

# FOUNDATIONS OF STATISTICAL ANALYSIS & MACHINE LEARNING

**Amric Trudel**  
amric.trudel@epita.fr



# INTRODUCTION

---

## Amric Trudel



- McGill University graduate (Montréal, Canada)
- École 42, former president of 42AI
- Consultant at OCTO Technology





# TODAY'S SCHEDULE

---

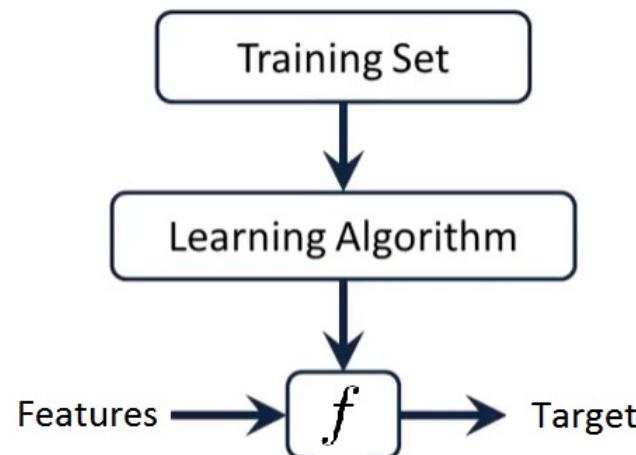
## Program

- Introduction to Machine Learning
- Tools & libraries
- Data exploration
  - Overview
  - Implementation
  - Practice

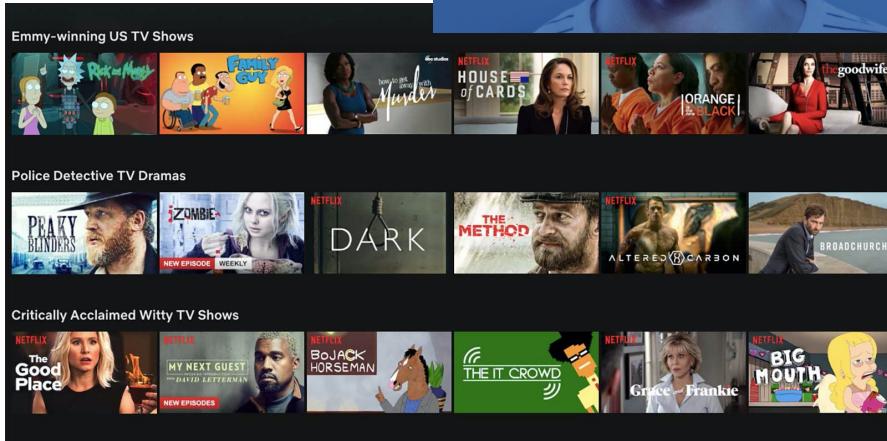
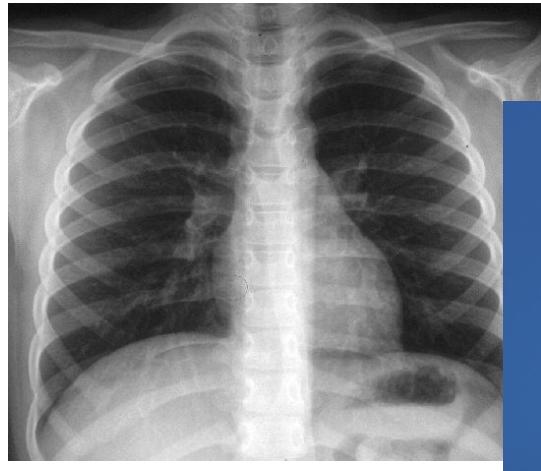
# WHAT IS ML?

A definition

- Machine Learning (ML) is a subset of artificial intelligence that encompasses techniques aiming at making predictions after having built models based on sample data.



