**Activity 1**

**Prediction with Back-Propagation and Linear Regression**

* **Git Repository**

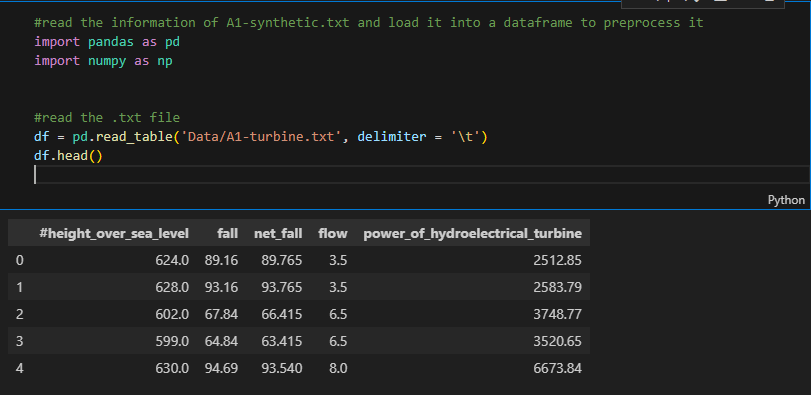
<https://github.com/YoussefEzz/Prediction-BP-and-LR>

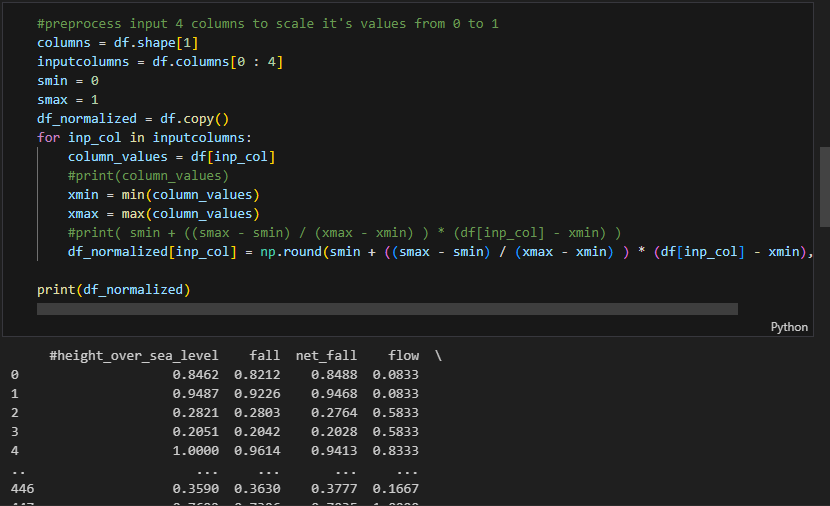
* **Part 1 : Selecting and analyzing the datasets**

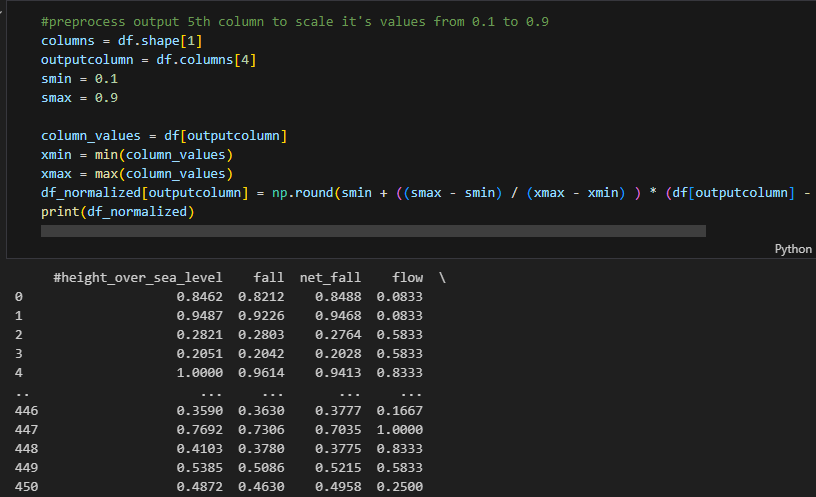
Since we do not want to give a priori more importance to some of the input variables w.r.t. the others, we should scale all of them to the same range of variation.

