Computació i Sistemes Intel·ligents

Part III: Machine Learning

Ramon Ferrer-i-Cancho
rferrericancho@upc.edu
(Marta Arias)

Dept. CS, UPC

Fall 2025

Website

Please go to http://www.cs.upc.edu/~csi for all course's material, schedule, lab work, etc.

Announcements through https://raco.fib.upc.edu

Class logistics

- Theory slots on Tuesdays.
- Laboratory slots on Thursdays.
- 1 exam: Tuesday Dec. 16th, in class
- 1 project (due after Christmas break, date TBD)

Check `http://www.cs.upc.edu/~csi` for details about the schedule.

Lab

Environment for practical work

We will use python3 and jupyter and the following libraries:

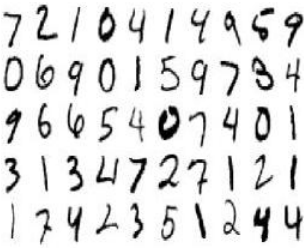
- pandas, numpy, scipy, scikit-learn, seaborn, matplotlib

During the first session we will cover how to install these in case you use your laptop. Libraries are already installed in the schools' computers.

… so, let's get started!

What is Machine Learning?

An example: digit recognition



Input: image e.g. 

Output: corresponding class label [0..9]

- Very hard to program yourself
- Easy to assing *labels*

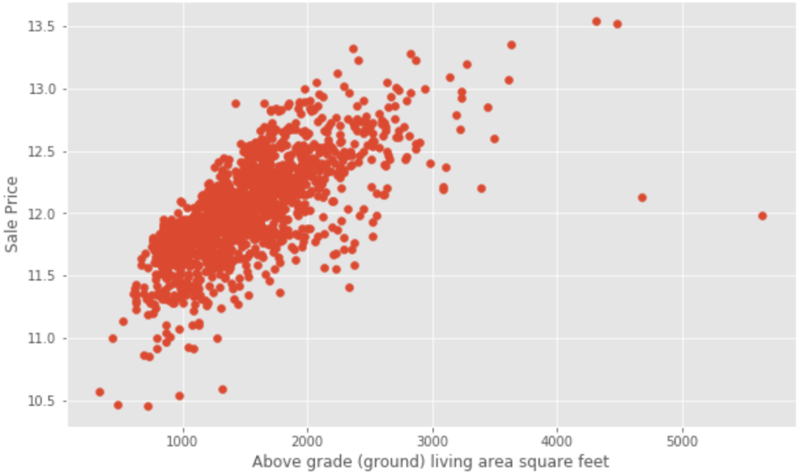
What is Machine Learning?

An example: flower classification (the famous “iris” dataset)

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
7.0	3.2	4.7	1.4	versicolor
6.1	2.8	4.0	1.3	versicolor
6.3	3.3	6.0	2.5	virginica
7.2	3.0	5.8	1.6	virginica
5.7	2.8	4.1	1.3	?

What is Machine Learning?

An example: predicting housing prices (regression)



Is Machine Learning useful?

Applications of ML

- Web search
- Computational biology
- Finance
- E-commerce (recommender systems)
- Robotics
- Autonomous driving
- Fraud detection
- Information extraction
- Social networks
- Debugging
- Face recognition
- Credit risk assessment
- Medical diagnosis
- ... etc

About this course

A gentle introduction to the world of ML

This course will teach you:

- Basic into concepts and intuitions on ML
- To apply off-the-shelf ML methods to solve different kinds of prediction problems
- How to use various python tools and libraries

This course will *not*:

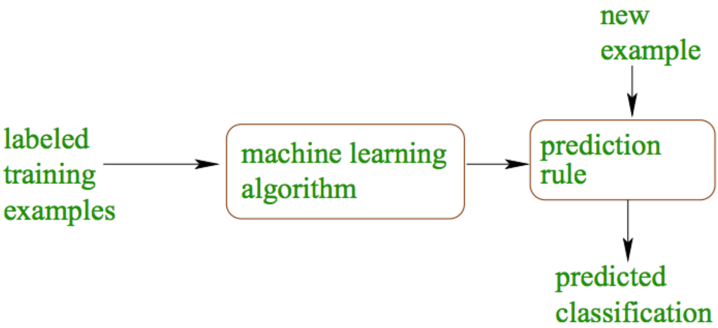
- Cover the underlying theory of the methods used
- Cover many existing algorithms, in particular will not cover neural networks or deep learning

Types of Machine Learning

- Supervised learning:
  - regression, classification
- Unsupervised learning:
  - clustering, dimensionality reduction, association rule mining, outlier detection
- Reinforcement learning:
  - learning to act in an environment

Supervised learning in a nutshell

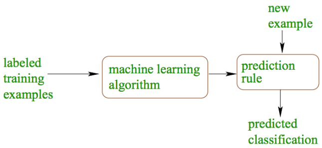
Typical “batch” supervised machine learning problem..



