# Lecture 4: Supervised learning

## Introduction to Machine Learning

Sophie Robert

L3 MIASHS | Semestre 2

2023-2024

# Reminders on previous session

## Question

Can anyone remind me of the definition of supervised learning ?
Can anyone give me some kind of problems that can be solved with supervised learning ?

This session is **hard but fundamental**: read it slowly and carefully, it is very important in order to understand the rest of the course.

# Context

# Context





To solve a prediction issue, there are two approaches:

- A field expertise approach

# Context

To solve a prediction issue, there are two approaches:

- A field expertise approach
- An automatic approach by finding patterns within the dataset and using these patterns to make decisions regarding incoming data

# Context





To solve a prediction issue, there are two approaches:

- A field expertise approach
- An automatic approach by finding patterns within the dataset and using these patterns to make decisions regarding incoming data

In this course, we will study **supervised learning**: **we are looking for a function that maps input vectors to labels based on example input-output pairs**.

# Context

To solve a prediction issue, there are two approaches:

- A field expertise approach
- An automatic approach by finding patterns within the dataset and using these patterns to make decisions regarding incoming data

In this course, we will study **supervised learning**: **we are looking for a function that maps input vectors to labels based on example input-output pairs**.

We will only be focusing on discrete problems, predicting **categorical variables**. Usually, this is referred to as **classification*** problems.

# Methodology

# Methodology

Unformally, supervised* learning consists of:

- Building a set of "examples", consisting of the data describing the individuals (**features***) and the corresponding label (**target***) acting as the ground **truth***

# Methodology

Unformally, supervised* learning consists of:

- Building a set of "examples", consisting of the data describing the individuals (**features***) and the corresponding label (**target***) acting as the **ground truth***
- Building a function mapping this data to this ground truth

# Methodology



Unformally, supervised* learning consists of:

- Building a set of ”examples”, consisting of the data describing the individuals (**features***) and the corresponding label (**target***) acting as the **ground truth***
- Building a function mapping this data to this ground truth
- Evaluating the performance of this method either on the example dataset or on some other data (test dataset)

# Methodology



## Supervised learning methodology

- Represent each individual $i$ by a tuple $(X_i, y_i), X_i \in \mathbb{R}^d, y_i \in \mathcal{Y} = \{1, ..., K\}$
- Define a function $f$ associating each $X_i$ to a label: $f(X_i) \in \mathcal{Y}$

# Example

For example, we will use the Pokemon dataset to train an algorithm to be able to predict if a Pokemon is legendary or not.

# Example

For example, we will use the Pokemon dataset to train an algorithm to be able to predict if a Pokemon is legendary or not.

## Features and target

What do you imagine will be some possible features ? will be the target to predict ?

# Evaluation

# Evaluation





How can I measure how well the algorithm performs ?

# Evaluation





How can I measure how well the algorithm performs ?

**Question**

Can you give possible evaluation metrics ? What metric did you use when your learned about regression ?

# Evaluation





How can I measure how well the algorithm performs ?

**Question**

Can you give possible evaluation metrics ? What metric did you use when your learned about regression ?

**We need objective metric(s) to assess the quality of the model we designed.**

# Confusion matrix



## Confusion* matrix

A **confusion matrix** is a specific table layout that allows visualization of **the performance of a classification algorithm**. Each row of the matrix represents the instances the predicted class while each column represents the instances of the actual class.

# Confusion matrix



## Confusion* matrix

A **confusion matrix** is a specific table layout that allows visualization of **the performance of a classification algorithm**. Each row of the matrix represents the instances the predicted class while each column represents the instances of the actual class.

The **confusion matrix** makes it easy to see when the algorithm is "confused".

| | | Actual | |
|---|---|---|---|
| | | Positive | Negative |
| Predicted | Positive | True Positive | False Positive |
| | Negative | False Negative | True Negative |

# Example: compute confusion metric

Build the confusion matrix for the following dataset:

| Actual class | Predicted class |
|---|---|
| 0 | 1 |
| 0 | 0 |
| 0 | 0 |
| 0 | 1 |
| 1 | 1 |
| 1 | 1 |
| 1 | 0 |

# Example: compute confusion metric

Build the confusion matrix for the following dataset:

| Actual class | Predicted class |
|---|---|
| 0 | 1 |
| 0 | 0 |
| 0 | 0 |
| 0 | 1 |
| 1 | 1 |
| 1 | 1 |
| 1 | 0 |

**Question**

Can you identify some examples of when you might want to treat false positives / false negatives differently ?

# Possible evaluation metrics





Many evaluation metrics rely on the confusion matrix.
Among the most famous:

- **Accuracy***: the proportion of true results among the total number of cases examined.
  $\text{Accuracy} = \frac{TP+TN}{TP+FP+FN+TN}$

# Possible evaluation metrics





Many evaluation metrics rely on the confusion matrix.
Among the most famous:

- **Accuracy***: the proportion of true results among the total number of cases examined.
  Accuracy $= \frac{TP+TN}{TP+FP+FN+TN}$
- **Precision***: the proportion of predicted positives that is truly positive.
  Precision $= \frac{TP}{TP+FP}$

