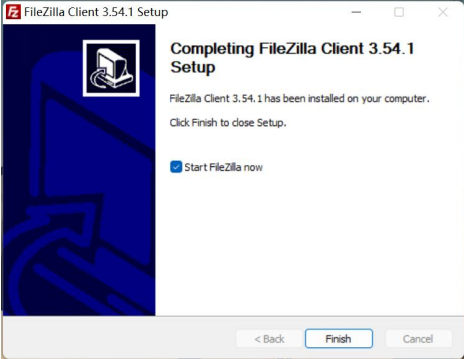
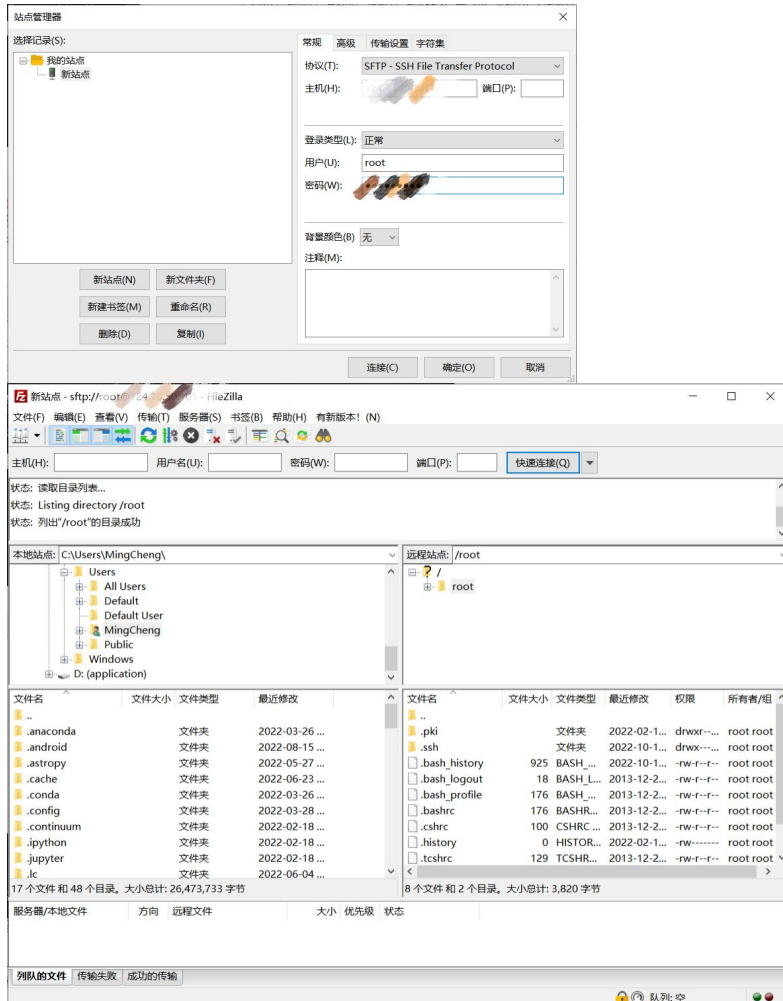


