数据预处理实验报告

2019211301班 池纪君 2019213688

实验内容

1. 通过爬虫取链家的新房数据，并进行预处理。
    - 最终的 csv 文件，应包括以下字段：名称地理位置（ 3个字段分别存 储），房型（只保留最小面积按照值总价万元，整数），均价（万元，保留小点后 4 位）；
    - 对于所有字符串段，要求去掉的前后空格；
    - 如果有缺失数据，不用填充。
    - 找出总价最贵和便宜的房子，以及总价的中位数
    - 找出单价最贵和便宜的房子，以及单价的中位数
2. 计算北京空气质量数据
    a. 汇总计算 PM 指数年平均值的变化情况
    b. 汇总计算 10 -15 年PM指数和温度月平均据的变化情况

实验过程

房屋信息抓取和处理

爬虫爬取

- 该爬虫和实验一中的爬虫相似，套用了之前的爬虫代码，其差异见下：
    1. 更改请求头：
        a. 请求一个网站的时候我们知道scrapy默认的请求头是`scrapy`，存在被浏览器封掉的可能，我们通过更换随机的请求头来模拟浏览器操作。
        b. 步骤：
            i. 在`settings.py`文件中添加一些UserAgent：

```
USER_AGENT_LIST=[
    "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.1 (KHTML, like Gecko) Chrome/22.0.1207.1 Safari/537.1",
    "Mozilla/5.0 (X11; CrOS i686 2268.111.0) AppleWebKit/536.11 (KHTML, like Gecko) Chrome/20.0.1132.57 Safari/536.11",
    "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.6 (KHTML, like Gecko) Chrome/20.0.1092.0 Safari/536.6",
    "Mozilla/5.0 (Windows NT 6.2) AppleWebKit/536.6 (KHTML, like Gecko) Chrome/20.0.1090.0 Safari/536.6",
    "Mozilla/5.0 (Windows NT 6.2; WOW64) AppleWebKit/537.1 (KHTML, like Gecko) Chrome/19.77.34.5 Safari/537.1",
    "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.9 Safari/536.5",
    "Mozilla/5.0 (Windows NT 6.0) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.36 Safari/536.5",
    "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.1063.0 Safari/536.3",
    "Mozilla/5.0 (Windows NT 5.1) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.1063.0 Safari/536.3",
    "Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Trident/4.0; SE 2.X MetaSr 1.0; SE 2.X MetaSr 1.0; .NET CLR 2.0.50727; SE 2.X MetaSr 1.0)",
    "Mozilla/5.0 (Windows NT 6.2) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.1062.0 Safari/536.3",
    "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.1062.0 Safari/536.3",
    "Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; 360SE)",
    "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.1061.1 Safari/536.3",
    "Mozilla/5.0 (Windows NT 6.1) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.1061.1 Safari/536.3",
    "Mozilla/5.0 (Windows NT 6.2) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.1061.0 Safari/536.3",
    "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/535.24 (KHTML, like Gecko) Chrome/19.0.1055.1 Safari/535.24",
    "Mozilla/5.0 (Windows NT 6.2; WOW64) AppleWebKit/535.24 (KHTML, l
ike Gecko) Chrome/19.0.1055.1 Safari/535.24"
]
```

ii. 在middleware.py中的process_request函数中随机更换请求头：

```
from lianjia.settings import USER_AGENT_LIST
import random
...
def process_request(self, request, spider):
        rand_use  = random.choice(USER_AGENT_LIST)
        if rand_use:
            request.headers.setdefault('User-Agent', rand_use)
        ...
```

2. 启用AutoThrottle扩展：

   在settings.py中设置以下属性：

```
# Enable and configure the AutoThrottle extension (disabled by default)
# See https://docs.scrapy.org/en/latest/topics/autothrottle.html
AUTOTHROTTLE_ENABLED = True
# The initial download delay
AUTOTHROTTLE_START_DELAY = 5
# The maximum download delay to be set in case of high latencies
AUTOTHROTTLE_MAX_DELAY = 60
# The average number of requests Scrapy should be sending in parallel to
# each remote server
AUTOTHROTTLE_TARGET_CONCURRENCY = 1.0
```

数据处理

分析程序`calc.py`将`result.csv`读取至`Dataframe`中，用`idxmax`，`idxmin`函数寻找最大（小）值的索引，再用iloc函数取出该值，球的总价和单价的最大（小）值。用`median`函数求得总价和单价的均值。具体实现见下。

空气质量信息处理

PM指数年平均值的计算

处理步骤如下：

1. 将每一行中北京四个地区的测量值取平均，作为该时刻北京的PM指数。
2. 将所有数据按照年份聚集，并用聚集函数mean进行计算后，导出到result.csv中

