Introduction to Machine Learning

DENSYS school

June 2021

Selim El Mekki, Bertrand Cornélusse

LIÈGE université
Sciences Appliquées

Outline

- Introduction
- Types of machine learning problems
- Supervised learning
  - Simple models, overfitting & underfitting, Cross validation
  - k-NN
  - Decision tree
- Steps to solve a SL problem
- Practice session: room occupancy prediction

For more info

Short intro in my PhD thesis: https://orbi.uliege.be/handle/2268/82167

An introductory course: https://people.montefiore.uliege.be/lwh/AIA/

What is Machine Learning

Machine learning is the study of computer algorithms that improve automatically through experience and by the use of data.

- Common applications:
  - Image recognition
  - Email Spam & Malware filtering
  - Speech recognition
  - Social media customized advertising
  - Netflix's recommandations
  - Medical diagnosis

Why Machine Leaning now?

- More Data
- Faster compute engines
- Algorithms (old and new)
- Machine learning softwares

Types of machine learning

- Supervised learning
- Unsupervised learning
- Reinforcement Learning
- And many variants ...

Supervised Learning

Goal: from the dataset, find a function f of the inputs that approximates at best the output

Inputs				Output
X1	X2	X3	X4	Y
-0.61	-0.43	Y	0.51	Healthy
-2.3	-1.2	N	-0.21	Disease
0.33	-0.16	N	0.3	Healthy
0.23	-0.87	Y	0.09	Disease
-0.69	0.65	N	0.58	Healthy
0.61	0.92	Y	0.02	Disease

training set

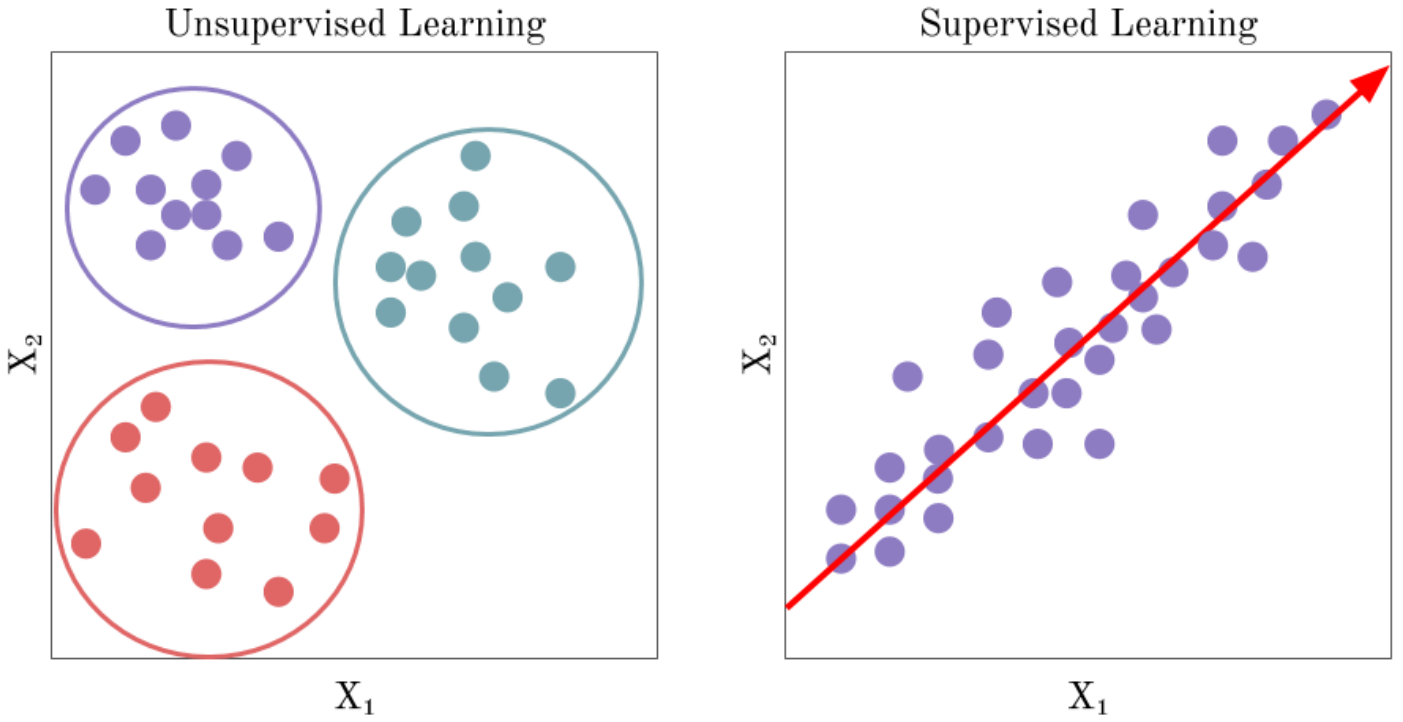
→ Learn a function f that best maps inputs to output from the training set

X1	X2	X3	X4	Y
-0.71	-0.27	T	-0.72	Healthy
-2.3	-1.2	F	-0.92	Disease
0.42	0.26	F	-0.06	Healthy
0.84	-0.78	T	-0.3	Disease
-0.55	-0.63	F	-0.02	Healthy
0.07	0.24	T	0.4	Disease
0.75	0.49	F	-0.88	?

→ Make a prediction using f

Unsupervised Learning

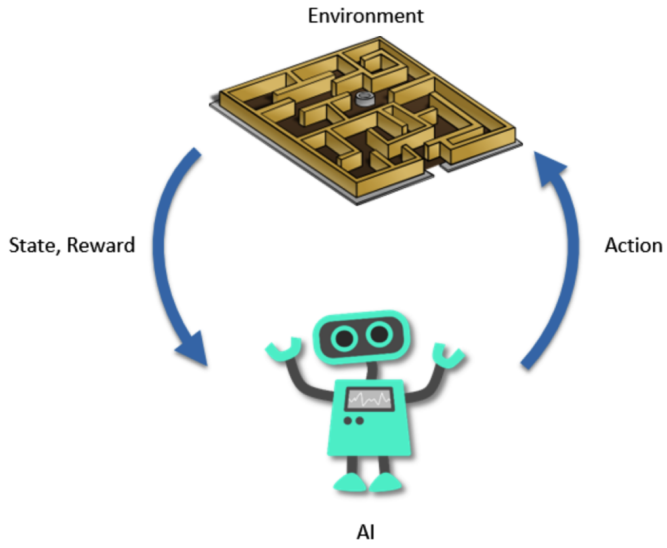
Unsupervised learning does not use labeled data.
Instead of making predictions, the goal of unsupervised learning is to find the underlying structure of dataset and group that data according to similarities.



