# Lab\_1 基于PCA/LDA降维方法的特征可视化

## 一、实习目的与要求

1、结合实际应用理解数据降维；

2、理论结合实践，采用PCA或者LDA对所提供的20664\*4097维数据进行降维，并进行可视化；

3、通过进一步查阅文献，了解相关研究方向的最新研究进展。

## 二、实习题目

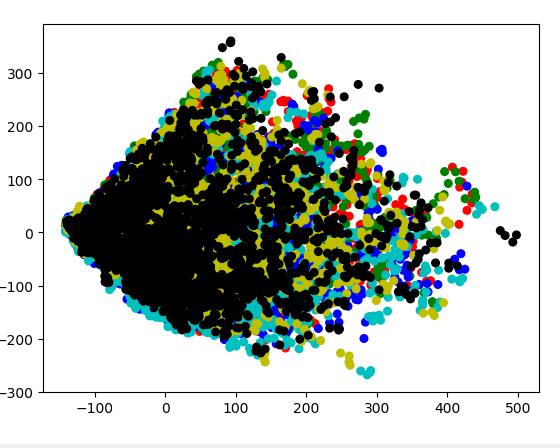
基于PCA/LDA降维方法的特征可视化

#### 【实验数据】

data.mat， 20664\*4097维， 每一行表示一个特征向量，每一个特征向量的最后一列（第4097列）为特征向量的类别，共20664个特征向量。

#### 【过程描述】

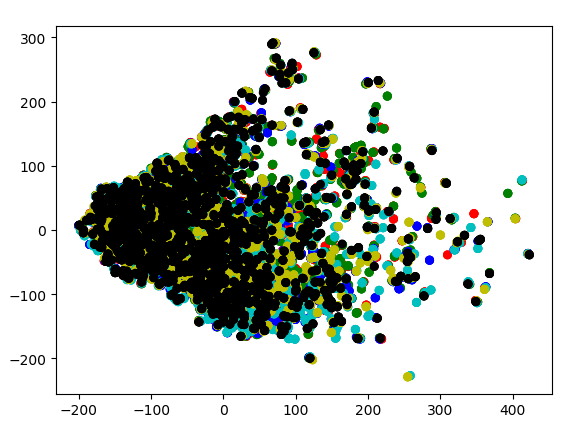
1. 数据处理：使用pyrhon的scipy库读取mat矩阵，并将最后一列单独提取出来记作label，前4096列的矩阵几位mat.
2. 降维方法：（1）只使用PCA降维的方法，将矩阵直接降维到2维。类之间分别不明显。



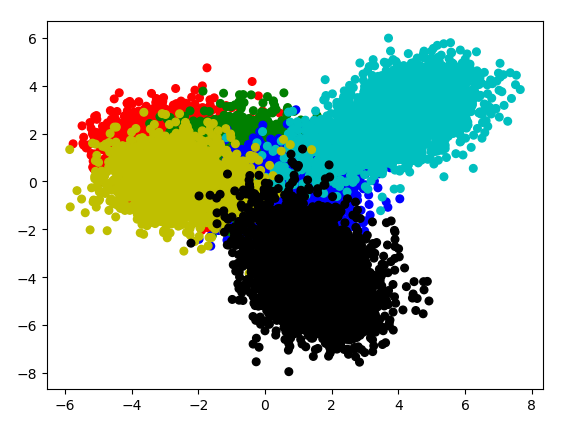
效果如上，类之间区别不明显，混作一团。

（2）采用SKLearn中的PCA降维法，时间消耗相对减少，但效果依然不好。（结果和上面一样）

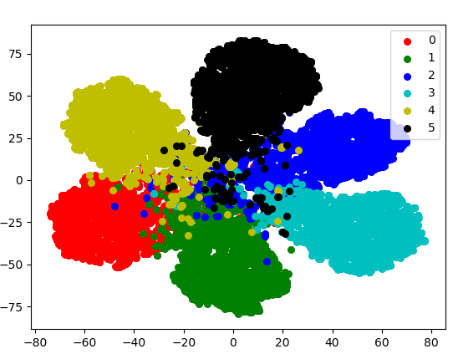
（3）PCA降到10维,T-SNE降到2维(分类效果相对较好)



