第三次作业 分类、预测与聚类

07111906-1120192461-刘冉

仓库地址Assignment3 at main · RIU-13

如果上方链接打不开，这是网址https://github.com/RIU-13/LRDataMining0327/tree/main/Assignment3

数据集介绍

20 NewsGroup是一个包含了20种话题的英文数据集，约18000条数据，分为训练集和测试集。为方便处理数据，本次作业采用sklearn数据接口获取数据。

```python
from sklearn.datasets import fetch_20newsgroups
from pprint import pprint
train_data = fetch_20newsgroups(subset='train')
pprint(list(train_data))#输出获取的训练集
print(train_data.target)
print(train_data.target_names)
```

```
['data', 'filenames', 'target_names', 'target', 'DESCR']
[7 4 4 ... 3 1 8]
['alt.atheism', 'comp.graphics', 'comp.os.ms-windows.misc',
'comp.sys.ibm.pc.hardware', 'comp.sys.mac.hardware', 'comp.windows.x',
'misc.forsale', 'rec.autos', 'rec.motorcycles', 'rec.sport.baseball',
'rec.sport.hockey', 'sci.crypt', 'sci.electronics', 'sci.med', 'sci.space',
'soc.religion.christian', 'talk.politics.guns', 'talk.politics.mideast',
'talk.politics.misc', 'talk.religion.misc']
```

数据预处理

去除停用词、数字、符号等

```python
from sklearn.feature_extraction.text import ENGLISH_STOP_WORDS
import re
import string
#获取英文停用词表
new_train_data = []
for doc in train_data.data:
#     words = [word for word in doc.split() if word.lower() not in ENGLISH_STOP_WORDS]#去除停用词
#     new_doc = " ".join(words)
    new_doc = re.sub('[\d]','',doc)#去除数字
    new_doc = re.sub('[{}]'.format(string.punctuation),"",new_doc)#去除符号
    new_train_data.append(new_doc)

print(type(new_train_data))
```

```
<class 'list'>
```

特征向量化

将预处理后的文本使用 TF-IDF 方法进行向量化。

```
from sklearn.feature_extraction.text import TfidfVectorizer

# 提取tfidf特征
vectorizer = TfidfVectorizer(stop_words='english')#去除停用词
vectors = vectorizer.fit_transform(new_train_data)
print(vectors.shape)
print(vectors.nnz / float(vectors.shape[0]))
```

```
(11314, 123106)
100.64035707972424
```

提取的TF-IDF 向量vectors是非常稀疏的，稀疏度大小为100

K-means模型训练

设定K值为20，构建K-means模型，对向量化的文本数据进行聚类

```
from sklearn.cluster import KMeans
number_of_clusters = 20
model = KMeans(n_clusters=number_of_clusters,
               init='k-means++',
               max_iter=100, # 每次最大迭代轮数
               n_init=1)

result_list = model.fit(vectors).labels_
```

```
print(result_list)
```

```
[14  2  4 ...  7 16  7]
```

```
order_centroids = model.cluster_centers_.argsort()[:, ::-1]
terms = vectorizer.get_feature_names_out()
```

```
for i in range(number_of_clusters):
    print("Cluster %d:" % i),
    for ind in order_centroids[i, :10]:
        print(' %s' % terms[ind])
```

```
Cluster 0:
 fbi
 clinton
 batf
 koresh
 waco
 atf
 compound
 writes
 article
 people
Cluster 1:
 msg
 food
 sensitivity
 chinese
 superstition
 glutamate
 restaurant
 reaction
 effects
 foods
Cluster 2:
 apple
 simms
 mouse
 mac
 mhz
 centris
 simm
 lc
 speed
 duo
Cluster 3:
 window
 server
 manager
 application
 xterm
 windows
 motif
 expose
 gl
 widget
Cluster 4:
 modem
 use
 power
 need
 amp
 audio
 lines
 mac
 sound
 subject
Cluster 5:
 turkish
 armenian
 armenians
 armenia
 serdar
 argic
 turks
 turkey
 genocide
 serazumauucp
Cluster 6:
 drive
 scsi
 ide
 drives
 controller
 hard
 disk
 mb
 floppy
 mac
Cluster 7:
 subject
 lines
 organization
 writes
 university
 article
 nntppostinghost
 distribution
 just
 know
Cluster 8:
 people
 gun
 dont
 think
 writes
 morality
 article
 keith
 like
 just
Cluster 9:
 hockey
 team
 game
 nhl
 players
 play
 games
 leafs
 season
 teams
Cluster 10:
 church
 clayton
 cramer
 homosexual
 gay
 men
 marriage
 people
 homosexuality
 catholic
Cluster 11:
 god
 jesus
 bible
 christians
 christian
 faith
 people
 believe
 gods
 christ
Cluster 12:
 banks
 gordon
 gebcspittedu
 gebcadredslpittedu
 njxp
 chastity
 shameful
 skepticism
 intellect
 surrender
Cluster 13:
 israel
 israeli
 jews
 arab
 arabs
 lebanese
 israelis
 policy
 lebanon
 peace
Cluster 14:
 car
 space
 cars
 writes
 moon
 article
 just
 like
 bike
 nasa
Cluster 15:
 card
 video
 drivers
 monitor
 vga
 cards
 diamond
 driver
 windows
 bus
Cluster 16:
 email
 thanks
 university
 subject
 lines
 organization
 graphics
 software
 help
 looking
Cluster 17:
 windows
 dos
 file
 files
 program
 nt
 use
 version
 lines
 subject
Cluster 18:
 baseball
 game
 year
 team
 games
 players
 runs
 braves
 pitching
 season
Cluster 19:
 key
 clipper
 encryption
 chip
 keys
 government
 escrow
 crypto
 algorithm
 nsa
```

```
from sklearn.metrics import silhouette_score
#计算轮廓系数
# result_list = kmeans.fit_predict(data)
# 将原始的数据data和聚类结果result_list
# 传入对应的函数计算出该结果下的轮廓系数
score = silhouette_score(vectors, result_list)
print("轮廓系数为",score)
```

```
轮廓系数为 0.006492968483480617
```

