【华子】机器学习039-NLP文本分类器 -

(本文所使用的Python库和版本号: Python 3.6, Numpy 1.14, scikit-learn 0.19, matplotlib 2.2， NLTK 3.3)

1. 20 Newsgroups数据集介绍

本文要使用NLP中非常经典的一个数据集：20 Newsgroups。这个数据集是国际标准数据集之一，专门用于文本分类，文本挖掘，和信息检索等领域，类似于传统机器学习所用的Iris鸢尾花数据集一样，可以通过官方网址来下载和了解具体内容。

20 Newsgroups包含有20个类别的已经标注好的样本，总样本数量大约2万。这20个类别分别为：

comp.graphics comp.os.ms-windows.misc comp.sys.ibm.pc.hardware comp.sys.mac.hardware comp.windows.x	rec.autos rec.motorcycles rec.sport.baseball rec.sport.hockey	sci.crypt sci.electronics sci.med sci.space
misc.forsale	talk.politics.misc talk.politics.guns talk.politics.mideast	talk.religion.misc alt.atheism soc.religion.christian

这个数据集有三个版本，其主要内容和区别为：

1. 20news-19997.tar.gz: 最原始的没有修改过的一个版本。
2. 20news-bydate.tar.gz: bydate版本，按照时间分类，分为训练集（60%）和测试集（40%）两部分，不包含重复文档和新闻组名。一共有18846个样本（或称为文档）
3. 20news-18828.tar.gz: 不包含重复样本，只有来源和主题，一共有18828个样本。

sklearn中有两种加载方式，第一种是sklearn.dataset.fetch_20newsgroups，返回一个可以被文本特征提取器（CountVectorizer)自定义参数提取特征的原始文本序列，第二种是sklearn.datasets.fetch_20newsgroups_vectorized，返回一个已提取特征的文本序列，即不需要使用特征提取器。

此处我们只下载第二个版本，下载后得到20news-bydate.tar.gz文件，解压后得到两个文件夹，如下：

这两个文件夹每一个都有20个子文件夹，对应于20个不同类别。每个类别下面有几百个文档，即样本，每个文档都不长，比如一个文档（样本）的内容为：

```
From: zack@netcom.com (Zack T. Smith)
Subject: Strange exposure problem
Organization: NETCOM On-line Communication Services (408 241-9760 guest)
Lines: 19

Hi,

I'm trying to write a Motif program on an Interactive Unix machine, and I
observing very strange behavior when my program attempts to expose a
DrawingArea. Namely, some Xlib operations work, and some do not. In
particular, the expose consist of two XFillRectangle calls followed by sc
XDrawPoint calls, and for reasons unknown to me the point calls are faili
whenever a pulldown or popup up menu is (clicked on and) moved in the
rightward direction over the drawing area, but after the move, is still c
some part of the drawing area. This also happens less consistently when t
pulldown/popup is moved in the leftward direction.

Assuming that my code is not doing anything incredibly odd, is this a
server
bug?
```

这个数据集的加载方式已经被sklearn集成到代码中了，主要的接口是sklearn.dataset.fetch_20newsgroups，其默认加载第二个版本。这个函数的参数有：subset有三个选择train、test、all，选择数据的类型。category是选择新闻的类型，remove是可以选择去除('headers', 'footers', 'quotes')这三个文章的选项。

(——IOPub_data_rate_limit报错解决方法)

```
# 认识20newsgroups数据集
from sklearn.datasets import fetch_20newsgroups
# dataset=fetch_20newsgroups(subset='all')
# 自动下载第二个版本20news-bydate.tar.gz

# train_set=fetch_20newsgroups(subset='train') # 仅仅提取中间的train set
# test_set=fetch_20newsgroups(subset='test')

# 如果仅仅需要其中的某几个类别，可以用
sample_cate = ['alt.atheism', 'soc.religion.christian',
               'comp.graphics', 'sci.med', 'rec.sport.baseball'] # 只取5个类别
#shuffle=True:打乱数据，移除邮件的header，footers，>符号
train_set = fetch_20newsgroups(subset='train',categories=sample_cate,
                               shuffle=True, random_state=42,
                               remove = ('headers', 'footers', 'quotes'))
test_set = fetch_20newsgroups(subset='test', categories=sample_cate,
                              shuffle=True, random_state=42,
                              remove = ('headers', 'footers', 'quotes'))
print(len(train_set.data), len(test_set.data)) # 2854 1899
print(train_set.target_names) # 只有五个类别
```