PM指数和温度月平均值的计算

处理步骤如下：

1. 将每一行中北京四个地区的测量值取平均，作为该时刻北京的PM指数。
2. 将TEMP列复制一份至TEMP_AVG中。
3. 将表格中的year列和month列的类型设置为int，TEMP_AVG列的类型为float。
4. 将所有数据按照年，月聚集，并用聚合函数mean对PM指数，温度进行计算后，导出到result.csv中。

实验结果

房屋数据

1. 爬虫爬取到的房屋数据：

```
name,location0,location1,location2,type,size,perunit,price
和光悦府,朝阳,朝阳其它,南皋路和光悦府,4室,120,8.8,1056
水岸壹号,房山,良乡,良乡大学城西站地铁南侧800米，刺猬河旁,3室,185,5.8,1073
观唐云鼎,密云,溪翁庄镇,溪翁庄镇密溪路39号院（云佛山度假村对面）,3室,172,3.0,516
运河铭著,通州,北关,商通大道与榆东一街交叉口，温榆河森林公园东500米,2室,100,4.9,490
万年广阳郡九号,房山,长阳,长阳清苑南街与汇商东路交汇处西北角,3室,166,5.0,830
首开璞瑅公馆,丰台,方庄,紫芳园五区,3室,203,10.6,2152
华远裘马四季,门头沟,大峪,增产路16号院,3室,156,5.5,858
御汤山熙园,昌平,昌平其它,北京市昌平区小汤山镇顺沙路99号院,4室,300,4.0,1200
华远和墅,大兴,南中轴机场商务区,南六环磁各庄桥沿南中轴向南2公里,5室,295,5.4,1593
天资华府,房山,长阳,房山区CSD政务大厅5号门,3室,115,3.8,437
檀香府,门头沟,门头沟其它,京潭大街与潭柘十街交叉口,3室,208,4.5,936
韩建·观山源墅,房山,良乡,阳光北大街与多宝路交汇处西南（理工大学北校区西侧）,3室,290,4.0,1160
首城汇景墅,平谷,平谷其它,"金河北街6号院， 金河北街8号院",3室,360,2.5,900
中国铁建花语金郡,大兴,瀛海,南海子公园西侧(南五环旧忠桥向南第二个红绿灯西300米),3室,150,7.0,1050
北辰墅院1900,顺义,马坡,顺兴街11号院望尊园,4室,251,4.2,1054
首创天阅西山,海淀,海淀北部新区,海淀区丰秀东路9号院，永丰路与北清路交汇处东北角，中关村壹号北侧,4室,17
5,8.0,1400
翡翠公园,昌平,北七家,北七家京承高速北七家出口向西3公里，七星路与七北路交汇处,4室,98,6.1,598
北科建泰禾丽春湖院子,昌平,沙河,中关村北延新核心，沙河水库边（地铁昌平线沙河站向南800米）,4室,379,5.
0,1895
绿地海珀云翡,大兴,大兴其它,兴亦路京开高速东侧（黄村镇第一中心小学对面）,2室,102,6.5,663
都丽华府,平谷,平谷其它,新平南路与林荫南街交汇处向西100米,2室,94,2.9,273
中粮京西祥云,房山,长阳,地铁稻田站北800米，西邻京深路,4室,115,5.8,667
燕西华府,丰台,丰台其它,"王佐镇青龙湖公园东1500米,",4室,60,4.2,252
水岸壹号,房山,良乡,良乡大学城西站地铁南侧800米，刺猬河旁,3室,122,4.3,525
紫辰院,丰台,岳各庄,岳各庄北桥东北角200米处,5室,266,12.8,3405
鲁能格拉斯小镇,通州,通州其它,北京市通州区宋庄镇格拉斯小镇营销中心,3室,246,6.0,1476
兴创荣墅,大兴,大兴新机场洋房别墅区,北京市大兴区育胜街,3室,240,2.3,552
温哥华森林,昌平,北七家,"北五环外紧邻立汤路，北七家建材城向北第一个路口200米路东， 枫树家园6区， 枫树家
园五区",4室,460,4.3478,2000
润泽御府,朝阳,北苑,北京市朝阳区北五环顾家庄桥向北约2.6公里,4室,540,11.0,5940
中骏西山天璟,门头沟,城子,西山永定楼北300米,4室,117,6.5,760
```

```
国瑞熙墅,昌平,北七家,北七家镇岭上西路与定泗路交汇处东南角,3室,314,4.8,1507
中冶德贤公馆,大兴,旧宫,德贤东路6号院（南四环榴乡桥东南角800米）,0室,134,7.7,1032
燕西华府,丰台,丰台其它,"王佐镇青龙湖公园东1500米， 泉湖西路1号院（七区）， 泉湖西路1号院（六区）",0室,
195,5.2,1014
京西悦府,房山,阎村,燕房线阎村地铁站东南角约189米,3室,120,3.3,396
首创伊林郡,房山,良乡,京港澳高速22B良乡机场出口即到，行宫西街1号院,2室,81,3.65,296
K2十里春风,通州,通州其它,永乐店镇漷小路百菜玛工业园对面,2室,74,2.45,181
奥园雲水院,密云,溪翁庄镇,溪翁庄镇,3室,120,2.5,300
北京城建·龙樾西山,门头沟,冯村,长安街西延线南约300米,4室,118,4.8,566
远洋新天地,门头沟,上岸地铁,长安街西延线与滨河路南延交汇处（东南侧）,1室,1118,2.5,2795
长海御墅,房山,房山其它,长沟国家湿地公园南侧,3室,224,2.3,515
棠颂璟庐,亦庄开发区,亦庄开发区其它,鹿华路7号院（南海子公园北侧500米）,4室,250,7.5,1875
金隅上城郡,昌平,北七家,北亚花园东路50米,4室,212,4.5,954
万科弗农小镇,密云,溪翁庄镇,密关路西侧（密云水库南岸2公里）,3室,140,2.5,350
中铁华侨城和园,大兴,瀛海,南五环南海子公园西侧约500米,3室,154,6.0,924
顺鑫颐和天璟,顺义,顺义其它,新城右堤路与昌金路交汇处向北200米,3室,110,3.3,363
誉天下盛寓,顺义,中央别墅区,中央别墅区榆阳路与林荫路交叉口,3室,120,6.0,720
泰禾金府大院,丰台,西红门,南四环地铁新宫站南800米,2室,175,8.2,1435
奥园雲水院,密云,溪翁庄镇,密云区Y753(走石路),3室,111,2.2,244
北京城建北京合院,顺义,顺义其它,燕京街与通顺路交汇口东800米(仁和公园南),3室,95,4.7,446
珠光御景西园,丰台,丰台其它,北京市丰台区长辛店长云路2号珠江御景营销中心,3室,117,3.9,456
北京城建北京合院,顺义,顺义其它,燕京街与通顺路交汇口东800米(仁和公园南),4室,210,4.5,945
```

