Multi-Layer Perceptron

Ishita Jaju
16BCE1059

1 Multi-Layer Perceptions

A perceptron that has a single layer of weights cannot solve problems like XOR. It can only approximate linear functions of the input, where the discriminant to be estimated is non-linear.
When used for classification, such multilayer perceptrons (MLP) can implement nonlinear discriminants and if used for regression, it can approximate nonlinear functions of the input.

1.1 How it works

Input feature vector X is fed to the input layer, then the activation propagates in the forward direction. The values of the hidden units are calculated.

Each hidden unit is a perceptron by itself and applies the nonlinear function to its weighted sum. The output y_i are perceptrons in the second layer taking the hidden unit as their inputs.

2 MLP on my dataset

```
from sklearn.neural_network import MLPClassifier
clf = MLPClassifier(solver='lbfgs', alpha=1e-5,
                    hidden_layer_sizes=(5, 2), random_state=1)
```

```
clf.fit(X_train,y_train)
y_pred = clf.predict(X_test)
```

```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

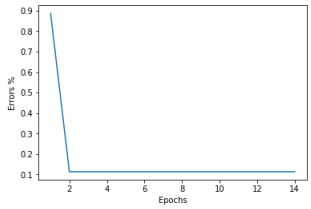
```
0.8867924528301887
```

The accuracy comes out to be 88%.

3 Training error over a number of Epochs

```
acc = []
for i in range(1,15):
    clf_test = MLPClassifier(solver = 'lbfgs',hidden_layer_sizes=(1,1),alpha=1e-5,random_state=1,max_iter=i)
    clf_test.fit(X_train,y_train)
    y_pred_ = clf_test.predict(X_test)
    acc.append(1-(accuracy_score(y_test,y_pred_)))
```

```
i = range(1,15)
plt.plot(i,acc)
plt.xlabel("Epochs")
plt.ylabel("Errors %")
plt.show()
```



The error becomes constant after just 2 iterations since the dataset is not very large.

4 Tests with multiple layers

```
for i in range(1,5):
    clf_test = MLPClassifier(solver = 'lbfgs',hidden_layer_sizes=(i,1),alpha=1e-5,random_state=1)
    clf_test.fit(X_train,y_train)
    y_pred_ = clf_test.predict(X_test)
    print("For no_of_layers=",i,"accuracy = ",(accuracy_score(y_test,y_pred_)))
```

```
For no_of_layers= 1 accuracy =  0.8867924528301887
For no_of_layers= 2 accuracy =  0.8867924528301887
For no_of_layers= 3 accuracy =  0.8867924528301887
For no_of_layers= 4 accuracy =  0.8867924528301887
```