**Practice #11: Machine Learning III**

**Objectives:**

* + Get use to dataframe and machine learning
  + Use a train and test set
  + Scale our data

**Steps:**

1. **Get the data and draw it**

*In this step, you will open, analyse and plot your data.*

* Open the file named “Data\_to\_analyse.parquet” with the read\_parquet() function of the pandas package.
* Print the dataframe.
* You can now pass to a numpy arrays if you are more comfortable with it. To pass from a dataframe column to a numpy array (1 column), use the following instruction:
  + x1 = df[“Age”].to\_numpy().reshape(-1,1)
* In this dataframe, every line is a person’s information. The first column is the money on the bank account of the person (in $), the second is the age of the person (in years) and the last is the amount of money given to charity by this person every year (in $/year). We want to make a model to predict how much a person will give to charity per year by using its age and its amount of money on its bank account. What are the features and what is the label?
* Make two scatter plots on two different figures (use plt.scatter() function from matplotlib.pyplot package):
  + Label = f(feature1)
  + Label = f(feature2)

1. **The creation of the train and test sets**

*You will now separate your data into two parts: a train part which will be used to train the model and a test part used at the end to validate the accuracy of your algorithm on unseen data.*

* Create four dataframes or matrix: a train and a test (for your features and labels) with the function train\_test\_split() (in the sklearn.model\_selection package). The size of the train dataframe (or matrix) must be 80 % of the size of the complete dataframe (or matrix).
* Print the four dataframes (or matrix)..

1. **The scaling of the data**

*Your data have different scale. In machine learning, data with the same scale are preferred. Use the standardization to get the same scale on all your data.*

* Create one standard scaler object (StandardScaler() function of sklearn.preprocessing package)
* Scale your features of the train set with the previously created scaler (use the fit\_transform() function). The fit\_transform() function accept a dataframe (or matrix). in parameter but always return a numpy matrix.
* Do the same with the label of the train set with another standard scaler object.
* Print your scaled features and labels.

1. **The linear regression**

*You can now use your features and label to make a linear regression model.*

* Create a linear regression model and train it with the train set (use the LinearRegression() function of the sklearn.linear\_model package and the fit() function of your model)
* Print the coefficients, the intercept and the score of your model (coef\_ and intercept\_ and score() fucntion).
* Write the equation of your model.

1. **The comparison with the test set.**

*You can now study the accuracy of your model on unseen data.*

* Scale your features and labels of your test set by using the same scaler objects than in step 3.
* Print the score of your model on the test set.

1. **Make predictions**

*You can now use your model to make predictions.*

* Predict the amount of money given to charity by these two persons:
  + **John:** Money on his account: 100 000$, age: 20
  + **Charles:** Money on his account: 100 000 000$, age: 90
* Remember than you have to scale your features before prediction and then do the inverse scale of the prediction after the prediction.
* Which one will give more money to charity?