# Possible evaluation metrics



**Recall***: the proportion of actual positives correctly classified.
$\text{Recall} = \frac{TP}{TP+FN}$

# Possible evaluation metrics



**Recall***: the proportion of actual positives correctly classified.
$\text{Recall} = \frac{TP}{TP+FN}$

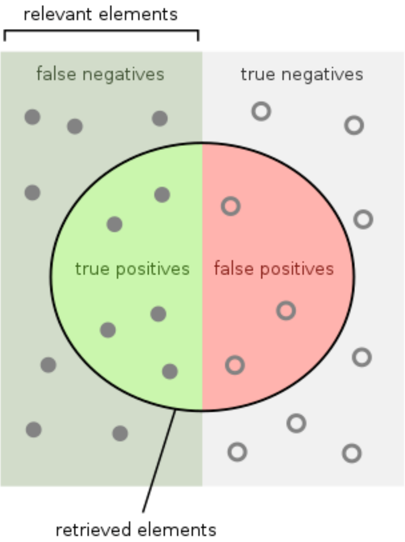
**F1-Score***: Harmonic mean of precision and recall.
$F_1 = 2 \times \frac{precision \times recall}{precision + recall}$

## Question

Can you identify at least one use-case where you want to favor recall *versus* precision rather than the other ?

# Schematic view of confusion matrix





# Example: compute metric value

Using the previous dataset, compute the following metrics:

- Accuracy:
- Precision:
- Recall:
- F1-score:

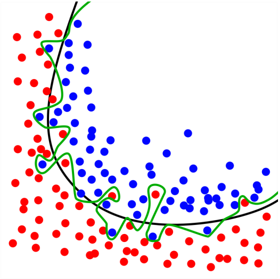
# Overfitting

# Overfitting

## Overfitting*

It is always possible to build a function that will match EXACTLY the training dataset, but it doesn't mean it will generalize well !

It is not always a good idea to have a **perfect fit** for $f$ (i.e, $f(X_i) = y_i$ for all $i$ in the training set).



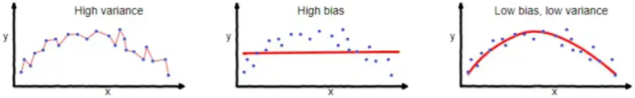
# Bias-variance trade-off

# Bias-variance trade-off

## Bias-variance trade-off

Selecting the right algorithm is a trade-off between **bias*** and **variance***.

- **Bias* error**: error from erroneous assumptions in the learning algorithm (high bias: the algorithm does not learn from the dataset)
- **Variance* error**: error from sensitivity to small fluctuations in the training set (high variance: the algorithm learns the noise from the dataset)




overfitting — underfitting — Good balance

# Training and testing dataset

# Training / testing dataset



If a 100% fit on the training set is not always the sign of a good model, how can we select the best model ?

# Training / testing dataset

If a 100% fit on the training set is not always the sign of a good model, how can we select the best model ?

## Train and test dataset

We split the dataset into two datasets:

- **Training* dataset**: dataset to build the model
- **Testing* dataset**: dataset to test the model (i.e. compute the scoring metric)

# Training / testing dataset



If a 100% fit on the training set is not always the sign of a good model, how can we select the best model ?

## Train and test dataset

We split the dataset into two datasets:

- **Training* dataset**: dataset to build the model
- **Testing* dataset**: dataset to test the model (i.e. compute the scoring metric)

## Question

What issue do you see with this approach ? What should we make sure of when splitting the dataset ?

# Cross validation

# Cross validation





## Cross validation

Cross-validation is a resampling method that uses **different portions of the data** to **test and train** a model on different iterations, in order to estimate how accurately a predictive model will perform in practice.

# Cross validation

## Cross validation

Cross-validation is a resampling method that uses **different portions of the data** to **test and train** a model on different iterations, in order to estimate how accurately a predictive model will perform in practice.

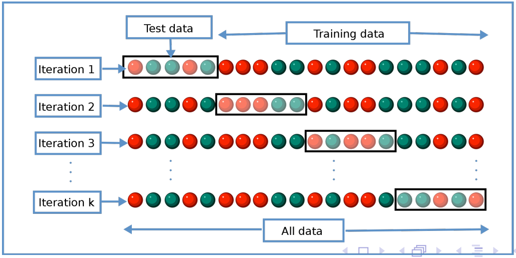
Popular approach is **k-fold cross validation**: divide the dataset into $k$ subsets, train on $k-1$ subsets, evaluate on the $k-th$ subset, repeat and aggregate performance score.

# Cross validation

## Cross validation

Cross-validation is a resampling method that uses **different portions of the data** to **test and train** a model on different iterations, in order to estimate how accurately a predictive model will perform in practice.

Popular approach is **k-fold cross validation**: divide the dataset into $k$ subsets, train on $k-1$ subsets, evaluate on the $k-th$ subset, repeat and aggregate performance score.



# Hyperparameter selection

# Hyperparameter selection

## Hyperparameters

Most algorithms have hyperparameters: parameters that condition how the algorithms behave. These parameters must be **optimized** for each different use-case.

How can we select the best hyperparameters ?

# Hyperparameter selection

## Hyperparameters

Most algorithms have hyperparameters: parameters that condition how the algorithms behave. These parameters must be **optimized** for each different use-case.

How can we select the best hyperparameters ?

A simplistic approach is simply factorial design using k-th cross validation. Other solutions:

- Genetic algorithms
- Simulated annealing
- ...

Beware of overfitting !

# Algorithms

# Algorithms





Many algorithms are available in the literature ! In this course, we'll study:

- K-nearest neighbors
- Naive Bayes
- Classification trees

# Questions



Questions ?