**Implementing Artificial Neural Network training process in Python**

**Problem Statement:-**

1. W have Dataset listed 517 fires from the Montesano natural park in Portugal. For each incident, weekday, month, coordinates, and the burnt area are recorded, as well as several meteorological data such as rain, temperature, humidity, and wind. Predict the burned area of forest fires with Neural Networks. The model is then used to predict the burnt area based on the current date and the coordinates, where the fire is spotted. This prediction can be used for calculating the forces sent to the incident.

Solution:

An Artificial Neural Network (ANN) is an information processing paradigm that is inspired the brain. ANNs, like people, learn by example. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process. Learning largely involves adjustments to the synaptic connections that exist between the neurons.

Steps

In this guide, we will follow the following steps:

Step 1 - Loading the required libraries and modules.

Step 2 - Loading the data and performing basic data checks.

Step 3 - Creating arrays for the features and the response variable.

Step 4 - Creating the training and test datasets.

Step 5 -Define Keras Model

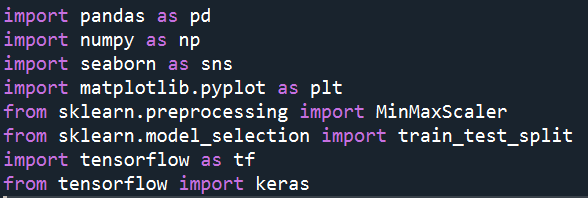
Step6- Compile Keras Model

Step7- Evaluate Keras Model

Step8- Graphical representation of metrics

The following sections will cover these steps.

# Step1-- Loading the required libraries and modules.

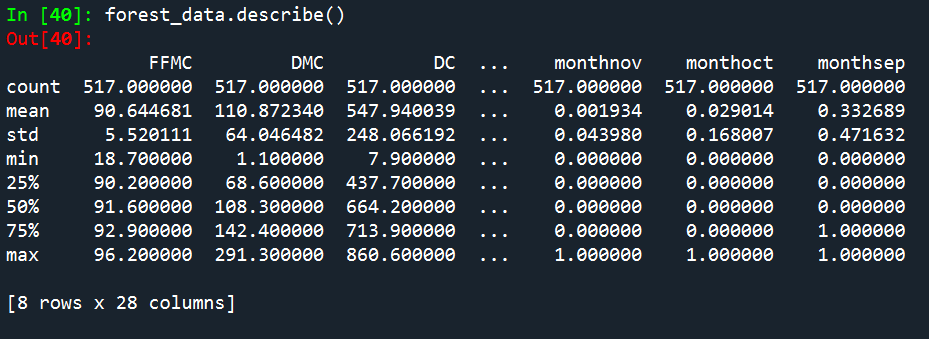


Step2- *Loading the data and performing basic data checks*

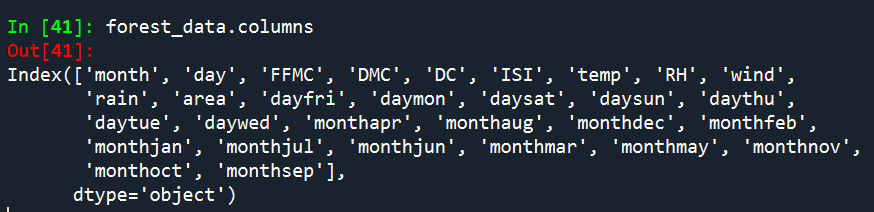
The required data is loaded as



Data is evaluated by using describe as

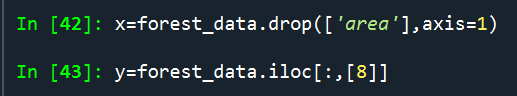


And columns are listed as



Step3- Creating arrays for the features and the response variable.

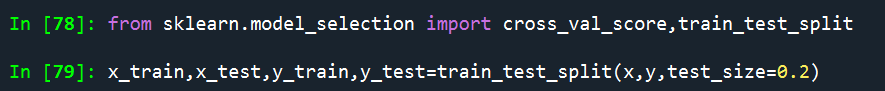
X is taken as feature variable and y is taken as response variable



Step 4 - Creating the Training and Test Datasets

We will build our model on the training set and evaluate its performance on the test set. This is called the holdout-validation method.

Split the data into training and test dataset, with the 'test\_size' argument specifying the percentage of data to be kept in the test data.



The training process consists of the following steps:

1. **Forward Propagation:**  
   Take the inputs, multiply by the weights (just use random numbers as weights)  
   Let Y = WiIi= W1I1+W2I2+W3I3  
   Pass the result through a sigmoid formula to calculate the neuron’s output. The Sigmoid function is used to normalise the result between 0 and 1:  
   1/1 + e-y
2. **Back Propagation**  
   Calculate the error i.e the difference between the actual output and the expected output. Depending on the error, adjust the weights by multiplying the error with the input and again with the gradient of the Sigmoid curve:  
   Weight += Error Input Output (1-Output) ,here Output (1-Output) is derivative of sigmoid curve.

 Repeat the whole process for a few thousands iterations.

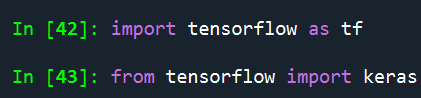
Step 5 -Define Keras Model

Models in Keras are defined as a sequence of layers.