2. 分析结果：

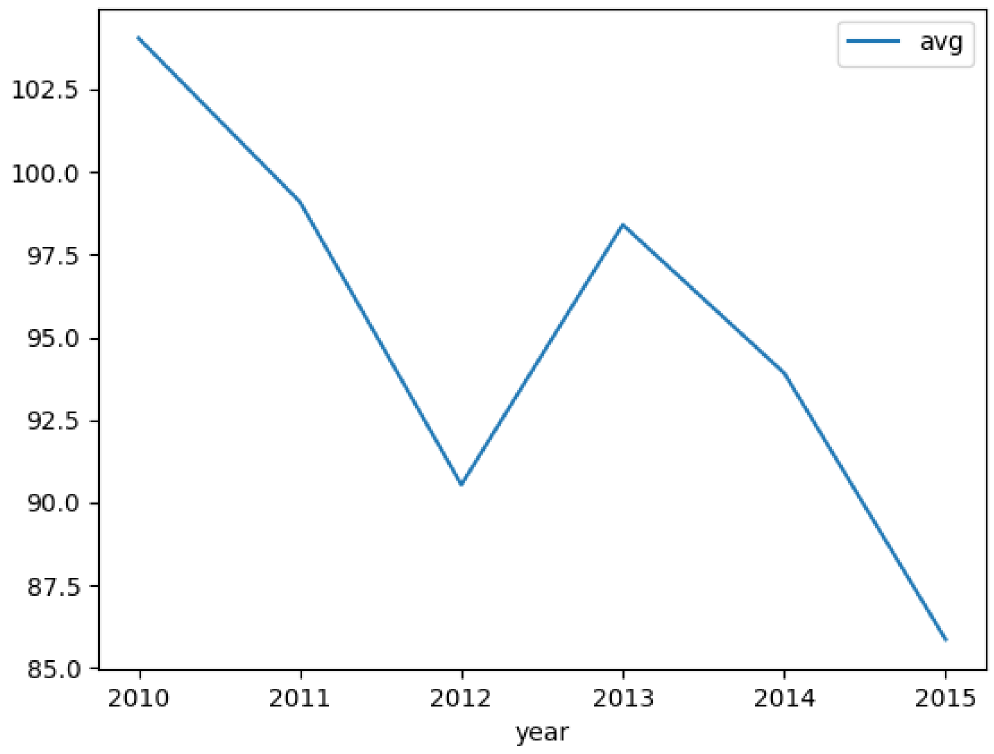
```
总价最贵：
name                          北京壹号总部
location0                         大兴
location1                         亦庄
location2     台湖镇光机电一体化产业基地科创东二街5号
type                              1室
size                            3127
perunit                          2.8
price                           8756
Name: 135, dtype: object

总价最便宜：name                  长海御墅
location0              房山
location1            房山其它
location2     长沟国家湿地公园南侧
type                   1室
size                   70
perunit               1.5
price                 105
Name: 143, dtype: object

总价中位数：559.0

均价最贵：
name                       北京庄园
location0                    顺义
location1                  顺义其它
location2     京承高速第11出口往东800米
type                         4室
size                        460
```

```
perunit                  16.7
price                    7682
Name: 126, dtype: object

均价最便宜：name                长海御墅
location0            房山
location1          房山其它
location2     长沟国家湿地公园南侧
type                 1室
size                 70
perunit             1.5
price               105
Name: 143, dtype: object

均价中位数：4.7
```