（3）采用LDA降维方式，直接降到2维。（因为LDA分类效果好，降维后相比于PCA类之间区分明显）效果如下（这里采用sklearn中的LDA降维方法）

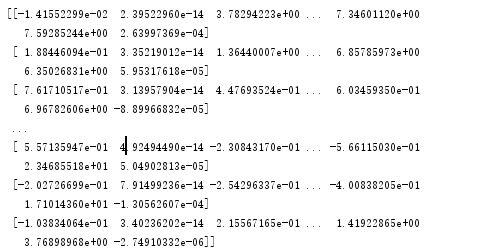


（3）采用T-SNE和PCA降维结合的方式，先使用PCA降维到10维，后采用T-SNE降维到2维的方式。



（4）SVD降维

这种降维方式是对原矩阵A(m\*n)进行进行分解得到奇异值向量S,然后类似于PCA的有效特征值提取过程，根据所需维度，或者有效百分比提取前k个奇异值转化为奇异值对角矩阵sigmal(k\*k),然后原来分解的左奇异矩阵减少为k列即U(m\*k)，右奇异矩阵减少到k行VT(k\*n).最后还原(U\*sigmal\*VT)百分比为0.93;但SVD降维不知道该怎么可视化出来



【源代码】

（附上源代码）

1. **数据预处理**

*#读取文件***def** readFile():  
 file=**"data.mat"**;  
 Data=scipy.loadmat(file);  
 Data=np.array(Data[**'data'**]);  
  
 column=Data.shape[1]-1;  
 arrayMat=Data[:,0:column];  
 label=Data[:,column]  
 **return** arrayMat,label;

1. **降维方法（要求至少选了一种降维方法：PCA、LDA、SVD等）**
2. **PCA降维自己编写版**

*#归一化过程***def** featureNormalize(X):  
 *'''（每一个数据-当前列的均值）/当前列的标准差'''* n = X.shape[1]  
 l=X.shape[0];  
 meanVal = np.zeros((1, n));  
 sigma = np.zeros((1, n))  
  
 meanVal = np.mean(X, axis=0) *# axis=0表示列* sigma=np.std(X,axis=0)  
 newData=X-meanVal;  
 **return** newData,meanVal;  
  
*#当前计算PCA要提取的特征值的数量***def** percentageCal(eigVals,percent):  
 arraySort=np.sort(eigVals);  
 arraySort=arraySort[-1::-1]; *#倒叙* sumVects=sum(eigVals);  
 num=0;  
 total=0;  
 **for** vect **in** eigVals:  
 num+=1;  
 total+=vect;  
 **if** total>=sumVects\*percent:  
 **return** num;  
  
*#PCA降维过程***def** PCADepress(mat,percent):  
 newData, meanVal = featureNormalize(mat);  
 covMat = np.cov(newData, rowvar=0); *# 求协方差矩阵,return ndarray；若rowvar非0，一列代表一个样本，为0，一行代表一个样本* eigVals, eigVects = np.linalg.eig(np.mat(covMat)); *# 求特征值和特征向量,特征向量是按列放的，即一列代表一个特征向量* n = percentageCal(eigVals, percent); *# 获取要选择的特征值数n* eigValIndice = np.argsort(eigVals); *# 对特征值从小到大排序，返回次序* n\_eigVals = eigValIndice[-1:-(n + 1):-1];  
  
 n\_eigVect = eigVects[:, n\_eigVals] *# 最大的n个特征值对应的特征向量* lowDDataMat = newData \* n\_eigVect *# 低维特征空间的数据  
 # 最后输出的低维矩阵* reconMat = (lowDDataMat \* n\_eigVect.T) + meanVal *# 重构数据* **return** lowDDataMat, reconMat;  
  
