实验四：NLP 在金融场景下的应用实验报告

数据获取

关键代码

```
ts.set_token("a6dae538a760f0b9e39432c1bff5e50a1c462a1a087e994dae18fa04")
pro = ts.pro_api()

# 找到出现频数大于80的6个industry
feature=['软件服务','元器件','电气设备','化工原料','专用机械','通信设备']

# 获取数据
data1 = pro.stock_basic(exchange='', list_status='L', fields='ts_code,industry')
data2 = pro.stock_company(exchange='SZSE', fields='ts_code,business_scope')

# 合并数据
data=pandas.merge(data1,data2)

# 处理NaN
data=data.dropna()

# 去除不考虑分类的industry，并将industry信息对应为数字0，1，2，3，4，5
data=data[data['industry'].isin(feature)]
for i in range(len(feature)):
    data.industry[data['industry']==feature[i]]=i

scope=data['business_scope']
industry=data['industry']

# 获取stop words信息
with open('stop_words.txt', 'r', encoding='utf8') as f:
    file = f.read().split('\n')
stop_words=set(file)
```

其中stop words信息为中文停用词表，下载地址https://github.com/goto456/stopwords

运行结果

输出读到的数据查看正确

	ts_code	industry	business_scope
15	000020.SZ	1	生产经营各种彩色电视机、液晶显示器、液晶显示屏(在分支机构生产经营)、收录机、音响设备、电子...
25	000032.SZ	1	研发、生产、销售通信设备、交通通讯设备(生产场地营业执照另行办理)、计算机及软件、办公自动化...
34	000045.SZ	1	生产、经营偏光片等光学膜产品；酒店、物业租赁与经营管理；生产、加工纺织品、针织品、服装、装饰...
37	000049.SZ	2	无汞碱锰电池、一次锂电池、锌空气电池、镍氢电池、锂聚合物电池、燃料电池及其他种类电池、电池材...
38	000050.SZ	1	从事显示器件及相关的材料、设备、产品的设计、制造、销售；提供相关技术开发、技术咨询、技术服务...
...	...	...	...
2235	300846.SZ	0	技术开发、技术推广、技术服务、技术咨询；计算机系统服务；数据处理；基础软件服务；应用软件服务...
2236	300847.SZ	3	光电材料及关联产品的研发、生产、销售，有机光导鼓、墨粉、鼓粉盒及零部件的研发、生产、销售；复...
2237	300848.SZ	3	弹性体、聚酯多元醇、改性异氰酸酯、聚氨酯、改性聚氨酯树脂、塑料制品的研发、制造、销售；不干胶...
2240	300851.SZ	0	技术开发、技术推广、技术咨询、技术转让、技术服务；计算机系统服务；应用软件服务；仪器仪表维修...
2241	300852.SZ	1	研发、制造、销售：双面、多层、刚挠结合、金属基、高频、HDI、元件嵌入式等电路板；电路板设计...

文本数值化

关键代码

```
# 检查是否包含汉语
def check(str):
    mark=False
    for i in str:
        if u'\u4e00' <= i <= u'\u9fff':
            mark=True
    return mark

# 去除stop word
def remove_stop_word(words):
    temp = list(words)
    words_list=[]
    for i in temp:
        if i in stop_words:
            continue
        elif check(i)==False:
            continue
        elif i.isdigit():
            continue
        words_list.append(i)
    return words_list

word_list=[]

# 注释部分为jieba提取关键词并拼接
for str in scope:
    temp1=remove_stop_word(jieba.cut(str))
    # temp2=remove_stop_word(jieba.analyse.extract_tags(str))
    word_list.append(" ".join(i for i in temp1))#+" "+" ".join((j for j in
temp2)))
v = TfidfVectorizer()
vector=v.fit_transform(word_list)
```