LIÈGE universit
Sciences Ap

Reinforcement Learning

- Learning from interactions with an environment
- Goal: learn to take the actions in any given state in order to maximize a reward function by interacting with an environment
- Example:
  - State: the robot position in the labyrinth
  - Actions: Go ahead, turn right, turn left
  - Reward: Get a reward each time the robot manages to get out of the labyrinth
  - Goal: learn from experience how to get out of the labyrinth as fast as possible (i.e. find the optimal action to pick for any given state)



Supervised Learning

Supervised learning (SL)

- In the SL paradigm, the data is organized as a set of N objects described by their input features and their output label,

$$\{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}.$$

- The features x_i are easily obtained by observation, while the output value y_i is provided by an expert, or is the result of a difficult observation and/or computation.

Mapping, loss function, expected loss

SL algorithms search for a **mapping** $h : \mathcal{X} \to \mathcal{Y}$ between the feature space and the output space that generalizes well to elements for which the output value has not been observed.

To this end, we define a **loss function**

$$\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$$

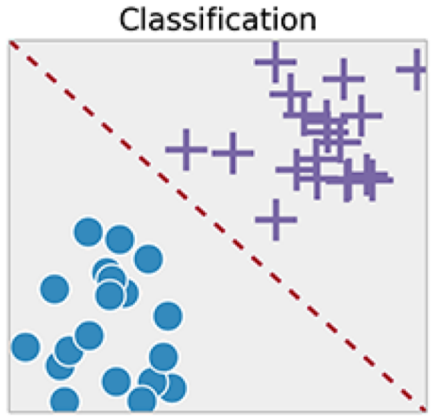
and search for a mapping h which minimizes the ***expected loss***

$$\mathbb{E}_{\mathbb{P}_{X,Y}} \{\ell(h(x), y)\}$$

Classification and Regression

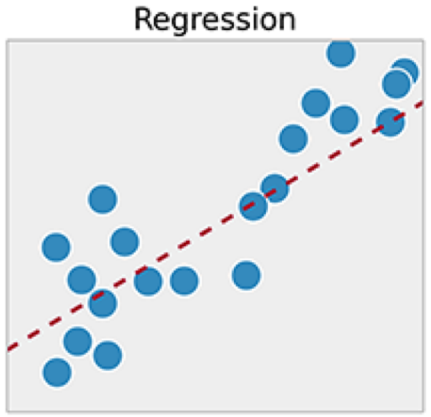
Classification is the task of approximating a mapping function (h) from input variables (X) to discrete output variables (y).

Examples: Predict if an email is a spam or not, predict if a room is occupied or not



Regression predictive modeling is the task of approximating a mapping function (h) from input variables (X) to a continuous output variable (y).

Examples: Predict the price of a house, timeseries forecasting



Empirical loss minimization

As we usually do not know the joint distribution of the objects, we instead search for a function that minimizes an estimate

$$\frac{1}{N}\sum_{i=1}^{N}\ell(h(x_i),y_i)$$

of the expected loss, also called the ***empirical loss***.

Model selection

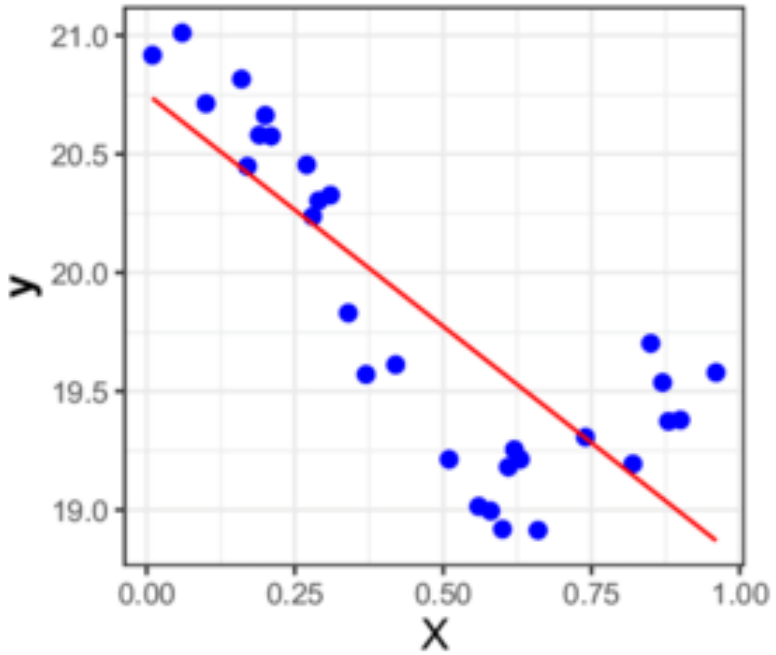
- By selecting a SL method (kNN, SVM, etc.), we restrict and parameterize the **hypothesis space** of functions $h_\theta(x)$
- But how do we pick up a method / model, and how do we optimize its parameters

Linear model

Polynomial of degree 1 regression model
with 1 input and 1 output

$$h(X) = \omega_0 + \omega_1 X$$

During the learning phase, find the values for ω_0, ω_1 in order to best fit the training set