# 《大数据导论》实验报告

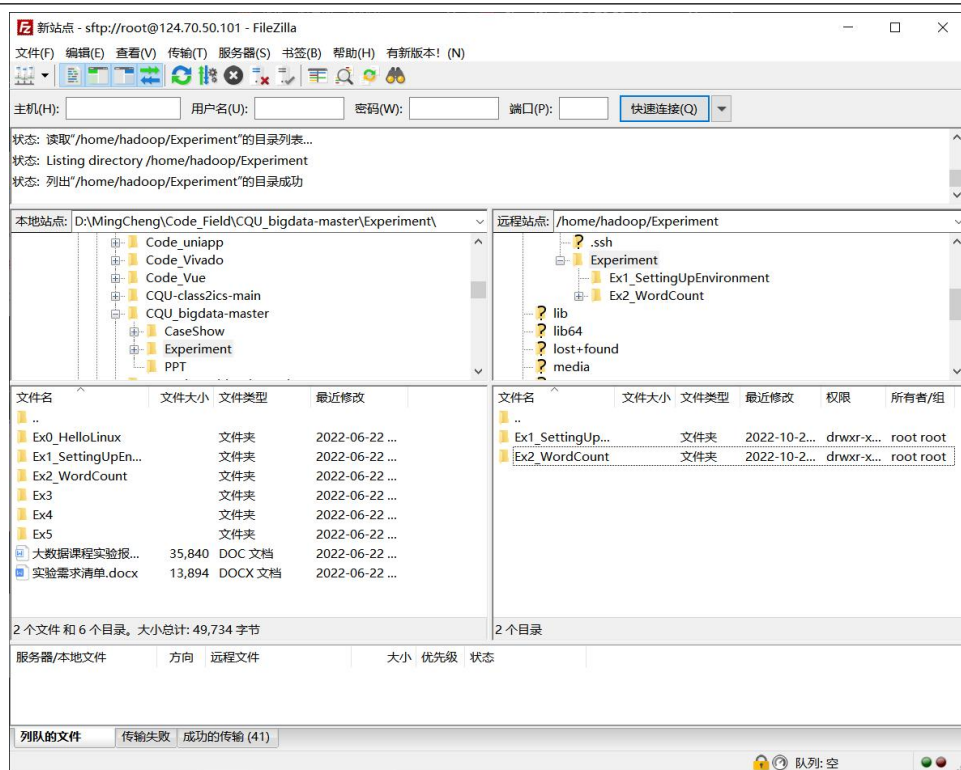
实验题目	词频统计与可视化
<p><b>一、实验目的</b></p> <p>通过本次实验，我们应当：</p> <ul style="list-style-type: none"><li>● 熟悉 hadoop+ Spark 下编程环境</li><li>● 掌握基于 Spark 的基本 MAP REDUCE 操作</li><li>● 掌握基本大数据可视化工具</li><li>● 独立完成本次青年群体择偶观分析实验</li><li>● 【新增】远程开发相关知识</li></ul>	
<p><b>二、实验项目内容</b></p> <p>本次实验需小组内分工合作完成两个任务：</p> <p>1、WordCount 词频统计</p> <p>你将会使用到 jieba 分词&amp;基于 pySpark 的基本 MAP REDUCE 操作进行词频统计，在指定数据集上大数据分析青年群体择偶观倾向。</p> <p>2、大数据可视化</p> <p>你将使用 Echarts &amp; WordCloud 两个可视化库来进行大数据可视化，小组独立完成核心代码编写、测试。</p>	
<p><b>三、实验过程或算法（源程序）</b></p> <p>本次实验我们采用了<b>云服务器</b>进行环境搭建，同时在本次实验基础上完成了<b>扩展点 1 和 3</b>。（使用分布式完成实验；新增柱状图、折线图等可视化）</p> <p><b>1. 上传文件</b></p> <p>开始实验前， 先将代码及相关资源上传到服务器, 本次实验我们利用 FTP 软件将本地（Windows）文件上传到服务器(Linux)。</p> <p>（1）下载软件-Filezilla</p> <p>下载安装即可，过程较为简单，在此不再赘述。</p>  <p>（2）连接服务器</p>	

依次点击：文件-->站点管理器-->新站点



(3) 上传文件

左侧为本地文件，右侧为服务器文件目录（默认为 /home/hadoop）



上传完毕后，可在服务器上查看文件：

```
[hadoop@master Experiment]$ cd /home/hadoop/Experiment/
[hadoop@master Experiment]$ ll
total 8
drwxr-xr-x 2 root root 4096 Oct 24 14:51 Ex1_SettingUpEnvironment
drwxr-xr-x 3 root root 4096 Oct 24 14:52 Ex2_WordCount
[hadoop@master Experiment]$
```

## 2. 远程开发

通过 SSH 工具和 FTP 工具，我们可以很方便在终端命令行操作远程服务器，上传/下载数据集和代码等。但是，因为代码运行环境在远程服务器上，当进行代码编写和调试时，我们只能在 Filezilla 上传数据集等资源，Xshell 连接服务器命令行操作，使用 vim 编写代码，关键地方 print，编写好后命令行下执行 python code.py，观察代码 print 输出 & 调试，效率十分低下。

对于实际开发来说，我们还是更希望能使用本地 IDE 去调试远程代码，而不是使用 vim 或者直接远程服务器安装 IDE。实现远程开发的步骤如下：

### 2.1 检查 SSH 服务

华为云等云厂商一般还会默认安装 ssh-server 服务，我们需要在此查看是否安装了 ssh-server 服务。

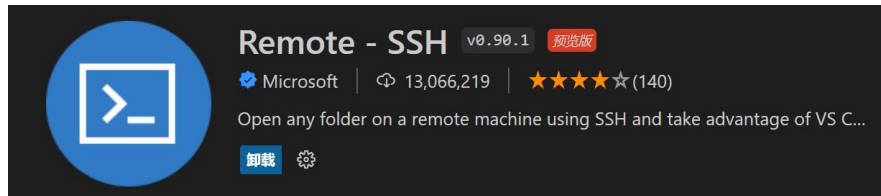
```
[root@master Experiment]# service sshd status
Redirecting to /bin/systemctl status sshd.service
● sshd.service - OpenSSH server daemon
```

可以看到已经启动 ssh 服务。

### 2.2 VSCode 远程开发实践

#### (1) 安装 Remote-SSH 插件

左侧 Extension 图标 ---> 输入 Remote-SSH ---> 安装即可, 需要注意的是远程和本地都进行安装。过程较简单, 在此不进行赘述。

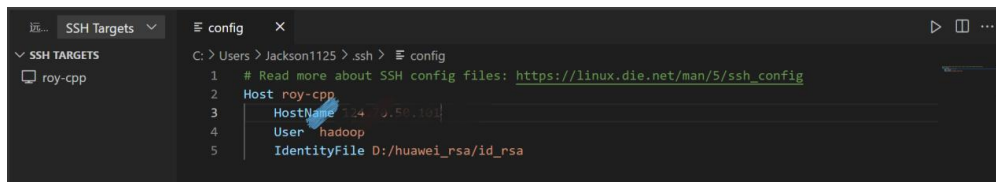


## (2) 配置 Remote-SSH 插件

选择 .ssh/config 文件进行配置, 打开文件后, 需要设置的字段如下:

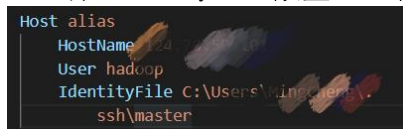
- Host: 自定义即可
- HostName: 云主机公网 IP
- User: 登陆的用户, 选择 root
- IdentityFile: 免密登录私钥地址, 如果没有配置则每次远程登录需要输入密码

如下, 为我们小组的配置:



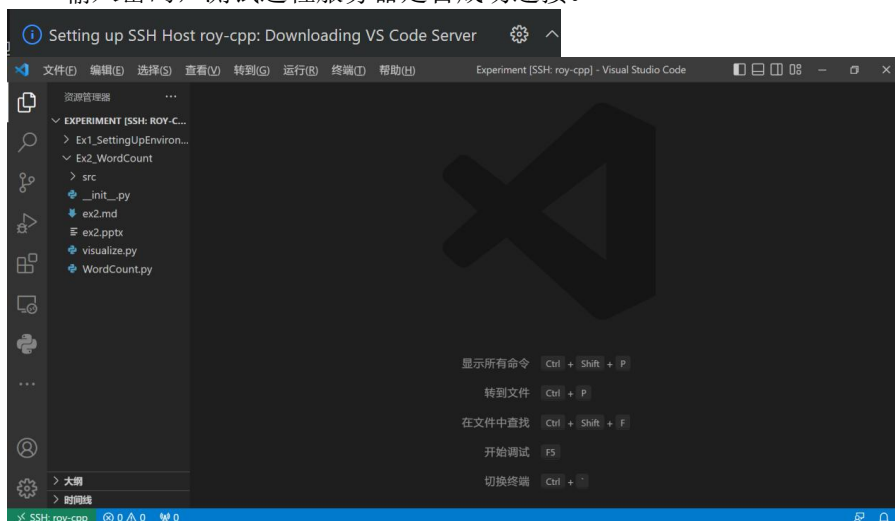
按照示例直接进行配置会报错 “too open”, 远程控制时无法修改文件。

将 IdentityFile 放置 .ssh 目录下可解决, 如下是成功运行的配置:



## (3) 登录测试

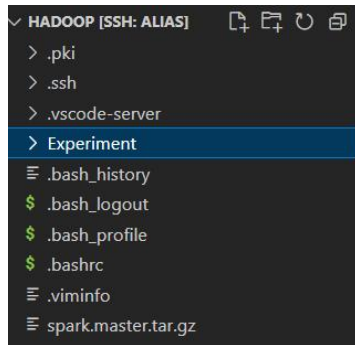
输入密码, 测试远程服务器是否成功连接。



可以看到远程服务器已经成功连接。

## (4) 远程开发实践

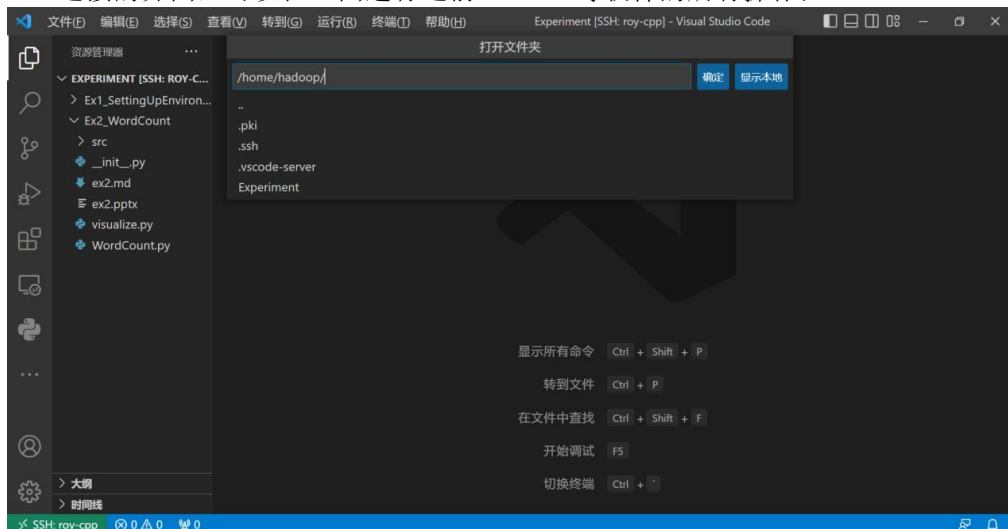
a、打开远程文件：File-->Open Folder-->指定文件夹路径，类似打开本地文件：



但是我们遇到了在 Experiment 目录下报错“无修改权限”的问题，对此，我们需要设置用户 hadoop 为目录 Experiment 拥有者，即可解决，代码如下：

```
1 | sudo chown -R hadoop:hadoop Experiment # 设置用户hadoop为目录Experiment拥有者
```

b、使用终端：在 VSCode 下方 Terminal 一栏可进行命令行操作，可认为是一个 ssh 连接的界面，可以在上面进行之前 Xshell 等软件的所有操作：



