Jayathilaka H.A.D.T.T.

E/16/156

CO542

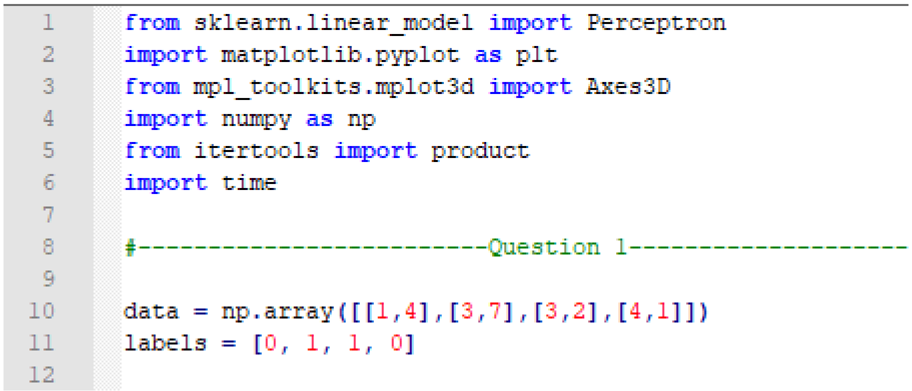
Neural Networks and Fuzzy Systems

2021

Lab 03 - Introduction to Perceptron

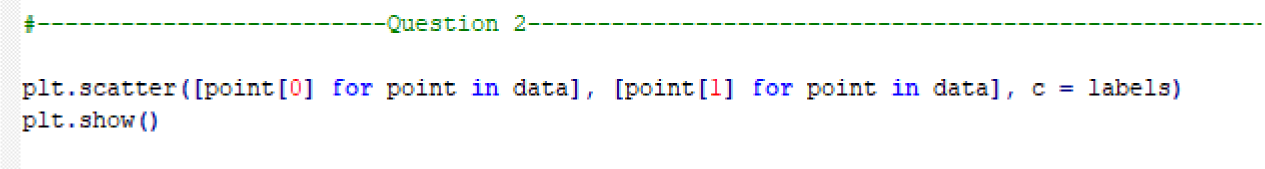
Exercise 2

1. **Demonstrate the behavior of a two input neuron, which is trained to classify 4 input vectors into two categories using single layer perceptron model. (Use smaller magnitudes as the values for the four input vectors).**

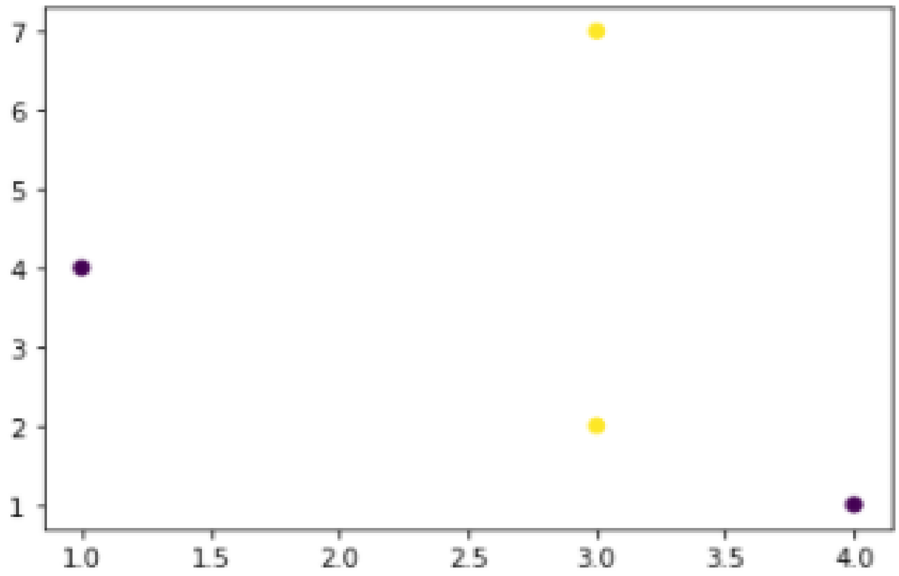


```python
from sklearn.linear_model import Perceptron
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
import numpy as np
from itertools import product
import time

#-------------------------Question 1--------------------

data = np.array([[1,4],[3,7],[3,2],[4,1]])
labels = [0, 1, 1, 0]