可视化

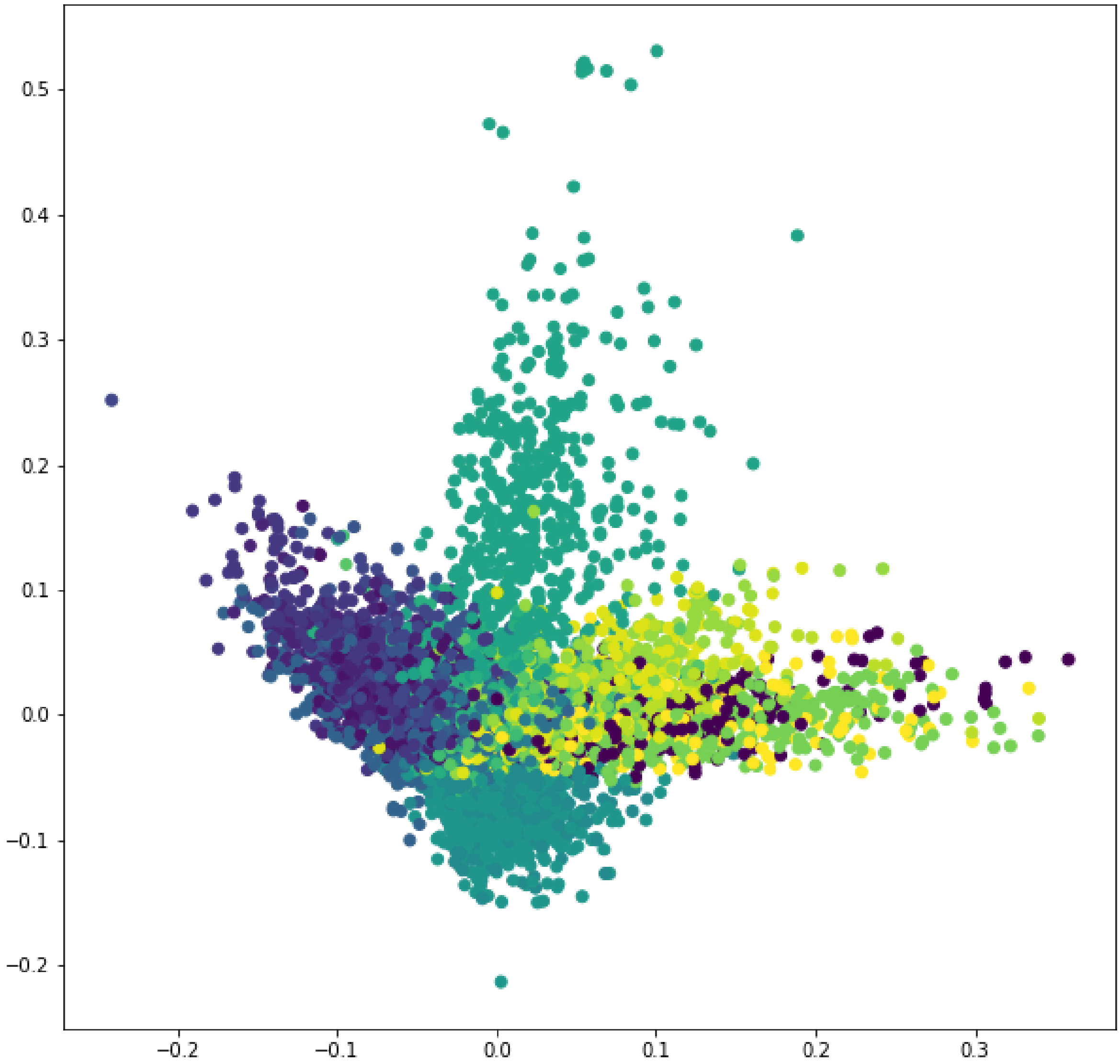
使用PCA对数据进行降维至2D

```
from sklearn.decomposition import PCA

pca = PCA(n_components=2)#投影到两个维度
pca.fit(vectors.toarray())
print('-'*20 + 'Explained variance ratio' + '-'*20)
print(pca.explained_variance_ratio_)
print('-'*20 + 'Singular value' + '-'*20)
print(pca.singular_values_)
```

```
--------------------Explained variance ratio--------------------
[0.00393553 0.00282597]
--------------------Singular value--------------------
[6.63651972 5.62370244]
```