PM数据

1. 年平均PM指数

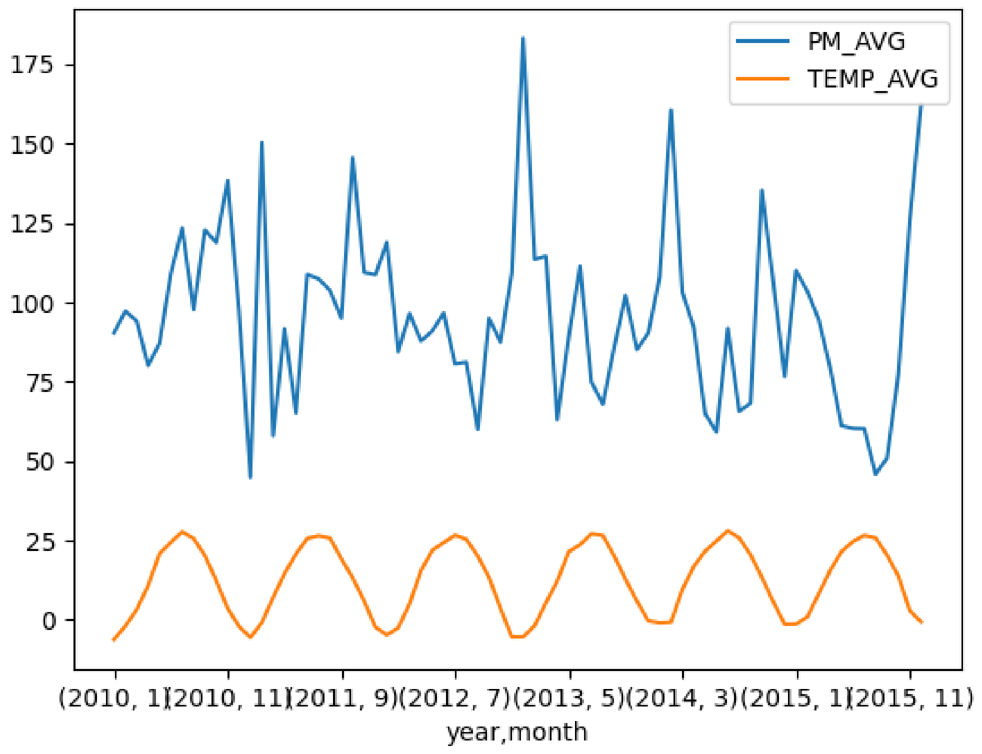
   ```
                avg
   year
   2010  104.045730
   2011   99.093240
   2012   90.538768
   2013   98.402683
   2014   93.917709
   2015   85.858937
   ```



