

Structural data will be given. From row data we start from a dataset and we define the first pre-processing pipeline. Initial dataset ⟹ dataset that can be use by the algorithm

There are subtask (**handling missing data**) some problems that needs to be fixed, have different options on how to handle the problem

**Feature extraction**: we can build new features to improve the performances of the learning algorithm.

**Selection**: select a subset of features to remove some that can reduce the performances of the algorithm

**SECOND STEP: TRAIN THE MODEL**

using a ML algorithm. In case of supervised task we split the dataset in two part

* training dataset
* test dataset

Once we have the training dataset we start pre processing (use same algorithm or a new one) take the data, transform it, split, take the train dataset and apply other transformation

Once we have finished the transformation we run the ML algorithm and obtain a set of candidate models. Once we have one we choose the best one (aka **hyperparameter optimization**) we choose the best set of parameters that maximize the performance measure.

Then we apply the train model on the test dataset: we want to measure how our model is able to generalize in unseen data. Most important part: we should discard the test dataset until the evaluation of the model.

The test data used only at the last evolution of the model.

Want to simulate the fact of having unseen data. This can be done only be use a part of the data we already have.

If you have 90% accuracy on training set but 20% on test set ⟹ overfitting. Unable to generalize in unseen data. If the performance is more or less the same and it is good ⟹ :)

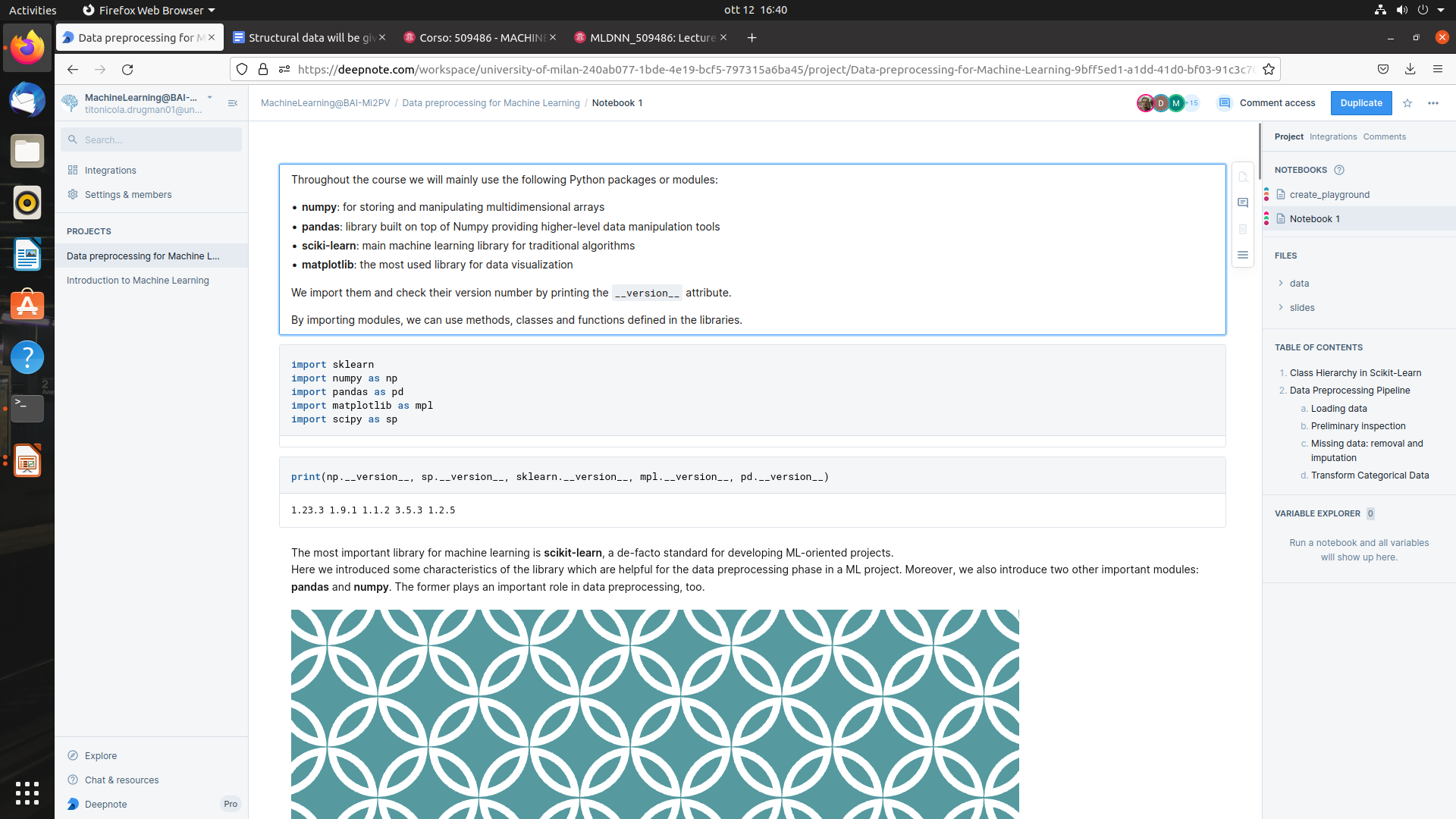
Cross-validation: sort of work around to mimic the presence of a test dataset exploiting the test dataset, is a recursive process. Split again the training dataset