# WHERE IS ML?

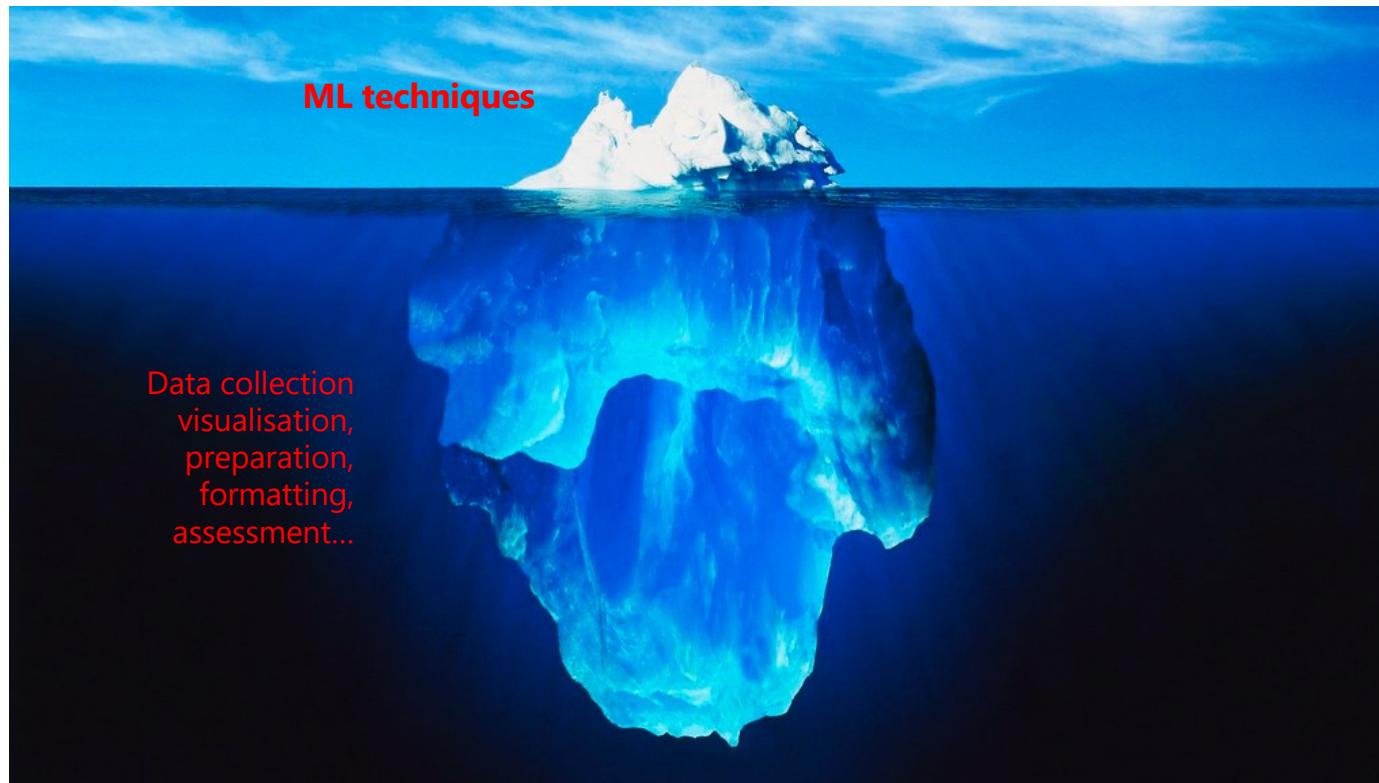


## Use cases



# HOW TO LEARN ML?

The underestimated part of ML





# ML WORKFLOW

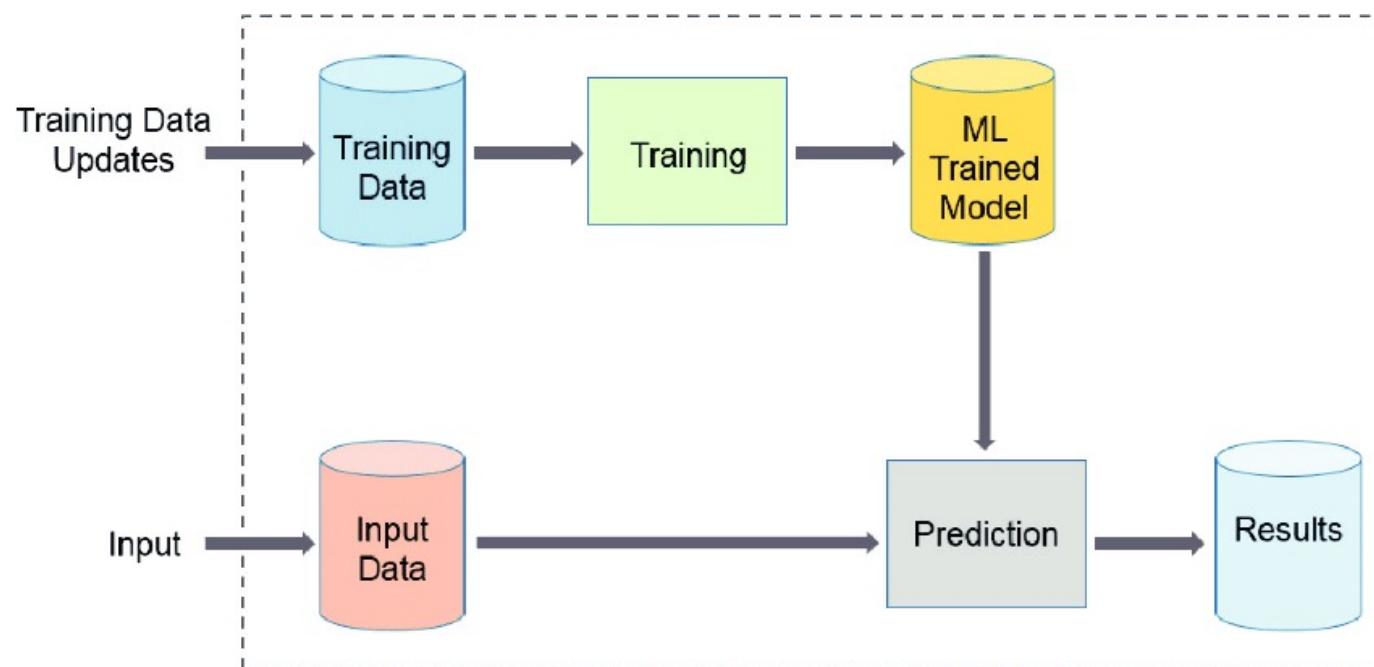
---

Usual process

- Collect data
- Clean and prepare the data
- Pick a model
- Train the model on the data
- Test and evaluate the model
- Improve the model
  
