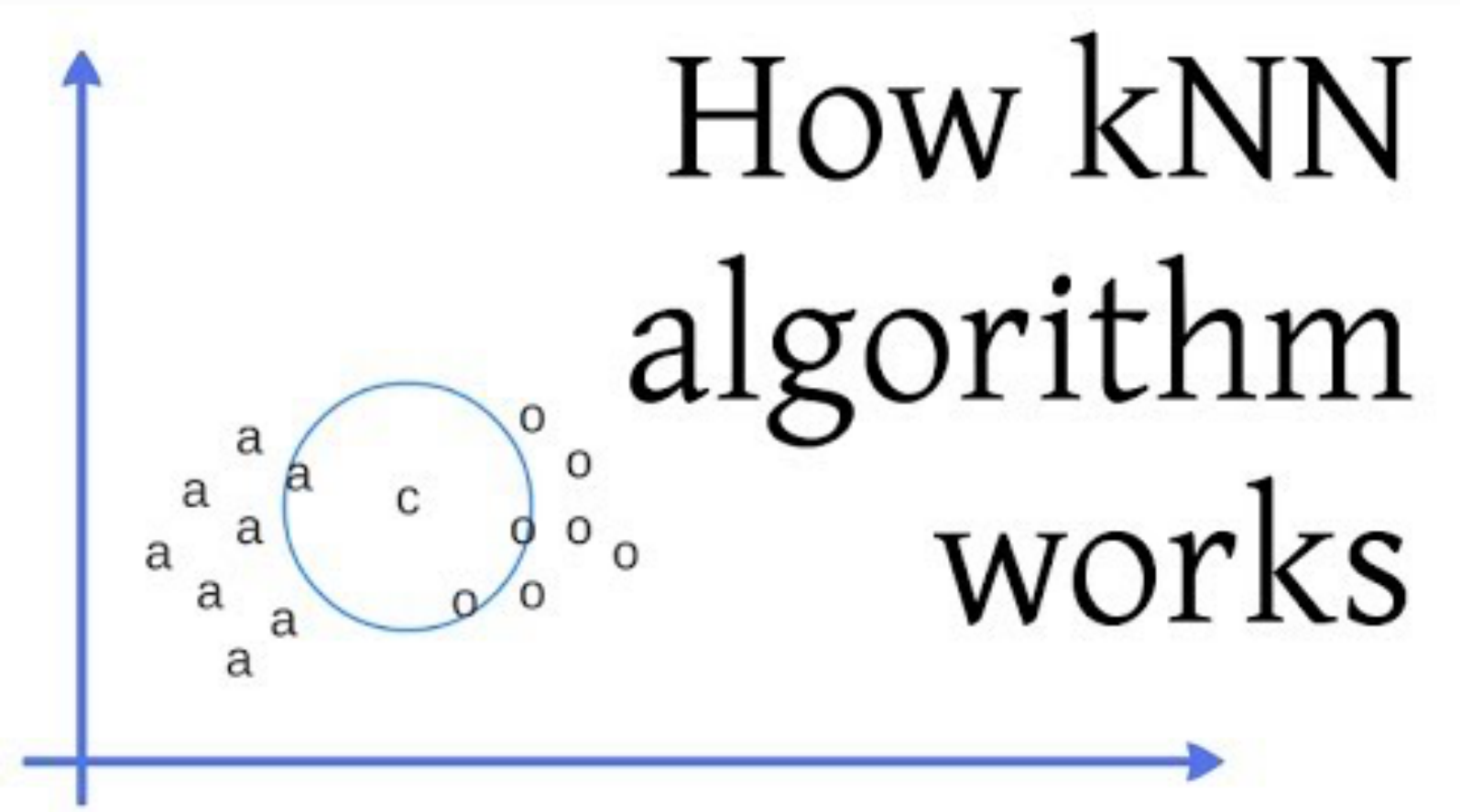
Supervised Learning Algorithms: K-Nearest Neighbor

The KNN algorithm find those k samples that are closest to the point to classify, basing its predictions on the distance metric.
The main advantage is that as it is a memory-based algorithm, it adapts to new training data.
The down-side is that the computational cost increases linearly with the size of the training data.