```

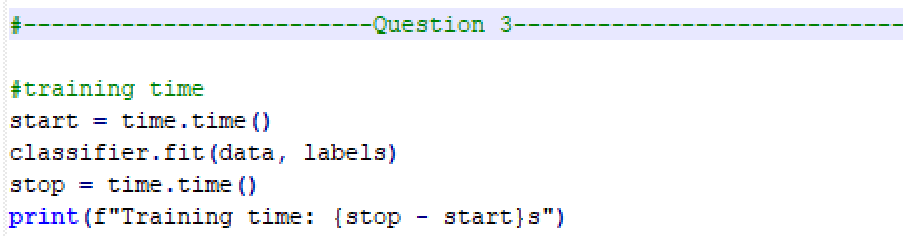
2. **Plot these vectors (Use a scatter plot).**



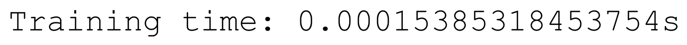
```python
#-------------------------Question 2-----------------------------------------------------

plt.scatter([point[0] for point in data], [point[1] for point in data], c = labels)
plt.show()
```



3. **Observe the training time.**

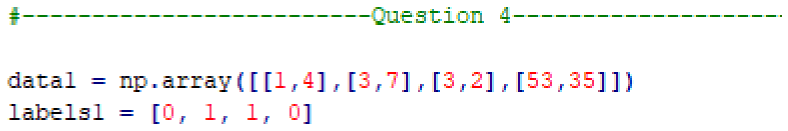


```
#-------------------------Question 3---------------------------

#training time
start = time.time()
classifier.fit(data, labels)
stop = time.time()
print(f"Training time: {stop - start}s")
```



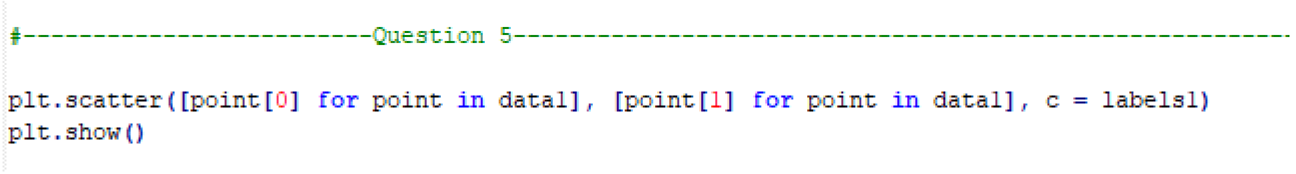
```
Training time: 0.00015385318453754s
```

4. **Modify the part Exercise 2.1 such that one input vector is much larger than all the others.**

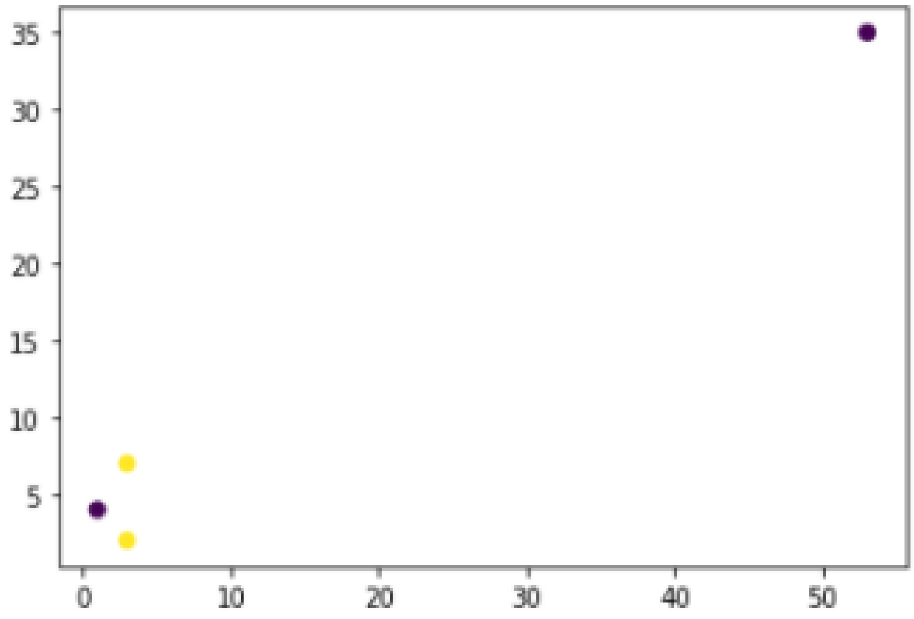


```
#-------------------------Question 4--------------------

data1 = np.array([[1,4],[3,7],[3,2],[53,35]])
labels1 = [0, 1, 1, 0]
```