在这里我没有采用jieba提取关键词的方式，因为

1.输出了jieba提取的关键词，查看之后认为提取效果并不好

2.在完成后面部分之后，对比了jieba提取关键词拼接在分词字符串后面的方法，发现对最终得到的关键词影响不大，对最终的分类正确率影响也不大

遇到的问题

1.对pandas，list，scipy matrix，generator等数据类型不够了解，在各个接口处都遇到问题

2.去除stop word处即使判断了空格，也无法真正去除空格

解决方法

1.由于各个库的说明文件虽然说明了参数，但是很少说明数据类型，采用了搜索他人的代码，输出数据类型查看各个接口要求的数据类型的方法查看要求的数据类型，并搜索各个类型间的转化方法

2.可能的原因是中文空白字符有多种编码。但是搁置这个问题进行到下一步之后发现空白字符的存在并不影响关键字提取。

运行结果

提取的关键字为：

```
'ic卡', 'ip电话', 'u盘', 'x射线', '一二次', '一体化', '一体机', '一次', '一次性', '一甲胺', '一类', '一铵', '丁二烯', '丁基', '丁烯', '丁烷', '丁腈', '丁苯', '丁酮', '丁酯', '丁酸', '丁醇',
......
'中药', '中药材', '中药饮片', '中试', '中转', '中间', '中间体', '临床', '为主', '为准', '主体', '主板', '主管部门', '主营', '乌洛托品', '乐清市', '乘客', '乘用', '乘用车',
......
'互感器', '互联', '互联网', '互联网服务', '互联网络', '互补', '互连', '五金', '五金交电', '五金产品', '五金件', '五金制品', '五金工具', '五金模具', '五金配件', '井下', '井口', '井盖', '亚硫酸', '亚硫酸钠', '亚磷酸', '亚锡', '交互', '交付', '交换', '交换机', '交易', '交流', '交流活动', '交电', '交直流', '交通', '交通管理', '交通系统', '交通设备', '产业', '产业化', '产业园', '产业政策', '产业链', '产品', '产品代理', '产品开发', '产品设计', '产品质量', '产品销售', '产地', '产盐', '产销', '亮化', '人事代理', '人体', '人力', '人力资源', '人员', '人工', '人工智能', '人才', '人才中介', '人造', '人造纤维', '人造草坪', '人造革', '人防', '仅限', '介护', '介质',
......
'供用电', '供电', '供电系统', '供配电', '供销业', '依托', '依法', '便携式', '保健食品', '保养', '保安', '保密', '保护', '保护区', '保护膜', '保洁', '保温', '保温材料', '保管', '保证', '保险', '保险业务', '保险代理', '保险柜', '保险箱', '保鲜', '信号', '信号灯', '信封', '信息', '信息产业', '信息加工', '信息化', '信息处理', '信息安全', '信息工程', '信息技术', '信息源', '信息系统', '信息网', '信息网络', '信托', '信用',
......
'光伏', '光卤石', '光学', '光学仪器', '光学材料', '光学玻璃', '光学薄膜', '光学镜片', '光导', '光感控', '光敏电阻', '光机电', '光污染', '光源', '光热', '光电', '光电子', '光纤', '光纤通信', '光缆', '光通信',
......
'取暖器', '受托', '受理', '受电弓', '变化', '变压器', '变换器', '变更', '变流器', '变电', '变电站', '变送器', '变配电', '变频', '变频器', '口罩', '口腔', '口腔科', '另办', '另发', '另择', '另有', '另行', '另设', '另附', '可编程', '可行性研究', '可见光', '可降解', '可靠性', '号文', '各类', '各项', '合作', '合同', '合同期', '合成', '合成树脂', '合成橡胶', '合成氨', '合成洗涤剂', '合法', '合资经营', '合路器', '合金', '合金材料',
......
'陶瓷', '陶瓷制品', '陶瓷器', '陶瓷材料', '隐身', '隔离', '隔离器', '隔膜', '隧道', '集中', '集中式', '集成', '集成电路', '集成系统', '集抄', '集群', '集资', '雕塑', '零件', '零售', '零售业', '零组件', '零部件', '零配件', '雷电', '雷达', '需凭', '需经', '需要', '露天开采', '静止', '非专业', '非专利', '非学历', '非标', '非标准', '非法', '非金属', '非金属材料', '非金属矿', '面向', '面料', '面板',
......
```