连接上后就和我们本地使用 VSCode 没有什么区别，可随意打开文件夹。

## 2.3 安装相关库

- 更新 pip，否则可能出现安装失败。

```
1 | sudo pip3 install --upgrade pip
```

- 修改 urlgrabber-ext-down, 第一行改为 python2/2.7

```
1 | sudo vim /usr/libexec/urlgrabber-ext-down
```

- 安装字体文件，需要 wqy-microhei.ttc 文件，但在 /usr/share/fonts 可能并不存在该文件。需要自行安装：

```
1 | sudo yum install wqy-microhei-fonts
```

- 安装 jieba

```
1 | sudo pip3 install jieba -i https://pypi.tuna.tsinghua.edu.cn/simple
```

- 安装 wordcloud

```
1 | sudo pip3 install wordcloud -i https://pypi.tuna.tsinghua.edu.cn/simple
```

- 安装 pyecharts ，需要指定安装 1.7.0 版本，否则下面代码 API 调用接口不正确。

```
1 | sudo pip3 install -i https://pypi.tuna.tsinghua.edu.cn/simple pyecharts==1.7.0
2 | sudo pip3 install snapshot-selenium
```

- 安装驱动：pyecharts 模块保存图片需要安装相应 chromedriver 和 google-chrome，并且版本号要一一对应。
- 安装 google-chrome，在此为避免版本安装出错，可在完成后进行版本检查。

```
1 | wget -O /etc/yum.repos.d/CentOS-Base.repo http://mirrors.aliyun.com/repo/Centos-7.repo
2 |
3 | curl https://intoli.com/install-google-chrome.sh | bash
4 | ldd /opt/google/chrome/chrome | grep "not found"
5 |
6 | google-chrome --no-sandbox --headless --disable-gpu --screenshot https://www.baidu.com/
```

查看 google-chrome 版本：

```
1 | google-chrome-stable --version
```

版本如下，安装正确：

```
● [hadoop@master spark]$ google-chrome-stable --version
Google Chrome 107.0.5304.110
```

- 安装 chromedriver，chromedriver 版本要和 google-chrome 对应，记录版本号后，去 <https://npm.taobao.org/mirrors/chromedriver/> 下载对应的驱动。

```
1 | sudo rm -f chromedriver_linux64.zip*
2 |
3 | wget https://registry.npmirror.com/-/binary/chromedriver/107.0.5304.62/chromedriver_linux64.zip
```

下载后，在下载的压缩文件所在的路径：

```
1 | # 2.unzip 解压
2 | # 解压出文件chromedriver
3 | unzip chromedriver_linux64.zip
4 |
5 | # 3.赋予777权限
6 | chmod 777 chromedriver
7 |
8 | # 4.mv 移动到/usr/bin/路径
9 | mv chromedriver /usr/bin/
```

## 2.4 设置日志级别

(1) 切换到 cof 目录

```
1 | cd /usr/local/spark/conf
```

(2) 设置配置文件

```
1 | cp log4j.properties.template log4j.properties
2 | vim log4j.properties
```

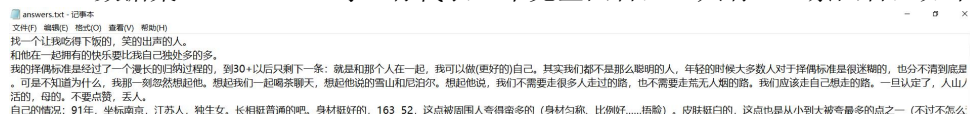


这里需要注意的是，修改 `log4j.rootCategory=WARN,console`

### 3. 实验流程

#### 3.1 数据集介绍

数据集 `answers.txt` 每一行代表一个完整回答，一共有 3W 条回答，如下图：



#### 3.2 词频统计 WordCount.py

##### (1) 完成编码

WordCount.py 有 3 个函数，它们的作用如下：

- `getStopWords`：读取 `stop_words.txt` 所有停用词，返回一个 python List
- `jiebaCut`：将所有答案合并，并进行分词，返回一个 python List
- `wordcount`：核心函数，利用 SparkRdd 完成词频统计

我们在此主要补全 `jiebaCut` 和 `wordcount` 代码，如下：

a、`jiebaCut`,使得 `str` 为所有答案拼接而成的字符串

```
1.     def jiebaCut(answers_filePath):
2.         """
3.         结巴分词
4.         :param answers_filePath: answers.txt 路径
5.         :return:
6.         """
7.         # 读取 answers.txt,answersRdd 每一个元素对应 answers.txt 每一行
8.         answersRdd = sc.textFile(answers_filePath)
9.
10.        # 利用 SpardRDD reduce()函数,合并所有回答
11.        str = answersRdd.reduce(lambda x, y: x + y )
12.
13.        # jieba 分词
14.        words_list = jieba.lcut(str)
15.        return words_list
```