Prediction rule = model

Try it!

Examples are animals

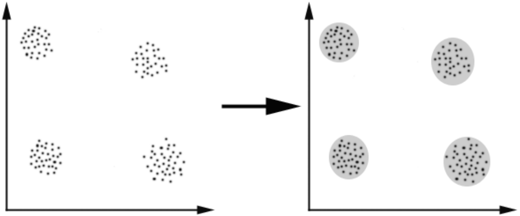


- positive training examples: bat, leopard, zebra, mouse
- negative training examples: ant, dolphin, sea lion, shark, chicken

Come up with a classification rule, and predict the “class” of: tiger, tuna.

Unsupervised learning

Clustering, association rule mining, dimensionality reduction, outlier detection



ML in practice

Actually, there is much more to it ..

- Understand the domain, prior knowledge, goals
- Data gathering, integration, selection, cleaning, pre-processing
- Create models from data (machine learning)
- Interpret results
- Consolidate and deploy discovered knowledge
- ... start again!

ML in practice

Actually, there is much more to it ..

- Understand the domain, prior knowledge, goals
- Data gathering, integration, selection, cleaning, pre-processing
- Create models from data (machine learning)
- Interpret results
- Consolidate and deploy discovered knowledge
- ... start again!

Representing objects

Features or attributes, and target values

Typical representation for supervised machine learning:

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	7.0	3.2	4.7	1.4	versicolor
4	6.1	2.8	4.0	1.3	versicolor
5	6.3	3.3	6.0	2.5	virginica
6	7.2	3.0	5.8	1.6	virginica

- Features or attributes: sepal length, sepal width, petal length, petal width
- Target value (class): species

Main objective in classification: predict class from features values

Some basic terminology

The following are terms that should be clear:

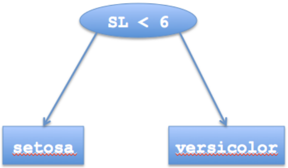
- dataset
- features
- target values (for classification)
- example, labelled example (a.k.a. sample, datapoint, etc.)
- class
- model (hypothesis)
- learning, training, fitting
- classifier
- prediction

Today we will cover decision trees and the nearest neighbors algorithm

Decision Tree: Hypothesis Space

A function for classification

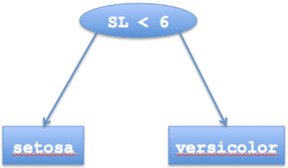
	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	7.0	3.2	4.7	1.4	versicolor
4	6.1	2.8	4.0	1.3	versicolor
5	6.3	3.3	6.0	2.5	virginica
6	7.2	3.0	5.8	1.6	virginica
7	5.7	2.8	4.1	1.3	?

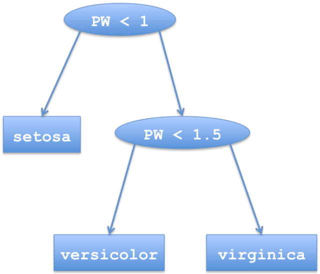


Decision Tree: Hypothesis Space

A function for classification

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	7.0	3.2	4.7	1.4	versicolor
4	6.1	2.8	4.0	1.3	versicolor
5	6.3	3.3	6.0	2.5	virginica
6	7.2	3.0	5.8	1.6	virginica
7	5.7	2.8	4.1	1.3	?

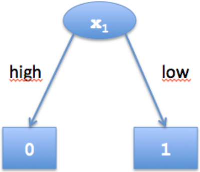


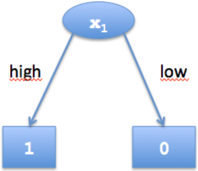


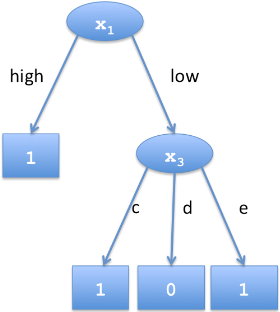
Decision Tree: Hypothesis Space

A function for classification

	x_1	x_2	x_3	x_4	class
1	high	1	c	good	0
2	high	0	d	bad	0
3	high	0	c	good	1
4	low	1	c	bad	1
5	low	1	e	good	1
6	low	1	d	good	0



