

湖南大学实训报告

课程名称: 计算与人工智能概论 实验类型: 团队实训
实验项目名称: 中国 500 强公司信息爬取
学生姓名: 周圣凯 班级: 2402 学号: 202410040215
同组姓名: 班级: 学号:

一、实训目的

1. 巩固计算与人工智能概念课程所学基础知识
2. 拓展训练学生计算和 AI 思维能力
3. 加强团队分工与合作, 培养学生团队协作能力

二、实训内容和团队分工

//1 任务描述: 在下列代码框中获取中国 500 强公司信息概况,具体要求如下:

1. 获取所有页面中的公司名、法定代表人、注册时间以及证券类别, 将获得到的内容保存到 csv 文件;
2. 获取数据之后, 分析 500 强公司的证券占比;
3. 绘制饼图展示分析的结果。

部分数据展示如下:

新华人寿保险股份有限公司, 万峰, 1996-09-28, A 股
中国民生银行股份有限公司, 洪崎, 1996-02-07, A 股
兴业银行股份有限公司, 高建平, 1988-08-22, A 股
上海浦东发展银行股份有限公司, 吉晓辉, 1992-10-19, A 股
苏宁云商集团股份有限公司, 张近东, 2001-06-29, A 股
中国太平保险集团有限责任公司, 王滨, 1982-02-13, 港股
华能国际电力股份有限公司, 曹培玺, 1994-06-30, A 股

本实训是一个中国 500 强公司信息爬取的案例, 主要是通过 https://top.chinaz.com/gongsitop/index_500top.html 该网站获取前 500 强公司的公司名、法定代表人、注册时间以及证券类别, 将获得到的内容保存至 csv 文件。

三、实训环境

如代码所示

四、实训方法和步骤

首先, 我们进入该网站: https://top.chinaz.com/gongsitop/index_500top.html, 可以看到前 500 强公司的大致信息。我们需要获取的是公司名、法定代表人、注册时间以及证券类别。既然我们已经了解了我们要爬取的内容, 那接下来我们开始分析网页结构。首先需要了解我们需要的数据所在的位置, 所以我们需要进入开发者模式 (也可以鼠标右击, 选择检查) 查看数据的来源, 这里我使用的是 Chrome 浏览器。我们可用通过全局搜索的方式来获取关键信息所在的位置。

找到了数据所在的位置之后, 我们可以发现数据就在当前页面链接的返回结果中, 接下来我们观察下一页的 url 是怎么变化的。

除了第一页是 index_500top 以外, 其余页面的 url 都是 index_500top 加页数的形式。现在我们只需要获取到每一页的内容然后对其进行解析即可。

首先我们需要导入 requests 库

```
import requests
```

若是本地没有安装 requests 的同学可以通过“pip install requests”来安装

第一步: 对目标 url 进行请求, 开发者工具中可以看到请求的具体信息, 获取请求返回的内容。

```

# 请求的 url
url = "https://top.chinaz.com/gongsitop/index_500top.html"
# 设置请求头信息
headers = {
    "user-agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/95.0.4638.69 Safari/537.36"
}
# 使用 requests 模块发起 GET 请求
response = requests.get(url, headers=headers)
# 获取请求的返回结果
html = response.text
html

```

第二步：解析返回内容。从上面的返回结果可以看出返回的是 HTML 类型的数据，如果我们需要获取到关键信息，就需要使用到 XPATH 或者正则表达式等解析工具，这里我们使用的是 正则表达式。正则表达式是一种非常高性能的清洗方式，Python 中内置了 re 模块来实现正则表达式。我们需要导入这个库。

In [3]:

```

# 导入 re 模块

# 使用 findall 函数来获取数据
# 公司名
company = re.findall('<a.*?target="_blank">(.*?)</a></h3>', html)
# 法定代表人
person = re.findall('法定代表人： </span>(.*?)</p>', html)
# 注册时间
signDate = re.findall('注册时间： </span>(.*?)</p>', html)
# 证券类别
category = re.findall('证券类别： </span>(.*?)</p>', html)
category
pageOne = list(zip(company, person, signDate, category))
pageOne
# 存储内容
message = []
# 总共 16 个页面的数据
for page in range(16):
    # 组装 url
    if page == 0:
        url = "https://top.chinaz.com/gongsitop/index_500top.html"
    else:
        url = "https://top.chinaz.com/gongsitop/index_500top_{}.html".format(page + 1)
    # 使用 requests 模块发起 GET 请求
    response = requests.get(url, headers=headers)
    html = response.text

```

```

# 使用 findall 函数来获取数据
# 公司名
company = re.findall('<a.*?target="_blank">(.*?)</a></h3>', html)
# 法定代表人
person = re.findall('法定代表人: </span>(.*?)</p>', html)
# 注册时间
signDate = re.findall('注册时间: </span>(.*?)</p>', html)
# 证券类别
category = re.findall('证券类别: </span>(.*?)</p>', html)
pageOne = list(zip(company, person, signDate, category))
# 合并列表
message.extend(pageOne)
message

```

第三步：保存内容。我们可以将 message 中的数据保存到数据库或者文件中，这里我们选择保存到 csv 文件。

导入 python 中的内置模块 csv

```
import csv
```

```
with open("content.csv", "w") as f:
```

```
    w = csv.writer(f)
```

```
    w.writerows(message)
```

```
!cat content.csv
```

数据可视化

```
import pandas as pd
```

读取数据

```
df = pd.read_csv("content.csv", names=["company", "person", "signDate", "category"])
```

```
df.head()
```

```
df.info()
```

根据证券类型进行分组

```
df1 = df.groupby("category").count()["company"]
```

```
df1
```

在 jupyter 中直接展示图像

```
%matplotlib inline
```

```
import matplotlib.pyplot as plt
```

用黑体显示中文

```
plt.rcParams['font.sans-serif'] = ['SimHei']
```

每个扇形的标签

```
labels = df1.index
```

每个扇形的占比

```
sizes = df1.values
```

```
fig1, ax1 = plt.subplots()
```

```
# 绘制饼图
ax1.pie(sizes, labels=labels, autopct='%d%%',radius=2,textprops={'fontsize': 20},
        shadow=False, startangle=90)
ax1.axis()

plt.show()
```

五、实训结果和分析

如代码所示

六、讨论与心得

1. 课堂上学到的正则表达式、HTTP 请求等知识在这个项目中得到了实际应用，让我深刻理解了理论如何转化为实践。
2. **反爬机制**：最初请求时遇到 403 错误，通过添加 **User-Agent** 请求头解决，意识到网站对爬虫有基础防护。
3. 同步请求（逐页爬取）速度较慢，未来可尝试 **aiohttp** 异步请求提升效率。
4. 仅统计证券类别占比略显单薄，若能结合注册时间分析企业成立年代分布，或关联行业类型（需额外爬取），结论会更丰富。
5. 某些公司证券类别为“-”（缺失值），直接统计可能影响准确性，需考虑填充或过滤。
6. 最初饼图的标签重叠，通过调整 **figsize** 和 **textprops** 优化显示效果。
7. 意识到颜色对比度的重要性：为突出主要类别（如 A 股），手动指定颜色或使用渐变色会更直观。
8. 正则表达式编译（**re.compile**）未使用，但在大规模数据中预编译模式能减少重复解析开销。