We have

* training
* validation
* test

**PREPROCESSING PIPELINE 1**

<https://deepnote.com/workspace/university-of-milan-240ab077-1bde-4e19-bcf5-797315a6ba45/project/Data-preprocessing-for-Machine-Learning-9bff5ed1-a1dd-41d0-bf03-91c3c700099d/notebook/Notebook%201-69f2f3fbee504001a7a55641322b783a>



import sklearn

import numpy as np

import pandas as pd

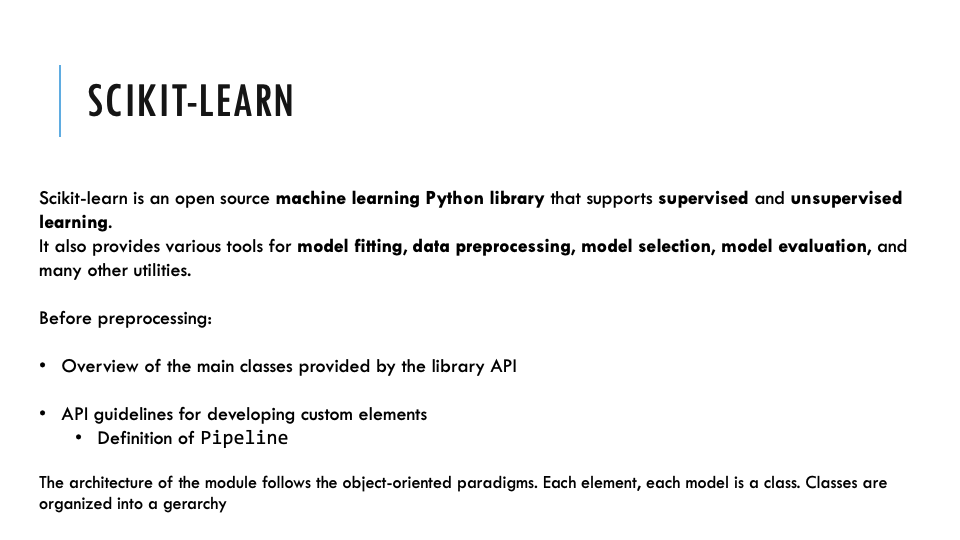
import matplotlib as mpl

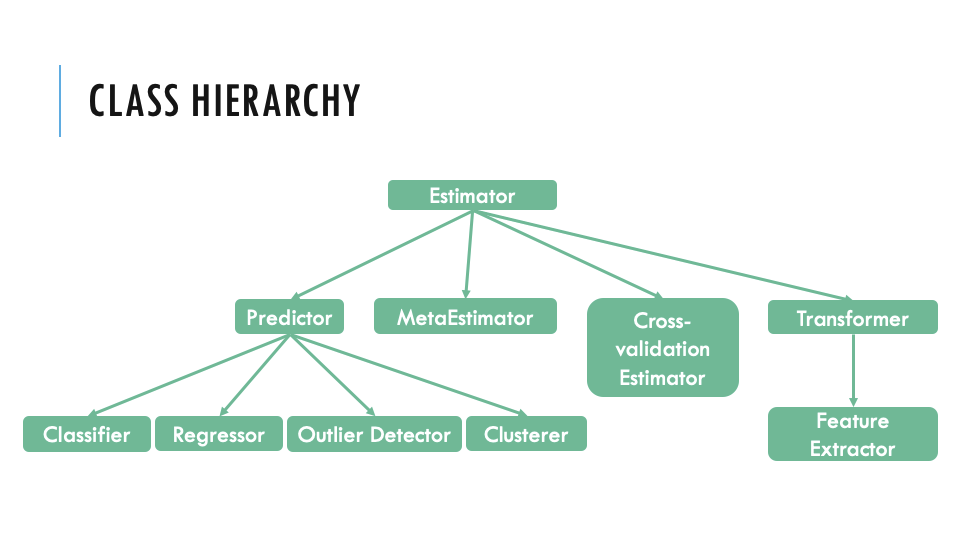
import scipy as sp

insead of numpy you write np, rename the module