- Deploy the model to production

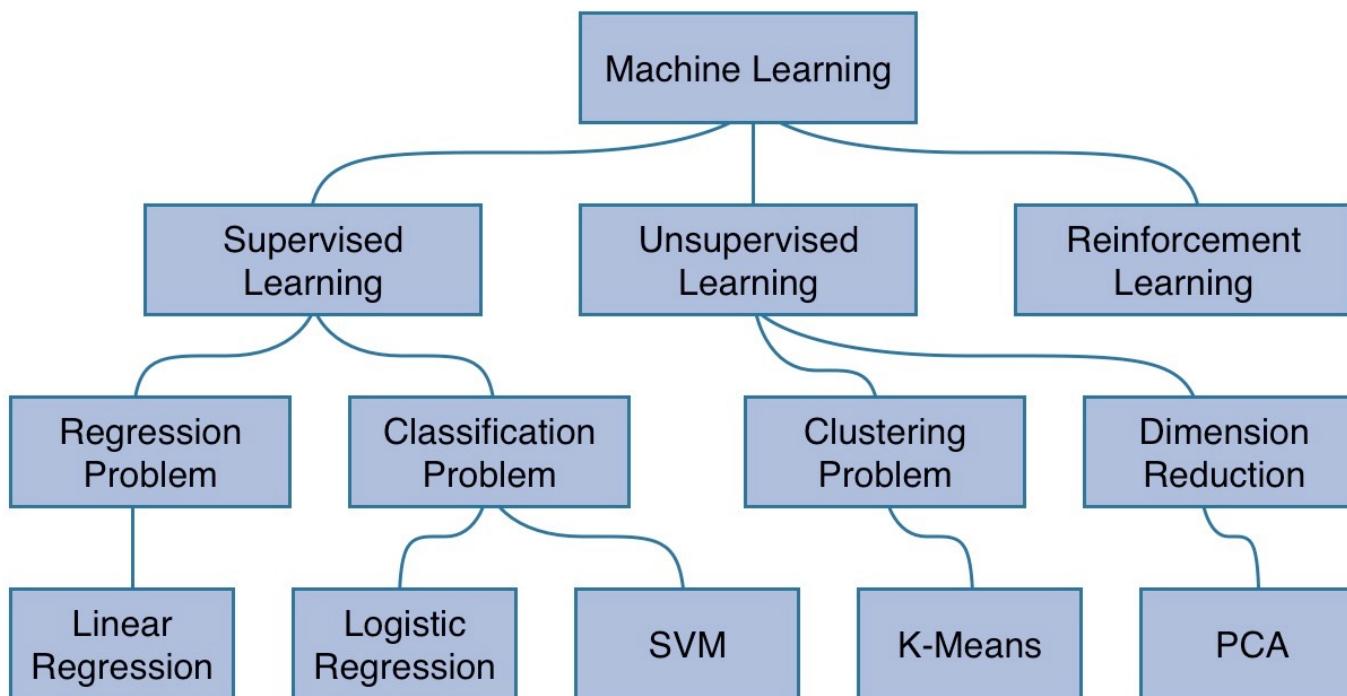
# ML WORKFLOW

Usual process



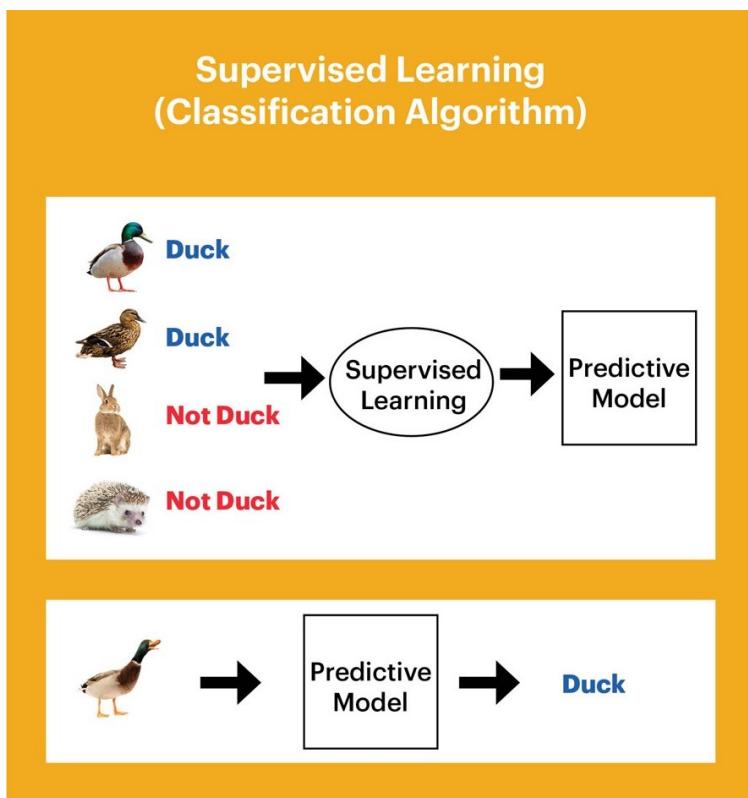
# MACHINE LEARNING

And its multiple branches



# SUPERVISED VS UNSUPERVISED LEARNING

Labeled vs. unlabeled

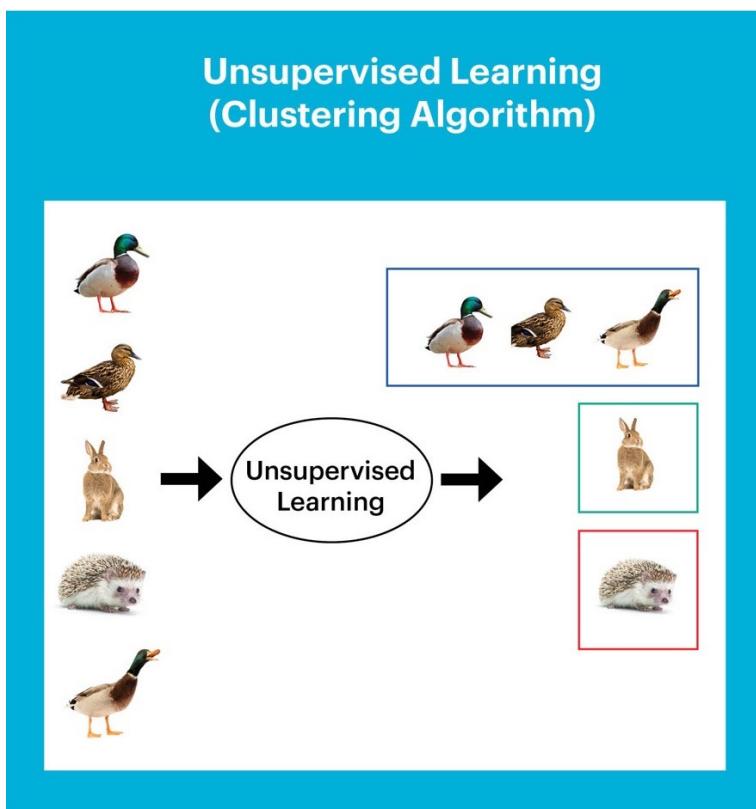


## Supervised Learning

- Data is **labeled**
- We want the algorithm to learn the **relationship** between an object's features and its label

# SUPERVISED VS UNSUPERVISED LEARNING

Labeled vs. unlabeled

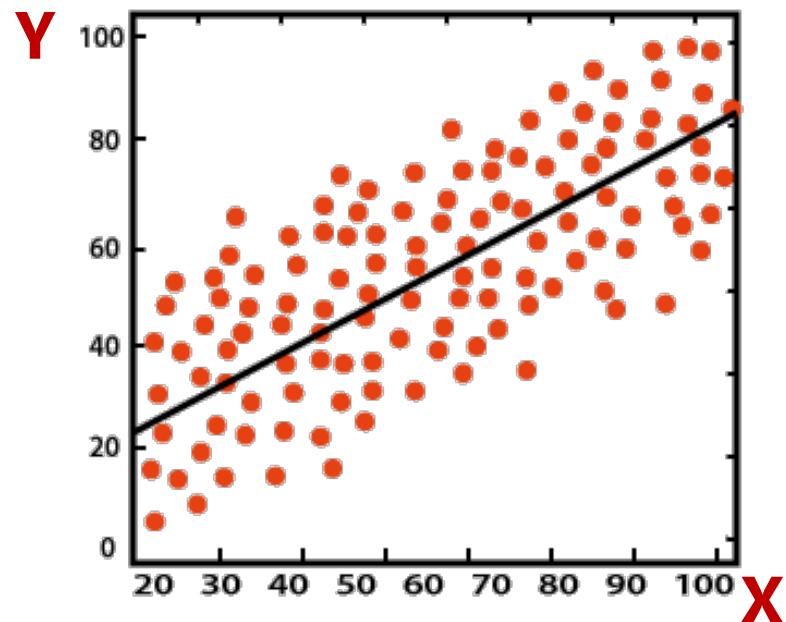


## Unsupervised Learning

- Data is **unlabeled**
- We don't know exactly what we are looking for
- We want the algorithm to **find patterns and structure** in the data