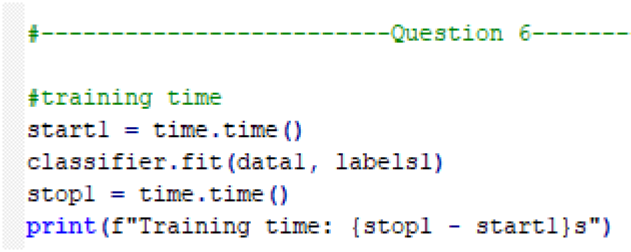
5. **Plot these vectors.**



```
#-------------------------Question 5-------------------------------------------------------

plt.scatter([point[0] for point in data1], [point[1] for point in data1], c = labels1)
plt.show()
```



6. **Observe the training time and compare it with the results you obtained in Exercises 2.3.**

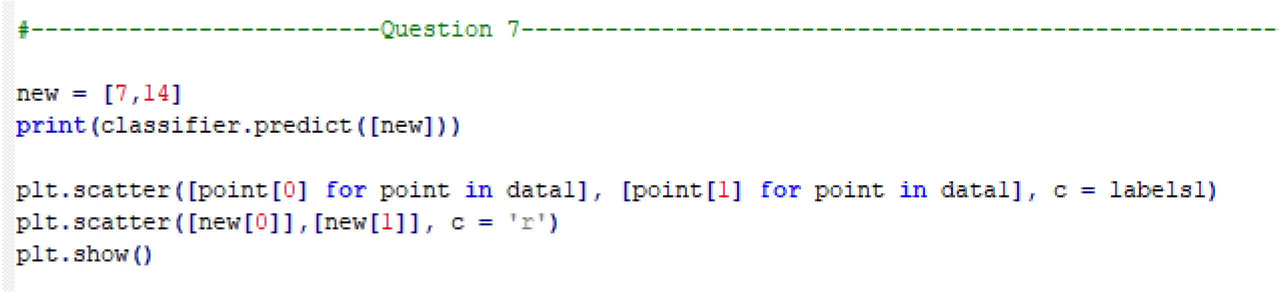


```
#-------------------------Question 6-----------------------

#training time
start1 = time.time()
classifier.fit(data1, labels1)
stop1 = time.time()
print(f"Training time: {stop1 - start1}s")
```

```
Training time: 0.011083841323852539s
```

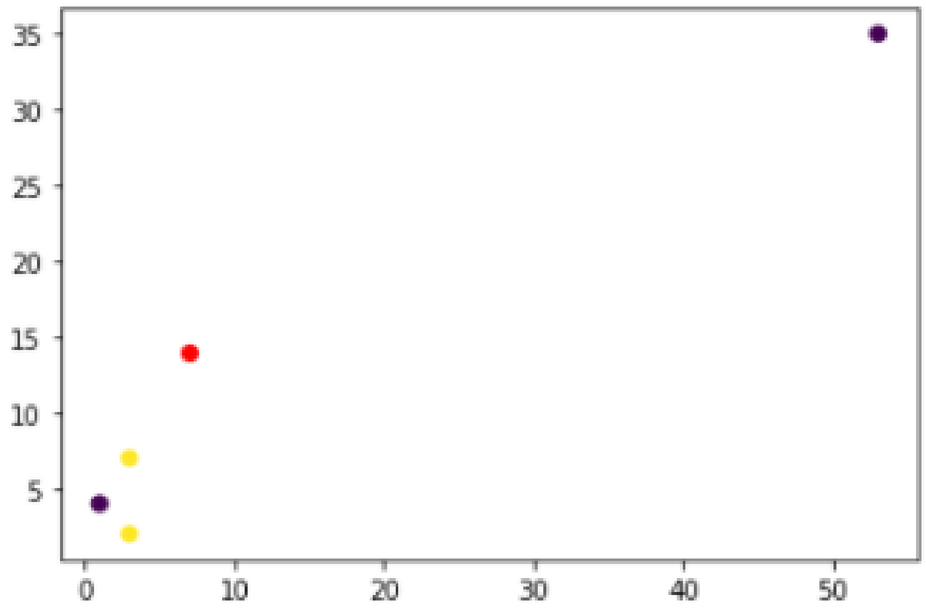
In exercise 3 training time is 0.00015385318453754s while in this exercise training time is 0.011083841323852539s. So the training time has been increased.

