chine-learning-challenges-aravindh

April 21, 2023

```
[ ]: import numpy as np
     import matplotlib.pyplot as plt
     import pandas as pd
     import h5py
     import scipy
     from PIL import Image
     from scipy import ndimage
     from sklearn.model_selection import train_test_split
     from sklearn import ensemble
     from sklearn.metrics import mean_absolute_error
     import sklearn.externals
     import tensorflow as tf
```

```
[ ]: test_data=pd.read_csv('mnist_test.csv')
     train_data=pd.read_csv('mnist_train.csv')
```

```
[ ]: test_data.head()
```

```
[ ]: test_data.info()
```

```
[ ]: train_data=pd.read_csv("mnist_train.csv")
```

```
[ ]: train_data.info()
```

```
[ ]: test_data.describe()
```

```
[ ]: train_data.describe()
```

```
[ ]: x=test_data['label']
     x=x.transpose()
     y=test_data['label']
     y=y.transpose()
```

```
[ ]: x=train_data['label']
     x=x.flatten()
     y=train_data['label']
     y=y.flatten()
```

[ ]:
```
# normalization
x=x/225
```

[ ]:
```
x_train = x[:785, :]
y_train = y[:785]
x_test = x[60000:, :]
y_test = y[60000:]
```

[ ]:
```
m = x.shape[0]
input_layer_size = 784
hidden_layer_size = 100
num_labels = 10
```

[ ]:
```
initial_Theta1 = initialise(hidden_layer_size, input_layer_size)
initial_Theta2 = initialise(num_labels, hidden_layer_size)
```

[ ]:
```
initial_nn_params = np.concatenate((initial_Theta1.flatten(), initial_Theta2.
  ↪flatten()))
maxiter = 100
lambda_reg = 0.1
myargs = (input_layer_size, hidden_layer_size, num_labels, x_train, y_train,␣
  ↪lambda_reg)
```

[ ]:
```
Theta1 = np.reshape(nn_params[:hidden_layer_size * (input_layer_size + 1)], (
                    (hidden_layer_size, input_layer_size + 1)) # shape = (100,␣
  ↪785)
Theta2 = np.reshape(nn_params[hidden_layer_size * (input_layer_size + 1):],
                    (num_labels, hidden_layer_size + 1)) # shape = (10, 101)
```

[ ]:
```
# Checking test set accuracy of our model
pred = predict(Theta1, Theta2, x_test)
print('Test Set Accuracy: {:f}'.format((np.mean(pred == y_test) * 100)))

# Checking train set accuracy of our model
pred = predict(Theta1, Theta2, x_train)
print('Training Set Accuracy: {:f}'.format((np.mean(pred == y_train) * 100)))
```

[ ]:
```
# Evaluating precision of our model
true_positive = 0
for i in range(len(pred)):
    if pred[i] == y_train[i]:
        true_positive += 1
false_positive = len(y_train) - true_positive
print('Precision =', true_positive/(true_positive + false_positive))
```

[ ]:

```
# Saving Thetas in .txt file
np.savetxt('Theta1.txt', Theta1, delimiter=' ')
np.savetxt('Theta2.txt', Theta2, delimiter=' ')
```