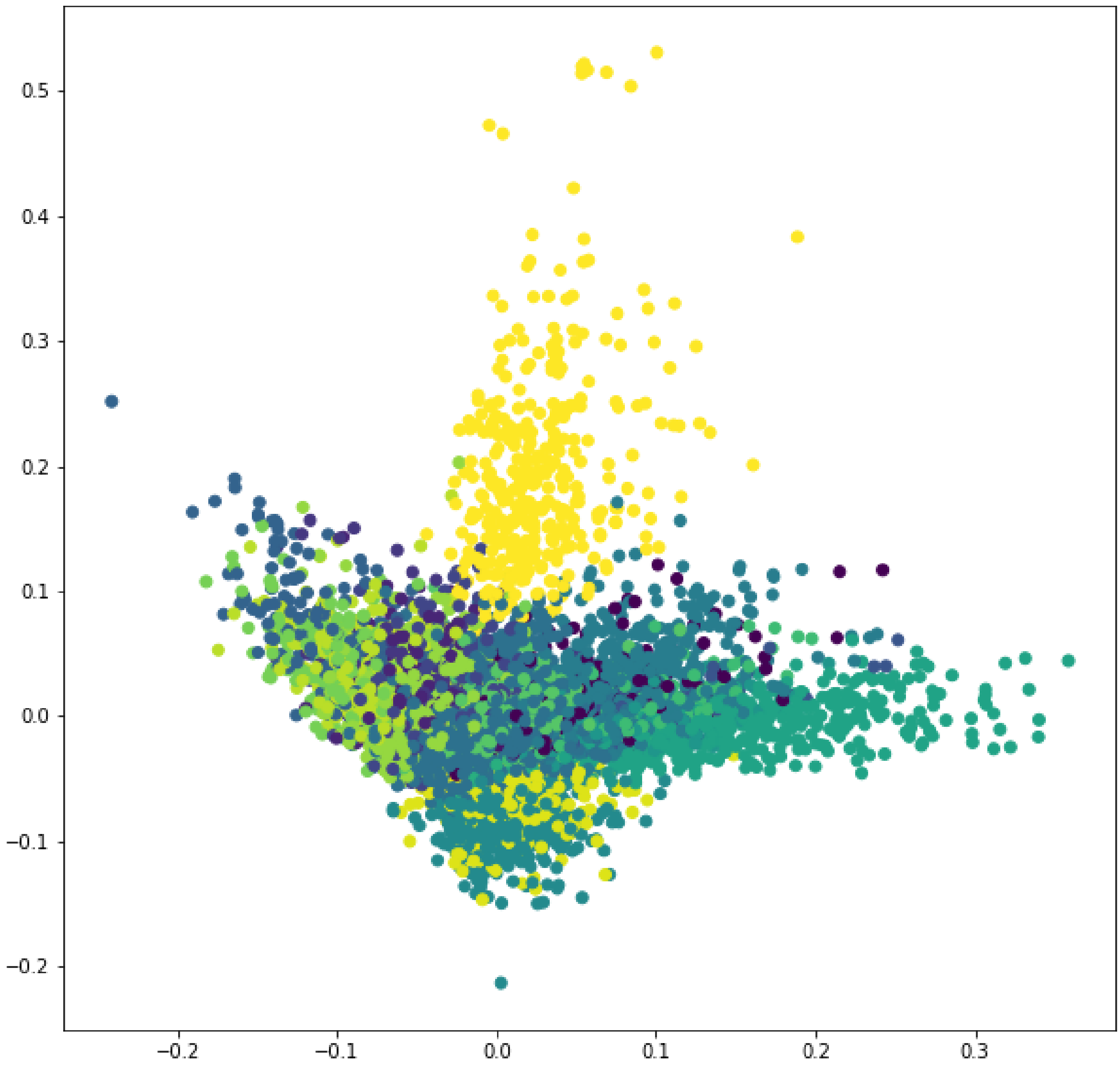
```
import matplotlib.cm as cm
import numpy as np
import matplotlib.pyplot as plt

PCA_view = pca.transform(vectors.toarray())
print(PCA_view.shape)

plt.figure(figsize=(10,10))
plt.scatter(PCA_view[:,0], PCA_view[:,1], c=train_data.target)
plt.show()
```

```
(11314, 2)
```



上面的图是用PCA对原数据进行降维得到的，下面对K聚类结果进行可视化

```
plt.figure(figsize=(10,10))
plt.scatter(PCA_view[:,0], PCA_view[:,1], c=result_list)
plt.show()
```



发现两者形状和颜色分布非常相似，可以看出聚类效果良好