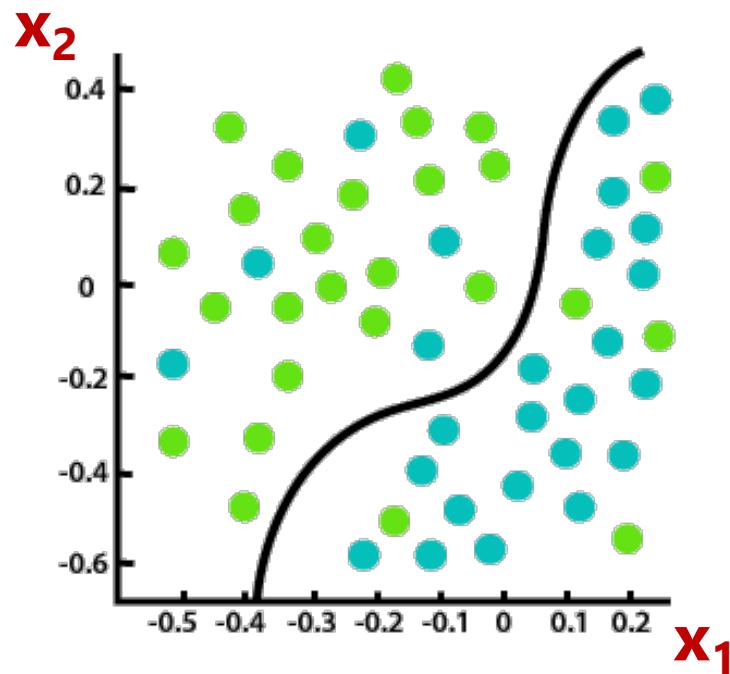
# SUPERVISED LEARNING

Continuous vs. discrete output



Regression

60.03

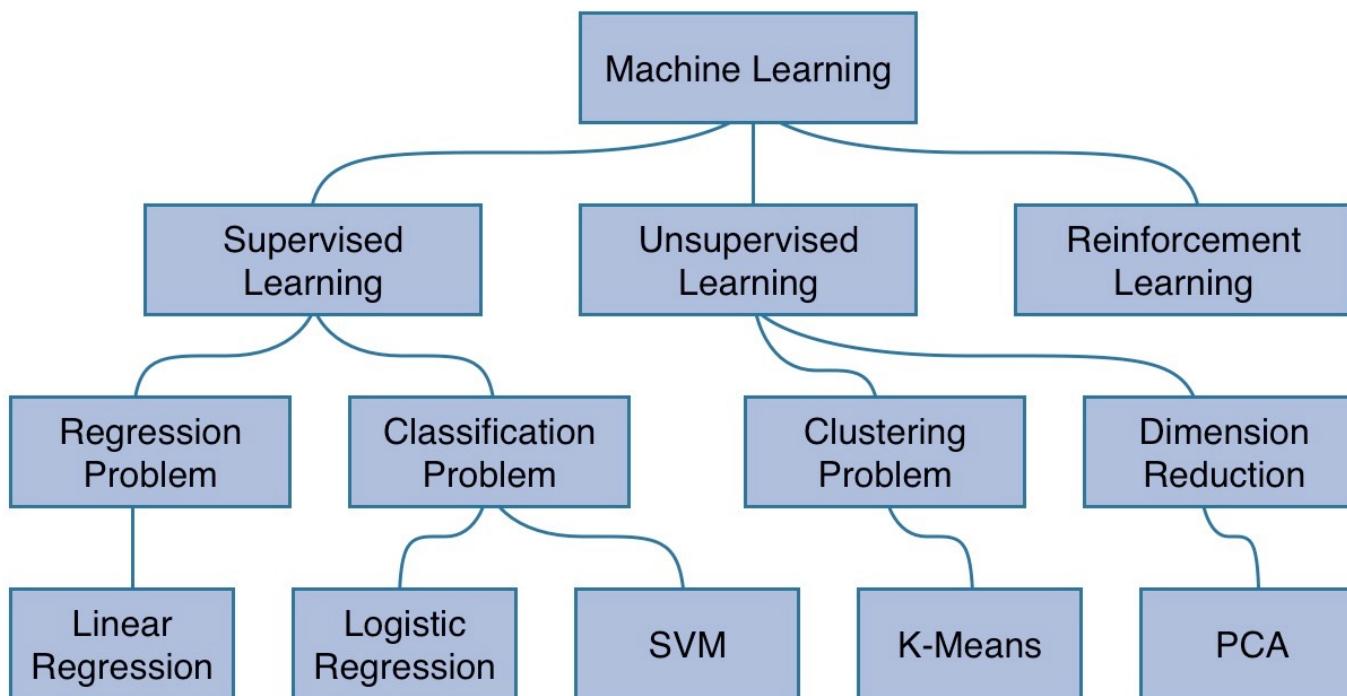


Classification

"Not SPAM"

# MACHINE LEARNING

And its multiple branches

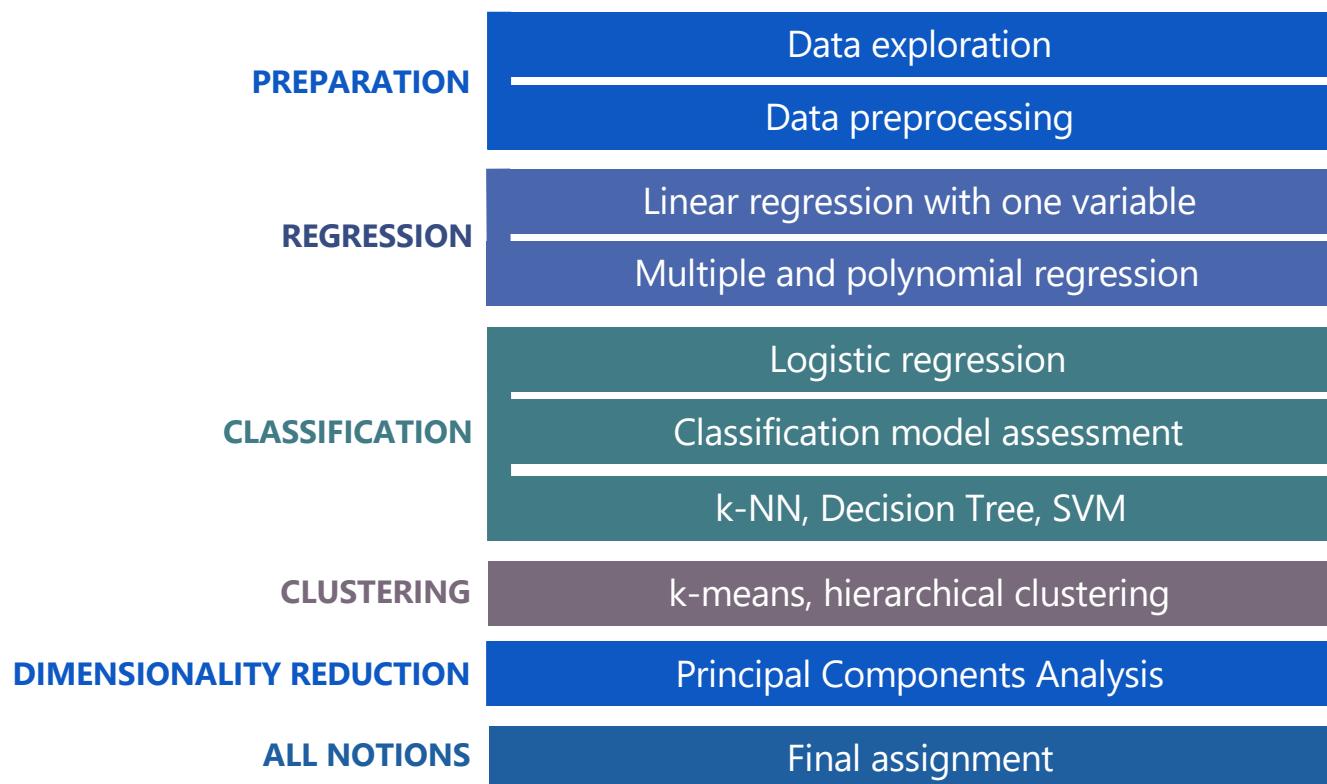




# COURSE PROGRAM

---

## Structure





# COURSE ORGANIZATION

---

- Class Schedule:
  - 20 min: homework correction
  - 50 min: theoretical background
  - 20 min: Python demo (by the teacher)
  - 60 min: practical exercise (by the students, to be finished at home)
- Material:
  - Slides and demo uploaded on Teams before each session
  - Practical exercise available on Teams as an assignment (with deadline)
  - Practical exercise correction uploaded on Teams after the deadline
- Grades:
  - Practical exercises: 40 %
  - Final exam: 60 %



# METHODS

---

## Interactions

- Don't hesitate to **ask questions** during the teacher presentation
- When asked a question by the teacher during class, **don't worry about providing a correct answer**. Just answer according to your current knowledge and intuition.
- You should take advantage as much as possible of the time dedicated to the **practical exercise** to ask specific questions to the teacher and make sure everything is sufficiently clear so that you can finish the assignment at home.



# TOOLS & LIBRARIES

---

## Installation

- Python 3.8 (or above). Make sure it includes pip (needed to install libraries).
- A development IDE: e.g. Visual Studio Code, Spyder.
- Jupyter notebook

```
pip install jupyterlab
```

- Libraries: scikit-learn, numpy, matplotlib, pandas

```
python -m pip install pandas
python -m pip install numpy
python -m pip install -U scikit-learn
python -m pip install -U matplotlib
```



# TOOLS & LIBRARIES

---

## Import

- Import the libraries you need before to use it in your code.

