Jayathilaka H.A.D.T.T.

E/16/156

CO542

Neural Networks and Fuzzy Systems

2021

Lab 04 - MLP

Task 2

In this exercise we will look at a real world scenario where neural networks are used: function fitting.

1. **Generate data (inputs and outputs) for the following function, y = −x 4 + x 3 + 23x 2 − 21x + 32 (1) x should be between -100 and 100; use an interval of 0.001 between the values.**

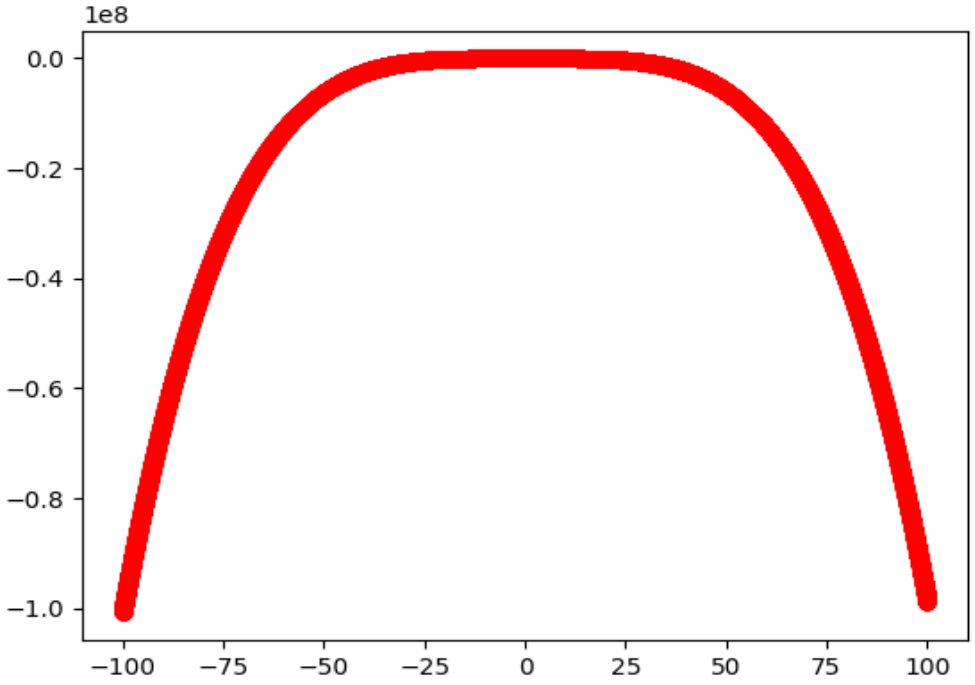
```
from sklearn.neural_network import MLPClassifier
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
import numpy as np
from itertools import product

# generate data for input
x = np.arange(-100, 100, 0.001)

# by using generated data for x, generate data for output
y = -x**4 + x**3 + 23*x**2 - 21*x + 32

# get the size of the input and output vectors
print(x.shape)
print(y.shape)

#generate the graph
plt.scatter(x, y)
plt.show()
```



2. **What is the number of inputs and outputs of the network?**
   Input : 1
   Output : 1

```
F:\Engineering\Third year\Sixth semester\CO542\Lab\Lab4\Task2>python exercise.py
(200000,)
(200000,)
```

Input vector shape : 200000
Output vector shape : 200000

3. **Model the MLP using MLPRegressor instead of MLPClassifier. (They are almost the same; except for very subtle differences. Try to find their differences).**
   Classification predicts a discrete class label while regression predicts a continuous quantity. That is the difference between the classification and regression.

```
#Model the MLP using MLPRegressor instead of MLPClassifier
x = x.reshape(-1,1)
model = MLPRegressor(hidden_layer_sizes=(128, 64, 32), max_iter=100)
model.fit(x,y)
print("------------------Model score----------------")
print(model.score(x,y))
```

```
F:\Engineering\Third year\Sixth semester\CO542\Lab\Lab4\Task2>python exercise.py
(200000,)
(200000,)
------------------Model score----------------
0.9999987625845264
```

4. **Generate a test set (of your choice) and test it with the network.**

```
#Generate a new test set and test it with the network.
xl = np.arange(-100, 100, 10)
xl=xl.reshape(-1, 1)
yl = model.predict(xl)
```

5. **Plot the train data and model predictions on a same plot and observe up to what extend the predicted values are fitting with the original data set.**

```
#Plot the train data and model predictions on a same plot
plt.scatter(x, y ,c='b', marker="s", label='Training')
plt.scatter(xl, yl, c='r', marker="o", label='Predictions')
plt.show()
```