then check the version of our modules. *.\_\_version\_\_* is a specific function

scikit-learn: standard for eveloping ML

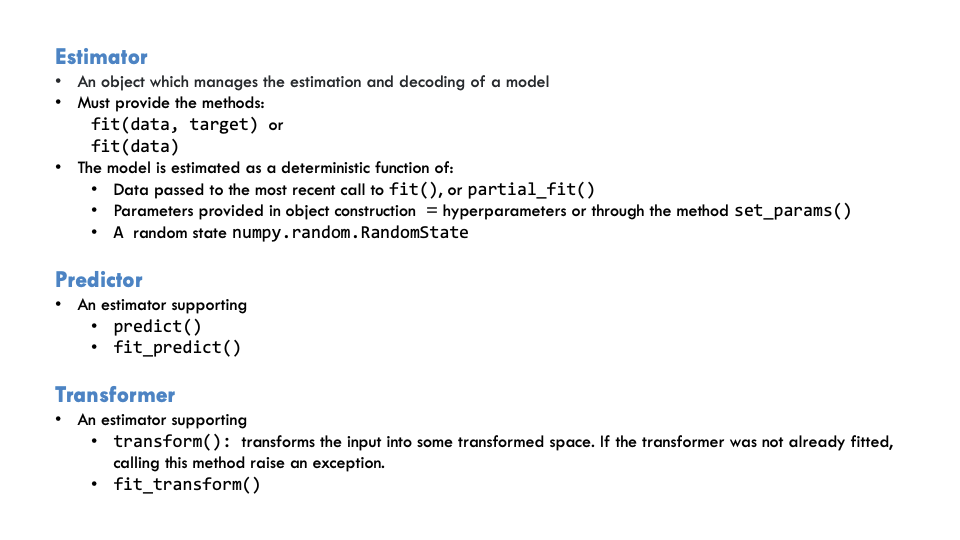




estimator: main type of object. Most classes are estimator

metaestimator: set of estimator

cross-validator estimator: another type of estimatoir



most important class

estimator

predictor

transformer

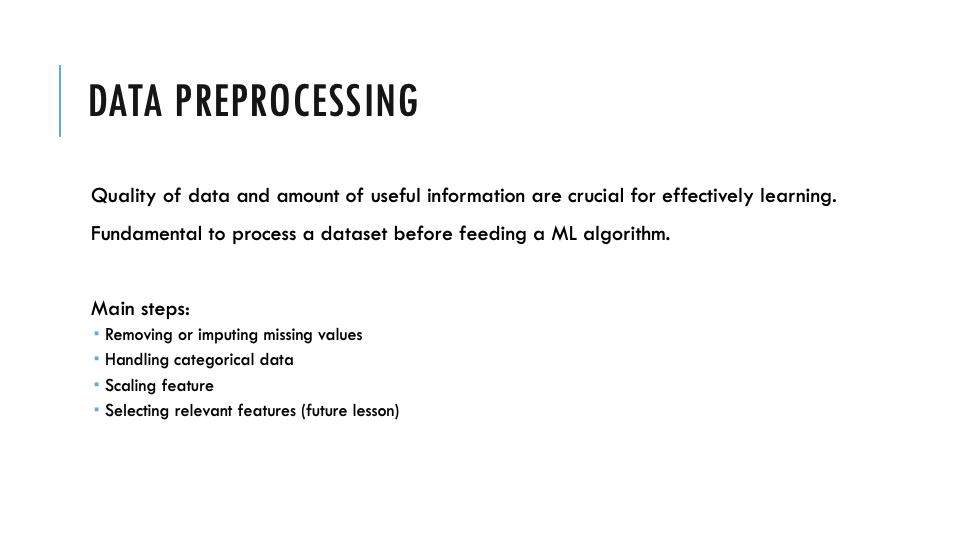
**estimator**: take data and target (supervised) or only data (unsupervised). Every class with a fit is an estimator. Can have also partial fit and the parameters can be provided by a specific method

