CS498 AML HW3 REPORT

Pengyu Cheng
pcheng11

- 50 MSE fitted in a table

0N	1N	2N	3N	4N	0c	1c	2c	3c	4c
4.54247067	0.38345031	0.175563	0.14178365	0.16083836	4.54311903	0.38461353	0.17781528	0.14444051	0.16083836
4.54247067	0.64109318	0.71562849	0.90839291	1.11565786	4.54953899	0.64864211	0.75062113	0.94197282	1.11565786
4.54247067	1.29037245	1.96724039	2.65084114	3.65327973	4.55747296	1.32346215	2.11974805	3.02737992	3.65327973
4.54247067	0.79994274	0.82808255	0.98494977	1.194	4.56619867	0.84061416	1.2070898	1.27119197	1.194
4.54247067	1.91776775	3.3317221	4.5482572	5.13926667	4.919928	2.83567943	4.6514345	4.97124727	5.13926667

CS498 AML HW3 REPORT
Pengyu Cheng
pcheng11

- Code (Used PCA library from scikit-learn)

```
def reconstruct_noise(data, mean_val, num_component):
    pca = PCA(n_components=num_component, svd_solver='full')
    scores = pca.fit_transform(data)
    re_data = pca.inverse_transform(scores)
    return re_data

def reconstruct_noiseless(data, non_noise_data, mean_val, num_component):
    pca = PCA(n_components=num_component, svd_solver='full')
    pca.fit(non_noise_data) #using non-noise to fit
    scores = pca.transform(data)
    re_data = pca.inverse_transform(scores)
    return re_data

def write_csv(dir, res, flag):
    if flag == 2:
        with open(dir + 'pcheng11-numbers.csv', 'w+') as f:
            writer = csv.writer(f)#, quoting=csv.QUOTE_NONNUMERIC)
            writer.writerow(["0N","1N","2N","3N","4N","0c","1c","2c","3c","4c"])
            for i in res:
                writer.writerow(i)
    else:
        with open(dir + 'pcheng11-recon.csv', 'w+') as f:
            writer = csv.writer(f)#, quoting=csv.QUOTE_NONNUMERIC)
            writer.writerow(["X1","X2","X3","X4"])
            for i in res:
                writer.writerow(i)
```

- Main function:

```
def main():
    data_dir = '/Users/pengyucheng/Desktop/cs498_aml/cs498_aml_hw3/hw3-data/'
    write_dir = '/Users/pengyucheng/Desktop/cs498_aml/cs498_aml_hw3/hw3_result/'
    datas = read_data(data_dir+'dataI.csv', data_dir+'dataII.csv', data_dir+'dataIII.csv',\
                      data_dir+'dataIV.csv', data_dir+'dataV.csv', data_dir+'iris.csv' )
    non_noise_data = datas[-1]
    noiseless_mean_val = np.mean(non_noise_data, axis=0)
    res = []
    for data in datas[:-1]:
        sub_res = []
        #noiseless
        for num_component in range(5): #0~4 pca
            if num_component == 0:
                re_data = np.zeros((150, 4))
                for k in range(4):
                    re_data[:,k] = noiseless_mean_val[k]
            else:
                re_data = reconstruct_noiseless(data, non_noise_data, noiseless_mean_val, num_component)
            MSE = 1/150 * np.sum((non_noise_data - re_data)**2)
            sub_res.append(MSE)
        #noise
        mean_val = np.mean(data, axis=0)
        for num_component in range(5): #0~4 pca

            if num_component == 0:
                re_data = np.zeros((150, 4))
                for k in range(4):
                    re_data[:,k] = mean_val[k]
            else:
                re_data = reconstruct_noise(data, mean_val, num_component)
            print(num_component)
            print(re_data[0,:])
            MSE = 1/150 * np.sum((non_noise_data - re_data)**2)
            sub_res.append(MSE)
        res.append(sub_res)
    print(res)
    #task 1
    data2 = reconstruct_noise(datas[1], np.mean(data[1], axis=0), 2)
    write_csv(write_dir, data2, 1)
    #task 2
    write_csv(write_dir, res, 2)
```