It is also crucial to stablish an appropiate distance metric. Usually, it is used the ‘Minkowski’ distance, which a generalization of the Eucledian and Manhattan distance.This distance is defined as follows:

$$d\left(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}\right) = \sqrt[p]{\sum_k \left|x_k^{(i)} - x_k^{(j)}\right|^p}$$

Supervised Learning Algorithms:
K-Nearest Neighbor Example

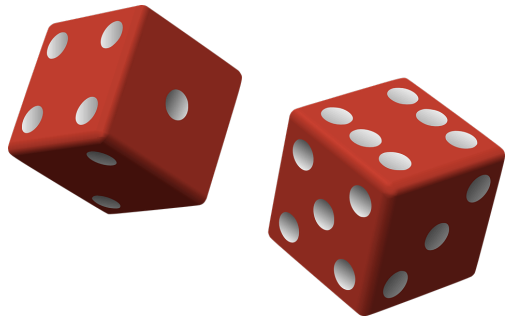


https://www.youtube.com/watch?v=UqYde-LULfs

Supervised Learning Algorithms: Naive Bayes Classifier

A Naive Bayes classifier is a probabilistic machine learning model that's used for classification task. The crux of the classifier is based on the Bayes theorem.

Bayes theorem: $$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$



Using Bayes theorem, we can find the probability of **A** happening, given that **B** has occurred. Here, **B** is the evidence and **A** is the hypothesis. The assumption made here is that the predictors/features are independent. That is presence of one particular feature does not affect the other. Hence it is called naive.
(This theorem is often used in «roll the dice» examples.)

Supervised Learning Algorithms: Naive Bayes Classifier

X is given (features) as $X = (x_1, x_2, x_3, ..., x_n)$

By substituting for X and expanding using the chain rule we get

$$P(y|x_1, ..., x_n) = \frac{P(x_1|y)P(x_2|y)...P(x_n|y)P(y)}{P(x_1)P(x_2)...P(x_n)}$$

For all entries in the dataset, the denominator does not change, it remain static. We can remove it and introduce proportionality. However, we need to find the class y with maximum probability.

Our formula will be: $y = argmax_y P(y) \prod_{i=1}^{n} P(x_i|y)$

There are different functions to estimate $P(x_i|y)$, so we can have: Multinomial Naive Bayes; Bernoulli Naive Bayes; Gaussian Naive Bayes.

(Gaussian Naive Bayes: $P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} exp\left(-\frac{(x_i-\mu_y)^2}{2\sigma_y^2}\right)$)

Supervised Learning Algorithms: Naive Bayes Classifier Example

$$y = argmax_y P(y) \prod_{i=1}^{n} P(x_i|y)$$

Example of Naïve Bayes Classifier

[Optional]

Name	Give Birth	Can Fly	Live in Water	Have Legs	Class
human	yes	no	no	yes	mammals
python	no	no	no	no	non-mammals
salmon	no	no	yes	no	non-mammals
whale	yes	no	yes	no	mammals
frog	no	no	sometimes	yes	non-mammals
komodo	no	no	no	yes	non-mammals
bat	yes	yes	no	yes	mammals
pigeon	no	yes	no	yes	non-mammals
cat	yes	no	no	yes	mammals
leopard shark	yes	no	yes	no	non-mammals
turtle	no	no	sometimes	yes	non-mammals
penguin	no	no	sometimes	yes	non-mammals
porcupine	yes	no	no	yes	mammals
eel	no	no	yes	no	non-mammals
salamander	no	no	sometimes	yes	non-mammals
gila monster	no	no	no	yes	non-mammals
platypus	no	no	no	yes	mammals
owl	no	yes	no	yes	non-mammals
dolphin	yes	no	yes	no	mammals
eagle	no	yes	no	yes	non-mammals

A: attributes

M: mammals

N: non-mammals

$$P(A|M) = \frac{6}{7} \times \frac{6}{7} \times \frac{2}{7} \times \frac{2}{7} = 0.06$$

$$P(A|N) = \frac{1}{13} \times \frac{10}{13} \times \frac{3}{13} \times \frac{4}{13} = 0.0042$$

$$P(A|M)P(M) = 0.06 \times \frac{7}{20} = 0.021$$

$$P(A|N)P(N) = 0.004 \times \frac{13}{20} = 0.0027$$

Give Birth	Can Fly	Live in Water	Have Legs	Class
yes	no	yes	no	?

P(A|M)P(M) > P(A|N)P(N)

=> Mammals

We know methods…
But how to use the data?