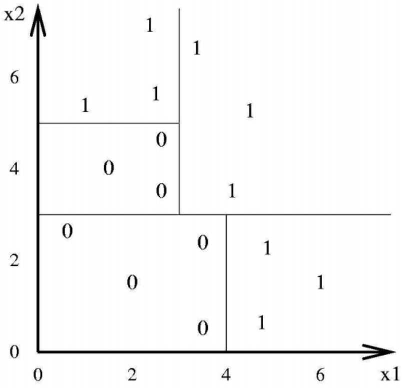


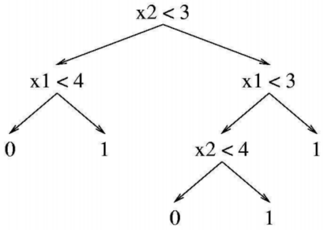


Exercise: Count many *classification errors* each tree makes.

Decision Tree Decision Boundary

Decision trees divide the feature space into axis-parallel rectangles and label each rectangle with one of the classes.





The greedy algorithm for boolean features

GrowTree(S)

  if $y = 0$ for all $(\mathrm{x}, y) \in S$ then

    return new *leaf*(0)

  else if $y = 1$ for all $(\mathrm{x}, y) \in S$ then

    return new *leaf*(1)

  else

    choose best attribute x_j

    $S_0 \leftarrow$ all (x, y) with $x_j = 0$

    $S_1 \leftarrow$ all (x, y) with $x_j = 1$

    return new *node*(GrowTree(S_0), GrowTree(S_1))

  end if

The greedy algorithm for boolean features

```
GrowTree(S)
  if y = 0 for all (x, y) ∈ S then
      return new leaf(0)
  else if y = 1 for all (x, y) ∈ S then
      return new leaf(1)
  else
      choose best attribute x_j
      S_0 ← all (x, y) with x_j = 0
      S_1 ← all (x, y) with x_j = 1
      return new node(GrowTree(S_0), GrowTree(S_1))
  end if
```

What about attributes that are non-boolean?

Multi-class categorical attributes

In the examples we have seen cases with *categorical* (a.k.a. discrete) attributes, in this case we can chose to

- Do a multiway split (like in the examples), or
- Test single category against others
- Group categories into two disjoint subsets

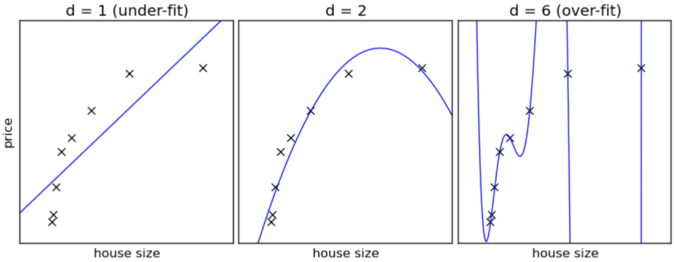
Numerical attributes

- Consider thresholds using observed values, and split accordingly

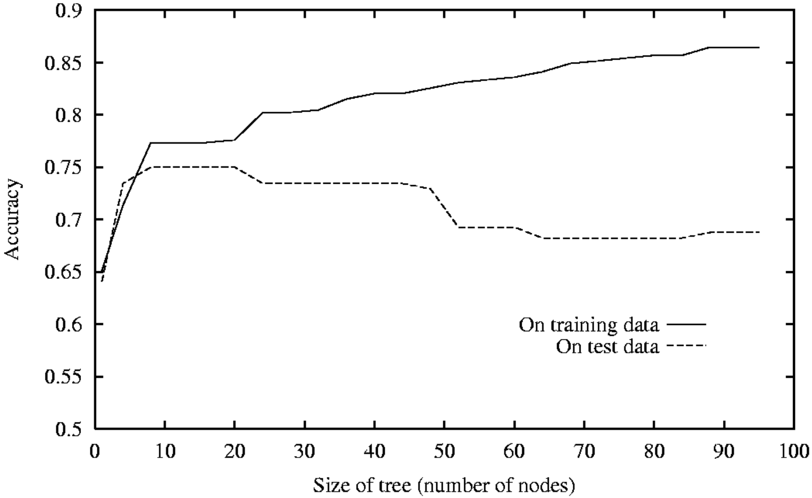
The problem of overfitting

- Define training error of tree T as the number of mistakes we make on the training set
- Define test error of tree T as the number of mistakes our model makes on examples it has not seen during training

Overfitting happens when our model has very small training error, but very large test error



Overfitting in decision tree learning



Avoiding overfitting

Main idea: prefer smaller trees over long, complicated ones.