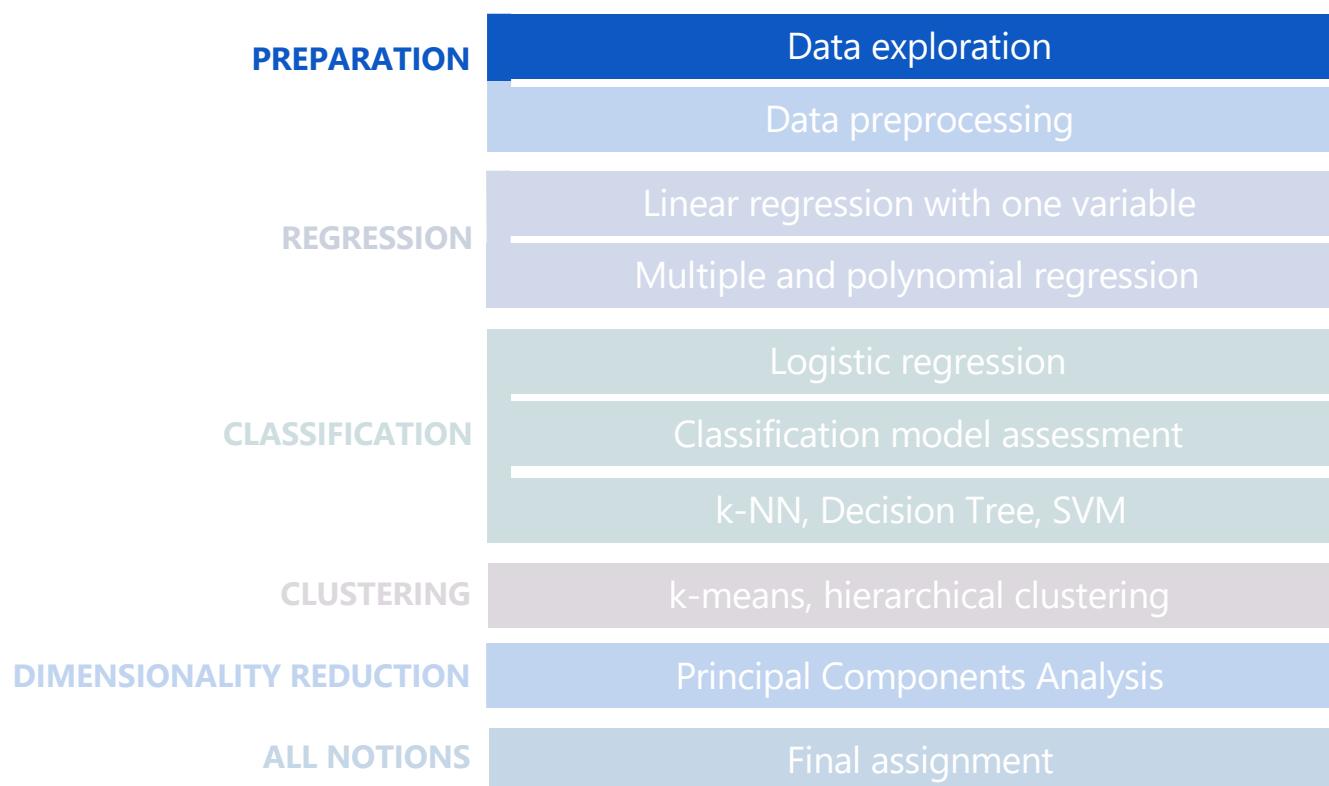
```
import numpy as np          ← operations on array  
import pandas as pd         ← data manipulation  
import sklearn              ← ML algorithms  
import matplotlib.pyplot as plt    ← plotting  
import ...
```



# COURSE PROGRAM

---

## Structure

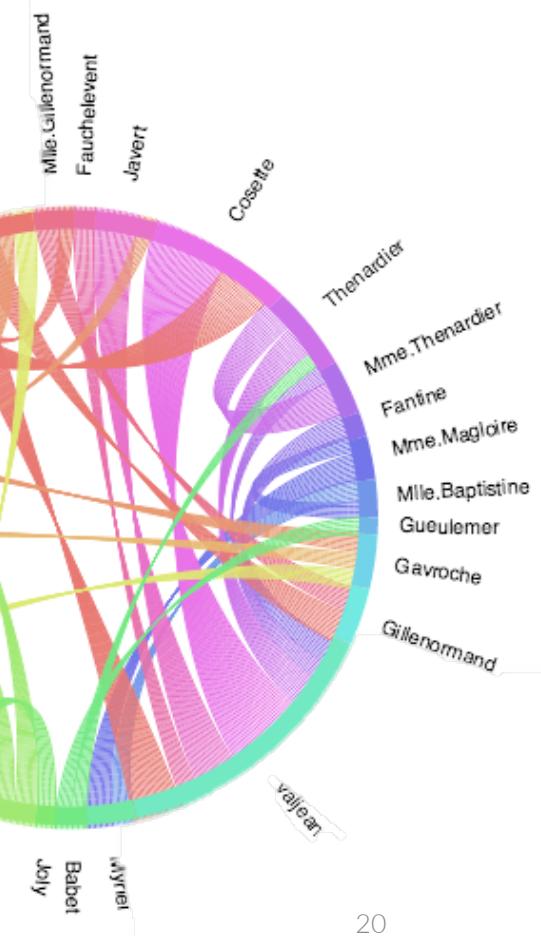


# EXPLORE THE DATA SET

Why?

- Understand your data
- Identify the variables
- Get some intuition (correlation, etc.)
- Identify problems in the data

2	15647311	Hill	608	Spain	Female	41	1	83807.86	1	0	1	112542.58	0
3	15619304	Onio	502	France	Female	42	8	159660.8	3	1	0	113931.57	1
4	15701354	Boni	699	France	Female	39	1	0	2	0	0	93826.63	0
5	15737388	Mitchell	950	Spain	Female	43	2	125510.82	1	1	1	79084.1	0
6	15574012	Chu	645	Spain	Male	44	8	113755.78	2	1	0	149756.71	1
7	15592531	Barnett	822	France	Male	50	7	0	2	1	1	100624.0	0
8	15656148	Obinna	376	Germany	Female	29	4	115046.74	4	1	0	119346.88	1
9	15792365	He	501	France	Male	44	4	142051.07	2	0	1	7940.0	0
10	15592389	H?	684	France	Male	27	2	134603.88	1	1	1	71725.73	0
11	15767321	Bearce	528	France	Male	31	6	102016.72	2	0	0	80181.12	0
12	15737173	Andrews	497	Spain	Male	24	3	0	2	1	0	75390.01	0
13	15632264	Kay	476	France	Female	34	10	0	2	1	0	28260.08	0
14	15691483	Chin	549	France	Female	25	5	0	2	0	0	190857.79	0
15	15690000	Scott	635	Spain	Female	35	7	0	2	1	1	65971.0	0
16	15643666	Soforth	616	Germany	Male	45	3	143129.41	2	0	1	0	0
17	15737452	Romeo	653	Germany	Male	58	5	1132602.88	1	1	0	0	0
18	15788218	Henderson	549	Spain	Female	24	0	0	2	1	0	0	0
19	15661507	Muldrow	587	Spain	Male	45	0	0	1	0	0	0	0
20	15569982	Hao	726	France	Female	24	6	0	2	1	0	0	0
21	15577657	McDonald	732	France	Male	41	8	0	2	1	0	0	0
22	15597945	Delluccio	636	Spain	Female	32	8	0	2	1	0	0	0
23	15699309	Gerasimov	510	Spain	Female	38	4	0	0	0	0	0	0
24	15725737	Mosman	659	France	Male	46	3	0	0	0	0	0	0
25	15625047	Yen	848	France	Female	38	5	0	0	0	0	0	0
26	15738191	Maclean	577	France	Male	25	3	0	0	0	0	0	0
27	15736816	Young	756	Germany	Male	36	2	136815.64	0	0	0	0	0
28	15700772	Nebachi	571	France	Male	44	9	0	0	0	0	0	0
29	15720993	McWilliams	574	Germany	Female	43	3	141349.43	0	0	0	0	0
30	15656300	Luccino	411	France	Male	29	0	59697.17	0	0	0	0	0
31	15584745	Azilone	591	Spain	Female	42	0	0	0	0	0	0	0
32	15706552	linakachukwu	533	France	Male	36	7	85311.7	0	0	0	0	0
33	15750181	Sanderson	553	Germany	Male	41	9	110112.54	0	0	0	0	0
34	15659426	Maggard	520	Spain	Female	42	6	0	0	0	0	0	0
35	15732963	Clements	722	Spain	Female	29	9	0	0	0	0	0	0
36	15794171	Lombardo	475	France	Female	45	0	134264.04	0	0	0	0	0
37	15788448	Watson	490	Spain	Male	31	0	0	0	0	0	0	0
38	15729599	Lorenzo	804	Spain	Male	33	0	0	0	0	0	0	0
39	15717426	Armstrong	950	France	Male	36	7	0	1	0	0	0	0
40	15595768	Cameron	582	Germany	Male	41	6	70349.48	2	0	0	0	0
41	15619360	Hsiao	472	Spain	Male	40	4	0	1	0	0	0	0
42	15738148	Clarke	465	France	Female	51	8	12522.32	1	0	0	0	0
43	15687946	Osborne	556	France	Female	61	2	117419.35	1	1	0	0	0
44	15755196	Lavine	834	France	Female	49	2	131304.56	1	0	0	0	0
45	15684171	Blanchi	660	Spain	Female	61	5	155931.11	1	1	1	0	0
46	15754840	Tyler	776	Germany	Female	32	4	109421.13	1	1	1	1265	0
47	15602280	Martin	929	Germany	Female	27	9	112045.67	1	1	1	119708.21	0
48	15771573	Ogakwe	637	Germany	Female	39	9	137100.0	1	1	1	117822.8	0
49	15766205	Yin	550	Germany	Male	38	2	10371.38	1	0	1	90876.13	0
50	15771873	Buccio	776	Germany	Female	37	2	103769.22	2	1	0	194099.0	0
51	15616550	Chidiebele	698	Germany	Male	44	10	116363.37	2	1	0	198059.0	0
52	15768193	Trevianni	585	Germany	Male	36	5	146050.97	2	0	0	110557.57	0
53	15683553	O'Brien	788	France	Female	33	5	10450.0	2	0	0	116818.19	0
54	15702298	Parkhill	655	Germany	Male	41	8	125561.97	1	0	0	164040.94	1
55	15605590	Yoo	601	Germany	Male	42	1	88405.72	1	1	0	400147.6	1
56	15760881	Phillips	619	France	Male	43	1	125211.92	1	1	1	113410.49	0
57	15630053	Tsao	656	France	Male	45	5	127864.4	1	1	0	87107.57	0