b、`Wordcount`,使得 `resRdd` 包含所有词频统计结果，且降序排列。

```
1. def wordcount(isvisualize=False):
2.     """
3.     对所有答案进行
4.     :param visualize: 是否进行可视化
5.     :return: 将序排序结果 RDD
6.     """
7.     # 读取停用词表
8.     stopwords = getStopWords(SRCPATH + 'stop_words.txt')
9.
10.    # 结巴分词
```

```

11. words_list = jiebaCut("file://" + SRCPATH + "answers.txt")
12.
13. # 词频统计
14. wordsRdd = sc.parallelize(words_list)
15.
16. # wordcount: 去除停用词等同时对最后结果按词频进行排序
17. # 完成 SparkRDD 操作进行词频统计
18. # 提示: 你应该依次使用
19. # 1.filter 函数分别进行停用词过滤、去除长度<=1 的词汇
20. # 2.map 进行映射, 如['a','b','a'] --> [('a',1),('b',1),('a',1)]
21. # 3.reduceByKey 相同 key 进行合并 [('a',2),('b',1)]
22. # 4.sortBy 进行排序, 注意应该是降序排序
23. resRdd = wordsRdd.filter(lambda word: word not in stopwords) \
24.
25. .filter(lambda word: len(word)>1) \
26. .map(lambda word:(word,1)) \
27. .reduceByKey(lambda a, b: a + b) \
28. .sortBy(ascending=False, numPartitions=None, keyfunc=lambda x: x[1]) \

```

注: 运行时可能出现如下报错:

```

[hadoop@master Experiment]$ /usr/bin/python3 /home/hadoop/Experiment/Ex2_WordCount/WordCount.py
Traceback (most recent call last):
  File "/home/hadoop/Experiment/Ex2_WordCount/WordCount.py", line 7, in <module>
    from pyspark import SparkConf, SparkContext
  File "/usr/local/spark/python/pyspark/_init_.py", line 53, in <module>
    from pyspark.rdd import RDD, RDDBarrier
  File "/usr/local/spark/python/pyspark/rdd.py", line 34, in <module>
    from pyspark.java_gateway import local_connect_and_auth
  File "/usr/local/spark/python/pyspark/java_gateway.py", line 29, in <module>
    from py4j.java_gateway import java_import, JavaGateway, JsonObject, GatewayParameters
ModuleNotFoundError: No module named 'py4j'

```

这可能是 py4j 版本不对, 对此的解决办法是在命令行中输入 `cd`

`$SPARK_HOME/python/lib`, 这表示进入到 py4j 所在的文件目录下, 然后再输入 `ls` 看到自己 py4j 所对应的版本, 如下:

```

[hadoop@master lib]$ ls
py4j-0.10.9-src.zip  PY4J_LICENSE.txt  pyspark.zip

```

最后输入 `vim ~/.bashrc`, 将 py4j 的版本修改为自己虚拟机中对应的版本号即可。

## (2) 提交代码运行结果

切换到 spark 目录运行后可以得到如下结果:

```

• [hadoop@master spark]$ cd /usr/local/spark
• [hadoop@master spark]$ bin/spark-submit /home/hadoop/Experiment/Ex2_WordCount/WordCount.py
22/10/24 23:22:14 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using built-in
java classes where applicable
Building prefix dict from the default dictionary ...
Loading model from cache /tmp/jieba.cache
Loading model cost 0.574 seconds.
Prefix dict has been built successfully.
22/10/24 23:22:29 WARN TaskSetManager: Stage 1 contains a task of very large size (10038 KiB). The maximum recommended task size is 1000 KiB.
[('身高', 2627), ('家庭', 2018), ('父母', 2002), ('性格', 1882), ('男生', 1640), ('朋友', 1618), ('条件', 1568), ('学历', 1445), ('女生', 1380), ('感情', 1301)]

```

## 3.3 可视化 visualize.py

### (1) 完成编码



visualize.py 有 3 个函数，它们的作用如下：

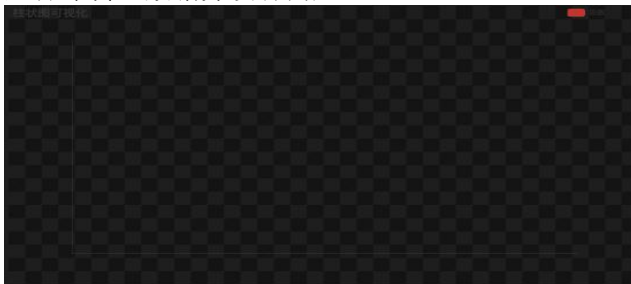
- rdd2dic：将 resRdd 转换为 python Dic，并截取指定长度 topK
- drawWorcCloud：进行词云可视化，同时保存结果
- drawPie：进行饼图可视化，同时保存结果

我们在此主要补全 rdd2dic 代码，如下：

**rdd2dic**,将 resRDD 转换为 python Dic，并截取指定长度

```
1. def rdd2dic(self, resRdd, topK):
2.     """
3.         将 RDD 转换为 Dic，并截取指定长度 topK
4.         :param resRdd: 词频统计降序排序结果 RDD
5.         :param topK: 截取的指定长度
6.         :return:
7.         """
8.         # 将 RDD 转换为 Dic
9.         resDic = resRdd.collectAsMap()
10.        # 截取字典前 K 个
11.        K = 0
12.        wordDicK = {}
13.        for key, value in resDic.items():
14.            if K >= topK:
15.                break
16.            wordDicK[key] = value
17.            K += 1
18.
19.        return wordDicK
```

