COMP809 – Data Mining and Machine Learning

Lab 6 – Neural Network/ MLP

This lab covers implementations related to Multi_Layer Perception (MLP) classifier using sklearn module.

In addition to the implementation of the classifiers you will learn how to display important curves/graphs using loss/accuracy.

For this excersice we use pima-indians-diabetes data set which uses eight numeric attributes to identify whether a patient has diabetes or not.

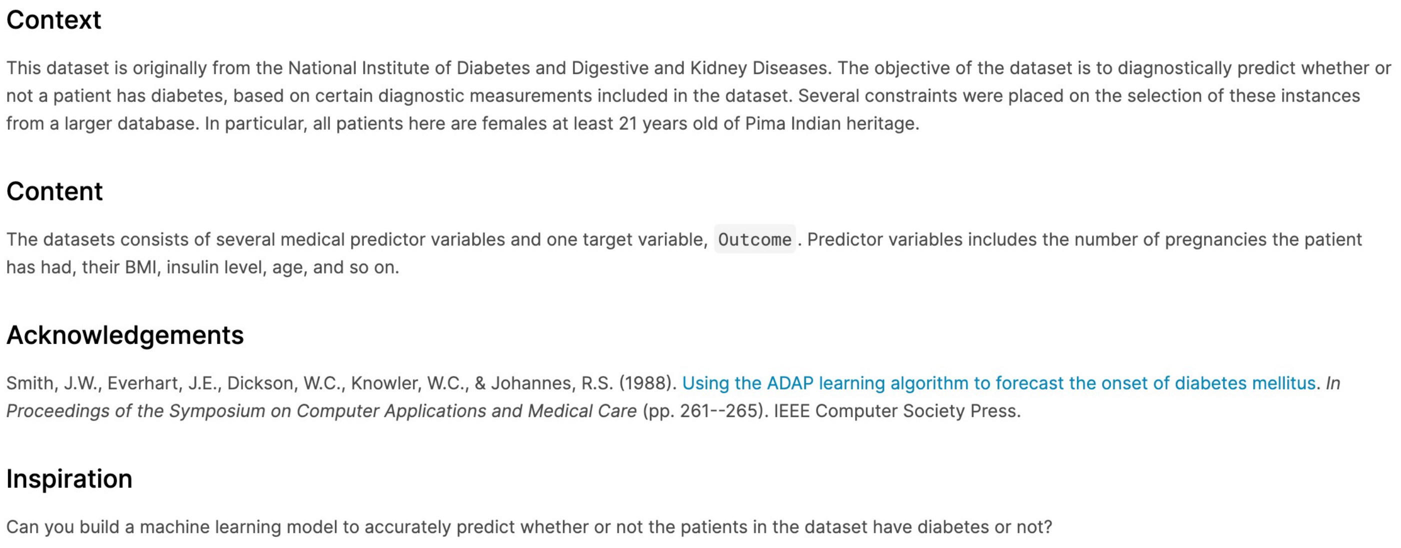
1. Importing libraries Numpy;

train_test_split; accuracy_score;

MLPClassifier; (from sklearn.neural_network) ;

matplotlib.pyplot; pandas;

2. Load Data



Context

This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. The objective of the dataset is to diagnostically predict whether or not a patient has diabetes, based on certain diagnostic measurements included in the dataset. Several constraints were placed on the selection of these instances from a larger database. In particular, all patients here are females at least 21 years old of Pima Indian heritage.

Content

The datasets consists of several medical predictor variables and one target variable, `Outcome`. Predictor variables includes the number of pregnancies the patient has had, their BMI, insulin level, age, and so on.

Acknowledgements

Smith, J.W., Everhart, J.E., Dickson, W.C., Knowler, W.C., & Johannes, R.S. (1988). Using the ADAP learning algorithm to forecast the onset of diabetes mellitus. *In Proceedings of the Symposium on Computer Applications and Medical Care* (pp. 261--265). IEEE Computer Society Press.