# EXPLORE THE DATA SET

## Load

- Load a data set from a local file:

```
dataset = pd.read_csv('iris.csv')
```

- Load a "reference" data set from a library:

```
dataset = sklearn.datasets.load_iris()
```

- Several bases of data sets:

This screenshot shows the scikit-learn documentation page for the datasets module. It includes a sidebar with navigation links like 'Home', 'Up', 'Next', and 'sklearn 0.23.2'. The main content area contains several code snippets under the heading 'Loaders'.

```
dataset.clear_data_home([data_home]) Delete all the content of the data home cache.  
datasets.dump_svmlight_file(X, y, f, ...) Dump the dataset in svmlight / libsvm file format.  
datasets.fetch_20newsgroups() Load the filenames and data from the 20 newsgroups dataset (classification).  
datasets.fetch_20newsgroups_tfidf_vectorizer([...]) Load the 20 newsgroups dataset and vectorize it into token counts (classification).  
datasets.fetch_california_housing() Load the California housing dataset (regression).  
datasets.fetch_covtype([data_home,...]) Load the covertype dataset (classification).  
datasets.fetch_kddcup99([subset,...]) Load the kddcup99 dataset (classification).  
datasets.fetch_lfw_pairs([subset,...]) Load the Labeled Faces in the Wild (LFW) pairs dataset (classification).  
datasets.fetch_lfw_people([..., data_home,...]) Load the Labeled Faces in the Wild (LFW) people dataset (classification).  
datasets.fetch_olivetti_faces([...]) Load the Olivetti faces data-set from AT&T (classification).  
datasets.fetch_rcv1([...] data_home, subset,...]) Fetch dataset from openml by name or dataset id.  
datasets.fetch_rcv1c() Load the RCV1 multilabel dataset (classification).  
datasets.fetch_species_distributions([...]) Loader for species distribution dataset from Phillips et al.  
datasets.get_data_home([data_home]) Return the path of the scikit-learn data dir.  
datasets.fetch_boston(), return_X_y() Load and return the boston house-prices dataset (regression).  
datasets.load_breast_cancer() Load and return the breast cancer wisconsin dataset (classification).
```

This screenshot shows the Kaggle datasets search results page. The search bar at the top has 'Search 60,541 datasets' and a filter icon. Below the search bar, there are several dataset cards with titles like 'COVID-19 data from John Hopkins University', 'US Election 2020 Tweets', 'US Election 2020 - Presidential Debates', 'Election, COVID, and Demographic Data by County 2020', '2020 United States presidential election', 'Netflix Movies and TV Shows', and 'COVID-19's Impact on Airport Traffic'. Each card includes a thumbnail, title, publisher, size, and download count.

This screenshot shows the public.opendataford.com/explore page. It features a grid of dataset cards. Some visible cards include 'JCDecaux bike Stations Data' (340 datasets), 'Geonames - All Cities with a population > 1000' (Territory map), 'USA 2016 Presidential Election by County' (221 datasets), and 'Titanic Passengers' (Public domain). Each card provides details like publisher, license, and file formats.

# EXPLORE THE DATA SET

Example dataset

- Iris data set: characteristics of three species of iris.

**iris setosa**



**iris versicolor**



**iris virginica**



# EXPLORE THE DATA SET

Load

Features/  
independent variables

Label/ target value /  
dependent variable

	sepal_length	sepal_width	petal_length	petal_width	species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5.0	3.4	1.5	0.2	setosa
9	4.4	2.9	1.4	0.2	setosa
10	4.9	3.1	1.5	0.1	setosa
11	5.4	3.7	1.5	0.2	setosa
12	4.8	3.4	1.6	0.2	setosa
13	4.8	3.0	1.4	0.1	setosa
14	4.3	3.0	1.1	0.1	setosa
15	5.8	4.0	1.2	0.2	setosa
16	5.7	4.4	1.5	0.4	setosa
17	5.4	3.9	1.3	0.4	setosa
18	5.1	3.5	1.4	0.3	setosa
19	5.7	3.8	1.7	0.3	setosa

# EXPLORE THE DATA SET

Get general information

- Information on format:

```
dataset.shape  
dataset.columns.values  
dataset.info()  
dataset.dtypes
```

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 150 entries, 0 to 149  
Data columns (total 5 columns):  
 #   Column      Non-Null Count  Dtype     
 ---    
 0   sepal_length  150 non-null   float64  
 1   sepal_width   150 non-null   float64  
 2   petal_length  150 non-null   float64  
 3   petal_width   150 non-null   float64  
 4   species       150 non-null   object    
 dtypes: float64(4), object(1)  
 memory usage: 6.0+ KB
```

- Information on content:

```
dataset.head()  
dataset.tail()  
dataset.sample(n=7)
```

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa



# EXPLORE THE DATA SET

---

## Basic operations

- Conversion:

```
pd.DataFrame(data=dataset, index=..., columns=...) # array to DataFrame  
df.to_numpy()      # DataFrame to array
```

- Selecting:

```
df['feature']  # column  
df[0:3]        # rows  
df[df['feature']>0]  # rows that satisfy a condition
```

- Manipulation:

```
df_copy = df.copy()  
df['new_feature'] = ['high' if x>10 else 'low' for x in df['feature']]  
df['feature'].values.reshape(-1,1)
```

# EXPLORE THE DATA SET

Compute basic statistics

- Count values:

```
dataset.isna().sum()  
dataset.duplicated().sum()  
dataset.nunique() #count unique elements  
(dataset['species']=='setosa').sum()
```

```
sepal_length      0  
sepal_width       0  
petal_length      0  
petal_width       0  
species           0
```

- Mean, min/max, standard deviation, etc.:

```
dataset.describe()
```

	sepal_length	sepal_width	petal_length	petal_width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

# EXPLORE THE DATA SET

## DataFrame vs array

### DataFrame (Pandas)

```
dataset  
  
  sepal_length  sepal_width  petal_length  petal_width  species  
0      5.1        3.5       1.4        0.2   setosa  
1      4.9        3.0       1.4        0.2   setosa  
2      4.7        3.2       1.3        0.2   setosa  
3      4.6        3.1       1.5        0.2   setosa  
4      5.0        3.6       1.4        0.2   setosa  
...     ...        ...       ...        ...    ...  
145     6.7        3.0       5.2        2.3  virginica  
146     6.3        2.5       5.0        1.9  virginica  
147     6.5        3.0       5.2        2.0  virginica  
148     6.2        3.4       5.4        2.3  virginica  
149     5.9        3.0       5.1        1.8  virginica
```

```
type(dataset)  
  
pandas.core.frame.DataFrame  
  
dataset['sepal_width'][0]  
  
3.5  
  
dataset.iloc[0]  
dataset.iloc[0,:]  
  
sepal_length      5.1  
sepal_width       3.5  
petal_length      1.4  
petal_width       0.2  
species           setosa  
Name: 0, dtype: object  
  
dataset.iloc[0][1]  
  
3.5
```

### Array (NumPy)

```
dataset.to_numpy()  
dataset.values  
  
array([[5.1,  3.5,  1.4,  0.2, 'setosa'],  
       [4.9,  3.0,  1.4,  0.2, 'setosa'],  
       [4.7,  3.2,  1.3,  0.2, 'setosa'],  
       [4.6,  3.1,  1.5,  0.2, 'setosa'],  
       [5.0,  3.6,  1.4,  0.2, 'setosa'],  
       [5.4,  3.9,  1.7,  0.4, 'setosa'],  
       [4.6,  3.4,  1.4,  0.3, 'setosa'],  
       [5.0,  3.4,  1.5,  0.2, 'setosa'],  
       [4.4,  2.9,  1.4,  0.2, 'setosa'],  
       [4.9,  3.1,  1.5,  0.1, 'setosa'],  
       [5.4,  3.7,  1.5,  0.2, 'setosa'],  
       [4.8,  3.4,  1.6,  0.2, 'setosa'],  
       [4.8,  3.0,  1.4,  0.1, 'setosa'],  
       [4.3,  3.0,  1.1,  0.1, 'setosa'],  
       [5.8,  4.0,  1.2,  0.2, 'setosa'],  
       [5.7,  4.4,  1.5,  0.4, 'setosa']])
```

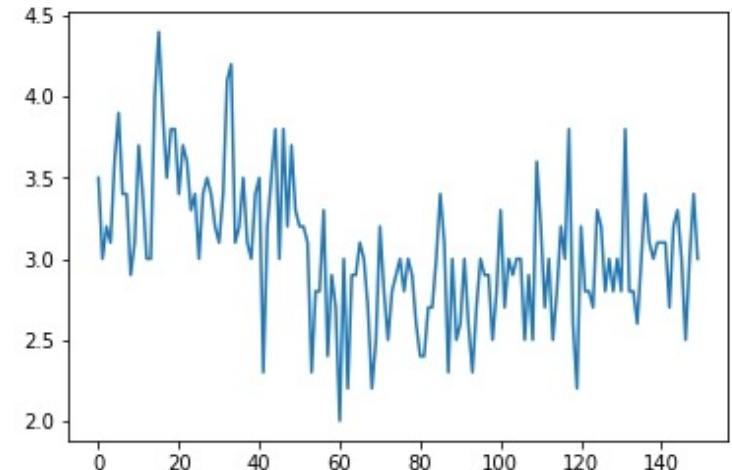
```
type(dataset.values)  
  
numpy.ndarray  
  
dataset.values[0][1]  
dataset.values[0,1]  
  
3.5  
  
type(dataset.values[0,1])  
  
float  
  
dataset.values[0,4]  
  
'setosa'  
  
type(dataset.values[0,4])  
  
str
```

# EXPLORE THE DATA SET

## Visualize

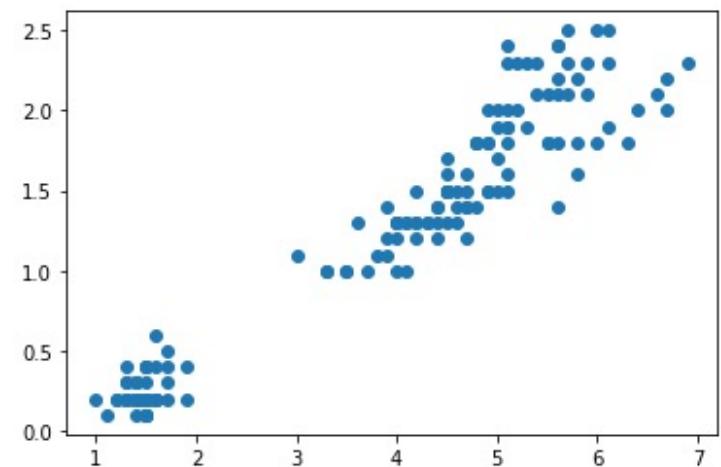
- Curve plot:

```
plt.plot(dataset['sepal_width'])  
plt.show()
```



- Scatter plot:

```
plt.scatter(dataset['petal_length'],  
           dataset['petal_width'])
```

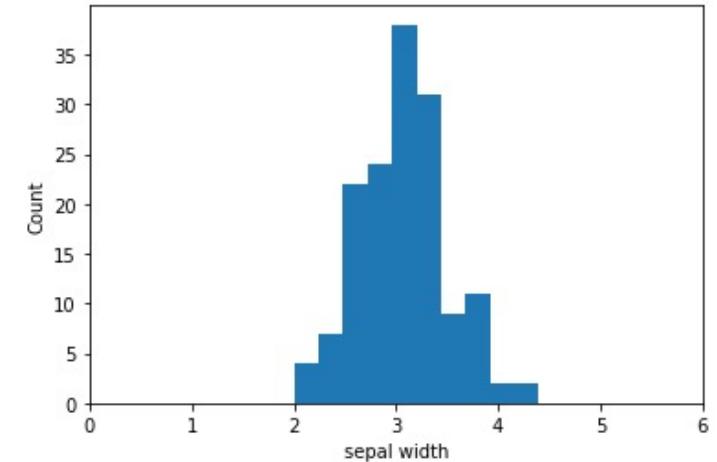


# EXPLORE THE DATA SET

## Visualize

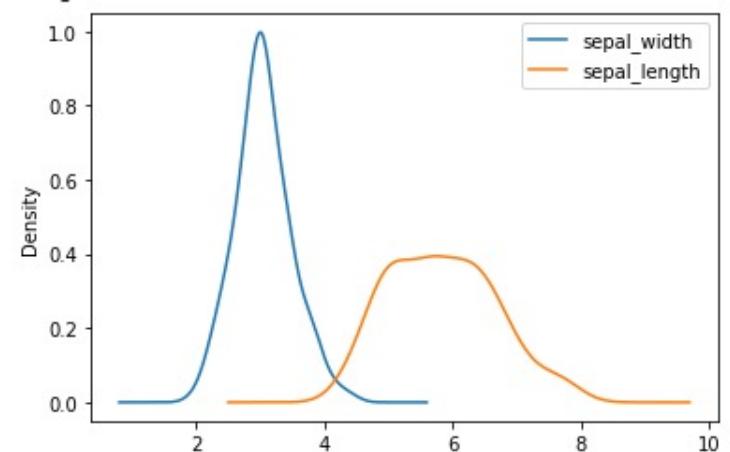
- Histogram:

```
plt.hist(dataset['sepal_width'])
```



- Density plot:

```
dataset[['sepal_width',  
        'sepal_length']].plot(kind='density')
```



# EXPLORE THE DATA SET

## Visualize

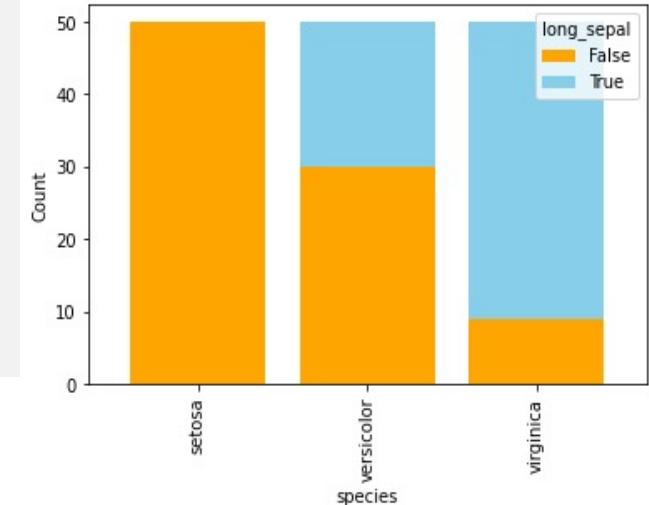
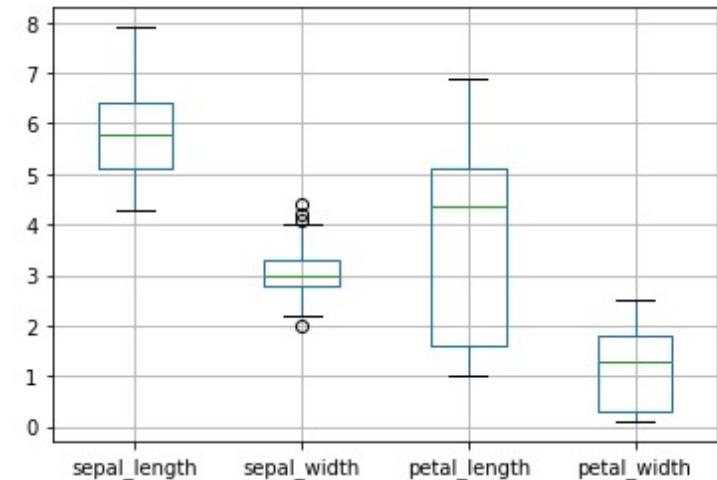
- Boxplot:

```
dataset[['sepal_length', 'sepal_width',
         'petal_length', 'petal_width']].boxplot()
```

- Stacked bars:

```
df_plot =
    dataset.groupby(['species', 'long_sepal']).size()\
    .reset_index()\
    .pivot(index='species', columns='long_sepal', values=0)

df_plot.plot(kind='bar', stacked=True,
              color=['orange', 'skyblue'], width=0.8)
```

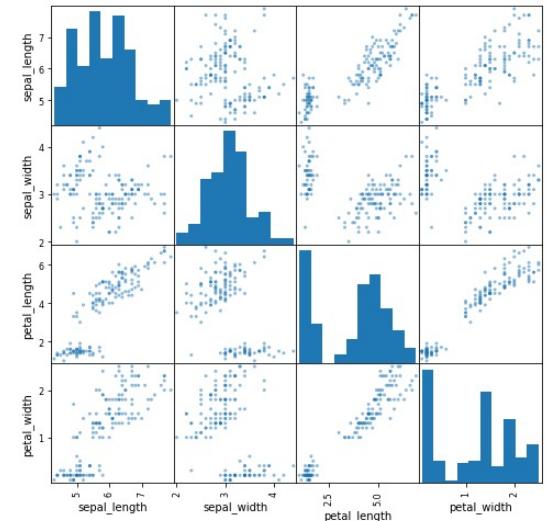


# EXPLORE THE DATA SET

## Visualize

- Scatter matrix:

```
scatter_matrix(dataset.loc[:, :4])
```

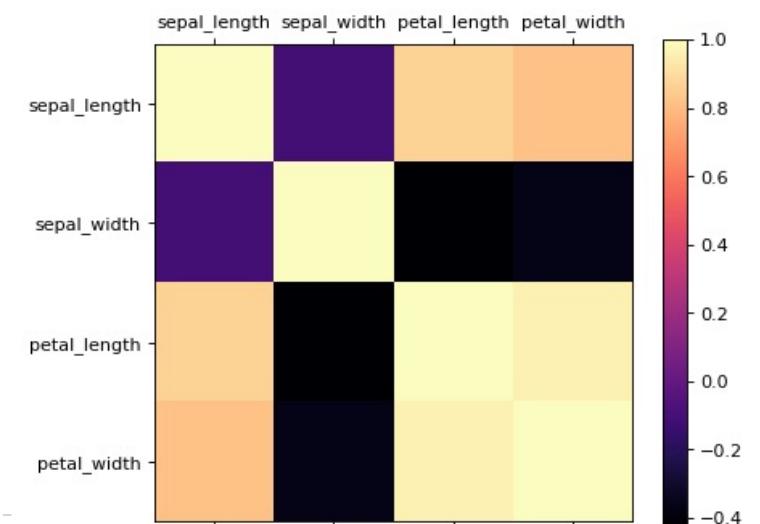


- Correlation matrix:

```
correlation = dataset.loc[:, :4].corr()  
matshow(correlation)
```

alternative

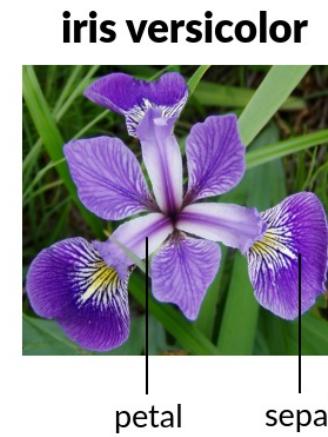
```
import seaborn as sns  
sns.heatmap(correlations, annot=True)
```



# EXPLORE THE DATA SET

## Implementation

- Iris data set: characteristics of three species of iris.
- Objectives:
  - Explore the data
  - Work with Jupyter notebook



# EXPLORE THE DATA SET

## Practice

- Titanic data set: “what sorts of people were more likely to survive?”
- Guidelines:
  - Data set is provided
  - Work with Jupyter notebook
  - Start during the course, finish at home
  - Report should be submitted before Monday 23:59 PM on Teams





## EXPLORE THE DATA SET

---

### Practical exercise

Advice for the assignment:

- Submit an executed Jupyter notebook
- Always check out the documentation of the libraries you use
- Don't do copy-paste from others' work (or from anywhere else)
- Always write code that you understand (you may be asked to demonstrate your understanding of the code you submitted)
- Provide written explanations (not only numbers) when an interpretation is needed