实验过程中出现了一个很奇怪的问题，数据可视化的图中只有框架，无数据，但是打印传入数据并没有问题：



后来发现 key\_list value\_list 返回的是迭代器而不是列表，需要转换成 list 才能正确绘图。

## (2) 提交代码运行结果

将主函数可视化代码切换为 **true**，切换到 spark 目录运行后可以在目录 /home/hadoop/Experiment/Ex2\_WordCount/results 得到如下结果：



## (5) 运行代码

```
1 cd /usr/local/spark
2 bin/spark-submit --master spark://master:7077 --executor-memory 2G
/home/hadoop/lab/Ex2_WordCount/WordCount.py
```

## (6) 得到分布式结果

```
● [hadoop@master spark]$ bin/spark-submit --master spark://master:7077 --e
xecutor-memory 2G /home/hadoop/lab/Ex2_WordCount/WordCount.py
22/11/10 22:21:40 WARN NativeCodeLoader: Unable to load native-hadoop li
brary for your platform... using builtin-java classes where applicable
Building prefix dict from the default dictionary ...
Loading model from cache /tmp/jieba.cache
Loading model cost 0.565 seconds.
Prefix dict has been built successfully.
22/11/10 22:21:55 WARN TaskSetManager: Stage 1 contains a task of very l
arge size (1259 KiB). The maximum recommended task size is 1000 KiB.
[('身高', 2627), ('家庭', 2018), ('父母', 2002), ('性格', 1882), ('男生'
, 1640), ('朋友', 1618), ('条件', 1568), ('学历', 1445), ('女生', 1380)
, ('感情', 1301)]
```



打开 UI，可以看到，分布式成功部署并实现。

### ▼ Completed Applications (1)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
app-20221110222142-0000	ex2	8	2.0 GiB		2022/11/10 22:21:42	hadoop	FINISHED	3.7 min

## 4.2 新增柱状图、折线图可视化

### (1) 柱状图可视化实现

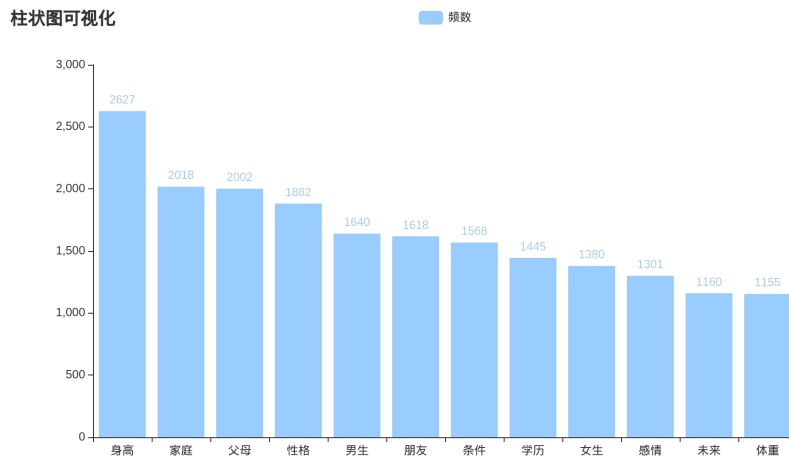
```
1. def drawBar(self, wordDic):
2.     """
3.     柱状图可视化
4.     :param wordDic: 词频统计字典
5.     :return:
6.     """
7.     key_list = wordDic.keys() # wordDic 所有 key 组成 list
8.     value_list = wordDic.values() # wordDic 所有 value 组成 list
9.     def bar_position() -> Bar:
10.         c = (
11.             Bar()
12.             .add_xaxis(list(key_list))
13.             .add_yaxis("频数",list(value_list))
14.             .set_global_opts
15.             (
```

```

16.         title_opts=opts.TitleOpts(title='柱状图可视化'),
17.         # 设置标题
18.         # legend_opts=opts.LegendOpts(pos_right="5%"),
19.         )
20.         .set_series_opts(itemstyle_opts=opts.ItemStyleOpts
    (color='#99ccff'))
21.         )
22.         return c
23.         # 保存结果
24.         if not os.path.exists(SAVAPATH):
25.             os.makedirs(SAVAPATH)
26.         make_snapshot(snapshot, bar_position().render(), SAVAPATH + '
    柱状图可视化.png')