7. **Classify a new input vector using model.predict() and plot this new point in the previous training set (Note: Use a different color to distinguish it from the training set).**

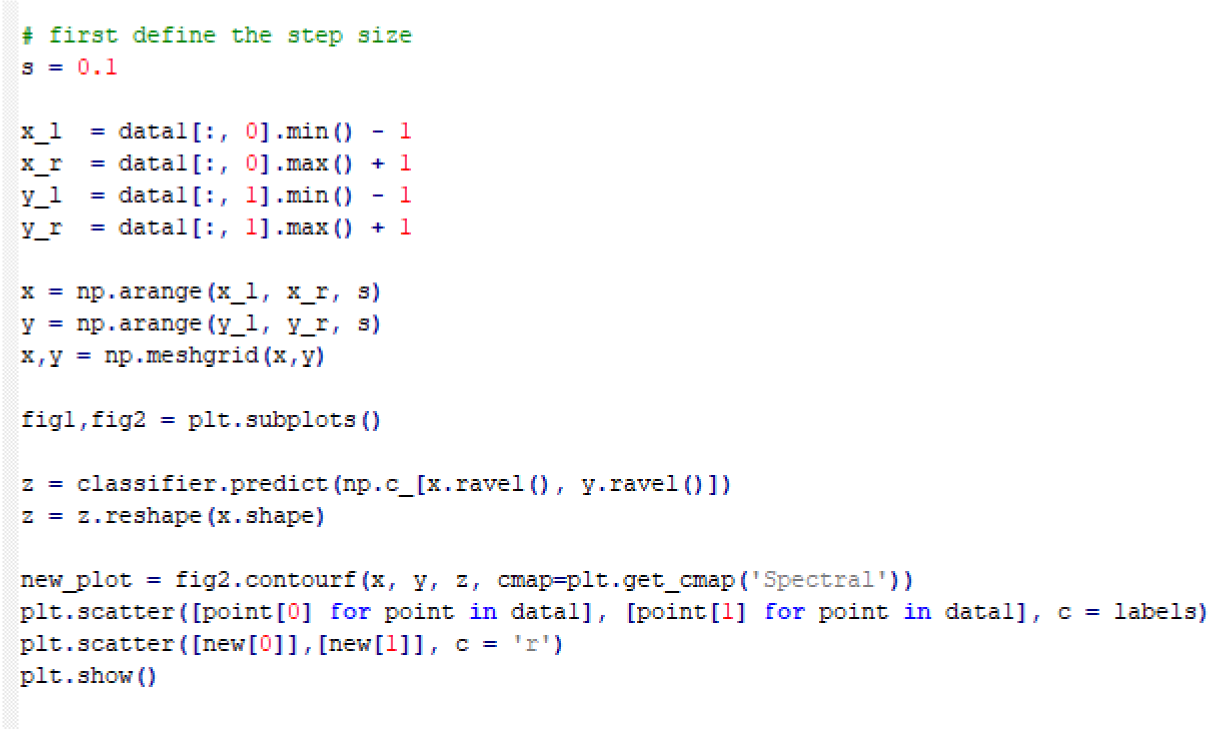


```
#-------------------------Question 7-------------------------------------------------------

new = [7,14]
print(classifier.predict([new]))

plt.scatter([point[0] for point in data1], [point[1] for point in data1], c = labels1)
plt.scatter([new[0]],[new[1]], c = 'r')
plt.show()
```

