CSPE 26/ECPE 26: DEEP LEARNING

Lab Assignment #1: Perceptron and non-separable data	September 7, 2017

Installation step: type the following command in the terminal: **pip install matlabplot –user**

1. Write a Python program that imports some functions and libraries as shown below, and initializes the training data to correspond to the "OR" function. Note that the vectors are augmented.

```
from random import choice
from numpy import array, dot, random
import numpy
import matplotlib.pylab as plt

unit_step = lambda x: 0 if x < 0 else 1

training_data = [(array([0, 0, 1]), 0),
                 (array([0, 1, 1]), 1),
                 (array([1, 0, 1]), 1),
                 (array([1, 1, 1]), 1), ]
```

What are the above lines doing?

2. Initialize the testing data to correspond to the "XOR" function, and also the transformed data to correspond to the "XOR" function.

3. Write a perceptron function (see the header below) that takes the following parameters: n - the maximum number of training data vectors it will examine, eta - the learning rate, and data - the input training data. In each iteration, it should randomly pick a training data vector, update the weight vector if it is mis-classified. When the number of iterations reaches 'n', it should print the output of the perceptron for each of the input vectors. It should finally return the weight vector.

```
def perceptron(n,eta,data):
```

You may use the "choice" function to pick a data vector randomly, as shown below:

```
x, expected = choice(data)
```

4. Write a function "transform(data)" to transform a given data set.

```
def transform(data):
```

It should take the two dimensional data set and compute two new dimensions as shown below:
$y_1 = \exp(-\frac{1}{2}||\boldsymbol{x} - \boldsymbol{a}||^2)$
$y_2 = \exp(-\frac{1}{2}||\boldsymbol{x} - \boldsymbol{b}||^2)$
where x is the original feature vector, $[y_1, y_2]T$ is the corresponding transformed feature vector, and
$\boldsymbol{a} = [0, 0]^T$ and $\boldsymbol{b} = [1, 1]^T$.

5. Use the perceptron function and run it on the training data to output the results.

6. Write a function "plot" to plot the data points and the decision boundary.

```
def plot(weights,data):
```

The plot function should separate the data points based on the labels, and use

```
plt.scatter()
```

to plot the points in different colours. It should also determine two end points of the decision boundary based on the weight vector, and use

```
plt.plot()
```

to plot the decision boundary.

7. Plot the training data and the decision boundary.
8. Run the perceptron algorithm on the testing data and plot the decision boundary. What do you observe?
9. Transform the data using the transform function.
10. Run the perceptron again to output the results on the transformed data. Is the transformed data separable?
11. Plot the transformed points and the decision boundary.