```

运行结果如下：



## (2) 折线图可视化实现

```

1.     def drawLine(self, wordDic):
2.         """
3.         折线图可视化
4.         :param wordDic: 词频统计字典
5.         :return:
6.         """
7.         key_list = wordDic.keys() # wordDic 所有 key 组成 list
8.         value_list = wordDic.values() # wordDic 所有 value 组成 list
9.         def line_position() -> Line:
10.             c = (
11.                 Line()
12.                 .add_xaxis(list(key_list))
13.                 .add_yaxis("频数", list(value_list))
14.                 .set_global_opts
15.                 (

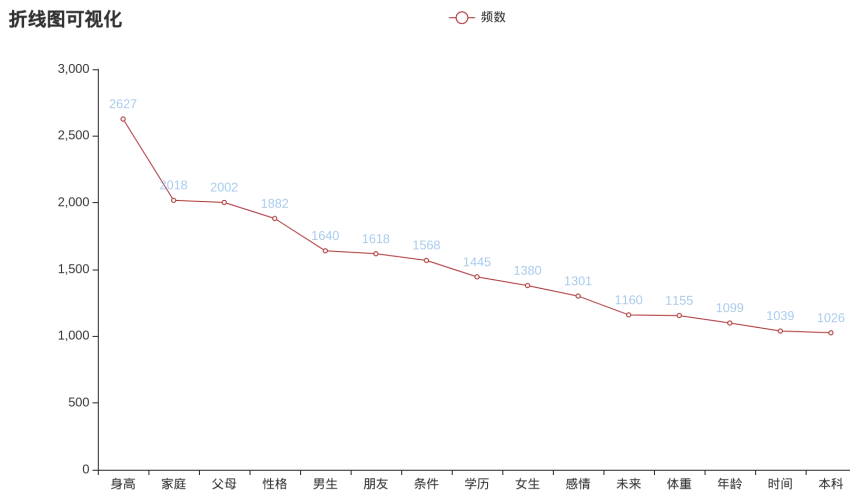
```

```

16.         title_opts=opts.TitleOpts(title='折线图可视化'),
17.         # 设置标题
18.         # legend_opts=opts.LegendOpts(pos_right="5%"),
19.         )
20.         .set_series_opts(label_opts=opts.LabelOpts(color='
#99ccff'))
21.     )
22.     return c
23.     # 保存结果
24.     if not os.path.exists(SAVAPATH):
25.         os.makedirs(SAVAPATH)
26.     make_snapshot(snapshot, line_position().render(), SAVAPATH + '
折线图可视化.png')

```

运行结果如下：



#### 四、实验结果及分析和（或）源程序调试过程

实验结果：

##### 1. 词频统计：

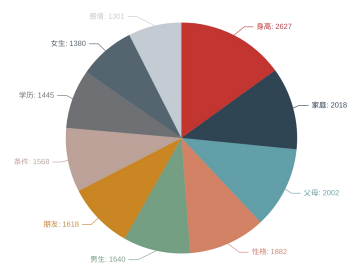
```

[hadoop@master spark]$ cd /usr/local/spark
[hadoop@master spark]$ bin/spark-submit /home/hadoop/Experiment/Ex2_WordCount/WordCount.py
22/10/24 23:22:14 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builti
n-java classes where applicable
Building prefix dict from the default dictionary ...
Loading model from cache /tmp/jieba.cache
Loading model cost 0.574 seconds.
Prefix dict has been built successfully.
22/10/24 23:22:29 WARN TaskSetManager: Stage 1 contains a task of very large size (10038 KiB). The maximum reco
mmended task size is 1000 KiB.
[('身高', 2627), ('家庭', 2018), ('父母', 2002), ('性格', 1882), ('男生', 1640), ('朋友', 1618), ('条件', 1568)
, ('学历', 1445), ('女生', 1380), ('感情', 1301)]

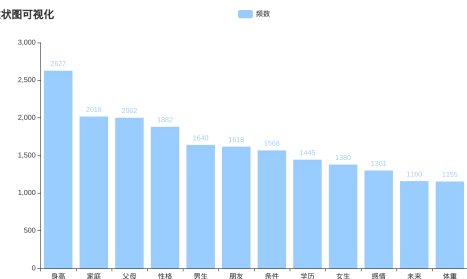
```

##### 2. 可视化：

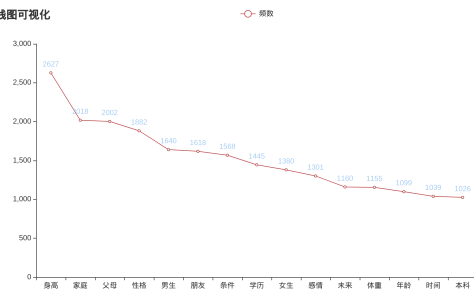
饼图可视化



柱状图可视化



折线图可视化



### 3. 分布式:

#### Completed Applications (1)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
app-20221110222142-0000	ex2	8	2.0 GiB		2022/11/10 22:21:42	hadoop	FINISHED	3.7 min

### 调试过程:

1. .ssh/config 文件的配置会报错 “too open”，远程控制时无法修改文件:

```
Host alias
  HostName 124.70.50.101
  User hadoop
  IdentityFile C:\Users\Mingcheng\.ssh\master
```

错误原因：路径配置未放在.ssh 下，需要将 IdentityFile 放置.ssh 目录下可解决。

2. 远程文件无修改权限:

```
1 sudo chown -R hadoop:hadoop Experiment # 设置用户hadoop为目录Experiment拥有者
```

