MLP_Regression

February 8, 2021

[ ]:

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.neural_network import MLPRegressor
import matplotlib.pyplot as plt
```

[ ]:

```
#Save the training data in the nparray data
data = np.loadtxt('data_function_approximation.txt')
```

[ ]:

```
#Performs MLP regression for a given activation and architecture and plots the␣
 ↪results into an coordinate system
def plot_mlp_regr(x, y, ax, activation, hidden_layer):
    #Initialize the model, perform the fit and make predictions
    reg = MLPRegressor(solver='lbfgs', activation=activation,␣
 ↪hidden_layer_sizes=hidden_layer, random_state=1)
    reg.fit(x.reshape(-1,1), y)
    y_pred = reg.predict(x.reshape(-1,1))

    #Plot the exact function and the estimate
    ax.set_title('architecture = ' + str(hidden_layer))
    ax.plot(x, y, label='exact')
    ax.plot(x, y_pred, label='appr')
    ax.legend()
```

[ ]:

```
#Test MLP for different architectures and for different activation functions
hidden_layer = [[(2), (2,1)], [(2,2), (3,2)]]
activation = ['tanh', 'logistic']

for activation in activation:
    fig, ax = plt.subplots(2,2)
    plt.subplots_adjust(hspace=0.5, wspace=0.5)
    print('\n' + str(activation) + ':')
    for i in range(2):
        for j in range(2):
            plot_mlp_regr(data[:,0], data[:,1], ax[i,j], activation,␣
 ↪hidden_layer[i][j])
```