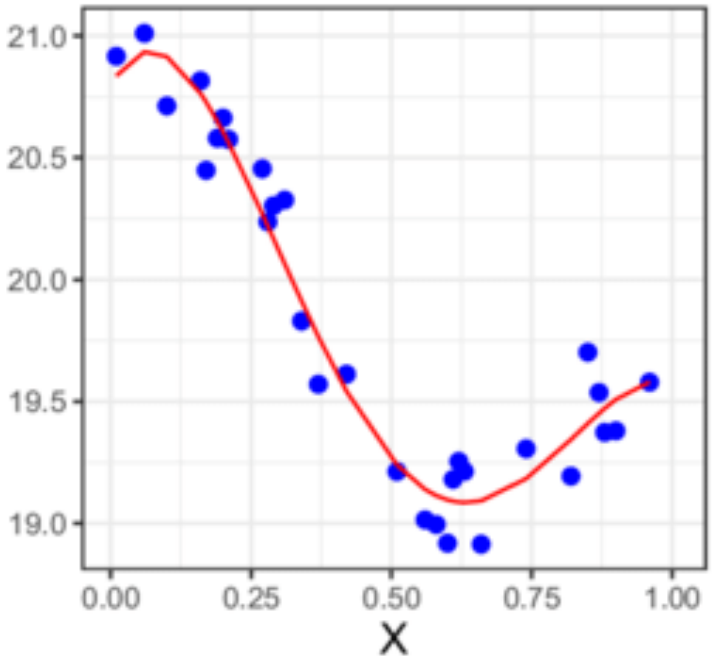


4th degree polynomial

Polynomial of degree 4 regression model
with 1 input and 1 output

$$h(X) = \omega_0 + \omega_1 X + \omega_2 X^2 + \omega_3 X^3 + \omega_4 X^4$$

During the learning phase, find the values for ω_0, ω_1, ω_2, ω_3, $\omega_4$in order to best fit the training set

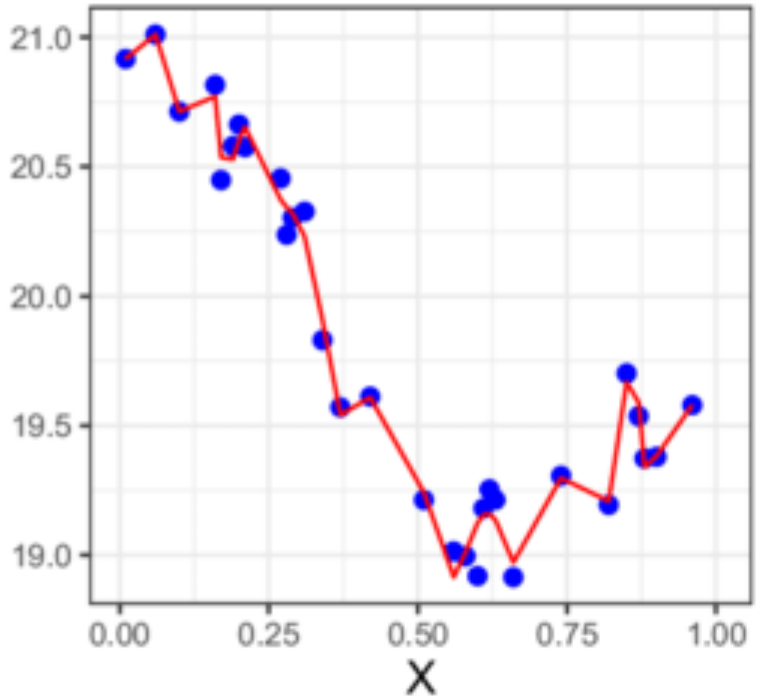


20th degree polynomial

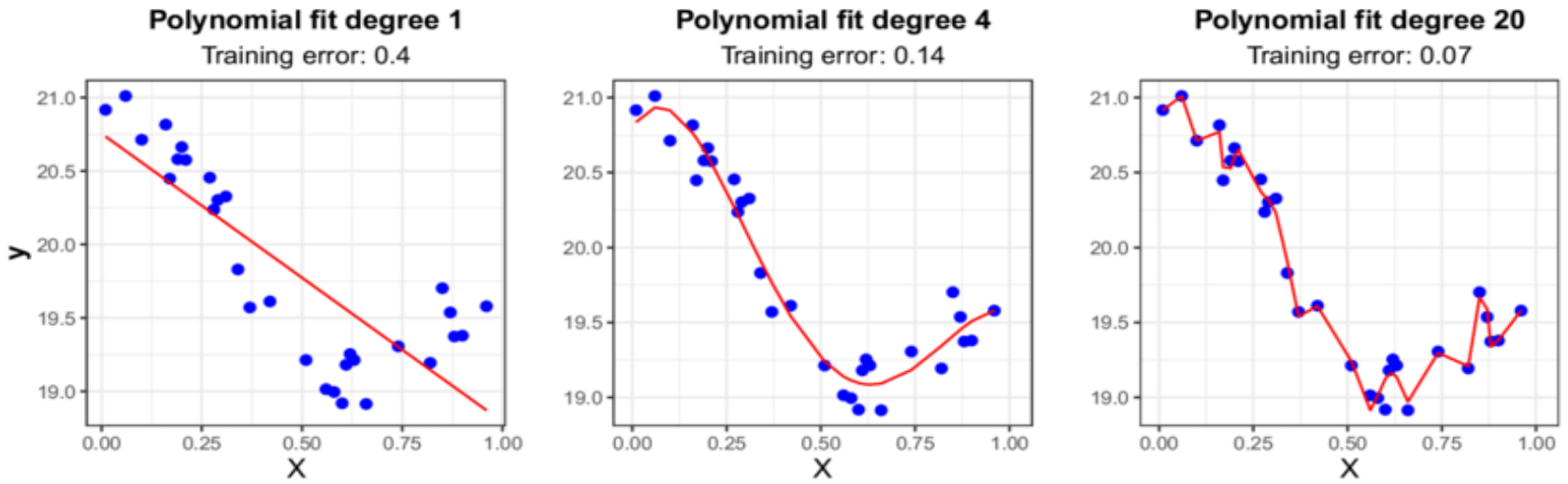
Polynomial of degree 20 regression model
with 1 input and 1 output

$$h(X) = \sum_{i=0}^{20} \omega_i X^i$$

During the training phase, find the values for ω_0, ..., ω_{20} in order to best fit the training set