在这里截取一部分查看。粗略看来可以发现提取基本成功，但是存在一些问题：

1.一部分同样作用的物品名称大量存在，如大量存在化学药品名，这些关键词如果可以被归类为“化学药品”能够提高分类的正确率

2.存在一部分无意义的词语，如大量动词（”交流“，”另有“，”需要“，”依托“），”中间“，”为主“，以及地名”乐清市“等，这些词语事实上对于分类没有意义，应该想办法除去。

对数值化后的稀疏矩阵的展示：

```
  (0, 2319) 0.11173609011648594
  (0, 987)  0.20823471898557883
  (0, 2189) 0.20823471898557883
  (0, 1371) 0.05775567064343222
  (0, 1459) 0.13035243337098926
  (0, 2287) 0.20823471898557883
  (0, 1048) 0.12132666918474704
  (0, 3108) 0.14914897931271393
  (0, 2546) 0.16614118873144268
  (0, 1943) 0.11237535717395687
  (0, 871)  0.20823471898557883
  (0, 193)  0.17019574443978203
  (0, 1964) 0.20823471898557883
  (0, 660)  0.16256481713686963
  (0, 3317) 0.20823471898557883
  (0, 2098) 0.1959231506247583
  (0, 2802) 0.12404765847730652
```

基于数值化文本向量进行分类器学习

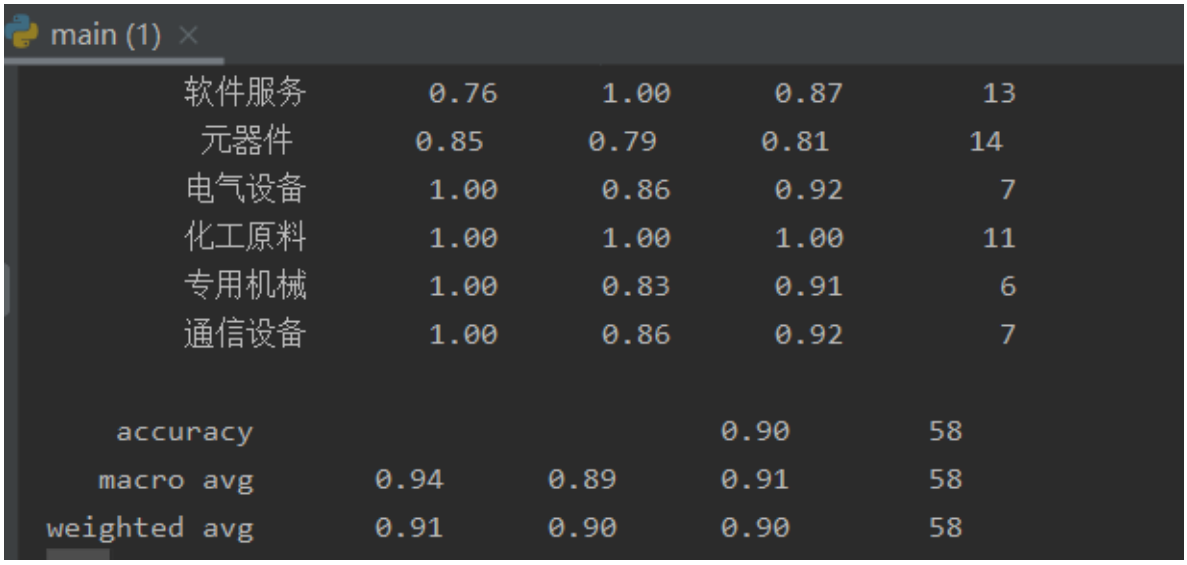
关键代码

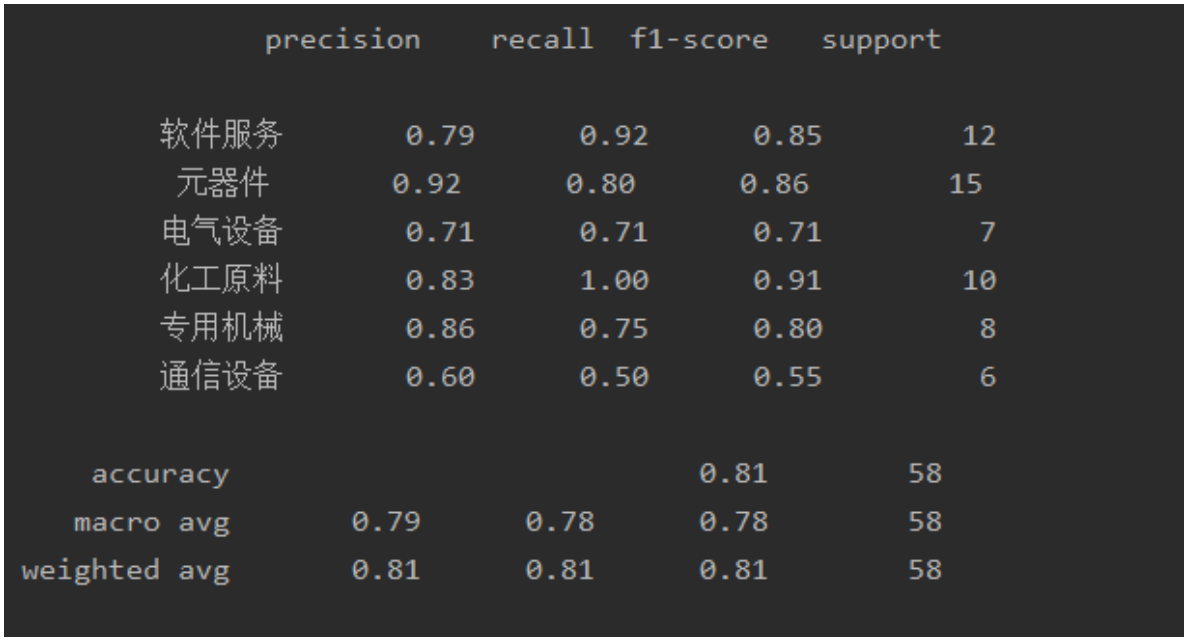
```
# 划分训练集与测试集
kf = KFold(n_splits=12,shuffle=True)
accuracy=0
for train_index, test_index in kf.split(target):
    X_train, X_test = vector[train_index], vector[test_index]
    y_train, y_test = [],[]
    for i in train_index:
        y_train.append(target[i])
    for i in test_index:
        y_test.append(target[i])

    # 这里采用了多种方式进行测试，其中能够得到较好的结果的为朴素的贝叶斯分类器与SVM分类器
    # 其余尝试因为效果过差不再放在代码里
    # clf = MultinomialNB(alpha=0.01)
    # clf.fit(X_train, y_train)
    # pred = clf.predict(X_test)
    svc = svm.SVC(kernel='sigmoid')
    svc.fit(X_train, y_train)
    pred = svc.predict(X_test)
    count = 0
    for l, r in zip(pred, y_test):
        if l == r:
            count += 1
    if(count/len(y_test)>accuracy):
        accuracy=count/len(y_test)
    # 输出测试结果
    print(classification_report(y_test, pred, target_names=feature))
print("max accuracy:",accuracy)
```

运行结果

最佳运行结果：

使用SVM分类器，1：11的训练数据和测试数据，启用shuffle。在最好的情况下能够达到接近90%的正确率，平均情况下能够达到75%左右的正确率。下图列数据分别为precision　recall　f1-score　support





其他运行结果：

分类方法	最佳正确率	平均正确率
朴素贝叶斯分类器	0.8305084745762712	0.7125462692382624
KNN分类器	0.6779661016949152	0.5736411455289304
Logistic Regression	0.864406779661017	0.6979836353009937
决策树	0.5517241379310345	0.454899668809663

除此之外，测试了SVM不同的核的效果，发现linear和sigmoid可以得到差不多好的结果，而其他核分类效果都很差。

观察训练集和测试集比例对结果的影响，下面两图分别为KF参数n_splits=x时对应的平局正确率和最高正确率。可以看到，在n_splits较小时，会因为训练集过小而导致正确率低；随着训练集比例增高，正确率有一定上升；但上升并不是无限逼近100%的，由于整个数据集有限，以及关键词提取能力和分类器能力有限，最终正确率在70%和83%上下稳定。