*estimator = fit*

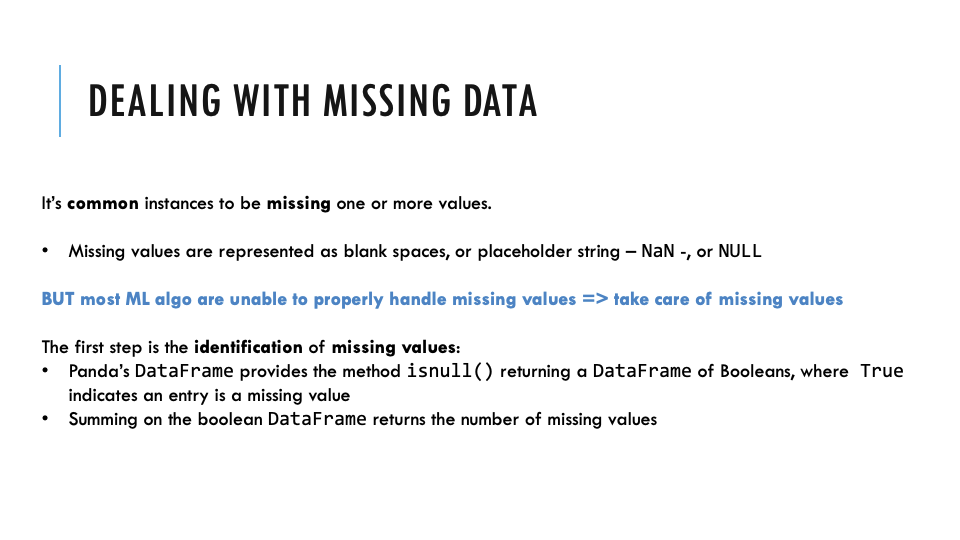
**predictor**: extension of estimator. is an estimator (have the fit method) and the predict method. There is also a combination of fit and predict.

fit = learn the parameters

**transformer**: “sibling” of predictor. Is an estimator with the method transform. It takes the input and returns a new type of dataset. Apply a transformation of the data.



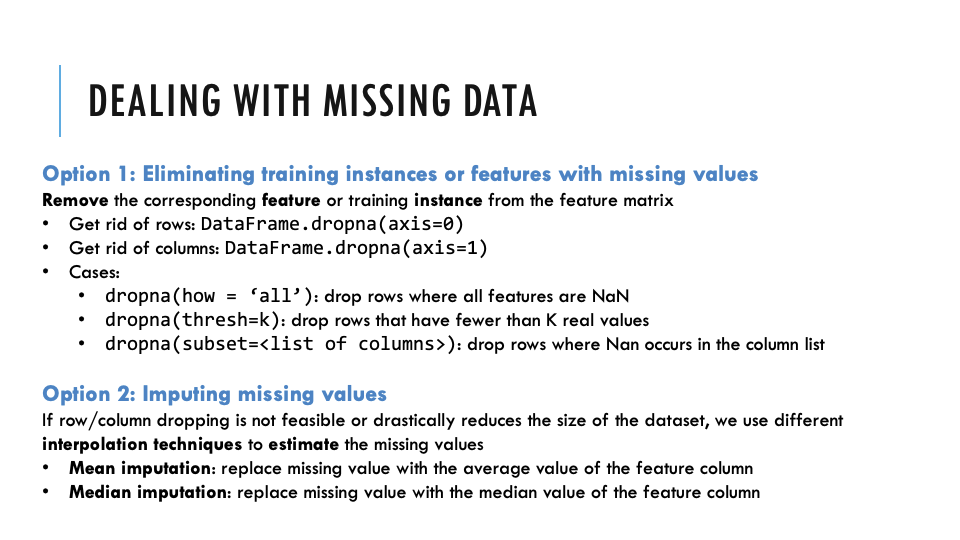
first step: load the data. Use PANDA



MISSING DATA: most algorithm are not able to handle missing data. They expect clean data

We need to identify where missing data are

**ISNULL()** return true ⟹ specific cell is missing. Missing data are indicated by an empty string or NaN o NULL



remove all the row/colums with empty

DROPNA

<https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.dropna.html>

**AXIS** (default 0 = row)

axis = 0 ⟹ row

axis = 1 ⟹ column

**HOW** (default any)

axis + how = “any” ⟹ If any NA values are present, drop that entire row or column.

axis + how = “all” ⟹ remove all the axis where all the items are missing