2. 10-15年PM指数和温度月平均：

```
year,month,PM_AVG,TEMP_AVG
2010,1,90.4037,-6.1626
2010,2,97.2399,-1.9226
2010,3,94.0465,3.2930
2010,4,80.0724,10.8069
2010,5,87.0719,20.8320
2010,6,109.0389,24.4347
2010,7,123.4261,27.7298
2010,8,97.6834,25.6116
2010,9,122.7927,20.2139
2010,10,118.7844,12.2997
2010,11,138.3840,3.6097
2010,12,97.1157,-2.0645
2011,1,44.8737,-5.5538
2011,2,150.2902,-0.8542
2011,3,57.9920,7.0685
2011,4,91.7207,14.6056
2011,5,65.1081,20.7137
2011,6,108.7947,25.6486
2011,7,107.3865,26.4691
2011,8,103.7338,25.7581
2011,9,94.9694,19.2319
2011,10,145.5568,13.2097
2011,11,109.4350,5.9806
```

```
2011,12,108.7214,-2.3024
2012,1,118.9224,-4.7581
2012,2,84.4420,-2.5115
2012,3,96.4743,5.0726
2012,4,87.8359,15.4736
2012,5,90.9667,21.8965
2012,6,96.6342,24.3375
2012,7,80.6497,26.6573
2012,8,81.1653,25.3737
2012,9,59.9522,20.0889
2012,10,94.9514,13.3172
2012,11,87.4370,3.6417
2012,12,109.1873,-5.4086
2013,1,185.4622,-5.3777
2013,2,116.7500,-1.8214
2013,3,116.9778,5.4059
2013,4,64.8573,12.2486
2013,5,93.1554,21.4556
2013,6,112.7762,23.6778
2013,7,75.1094,27.0860
2013,8,67.7270,26.5712
2013,9,86.8729,20.1250
2013,10,103.9557,12.8212
2013,11,85.8356,5.9139
2013,12,91.0717,-0.2930
2014,1,108.3519,-0.9140
2014,2,161.4057,-0.7024
2014,3,103.9967,9.5645
2014,4,93.8053,16.8444
2014,5,65.2606,21.6129
2014,6,59.9310,24.8333
2014,7,93.9471,28.0444
2014,8,67.3818,25.8011
2014,9,69.6711,20.5042
2014,10,134.2112,13.3414
2014,11,105.0890,5.6764
2014,12,76.1756,-1.4194
2015,1,110.4810,-1.3266
2015,2,104.3684,0.9420
2015,3,94.6081,8.2651
2015,4,80.1885,15.5389
2015,5,62.0695,21.4933
2015,6,59.0954,24.6745
2015,7,61.3640,26.5672
2015,8,47.4723,25.8291
2015,9,50.8595,20.4083
2015,10,76.4196,13.8280
2015,11,125.4299,2.8971
2015,12,162.1839,-0.6178
```

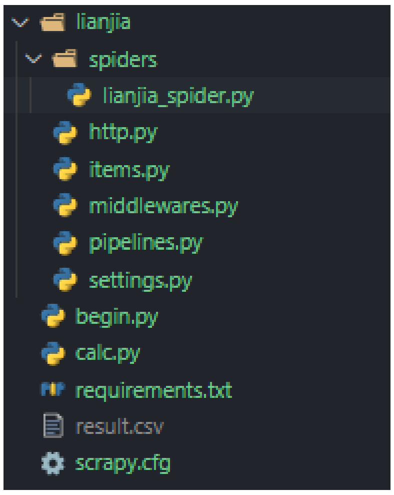


源代码

房屋数据抓取和处理

爬虫

文件目录树：


```
lianjia
  spiders
    lianjia_spider.py
  http.py
  items.py
  middlewares.py
  pipelines.py
  settings.py
begin.py
calc.py
requirements.txt
result.csv
scrapy.cfg
```

lianjia_spider.py

```
import scrapy
from lianjia.http import MyRequest
from lianjia.items import LianjiaItem


class lianjiaSpider(scrapy.Spider):
    name = "lianjia"
    max_page = 20
    id = 1
    start_urls = ["https://bj.fang.lianjia.com/loupan/"]

    def start_requests(self):
        for url in self.start_urls:
            yield MyRequest(url=url,
                            callback=self.parse,
                            dont_filter=True,
                            cb_kwargs={'page': 1})

    def parse(self, response, **kwargs):
        houses = response.xpath("/html/body/div[3]/ul[2]/li")

        for house in houses:
            try:
                print(house)
                item = LianjiaItem()
                item['name'] = house.xpath(
                    "./div/div[1]/a/text()").get().strip()
                locations = house.xpath("./div/div[2]")
                item['location0'] = locations.xpath(
                    "./span[1]/text()").get().strip()
                item['location1'] = locations.xpath(
                    "./span[2]/text()").get().strip()
                item['location2'] = locations.xpath("./a/text()").get().strip()
                item['type'] = house.xpath(
                    "./div/a/span[1]/text()").get().strip()
                item['size'] = int(
                    house.xpath("./div/div[3]/span/text()").get().split(' ')
                    [1].split('-')[0].strip('㎡'))

                flag = house.xpath(
                    "./div/div[6]/div[1]/span[2]/text()").get().strip()
                if flag == '元/㎡(均价)':
                    item['perunit'] = int(
                        house.xpath("./div/div[6]/div[1]/span[1]/text()").get(
                        ).split('-')[0])
                    item['price'] = round(item['perunit'] * item['size'] /
                                          10000)
                    item['perunit'] = round(item['perunit'] / 10000, 4)
                else:
                    print('\n\n\nasdfasdfasdfasdf\n\n')
                    print(flag)
                    item['price'] = int(
                        house.xpath("./div/div[6]/div[1]/span[1]/text()").get(
                        ).split('-')[0])
                    item['perunit'] = round(item['price'] / item['size'], 4)

                self.id += 1
                yield item
            except:
                print('Error occured.')

        if kwargs["page"] + 1 <= self.max_page:
            nxt_page_url = self.start_urls[0] + f'pg{kwargs["page"]+1}'
            yield MyRequest(url=nxt_page_url,
                            callback=self.parse,
                            dont_filter=True,
                            cb_kwargs={'page': kwargs["page"] + 1})
```