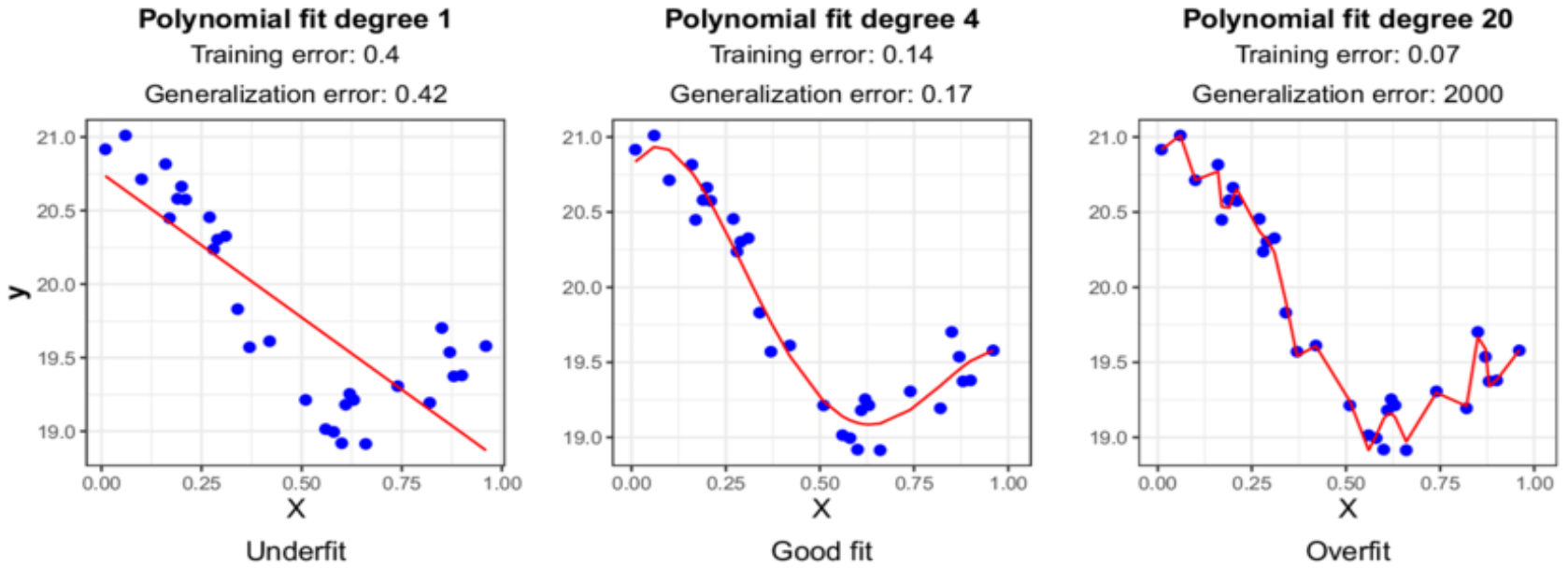
Model selection



- Linear model error on the training set: 0.4
- 4th degree polynomial error on the training set: 0.14
- 20th degree polynomial model error on the training set: 0.07

You need to evaluate your model on unseen data ⟶ Test set

Model selection



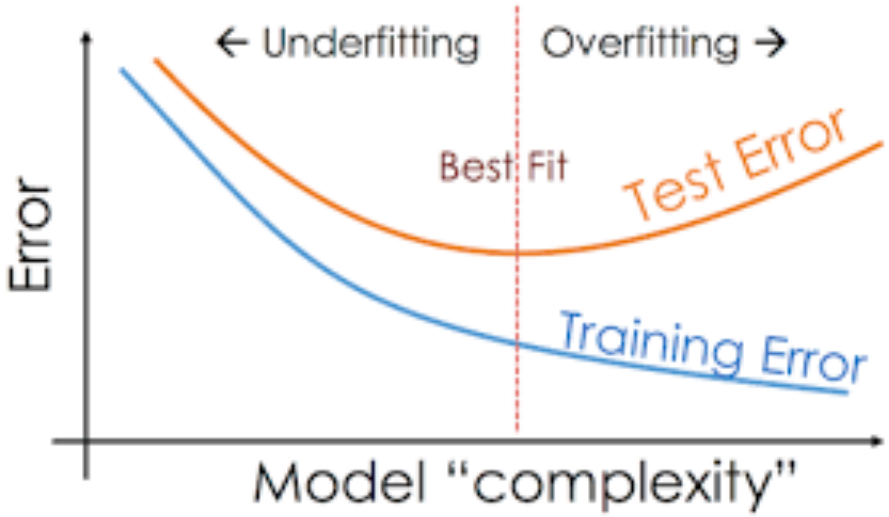
- Linear model accuracy on the test set: 0.42
- 4th Degree polynomial accuracy on the test set: 0.17
- 20th Degree polynomial accuracy on the test set: 2000

Linear model is underfitting while 20th degree polynomial is overfitting, the best fit for this problem is the 4th degree polynomial

Overfitting and underfitting

We can consider that data is composed of :

- A main signal
- A noise signal



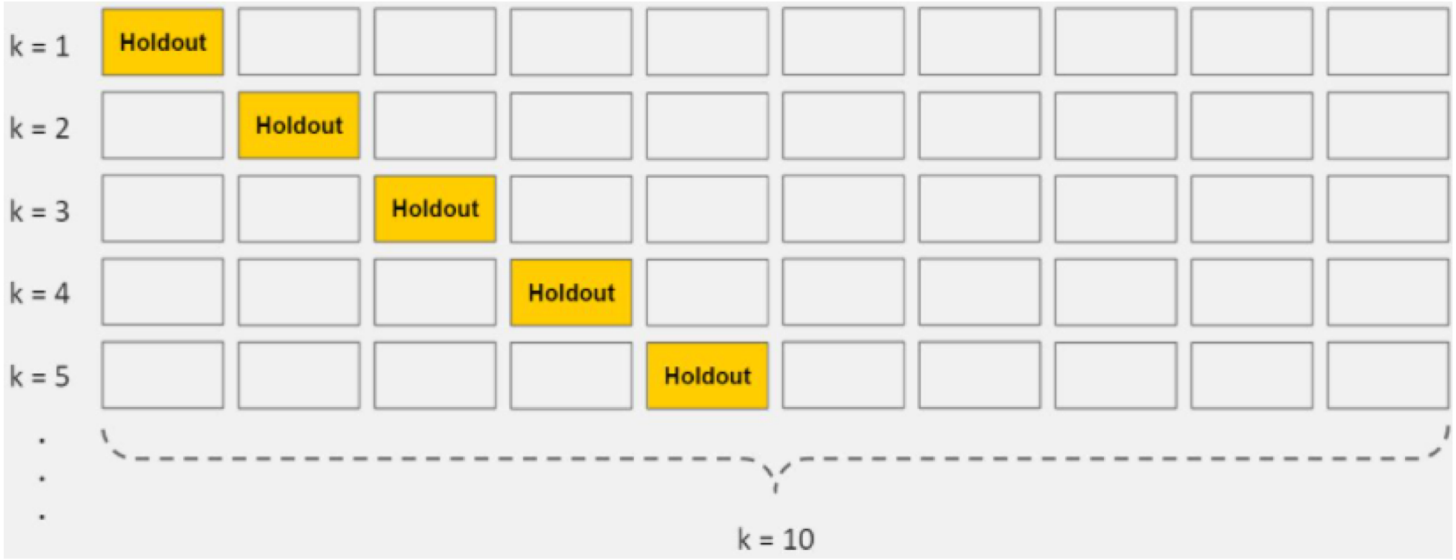
Overfitting happens when the model captures the noise along with the main signal in data. (Linear model)