Inspiration

Can you build a machine learning model to accurately predict whether or not the patients in the dataset have diabetes or not?

url =

"https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima -indians-diabetes.data.csv"

3. **Assign the column names to the dataframe**

3.1 Features:
'preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age',
3.2 Class Label: 'class'

3.3 Assign variable *predictors* = your feature dataset

3.4 Assign variable target = your class label dataset

4. **Split you dataset to :** features dataset for training
   features dataset for testing
   class label dataset for training
   class label dataset for testing

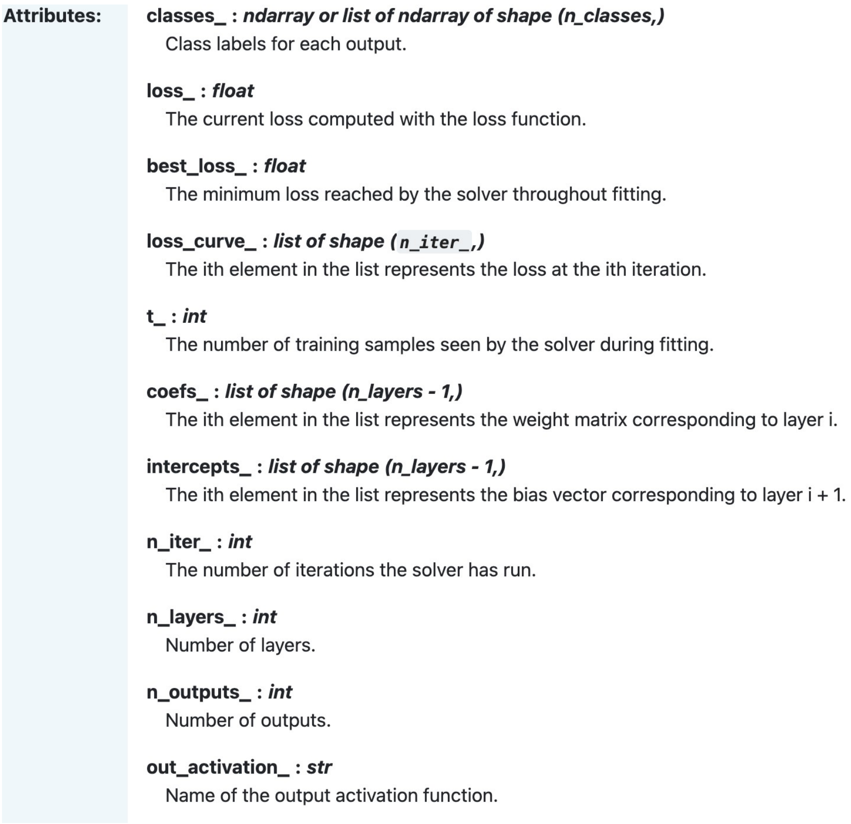
   training : 70% | testing : 30%

5. **Declare your classifier as a MLPClassifier**
   Let's start with:
   - Activation function = **'logistic'**,
     (Can you explain different activation functions?)
   - solver for weight optimization = **'adam'**,
   - learning rate = **0.01**,
   - 2 hidden layers o number of neuron units for the first hidden layer: **10** o number of neuron units for the second hidden layer: **5**
   - Maximum number of iteration = **200**,

6. **train your model and evaluation your model**
   - fit
   - predict
   - accuracy_score

   *** tell us your accuracy**

7. **Understand your models.**
https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html?highlight=loss



Attributes:

classes_ : ***ndarray or list of ndarray of shape (n_classes,)***
Class labels for each output.

loss_ : ***float***
The current loss computed with the loss function.

best_loss_ : ***float***
The minimum loss reached by the solver throughout fitting.

loss_curve_ : ***list of shape (`n_iter_`,)***
The ith element in the list represents the loss at the ith iteration.

t_ : ***int***
The number of training samples seen by the solver during fitting.

coefs_ : ***list of shape (n_layers - 1,)***
The ith element in the list represents the weight matrix corresponding to layer i.

intercepts_ : ***list of shape (n_layers - 1,)***
The ith element in the list represents the bias vector corresponding to layer i + 1.

n_iter_ : ***int***
The number of iterations the solver has run.

n_layers_ : ***int***
Number of layers.

n_outputs_ : ***int***
Number of outputs.

out_activation_ : ***str***
Name of the output activation function.

See the attributes of your MLP classifier Let's print out :

loss_
best_loss_
n_iter_

… … …

As well as **loss_curve_**

As you can see, loss curve is a **list**

8. **Plot this loss curve and explore how this function is converged**

9. **finally, adjust those hyperparameters, to improve your model, and see the how the loss curve get changed?**