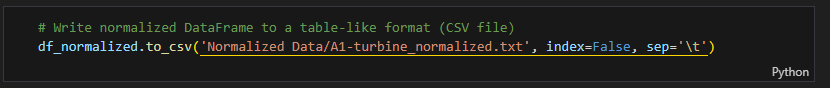
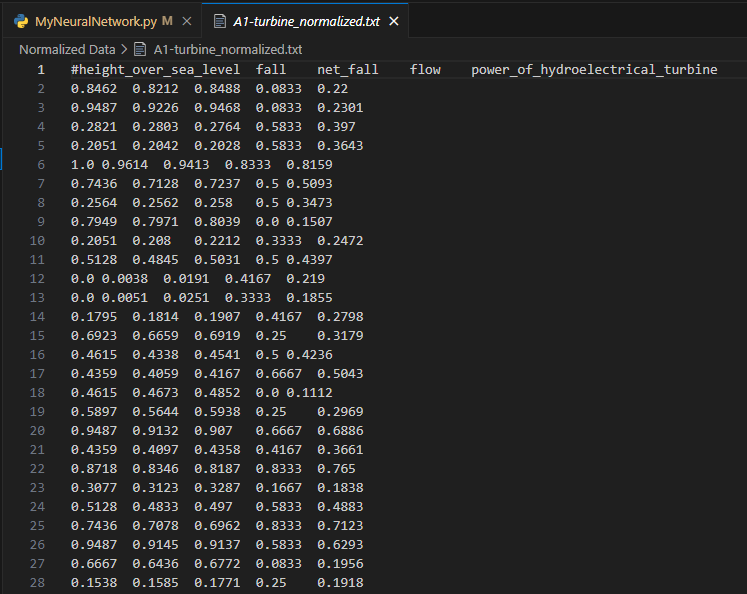
The scaling of the output variables has an additional requirement: since the output of a sigmoid lays in the range (0.0, 1.0), the desired output values must strictly fall within these limits. For predictions tasks (e.g. A1), where the output variable takes values in a certain [min, max] range, a convenient choice is its linear scaling to a range like [0.1, 0.9]

**Preprocess of Dataset 1 and 2 A1-synthetic.txt and A1-turbine.txt**

1. read data from “Data\A1-synthetic.txt” and “Data\A1-turbine.txt”
2. separate linear scaling of each input variable v1 to v9 for A1-synthetic - v3 and v8 are already between [0.0, 1.0] – and [ height\_over\_sea\_level fall net\_fall flow ] for A1-turbine from its [min, max] range to [0.0, 1.0] .



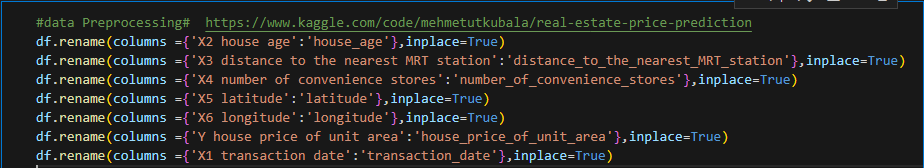
1. separate linear scaling of each output variable to [ 0.1, 0.9 ] since the output of a sigmoid lies in the range (0.0, 1.0) .
2. write normalized csv data to “Normalized Data\A1-synthetic\_normalized.txt” and “Normalized Data\A1-turbine\_normalized.txt”



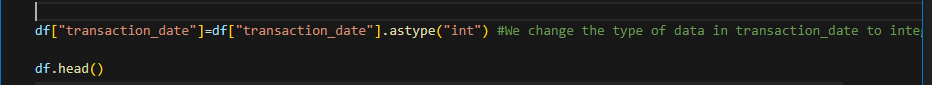
**Preprocess of Dataset 3 real estate price prediction**

Source : <https://www.kaggle.com/code/mehmetutkubala/real-estate-price-prediction>

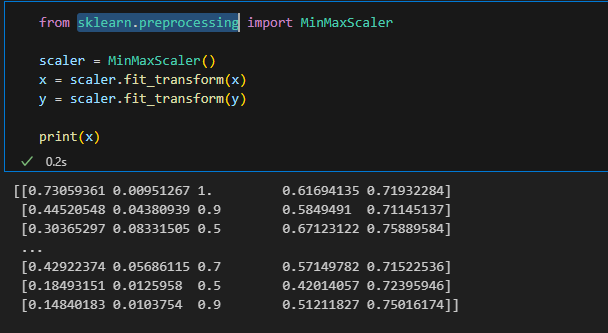
1. Get the real\_estate.csv from above source
2. Change column names to be more readable using Jupiter Notebook



1. Change column transaction date to integer data type



1. Linearize to the range [0, 1] using sklearn.preprocessing MinMaxscaler



* **Part 2: Implementation of BP**

1. **Read and parse the normalized data**

normalized files **A1-turbine\_normalized** that are going to be the input of your analysis part is read

#read and parse the .csv features file

df = pd.read\_csv('Normalized Data/A1-turbine\_normalized.txt', delimiter = '\t')

df.head()

columns = df.shape[1]

select the first 85% rows as training features an array of arrays

# construct an array of arrays size (451, 4) for all features input values

inputcolumns = df.columns[0 : 4]

features = df[inputcolumns].values

#select the first 85% as training features an array of arrays size (383, 4)

num\_training\_features = int(85 \* features.shape[0] / 100)

training\_features = features[0 : num\_training\_features]

select the first 85% rows as training targets as an array

# construct an array of size (451) for all features target values

outputcolumn = df.columns[4]

targets = df[outputcolumn].values

##select the first 85% as training tsrgets an array  size (383)

training\_targets = targets[0 : num\_training\_features]

call fit function to begin the training

# call fit function with features (n\_samples,n\_features) and targets (n\_samples)

nn.fit(training\_features, training\_targets)