MSBD6000B Project_1

Data Preprocessing

1. Data Reading

In this part, I defined a class for data reading named Data, the library of this part is mainly the numpy. For data structures I applied in this part, I used different lists for the features, tags, and the test data. For the most out layer, I used a dictionary to store them all. In the end, for all the lists in the dictionary, I used numpy.array to transform the list.

In order to avoid unexpected errors standing out in this session, I used the try structure for errors. The followings are some code details:

```
class Data:
    def __init__(self):
        self.data_dic = {}
        self.Feature = []
        self.Class = []
        self.testData = []
        # read traindata.csv
```

```
try:
    self.data = open('traindata.csv')
    for line in self.data:
        line = line.replace('\n', '').split(',')
        self.Feature.append(line)
    self.Feature = np.array(self.Feature)
    self.data_dic['Feature'] = self.Feature
except Exception, e:
    print 'Reading data error!'
    print e
# read trainlabel.csv
```

2. Data Preprocessing

In this part, I applied the processing.normalize to do the preprocessing. The library used here is the sklearn.preprocessing. The following is the details:

```
def preDeal(self):
    self.X = preprocessing.normalize(self.data_dic['Feature'])
    self.Y = self.data_dic['Class']
```

Training Models

1. Neural Network

In this model, I used the TensorFlow for constructing the deep learning model, BP Neural Net. The followings are the parameters for this deep learning model:

Parameters:

The input layer: 57 neutrals for the input

The hidden layer: 45 neutrals, activation function: tf.nn.relu

45 neutrals, activation function: tf.sigmoid

The output layer: 1 neutrals, activation function: tf.sigmoid

```
# two hiden layer
hidden_layer = add_layer(xs, 57, 45, activation_function = tf.nn.relu)
hidden_layer1 = add_layer(hidden_layer, 45, 45, activation_function = tf.sigmoid)
# output layer
prediction = add_layer(hidden_layer1, 45, 1, activation_function = tf.sigmoid)
```

Learning Rate: 0.1, AdamOptimizer

```
train_step = tf.train.AdamOptimizer(0.1).minimize(loss)
```

Training Times: 3000

```
for i in range(3000):
    sess.run(train_step, feed_dict = {xs: x_data, ys: y_data})
```

The accuracy result: 0.990

```
0.990062111801
```

Prediction:

```
temp = sess.run(prediction, feed_dict = {xs: preprocessing.normalize(ob.testData)})
```

2. Output into File

In this file, I just output the prediction result to the file. The followings are the output details:

```
path = os.path.abspath('.')
path = path + '/'+ 'project1_20453306'
try:
    with open(path, "wb") as file:
        for item in result:
            #print item
            file.write(str(float(item)))
            file.write('\n')
    file.close()
except Exception, e:
    print e
```

Code Resources

GitHub: https://github.com/NeroJ/MSBD-6000B-Proj_1