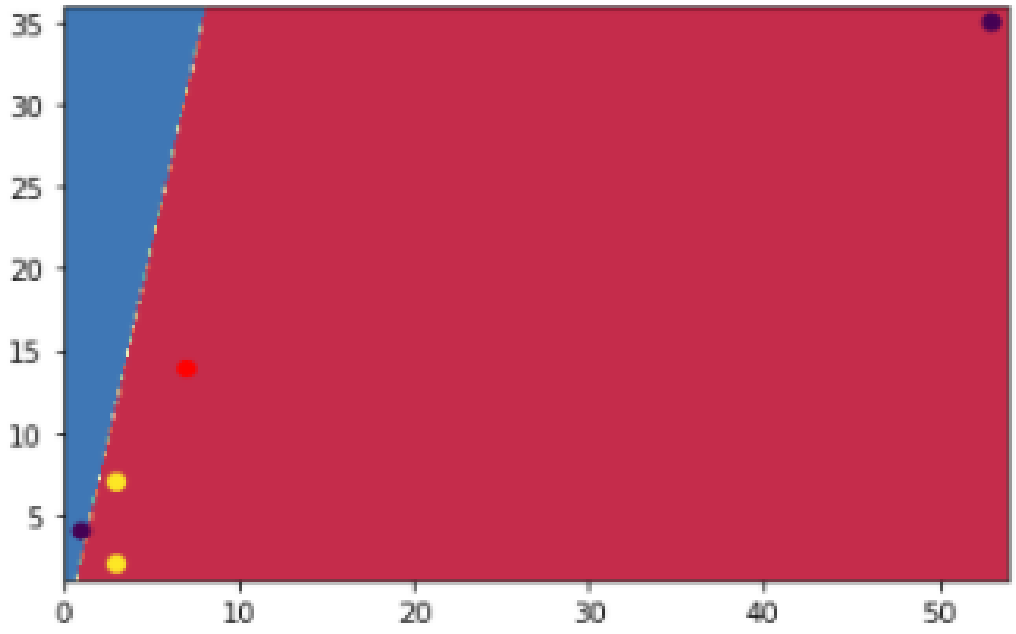
```
[0]
```



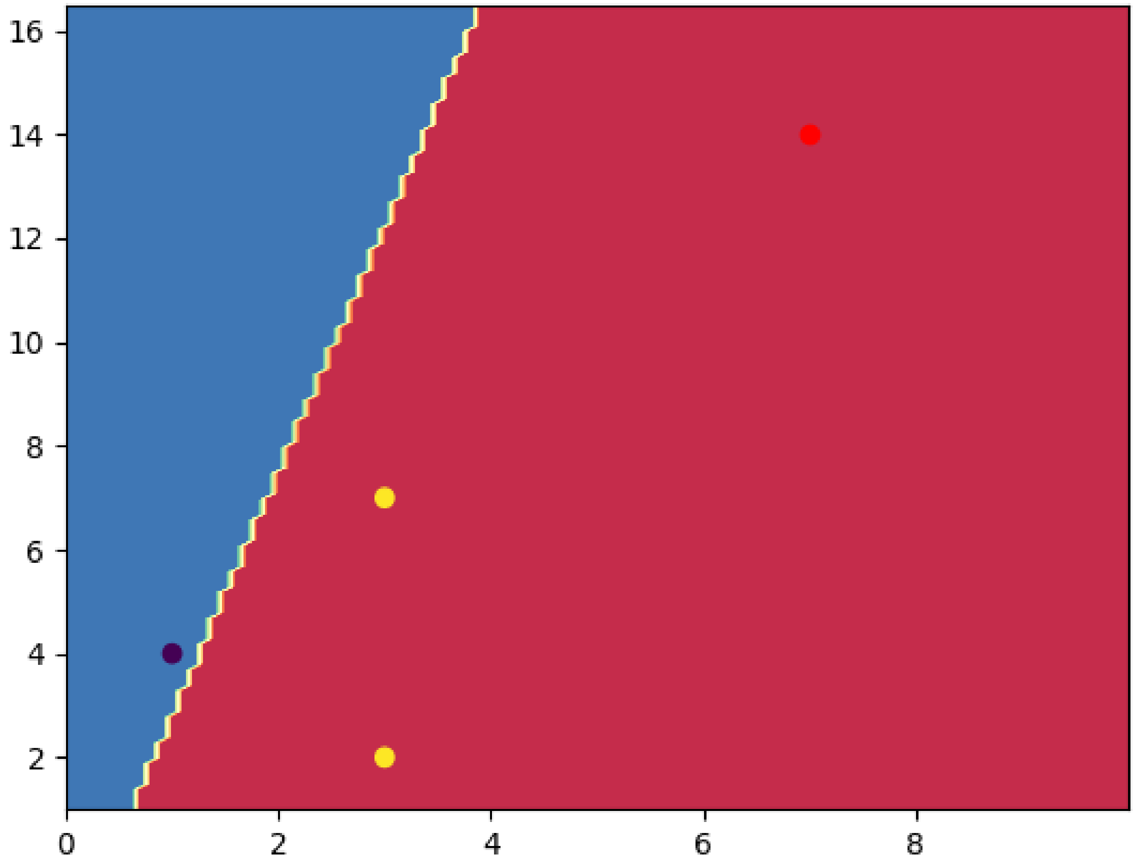
8. Add the training set and the classification line to the plot.



```
# first define the step size
s = 0.1

x_l  = data1[:, 0].min() - 1
x_r  = data1[:, 0].max() + 1
y_l  = data1[:, 1].min() - 1
y_r  = data1[:, 1].max() + 1

x = np.arange(x_l, x_r, s)
y = np.arange(y_l, y_r, s)
x,y = np.meshgrid(x,y)

fig1,fig2 = plt.subplots()

z = classifier.predict(np.c_[x.ravel(), y.ravel()])
z = z.reshape(x.shape)

new_plot = fig2.contourf(x, y, z, cmap=plt.get_cmap('Spectral'))
plt.scatter([point[0] for point in data1], [point[1] for point in data1], c = labels)
plt.scatter([new[0]],[new[1]], c = 'r')
plt.show()
```



9. **Zoom in the plot on an area of interest.**


16
14
12
10
8
6
4
2
0
2
4
6
8