http.py

```
from scrapy import Request

class MyRequest(Request):

    def __init__(self, wait_time=None, wait_until=None, *args, **cb_kwargs):
        self.wait_time = wait_time
        self.wait_until = wait_until
        super().__init__(*args, **cb_kwargs)
```

items.py

```
# Define here the models for your scraped items
#
# See documentation in:
# https://docs.scrapy.org/en/latest/topics/items.html

import scrapy

class LianjiaItem(scrapy.Item):
    # define the fields for your item here like:
    # name = scrapy.Field()
    name = scrapy.Field()
    type = scrapy.Field()
    location0 = scrapy.Field()
    location1 = scrapy.Field()
    location2 = scrapy.Field()
    price = scrapy.Field()
    size = scrapy.Field()
    perunit = scrapy.Field()
```

middlewares.py

```
from scrapy import signals
from selenium import webdriver
from selenium.webdriver.support.ui import WebDriverWait
from scrapy.http import HtmlResponse
from lianjia.settings import USER_AGENT_LIST
import random
# from selenium.webdriver.support import expected_conditions as Expect
# from selenium.webdriver.common.by import By

class LianjiaDownloaderMiddleware:
    """Scrapy middleware handling the requests using selenium"""

    def __init__(self):
        driver_executable_path = 'V:\Code\Py_projects\spiders\chromedriver.exe'
        driver_options = webdriver.ChromeOptions()
        driver_options.add_argument('--headless')
        driver_options.add_argument('--ignore-certificate-errors-spki-list')
        driver_options.add_argument('log-level=3')

        self.driver = webdriver.Chrome(executable_path = driver_executable_path, optio
ns = driver_options)

    @classmethod
    def from_crawler(cls, crawler):
        """Initialize the middleware with the crawler settings"""

        middleware = cls()
        crawler.signals.connect(middleware.spider_closed, signals.spider_closed)
        return middleware

    def process_request(self, request, spider):
        rand_use  = random.choice(USER_AGENT_LIST)
        if rand_use:
            request.headers.setdefault('User-Agent', rand_use)
        self.driver.implicitly_wait(30)
        self.driver.get(request.url)
        self.driver.find_elements_by_class_name('resblock-list-wrapper')
        # x = WebDriverWait(self.driver, 30).until(Expect.presence_of_element_located
((By.CLASS_NAME, "result")))

        for cookie_name, cookie_value in request.cookies.items():
            self.driver.add_cookie(
                {
                    'name': cookie_name,
                    'value': cookie_value
                }
            )
        # if request.wait_until:
        #     WebDriverWait(self.driver, request.wait_time).until(
        #         request.wait_until
        #     )

        body = str.encode(self.driver.page_source)

        # Expose the driver via the "meta" attribute
        request.meta.update({'driver': self.driver})

        return HtmlResponse(
            self.driver.current_url,
            body=body,
            encoding='utf-8',
            request=request
        )

    def spider_closed(self):
        """Shutdown the driver when spider is closed"""
        self.driver.quit()
```

pipelines.py

```
# Define your item pipelines here
#
# Don't forget to add your pipeline to the ITEM_PIPELINES setting
# See: https://docs.scrapy.org/en/latest/topics/item-pipeline.html

import csv
# useful for handling different item types with a single interface
# from itemadapter import ItemAdapter


class LianjiaPipeline:
    items = []
    header = ['name', 'location0','location1', 'location2', 'type', 'size', 'perunit',
'price']

    def open_spider(self, spider):
        try:
            self.file = open('result.csv', "w", encoding="utf-8", newline='')
        except Exception as err:
            print(err)
        self.f_csv = csv.DictWriter(self.file, self.header)
        self.f_csv.writeheader()

    def process_item(self, item, spider):
        self.items.append(dict(item))
        return item
    def close_spider(self, spider):
        self.f_csv.writerows(self.items)
        self.file.close()
```