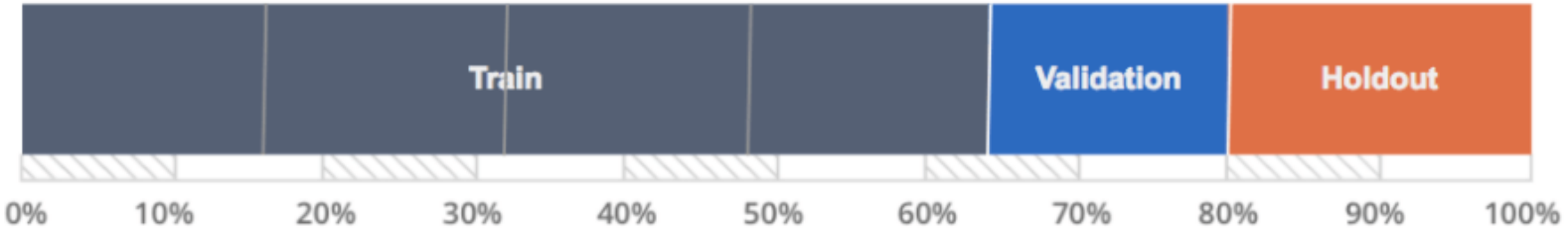
Supervised Learning: Data partition

The data used to build the final model usually comes from multiple datasets. In particular, three data sets are commonly used in different stages of the creation of the model.

- **training dataset**, that is a set of examples used to fit the parameters (e.g. weights of connections between neurons in artificial neural networks) of the model. In practice, the training dataset often consist of pairs of an input vector (or scalar) and the corresponding output vector (or scalar), which is commonly denoted as the target (or label).
- **validation dataset** provides an unbiased evaluation of a model fit on the training dataset while tuning the model's hyperparameters (e.g. Maximum Depth in decision tree, K in KNN)
- **test dataset** is a dataset used to provide an unbiased evaluation of a final model fit on the training dataset. Usually data in test dataset has never been used in training (called holdout).

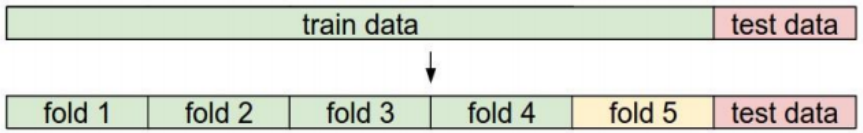
Supervised Learning: Data partition

Usually in machine learning we need a lot of data for training dataset…



It is not always given to us a fixed partition. We can proceed as follows:
e.g. 12000 labeled samples

- **Disjoint Sets**: 10000 Train, 1000 Valid, 1000 Test.
- **K-fold Cross-Validation**: 2000 Test, K = 5 fold of 2000 samples. We perform 5 times training by choosing one of the Fold as Valid and 4 as Test. The final performance is the mean.
- **Leave-one-out:** A subset of cross-validation were folds has lenght 1.



We know methods, we know how to use data…
But how to know if the training is going fine?

Supervised Learning: Loss and Accuracy

Our evaluation parameters are:

- Loss Function
- Accuracy Function

Loss is defined as the difference between the predicted value by your model and the true value. The most common loss function is cross-entropy. It's defined as:

$$\text{Cross-entropy} = -\sum_{i=1}^{n}\sum_{j=1}^{m} y_{i,j}\log(p_{i,j})$$

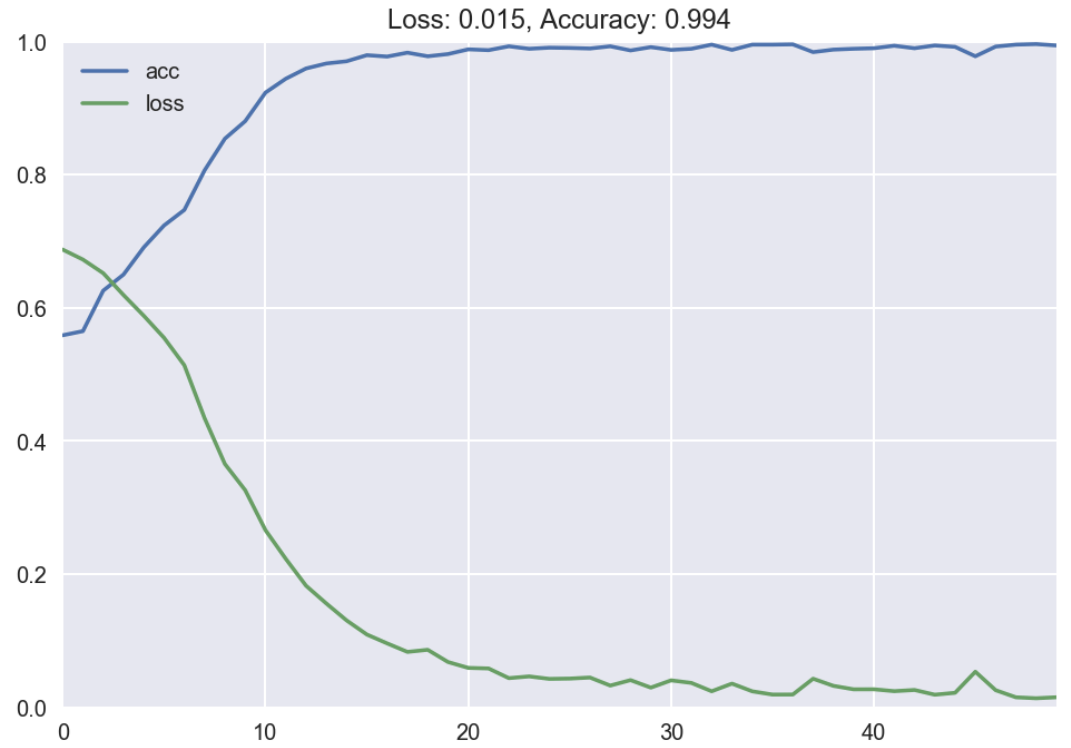
Accuracy is one of the metrics to measure the performance of your model. It's defined as:

$$Accuracy = \frac{\text{No of correct predictions}}{\text{Total no of predictions}}$$

Supervised Learning: Convergence

The first objective to pursue during training is the convergence on the Train set.
Consider a classifier whose training involves an iterative process. It has convergence when:

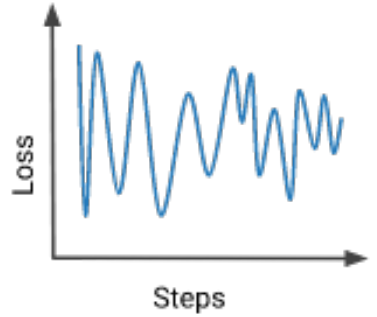
- **Loss** function becomes **decreasing**
- **Accuracy** function becomes **increasing**



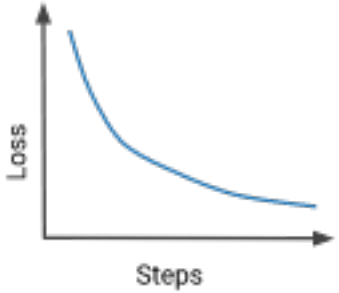
iterations or «epoch»

Supervised Learning: Convergence

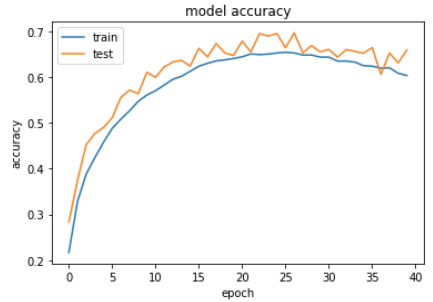
If the loss does not decrease (or significantly oscillate) the system does not converge. We have to change something in optimization function, hyperparameters etc.



If the loss decreases but the accuracy does not grow, probably a wrong loss-function was chosen.



If accuracy does not approach 100% on the Train, the degrees of Freedom of the classifier is not enough to manage the complexity of the problem.