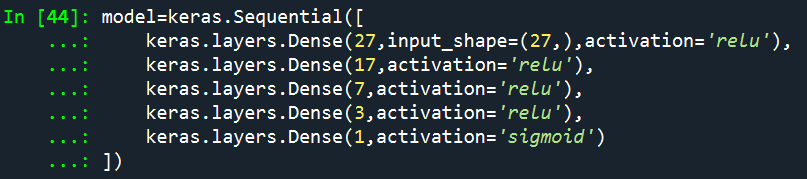
We create a [Sequential model](https://keras.io/models/sequential/) and add layers one at a time until we are happy with our network architecture.

The first thing to get right is to ensure the input layer has the right number of input features. This can be specified when creating the first layer with the **input\_dim** argument and setting it to n for the n input variables.

Keras is imported from tensorflow framework



ANN model is build using keras.sequential using activation functions as relu and sigmoid



Step6- Compile Keras Model

Now that the model is defined, we can compile it.

We must specify the loss function to use to evaluate a set of weights, the optimizer is used to search through different weights for the network and any optional metrics we would like to collect and report during training.

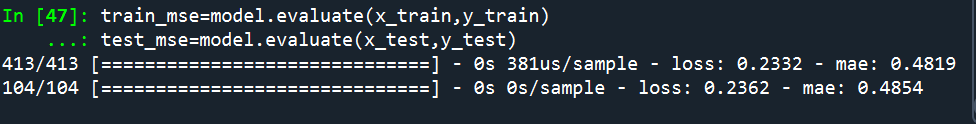
In this case, we will use cross entropy as the **loss** argument. This loss is for a regression problems and is defined in Keras as “loss='mse',metrics=['mae'] “.

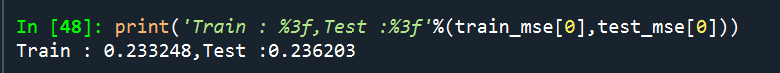


Step7- Evaluate Keras Model

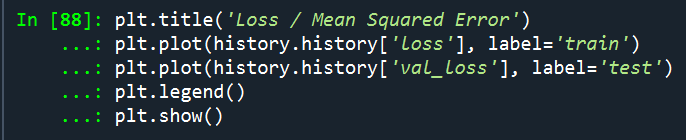
We have trained our neural network on the entire dataset and we can evaluate the performance of the network on the same dataset.

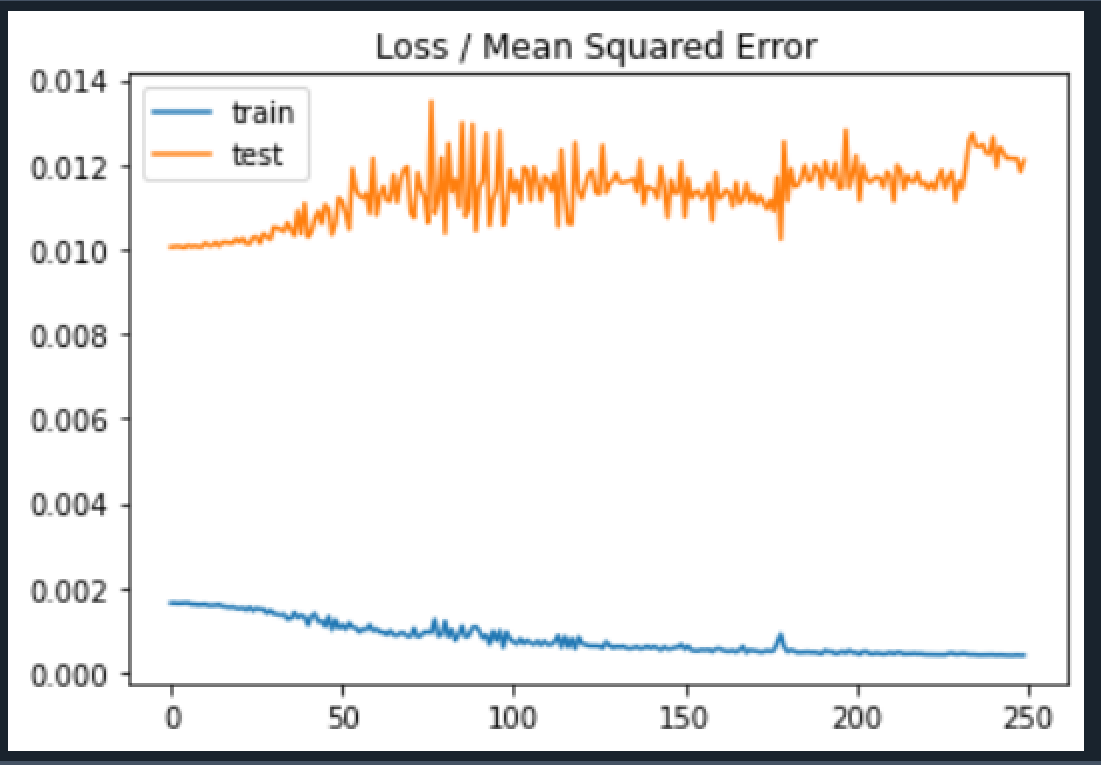
You can evaluate your model on your training dataset using the evaluate() function on your model and pass it the same input and output used to train the model.





Step8- Graphical representation of metrics





A line plot is also created showing the mean squared error loss over the training epochs for both the train (blue) and test (orange) sets.