Underfitting happens when a model is unable to capture the main signal in data. (20th degree polynomial)

Cross-validation

In general, this is bad practice to adjust your model based on your test set performance. Cross-validation is solution to prevent overfitting without evaluating its model on the test set.

- Split your initial training data into k subsets.
- Iteratively train the algorithm on k-1 subsets while using the remaining fold as the “test” set (called validation set) to calculate the prediction error.
- Report the mean error over the k subsets.
- Cross-validation allows you to tune hyperparameters with only your original training set. This allows you to keep your test set as a truly unseen dataset for selecting your final model.



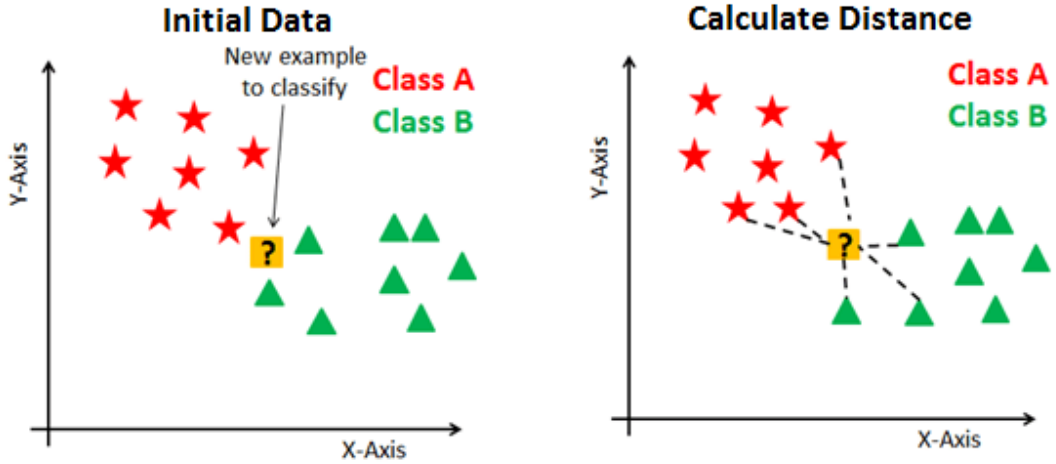
There are many SL methods

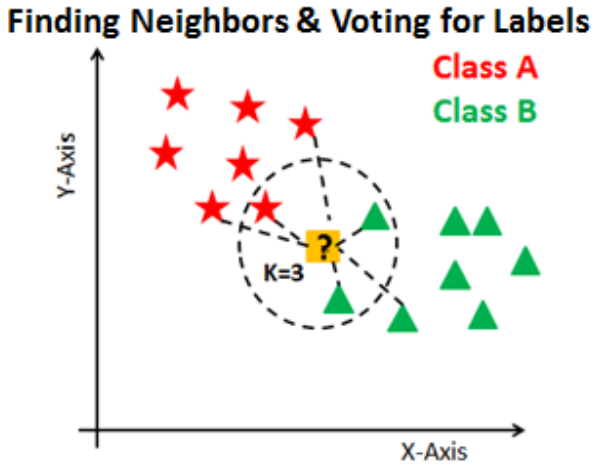
- (Linear) regression
- Nearest neighbors
- Regression and decision trees
- Support vector machines / support vector regression
- Artificial neural networks => deep learning
- Ensemble methods
- ...

k-NN

- Find the k nearest neighbors with respect to Euclidian distance

- Output the most frequent class (classification) or the average outputs (regression) among the k neighbors