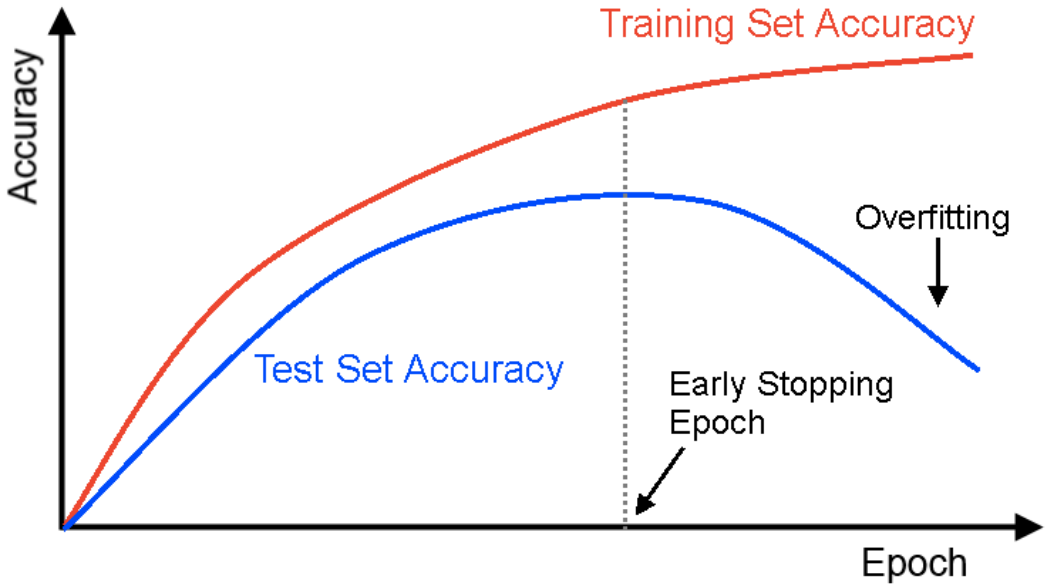


Supervised Learning: Overfitting and Generalization

Let us remember that our goal is to maximize accuracy on Test. In the hypothesis that Valid is representative of Test, we set ourselves the goal of maximizing accuracy on Valid.

By **generalization** we mean the ability to transfer the high accuracy achieved on Train to Valid.

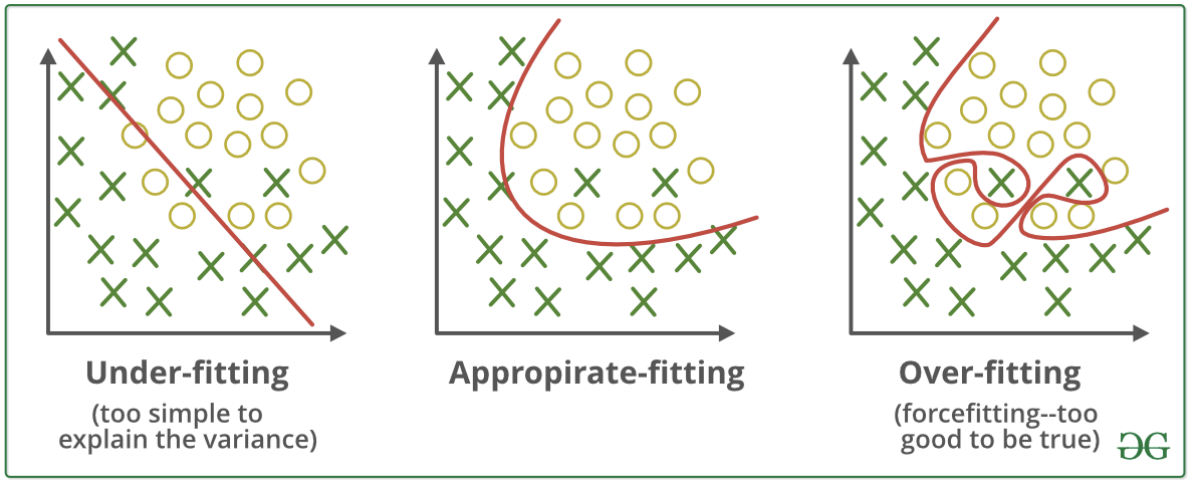
If the classifier's degrees of freedom are excessive, high accuracy is achieved on Train, but not on Valid (poor generalization). In this case we talk about **overfitting** of Train. This situation occurs very easily when Train is small.



Supervised Learning: Underfitting

Because of overfitting, we want the model to learn from the training data, but we don't want it to learn too much.
However, this could lead the model to not learn enough patterns from the training data, and possibly not even capture the dominant trend. This case is called **underfitting**.

As you probably expected, underfitting is just as bad for generalization of the model as overfitting.






Supervised and Unsupervised Learning

Warning

Overfitting is possible also in unsupervised learning, but it appears differently.

Example: If you fit n clusters to n cases, then you'd end up with (useless) clustering solution that does not translate to external data. In such case, clustering would overfitt by design.

It's more often discussed as "automatic determination of optimal cluster number", or model selection. Hence, cross-validation is not applicable in this setting.

How to deal with overfitting, underfitting and hyperparameters…
In Next Lessons!