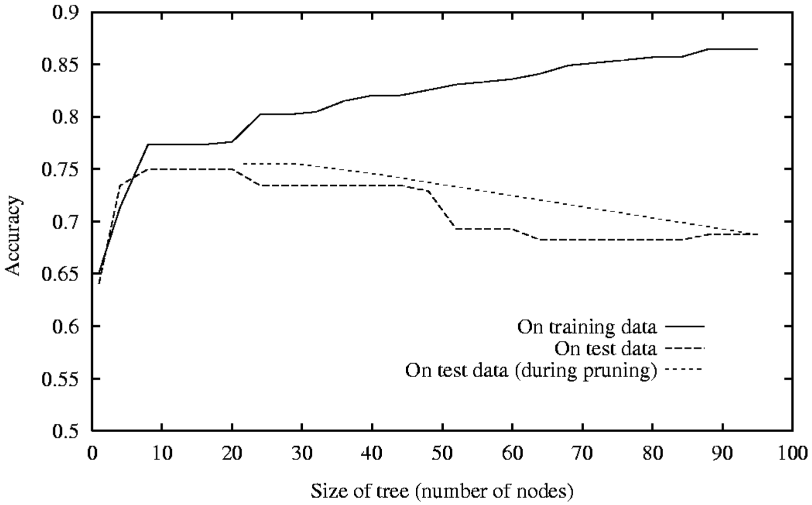
Two strategies

- Stop growing tree when split is not *statistically significant*
- Grow full tree, and then post-prune it

Reduced-error pruning

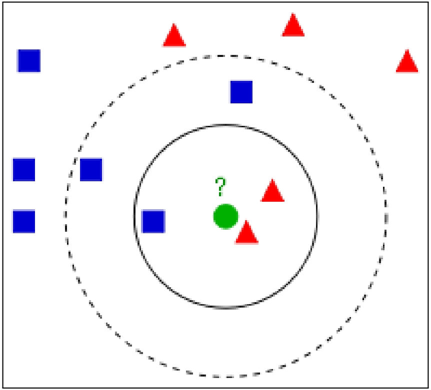
1. Split data into disjoint training and validation set
2. Repeat until no further improvement of validation error
   - Evaluate validation error of removing each node in tree
   - Remove node that minimizes validation error the most

Pruning and effect on train and test error



Nearest Neighbor

- k-NN, parameter k is number of neighbors to consider
- prediction is based on majority vote of k closest neighbors



How to find "nearest neighbors"

Distance measures

Numeric attributes

- Euclidean, Manhattan, L^n-norm

$$L^n(\mathrm{x}^1, \mathrm{x}^2) = \sqrt[n]{\sum_{i=1}^{dim} \left|\mathrm{x}_i^1 - \mathrm{x}_i^2\right|^n}$$

- Normalized by range, or standard deviation

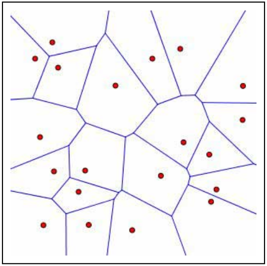
Categorical attributes

- Hamming/overlap distance
- Value Difference Measure

$$\delta(val_i, val_j) = \sum_{c \in classes} \left|P(c|val_i) - P(c|val_j)\right|^n$$

Decision boundary for 1-NN

Voronoi diagram



- Let S be a training set of examples
- The Voronoi cell of $\mathrm{x} \in S$ is the set of points in space that are closer to x than to any other point in S
- The Region of class C is the union of Voronoi cells of points with class C

Distance-Weighted k-NN

A generalization

Idea: put more weight to examples that are close

$$\hat{f}(\mathrm{x}') \leftarrow \frac{\sum_{i=1}^{k} w_i f(\mathrm{x}^i)}{\sum_{i=1}^{k} w_i}$$

where

$$w_i \stackrel{\text{def}}{=} \frac{1}{d(\mathrm{x}', \mathrm{x}^i)^2}$$

Avoiding overfitting

- Set k to appropriate value
- Remove noisy examples
  - E.g., remove x if all k nearest neighbors are of different class
- Construct and use prototypes as training examples

What k is best?

This is a hard question ... how would you do it?

What k is best?

This is a hard question ... how would you do it?

- Typically, we need to "evaluate" classifiers, namely, how well they make predictions on unseen data
- One possibility is by splitting available data into training (70%) and test (30%) – of course there are other ways
- Then, check how well different options work on the test set

... more on this this Friday in the lab session!