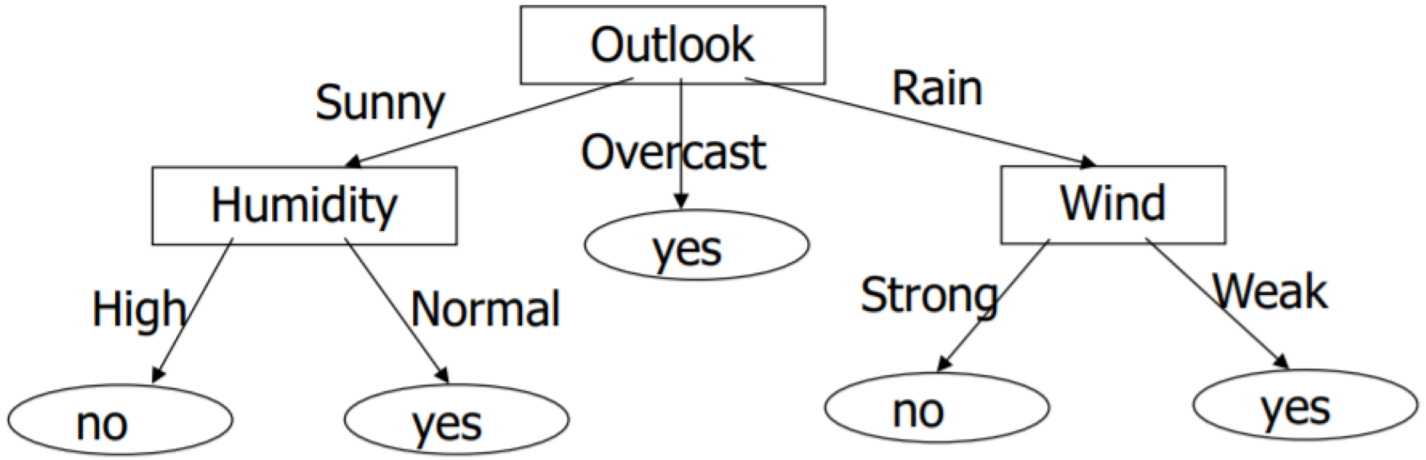




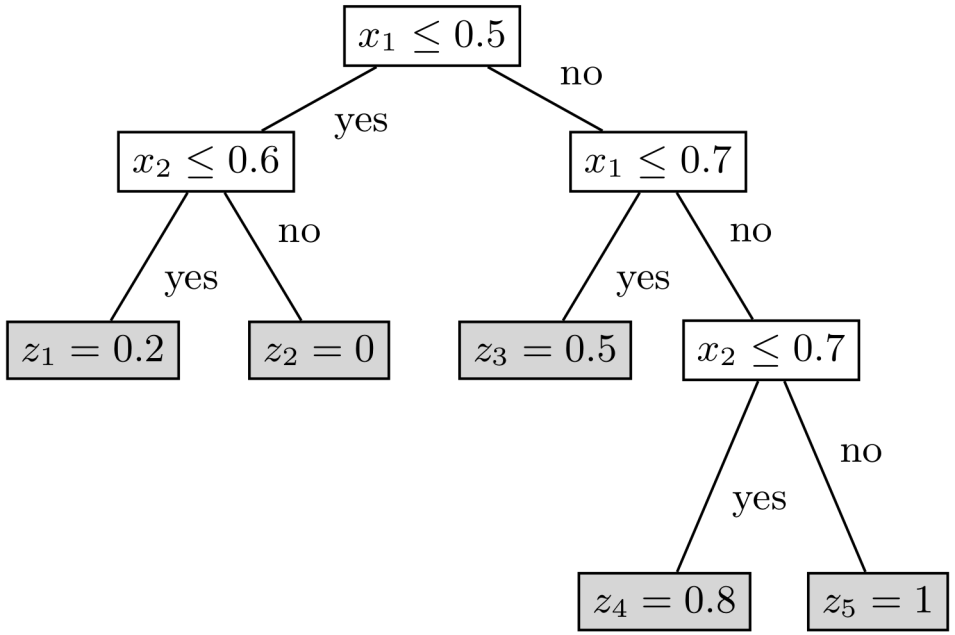
Decision Tree

- Choose the input that best separate the training set
- Split the training set according to the chosen input
- Proceed recursively until each object is correctly classified

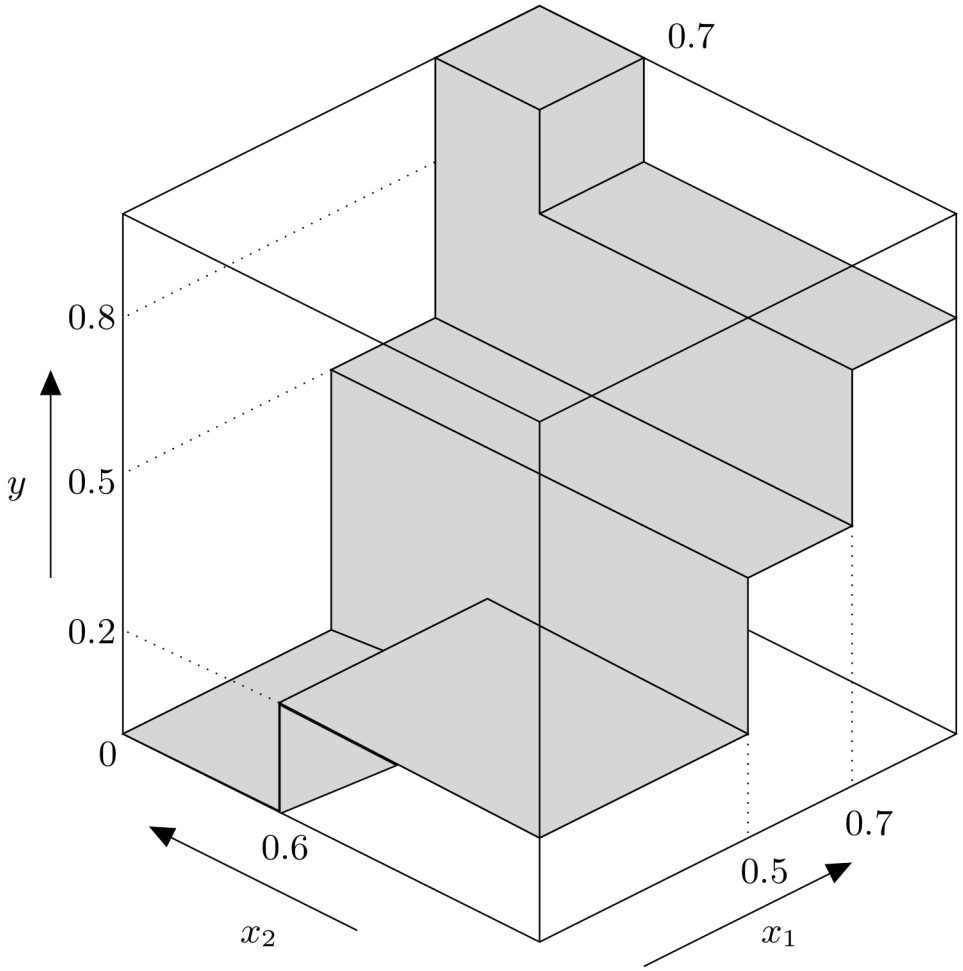
Day	Outlook	Temperature	Humidity	Wind	Play Tennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	Normal	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	High	Strong	Yes
D8	Sunny	Mild	Normal	Weak	No
D9	Sunny	Hot	Normal	Weak	Yes



A single tree example with $\mathcal{X} \in \mathbb{R}^2$ and $\mathcal{Y} \in \mathbb{R}$.