settings.py

```
# Scrapy settings for lianjia project
#
# For simplicity, this file contains only settings considered important or
# commonly used. You can find more settings consulting the documentation:
#
#     https://docs.scrapy.org/en/latest/topics/settings.html
#     https://docs.scrapy.org/en/latest/topics/downloader-middleware.html
#     https://docs.scrapy.org/en/latest/topics/spider-middleware.html

BOT_NAME = 'lianjia'

SPIDER_MODULES = ['lianjia.spiders']
NEWSPIDER_MODULE = 'lianjia.spiders'


# Crawl responsibly by identifying yourself (and your website) on the user-agent
#USER_AGENT = 'lianjia (+http://www.yourdomain.com)'

# Obey robots.txt rules
ROBOTSTXT_OBEY = True

# Configure maximum concurrent requests performed by Scrapy (default: 16)
# CONCURRENT_REQUESTS = 32

# Configure a delay for requests for the same website (default: 0)
# See https://docs.scrapy.org/en/latest/topics/settings.html#download-delay
# See also autothrottle settings and docs
DOWNLOAD_DELAY = 60
# The download delay setting will honor only one of:
#CONCURRENT_REQUESTS_PER_DOMAIN = 16
#CONCURRENT_REQUESTS_PER_IP = 16

# Disable cookies (enabled by default)
COOKIES_ENABLED = False

# Disable Telnet Console (enabled by default)
#TELNETCONSOLE_ENABLED = False

# Override the default request headers:
# DEFAULT_REQUEST_HEADERS = {
#    'Accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8',
#    'Accept-Language': 'en',
# }
USER_AGENT_LIST=[
    "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.1 (KHTML, like Gecko) Chrome/
22.0.1207.1 Safari/537.1",
    "Mozilla/5.0 (X11; CrOS i686 2268.111.0) AppleWebKit/536.11 (KHTML, like Gecko) Ch
rome/20.0.1132.57 Safari/536.11",
    "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.6 (KHTML, like Gecko) Chrome/
20.0.1092.0 Safari/536.6",
    "Mozilla/5.0 (Windows NT 6.2) AppleWebKit/536.6 (KHTML, like Gecko) Chrome/20.0.10
90.0 Safari/536.6",
    "Mozilla/5.0 (Windows NT 6.2; WOW64) AppleWebKit/537.1 (KHTML, like Gecko) Chrome/
19.77.34.5 Safari/537.1",
    "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.
0.1084.9 Safari/536.5",
    "Mozilla/5.0 (Windows NT 6.0) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.10
84.36 Safari/536.5",
    "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/
19.0.1063.0 Safari/536.3",
    "Mozilla/5.0 (Windows NT 5.1) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.10
63.0 Safari/536.3",
    "Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Trident/4.0; SE 2.X MetaSr 1.
0; SE 2.X MetaSr 1.0; .NET CLR 2.0.50727; SE 2.X MetaSr 1.0)",
    "Mozilla/5.0 (Windows NT 6.2) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.10
62.0 Safari/536.3",
    "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/
19.0.1062.0 Safari/536.3",
    "Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; 360SE)",
    "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/
19.0.1061.1 Safari/536.3",
    "Mozilla/5.0 (Windows NT 6.1) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.10
61.1 Safari/536.3",
    "Mozilla/5.0 (Windows NT 6.2) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.10
61.0 Safari/536.3",
    "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/535.24 (KHTML, like Gecko) Chrome/19.
0.1055.1 Safari/535.24",
    "Mozilla/5.0 (Windows NT 6.2; WOW64) AppleWebKit/535.24 (KHTML, like Gecko) Chrom
e/19.0.1055.1 Safari/535.24"
]

# Enable or disable spider middlewares
# See https://docs.scrapy.org/en/latest/topics/spider-middleware.html
#SPIDER_MIDDLEWARES = {
#    'lianjia.middlewares.LianjiaSpiderMiddleware': 543,
#}

# Enable or disable downloader middlewares
# See https://docs.scrapy.org/en/latest/topics/downloader-middleware.html
DOWNLOADER_MIDDLEWARES = {
   'lianjia.middlewares.LianjiaDownloaderMiddleware': 543,
}

# Enable or disable extensions
# See https://docs.scrapy.org/en/latest/topics/extensions.html
#EXTENSIONS = {
#    'scrapy.extensions.telnet.TelnetConsole': None,
#}

# Configure item pipelines
# See https://docs.scrapy.org/en/latest/topics/item-pipeline.html
ITEM_PIPELINES = {
   'lianjia.pipelines.LianjiaPipeline': 300,
}

# Enable and configure the AutoThrottle extension (disabled by default)
# See https://docs.scrapy.org/en/latest/topics/autothrottle.html
```

```
AUTOTHROTTLE_ENABLED = True
# The initial download delay
AUTOTHROTTLE_START_DELAY = 5
# The maximum download delay to be set in case of high latencies
AUTOTHROTTLE_MAX_DELAY = 60
# The average number of requests Scrapy should be sending in parallel to
# each remote server
AUTOTHROTTLE_TARGET_CONCURRENCY = 1.0
# Enable showing throttling stats for every response received:
#AUTOTHROTTLE_DEBUG = False

# Enable and configure HTTP caching (disabled by default)
# See https://docs.scrapy.org/en/latest/topics/downloader-middleware.html#httpcache-mi
ddleware-settings
#HTTPCACHE_ENABLED = True
#HTTPCACHE_EXPIRATION_SECS = 0
#HTTPCACHE_DIR = 'httpcache'
#HTTPCACHE_IGNORE_HTTP_CODES = []
#HTTPCACHE_STORAGE = 'scrapy.extensions.httpcache.FilesystemCacheStorage'
```