**def** PCA(mat, N):  
 newData, meanVal = featureNormalize(mat);  
 covMat = np.cov(newData, rowvar=0); *# 求协方差矩阵,return ndarray；若rowvar非0，一列代表一个样本，为0，一行代表一个样本* eigVals, eigVects = np.linalg.eig(np.mat(covMat)); *# 求特征值和特征向量,特征向量是按列放的，即一列代表一个特征向量  
 #n = percentageCal(eigVals, percent); # 获取要选择的特征值数n* eigValIndice = np.argsort(eigVals); *# 对特征值从小到大排序，返回次序* n\_eigVals = eigValIndice[-1:-(N + 1):-1];  
  
 n\_eigVect = eigVects[:, n\_eigVals] *# 最大的n个特征值对应的特征向量* lowDDataMat = newData \* n\_eigVect *# 低维特征空间的数据  
 # 最后输出的低维矩阵* reconMat = (lowDDataMat \* n\_eigVect.T) + meanVal *# 重构数据* **return** lowDDataMat, reconMat;

1. **PCA SKLean库版**

**def** pca(arrayMat,percentage):  
 pca = PCA(n\_components=percentage); *# ;读取arrayMat时'mle'的情况不行* n\_mat=pca.fit\_transform(arrayMat);  
 ratio=pca.explained\_variance\_ratio\_;  
  
 **return** n\_mat,ratio;

1. **LDA降维**

**def** LDA(mat,label):  
 lda=LinearDiscriminantAnalysis(n\_components=10);  
 lda.fit(mat,label);  
 n\_mat=lda.transform(mat);  
 ratio=lda.explained\_variance\_ratio\_;  
 **return** n\_mat,ratio;

**（4）LDA结合T-SNE降维**

**def** LDA(mat,label):  
 lda=LinearDiscriminantAnalysis(n\_components=10);  
 lda.fit(mat,label);  
 n\_mat=lda.transform(mat);  
 ratio=lda.explained\_variance\_ratio\_;  
 **return** n\_mat,ratio;  
  
*#T-SNE降维方法***def** SneDepress(X\_mat):  
 tsne=TSNE(n\_components=2,init=**"pca"**,random\_state=0);  
 X\_tsne=tsne.fit\_transform(X\_mat);  
  
 **return** X\_tsne;

**（3）模型训练**

**（4）可视化评价**

*#可视化过程***def** paint(n\_mat,label):  
 n\_mat=np.mat(n\_mat);  
 fig=plt.figure();  
 color=colors.ListedColormap([**'r'**,**'g'**,**'b'**,**'c'**,**'y'**,**'k'**]);  
 ax=fig.add\_subplot(111);  
   
 ax.scatter(n\_mat[:, 0].flatten().A[0], n\_mat[:, 1].flatten().A[0], marker=**'o'**, s=30, c=label,cmap=color);  
 *#plt.legend();* plt.show();

#### 【改进设想】

（或查阅文献提出改进设想；或考虑与第三方机器学习库分类结果的比较；或与采用其他分类算法实现的结果比较等）。

1. 针对T-SNE时间复杂度相对较大，可采用LDA降到更低的维度，再使用T-SNE降维会好一些。
2. 暂时未尝试基于SVM的降维，后期可采用SVM降维的方式于LDA降维做对比，选取出最优的方法。

## 三、本次实习小结

通过第一次实习，了解了多种降维方式，也简单了解了他们的优缺点，其中主要是PCA和LDA的对比，PCA降维时间复杂度底，但分类效果很差，LDA分类效果好，但时间复杂度相对较大。而TSNE的降维时间复杂度最大，但分类效果最明显。最后尝试了一下SVD，不同于以上几种，它是对原矩阵的分解，对其分解的奇异值向量进行降维。最后再用降维后的矩阵和向量还原，得到相似矩阵。但是很难表达出降维的含义。

=======================================================

我的联系方式：

电话：13349830890

邮箱：liuyy@cug.edu.cn