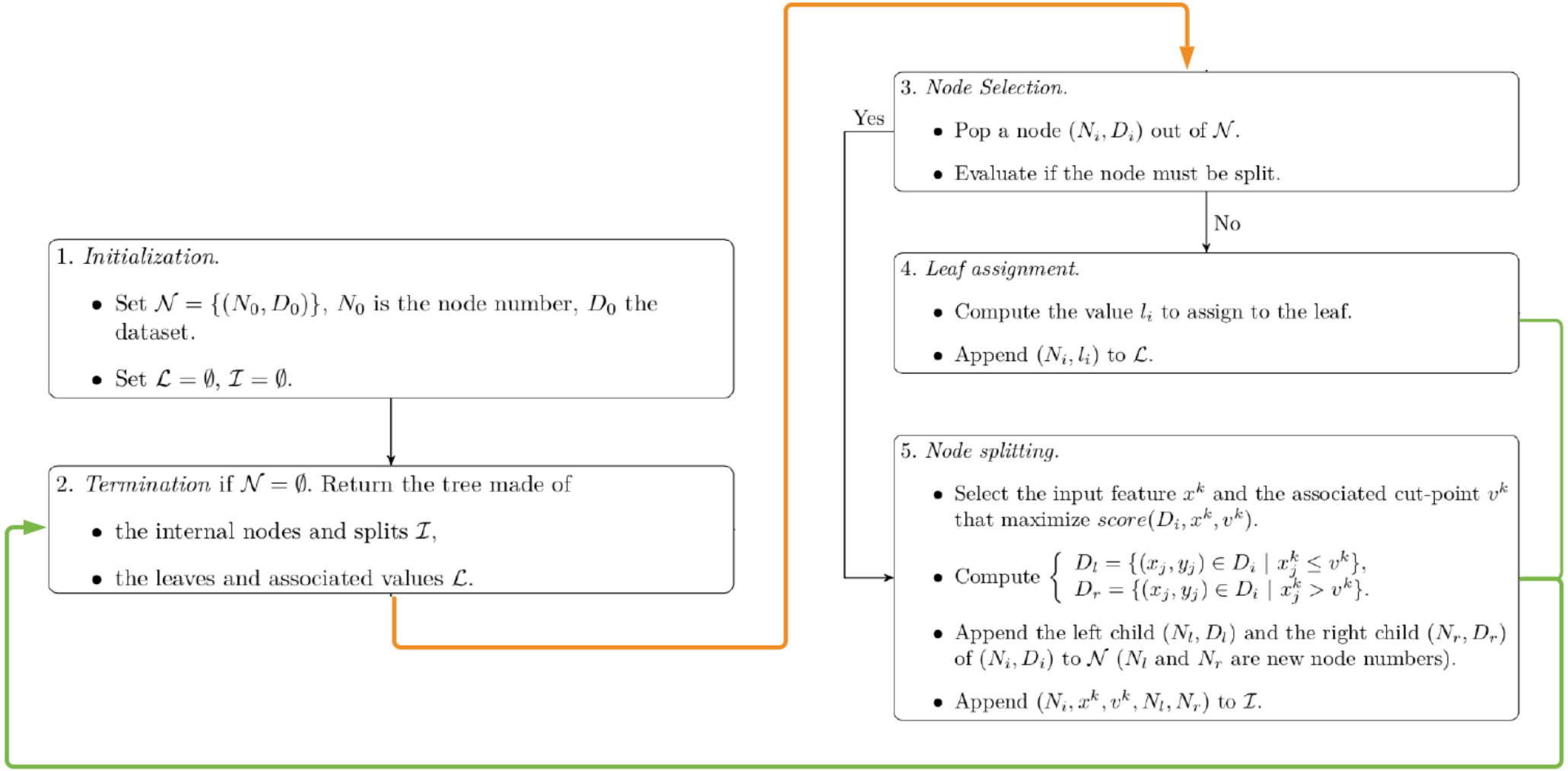
(a) A graphical model.



(b) The 3D corresponding representation.

LIÈGE université
Sciences Appliquées

Top down induction of a regression tree (TDIRT)



Comments

- If we use the mean squared error (**MSE**) **criterion** to estimate the **accuracy** of the tree predictor, defining l_i (step 4) as the **mean value** of the outputs of the objects constituting a leaf i is **optimal** with respect to empirical prediction error
- To **split** a node i (step 5), a test is defined by a **feature** x_k and a **cut-off value** v_k.
  - All the objects satisfying the test $x_k > v_k$ are assigned to the right successor node and the remaining ones are assigned to the left successor node

Splitting score

- To find the best test, a score is computed for every input feature and for every possible cut-off value. For regression trees, a typical score measure is the **relative decrease of output variance** in the two successor nodes with respect to the output variance of the current node:

$$score(D_i, x^k, v^k) = \frac{var\{y|D_i\} - \frac{|D_l|}{|D_i|}var\{y|D_l\} - \frac{|D_r|}{|D_i|}var\{y|D_r\}}{var\{y|D_i\}}.$$

- The test with the highest variance reduction is chosen

Until when do we split?

- It is more complicated to assess if a node should be split (step 3), i.e. to mitigate the complexity of the model.
- A classical way is to stop splitting when the number of objects in a node is below a threshold value,
  - but many other techniques have been developed to identify the tree of optimal com- plexity (**pruning** methods).

Steps to solve a supervised learning problem

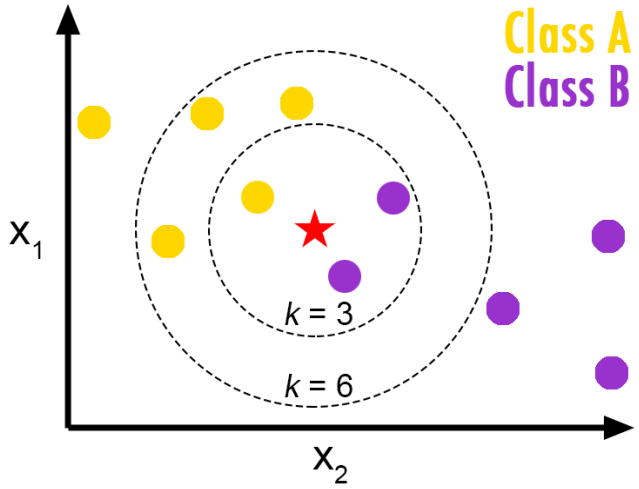
- Collect and clean the data
- Select a suitable model
- Fit your model on the training set
- Evaluate it on a validation set
- Adjust and improve your model (Overfitting? Underfitting?)