错误原因：没有设置用户为该目录的拥有者，利用 `sudo chown -R hadoop:hadoop Experiment` 指令设置即可解决。

3. wordCount 函数运行时出现版本问题的报错:



```
[hadoop@master Experiment]$ /usr/bin/python3 /home/hadoop/Experiment/Ex2_WordCount/WordCount.py
Traceback (most recent call last):
  File "/home/hadoop/Experiment/Ex2_WordCount/WordCount.py", line 7, in <module>
    from pyspark import SparkConf, SparkContext
  File "/usr/local/spark/python/pyspark/_init__.py", line 53, in <module>
    from pyspark.rdd import RDD, RDDBarrier
  File "/usr/local/spark/python/pyspark/rdd.py", line 34, in <module>
    from pyspark.java_gateway import local_connect and auth
  File "/usr/local/spark/python/pyspark/java_gateway.py", line 29, in <module>
    from py4j.java_gateway import java_import, JavaGateway, JavaObject, GatewayParameters
ModuleNotFoundError: No module named 'py4j'
```

错误原因：py4j 版本与自己虚拟机中的版本号对应不上，可以通过在命令行中输入

`cd $SPARK_HOME/python/lib` 进入到 py4j 所在的文件目录下，然后再输入

`ls` 看到自己 py4j 所对应的版本，最后输入 `vim ~/.bashrc`，将 py4j 的版本修改为自己虚拟机中对应的版本号即可。

4. 进行可视化时，运行代码出现报错信息 “No such file or directory”：

```
if not os.path.exists(SAVAPATH):
    os.makedirs(SAVAPATH)
make_snapshot(snapshot, pie_position().render(), SAVAPATH + '饼图可视化.png')
```

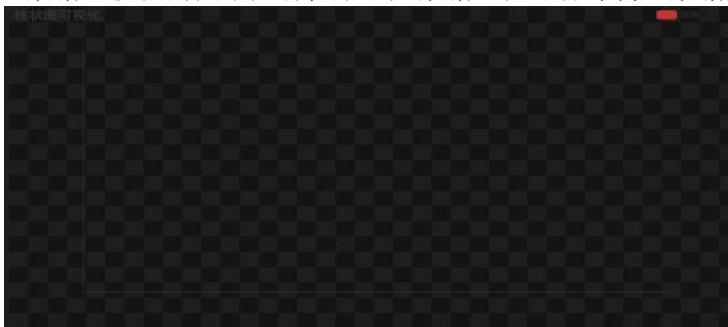
错误原因：保存文件前没有验证文件夹是否存在。

5. 内存问题导致报错：

```
File "/usr/local/lib/python3.6/site-packages/snapshot_selenium/snapshot.py", line 58, in get_chrome_driver
    return webdriver.Chrome(options=options)
File "/usr/local/lib/python3.6/site-packages/selenium/webdriver/chrome/webdriver.py", line 73, in __init__
    self.service.start()
File "/usr/local/lib/python3.6/site-packages/selenium/webdriver/common/service.py", line 76, in start
    stdin=PIPE)
File "/usr/lib64/python3.6/subprocess.py", line 729, in __init__
    restore_signals, start_new_session)
File "/usr/lib64/python3.6/subprocess.py", line 1295, in _execute_child
    restore_signals, start_new_session, preexec_fn)
OSError: [Errno 12] Cannot allocate memory
```

错误原因：服务器运行集群占用空间，将集群关闭后即可成功运行。

6. 数据可视化的图中只有框架，无数据，但是打印传入数据并没有问题：



错误原因：key\_list、value\_list 返回的是迭代器而不是列表，需要转换成 list 才能正确绘图。

**实验感悟：**

本次实验中，我们小组利用 hadoop 与 spark 完成了我们的第一个项目，虽然很欣喜，但是在过程中也遇到了很多困难。在最开始的文件配置中，我们按照操作指导

的示例进行配置，但是遇到了 `too open` 的报错，几番查找后发现应该放在自己对应的 `.ssh` 目录下；类似的问题还有远程文件无修改权限，都让我们困惑迷茫，最后几经探索和失败的尝试，终于意识到应该是目录拥有者的设置，好在最后成功解决。

在进行相关配置时，有一点值得注意的是版本的问题。虽然操作指导中已经提示要安装一致的版本，但是实际操作时还是会遇到这方面的报错，就比如本次实验运行 `wordCount` 时，就会出现版本问题报错。根据我们的经验总结，类似问题可以通过输入 `cd $SPARK_HOME/python/lib` 和 `ls` 先进行版本查看，最后将版本修改为与自己虚拟机相对应的版本即可。

此外，还有一些代码方面的报错。其中，在数据可视化时 `key_list`、`value_list` 返回的是**迭代器**而不是列表让我们 debug 了很久，最后真是恍然大悟。

上述 bug 都解决后，为了丰富数据的直观性与美观性，又在给定饼图和云图的基础上加入了柱状图和折线图。同时，为了更深入的理解 `hadoop+spark` 的分布式集群运算特点，我们还将原先的单机版改为了分布式运算，主机与从机间配置要求较高，但计算更加快捷。

经过团队 3 人两天的努力后，终于完成了本次实验。在实验要求基础上，我们进一步尝试了分布式部署运行和新增可视化分析，收获颇丰。