----------**输--------出**----------

2854 1899 ['alt.atheism', 'comp.graphics', 'rec.sport.baseball', 'sci.med', 'soc.religion.christian']

----------**完**----------

2. 构建分类器

2.1 准备数据集

```
# 1, 准备数据集
category_map = {'misc.forsale': 'Sales', 'rec.motorcycles': 'Motorcycles',
        'rec.sport.baseball': 'Baseball', 'sci.crypt': 'Cryptography',
        'sci.space': 'Space'}
from sklearn.datasets import fetch_20newsgroups
train_set=fetch_20newsgroups(subset='train',categories=category_map.keys(),
                             shuffle=True,remove = ('headers', 'footers', 'quotes'))
test_set=fetch_20newsgroups(subset='test',categories=category_map.keys(),
                             shuffle=True,remove = ('headers', 'footers', 'quotes'))
# 获取到的train_set包含有2968个样本,
print('train sample num: ', len(train_set.data)) # 2968
print(train_set.target_names) # 确保是我们要提取的这五个类别

print('test sample num: ', len(test_set.data)) # 1975
```

----------**输**--------**出**----------

train sample num: 2968 ['misc.forsale', 'rec.motorcycles', 'rec.sport.baseball', 'sci.crypt', 'sci.space'] test sample num: 1975

----------**完**----------

2.2 特征提取

直接上代码:

```
from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer = TfidfVectorizer(stop_words='english',lowercase=True)
train_vector = vectorizer.fit_transform(train_set.data)
print(train_vector.shape) # (2968, 31206)
# 此处相当于有2968个词袋，对这些词袋进行TfidfVectorizer进行特征提取,
# 得到最具典型的一些单词，这些单词的个数有31206个，故而得到(2968, 30206)矩阵
# 矩阵中的元素表示这个单词在该词袋中出现的tf-idf权重，值越大，表示该单词越重要。
```

2.3 定义模型，训练模型

```
# 定义模型，训练特征
from sklearn.naive_bayes import MultinomialNB
# alpha：拉普拉斯平滑参数，默认为1，0表示无平滑;fit_prior:是否使用先验概率
classifier=MultinomialNB(alpha=.01, fit_prior = False)
classifier.fit(train_vector,train_set.target)
```

2.4 查看模型在测试集上的表现

```
# 查看这个数据集在test_set上的表现
from sklearn import metrics
test_vector=vectorizer.transform(test_set.data)
print(test_vector.shape)
pred=classifier.predict(test_vector)
# macro:宏平均，micro：微平均（具体见常见名词）
F1_score=metrics.f1_score(test_set.target, pred, average='micro')
print('test set F1 score: ',F1_score)
```

3. 用GridSearch优化参数

```
# 用GridSearchCV优化参数
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import classification_report

parameters = {'fit_prior':(True, False), 'alpha':
(0.01,0.05,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1.0)}
clf = GridSearchCV(classifier,parameters,cv=5,scoring='precision_macro',n_jobs=-1)
clf.fit(train_vector, train_set.target)
print("Best param_set found on train set: {}".format(clf.best_params_))
# 分类模型报告
y_true, y_pred = test_set.target, clf.predict(test_vector)
print(classification_report(y_true, y_pred))
```

----------**输**--------**出**----------

Best param_set found on train set: {'alpha': 0.05, 'fit_prior': True} precision recall f1-score support

```
      0       0.92      0.89      0.91       390
      1       0.80      0.91      0.85       398
      2       0.93      0.88      0.91       397
      3       0.90      0.88      0.89       396
      4       0.91      0.88      0.89       394
```

avg / total 0.89 0.89 0.89 1975

----------**完**----------

从分类报告中可以看出，结果最好的是第0类和第2类，F1为0.91，最差的是第1类，F1值只有0.85。

########################**小**********结#################################

1，用NLP进行文本分类，和传统机器学习的主要区别在于前面特征的提取，一旦提取特征，后面模型的建立，训练，测试，分类报告等都一样。

2，对文本进行特征的提取有两种：CountVectorizer和TfidfVectorizer，但是TfidfVectorizer使用的最多，对文本量非常大的情况更加准确，故而此处我只用TfidfVectorizer来提取特征。

3，有一个地方很容易忽视：测试集在用predict之前，一定要用vectorizer.transform进行转换，这个过程就像是对数据进行归一化等，需要对train_X和test_X都要进行处理。

###

参考资料:

1, Python机器学习经典实例，Prateek Joshi著，陶俊杰，陈小莉译