- Evaluate your model on the test set

Data collection & cleaning

- Quantity: the more, the better
- Quality: Check if the data is well distributed
- Data rescaling (faster learning)
- Check missing values, wrong labeling, bad encoding
- Feature engineering

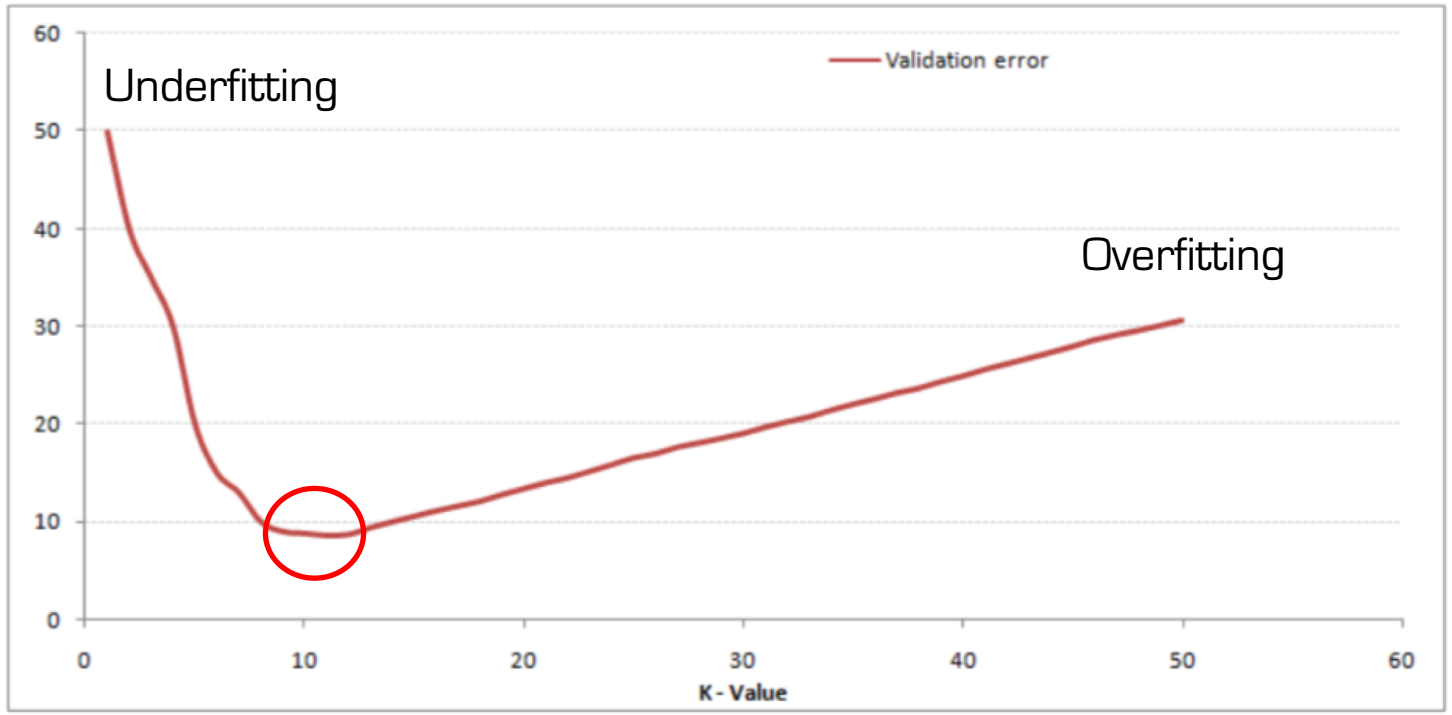
Hyperparameter tuning



- Example: K-nn model

  Which number of neighbors k to choose ?

  Use cross validation: try several values and pick the one that generalizes the best on the validation set



Classification metrics

- Confusion Matrix

n=165	**Predicted: NO**	**Predicted: YES**
Actual: NO	50	10
Actual: YES	5	100

- Accuracy

$$Accuracy = \frac{Number\ of\ Correct\ predictions}{Total\ number\ of\ predictions\ made} = \frac{TruePositive + TrueNegative}{TotalSample} = \frac{100 + 50}{165} = 0.91$$

Regression metrics

- Mean absolute error

$$\frac{1}{N}\sum_{j=1}^{N}|y_j - \hat{y}_j|$$

- Mean squared error

$$\frac{1}{N}\sum_{j=1}^{N}(y_j - \hat{y}_j)^2$$

Basic example with Sklearn

```
from sklearn import neighbors, datasets, preprocessing
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
iris = datasets.load_iris()
X, y = iris.data[:, :2], iris.target
X_train, X_test, y_train, y_test= train_test_split(X, y, random_state=33)
scaler = preprocessing.StandardScaler().fit(X_train)
X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
knn = neighbors.KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train, y_train)
y_pred = knn.predict(X_test)
accuracy_score(y_test, y_pred)
```

Imports

Collect data and spilt train/test

Rescale data

Model selection and training

Model evaluation on test set

Exercise: Room occupancy prediction

Predict the occupancy of a room (1 for occupied or 0 for not occupied) based on the following inputs:

- Temperature
- Relative humidity
- Light intensity
- CO2 concentration
- Humidity ratio, derived from temperature and relative humidity

	date	Temperature	Humidity	Light	CO2	HumidityRatio	Occupancy
2015-02-04 17:51:00	23.18	27.2720	426.0	721.25	0.004793	1
2015-02-04 17:51:59	23.15	27.2675	429.5	714.00	0.004783	1
2015-02-04 17:53:00	23.15	27.2450	426.0	713.50	0.004779	1
2015-02-04 17:54:00	23.15	27.2000	426.0	708.25	0.004772	1
2015-02-04 17:55:00	23.10	27.2000	426.0	704.50	0.004757	1
...	...	...	...	...	...	...
2015-02-04 19:25:59	22.10	27.2000	0.0	576.50	0.004475	0
2015-02-04 19:27:00	22.10	27.2000	0.0	580.00	0.004475	0
2015-02-04 19:27:59	22.10	27.2000	0.0	572.50	0.004475	0
2015-02-04 19:28:59	22.10	27.2000	0.0	567.00	0.004475	0
2015-02-04 19:30:00	22.10	27.2000	0.0	568.00	0.004475	0