begin.py

```
from scrapy import cmdline

cmdline.execute("scrapy crawl lianjia".split())
```

数据处理

calc.py

```
import pandas as pd

f = pd.read_csv(r'V:\Code\Py_projects\spiders\homework2\1_lianjia\result.csv')

mxid = f['price'].idxmax()
miid = f['price'].idxmin()
mid = f['price'].median()
print(f"\n总价最贵：\n{f.iloc[mxid]}")
print(f"\n总价最便宜：{f.iloc[miid]}")
print(f"\n总价中位数：{mid}")

mxid = f['perunit'].idxmax()
miid = f['perunit'].idxmin()
mid = f['perunit'].median()
print(f"\n均价最贵：\n{f.iloc[mxid]}")
print(f"\n均价最便宜：{f.iloc[miid]}")
print(f"\n均价中位数：{mid}")
```

空气质量和温度数据处理

1_avg.py

```
import pandas as pd
import numpy as np
from tqdm import tqdm

df = pd.read_csv('BeijingPM20100101_20151231.csv', encoding='utf-8', dtype=str)

for i in tqdm(range(len(df['No']))):
    sum = count = 0
    if not (df['PM_Dongsihuan'][i] is np.nan):
        sum += int(df['PM_Dongsihuan'][i])
        count += 1
    if not (df['PM_US Post'][i] is np.nan):
        sum += int(df['PM_US Post'][i])
        count += 1
    if not (df['PM_Dongsi'][i] is np.nan):
        sum += int(df['PM_Dongsi'][i])
        count += 1

    if not (df['PM_Nongzhanguan'][i] is np.nan):
        sum += int(df['PM_Nongzhanguan'][i])
        count += 1

    if count != 0:
        df.loc[i, 'avg'] = round(sum / count, 2)
    else:
        df.loc[i, 'avg'] = None

df = df.groupby("year").mean()
print(df)
```

2_air.py

```
import pandas as pd
import numpy as np
from tqdm import tqdm

df = pd.read_csv('BeijingPM20100101_20151231.csv', encoding='utf-8', dtype=str)
print(df.info())

for i in tqdm(range(len(df['No']))):
    sum = count = sumt = countt = 0
    if not (df['PM_Dongsihuan'][i] is np.nan):
        sum += int(df['PM_Dongsihuan'][i])
        count += 1
    if not (df['PM_US Post'][i] is np.nan):
        sum += int(df['PM_US Post'][i])
        count += 1
    if not (df['PM_Dongsi'][i] is np.nan):
        sum += int(df['PM_Dongsi'][i])
        count += 1
    if not (df['PM_Nongzhanguan'][i] is np.nan):
        sum += int(df['PM_Nongzhanguan'][i])
        count += 1

    try:
        df.loc[i, 'PM_AVG'] = round(sum / count, 2)
    except:
        pass

df['year'] = df['year'].astype('int')
df['month'] = df['month'].astype('int')
df['TEMP_AVG'] = df['TEMP'].astype('float')

newdf = df.groupby(by=['year', 'month']).agg(
    {
        'PM_AVG': 'mean',
        'TEMP_AVG': 'mean'
    }, sort=True)
newdf.to_csv('result.csv', float_format='%.4f', encoding='utf-8')
print(newdf)
```