Skfeature 使用文档

Loading an dataset（加载数据集）

在 shell 中输入 python，这里我们用到的测试数据集是：COIL20.mat.（matlab 格式文件），输入如下命令加载数据集 COIL20.mat

```
$ python

>>>import scipy.io

>>>mat = scipy.io.loadmat("COIL20.mat")
```

通过上述步骤加载数据集后，mat['X']存放所有样本所对应的特征向量, mat['Y']

存放有所有样本对应的真实类标签.

```
>>>X=mat['X'] // 数据集 COIL20 中特征值存在数组 X 中

>>>print X // 打印数组 X 的每个元素



>>>y = mat['Y'][:, 0] //数据集 COIL20 中真实类标签存在数组 y 中

>>>print y // 打印数组 y 的每个元素
>>
```

Shape of the data arrays（特征矩阵表示）

特征矩阵可以表示为一个 2 维数组.例如，在数据集中 COIL20.mat，输入如下命

令：

```
>>>import numpy as np

>>>n_samples, n_features = np.shape(X) // n_samples 表示样本量，n_features 表示特征个数

>>>print n_samples, n_features // 打印样本量和特征个数
```

```
>>>1440 1024  //打印的结果

>>>n_labels = np.shape(y) // n_labels 表示类标签个数

>>>print n_labels //打印类标签个数

>>>(1440L,) //打印的结果
```

上面我们做了数据集的加载操作，接下来我们对数据集进行特征选择：

- ## 将数据集分为训练集和测试集

通过类 sklearn.cross_validation 中的函数 train_test_split 对数据集进行切割．例如，在数据集 COIL20.mat 中，我们设 20%的数据为测试集，输入如下命令：

```
>>> from sklearn.cross_validation import train_test_split

>>> X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=40) // X_train, X_test, y_train, y_test 分别表示训练样本，测试样本，训练集类标签，以及测试集类标签
```

- ## 对训练集进行特征选择

这里以 Fisher Score 特征算法为例进行讲解．首先，我们计算训练集中所有特征的 Fisher Score 分数：

```
>>>from skfeature.function.similarity_based import fisher_score
>

>>>score = fisher_score.fisher_score(X_train, y_train) // score 表示训练集中不同特征对应 Fisher Score 分数的数组

>>>print score // 打印数组 score

>>>[ 13.96904931   0.5376816    0.19923194 ...,   3.71944606  14.01720752  14.05075518] // 打印结果
```

输入如下命令：对 Score 数组进行降序排列，得到新的数组 idx，

```
>>>idx = fisher_score.feature_ranking(score)
>>
>>>print idx
>>>[1023 1022   31 ...,   34   97  897]
```

接下来，对训练集和测试集的样本进行特征选择。我们选择前 n 个“最重要”特征（这里，假设我们选了前 5 个“最重要”特征）

```
>>>num_fea = 5
>>>selected_features_train = X_train[:, idx[0:num_fea]]
>>>selected_features_test = X_test[:, idx[0:num_fea]]
>>>print selected_features_train // 打印训练样本经过特征选择后的特征矩阵
>>>print selected_features_test //打印测试样本经过特征选择后的特征矩阵
```

•基于特征选择后的训练样本训练分类模型

这里以 SVM（分类器）为例：

```
>>>from sklearn import svm
>>>clf = svm.LinearSVC() // SVM 分类器中，选择线性核
>>>clf.fit(selected_features_train, y_train) // 利用 svm，对特征选择后的训练样本训练模型
```

• 模型训练完，对测试样本进行预测

```
>>>y_predict = clf.predict(selected_features_test) // 对测试集进行预测
>>>print y_predict // 打印所预测的类标签

>>>[19 19  2 ..., 19 19 19] // 打印结果
```

- 性能评估（看看特征选择的效果）

```
>>>from sklearn.metrics import accuracy_score // 加载统计准确率的函数
>>>acc = accuracy_score(y_test, y_predict) // 统计准确率

>>>print acc // 打印准确率
>>>0.09375 // 打印结果
```
