Python 数据预处理作业

班级：2018211313 班　　学号：2018211366　　姓名：蒋潇逸

版本：1.0

更新：*November 24, 2020*

本文档是 Python 数据预处理作业报告。

目录

1 作业要求及内容

1.1 作业 1

通过爬虫爬取链家的新房数据，并进行预处理。

- 对于所有字符串字段，要求去掉所有的前后空格
- 如果有缺失数据，不用填充
- 找出总价最贵和最便宜的房子，以及总价的中位数
- 找出单价最贵和最便宜的房子，以及单价的中位数

1.2 作业 2

计算北京空气质量数据

- 汇总计算 PM 指数年平均值的变化情况
- 汇总计算 10-15 年 PM 指数和温度月平均数据的变化情况

2 实验设备环境

Windows 10 专业版　　PyCharm 2019.3.3 x64　　编程语言 Python

3 作业 1

3.1 爬取链家新房的数据

爬取新房时有三种情况

- 有新房的建面和总价数据，算出均价即可
- 有新房的建面和均价数据，算出总价即可
- 有新房的均价数据没有建面数据，需爬取下一行的总价数据

爬虫代码如下所示：

```
import scrapy
from data.items import lianjiaItem
from scrapy.http import Request
```

```
4
5
6  class MySpider(scrapy.Spider):
7      name = "lianjia"
8      start_urls = ["https://bj.fang.lianjia.com/loupan/pg1/"]  # 初始
           url
9
10     def parse(self, response):
11         item = lianjiaItem()
12         for each in response.xpath('/html/body/div[4]/ul[2]/li'):
13             item['name'] = each.xpath("div/div[1]/a/text()").extract
                   ()[0]
14             item['position_1'] = each.xpath("div/div[2]/span[1]/text
                   ()").extract()[0]
15             item['position_2'] = each.xpath("div/div[2]/span[2]/text
                   ()").extract()[0]
16             item['position_3'] = each.xpath("div/div[2]/a/text()").
                   extract()[0]
17             if each.xpath("div/a/span/text()").extract():
18                 item['room'] = each.xpath("div/a/span/text()").
                       extract()[0]
19             else:
20                 item['room'] = None
21             if each.xpath("div/div[3]/span/text()").extract():
22                 temp = str(each.xpath("div/div[3]/span/text()").
                       extract()[0])
23                 temp = temp.replace("建面 ", "")
24                 temp = temp.replace("㎡", "")
25                 temp = temp.replace("[\'", "")
26                 temp = temp.replace("\']", "")
27                 item['area'] = int(temp.split("-")[0])
28                 temp = str(each.xpath("div/div[6]/div[1]/span[2]/text
                       ()").extract()[0])
29                 if temp.find("均价") > 0:
30                     item['unitPrice'] = int(each.xpath("div/div[6]/
                           div[1]/span[1]/text()").extract()[0])
31                     item['totalPrice'] = round(float(item['unitPrice'
                           ] * item['area'] / 10000), 4)
32                     item['totalPrice'] = "{:.4f}".format(item['
```

```
                        totalPrice']) + "\t"
                elif temp.find("总价") > 0:
                    item['totalPrice'] = int(each.xpath("div/div[6]/
                        div[1]/span[1]/text()").extract()[0])
                    item['unitPrice'] = round(float(item['totalPrice'
                        ] * 10000 / item['area']), 4)
                    item['unitPrice'] = "{:.4f}".format(item['
                        unitPrice']) + "\t"
            else:
                item['area'] = None
                temp = each.xpath("div/div[6]/div[2]/text()").extract
                    ()[0]
                temp = temp.replace("总价", "")
                temp = temp.replace("万/套", "")
                item['totalPrice'] = temp
                item['unitPrice'] = int(each.xpath("div/div[6]/div
                    [1]/span[1]/text()").extract()[0])
            yield (item)

        for i in range(2, 20):
            url = 'https://bj.fang.lianjia.com/loupan/pg{}/'.format(
                str(i))
            yield Request(url, callback=self.parse)  # 回调
```

pipeline.py 中代码如下所示：

```
# 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


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

import json


class MyPipeline(object):
    def open_spider(self, spider):
```

```
15          try:  # 打开json文件
16              self.file = open('MyData.json', "w", encoding="utf-8")
17          except Exception as err:
18              print(err)
19
20      def process_item(self, item, spider):
21          dict_item = dict(item)  # 生成字典对象
22          json_str = json.dumps(dict_item, ensure_ascii=False) + "\n"
                # 生成json串
23          # self.file = open('MyData.json', "w", encoding="utf-8")
24          self.file.write(json_str)  # 将json串写入到文件中
25          return item
26
27      def close_spider(self, spider):
28          self.file.close()  # 关闭文件
```

3.2 爬虫结果 (前 50 条数据)

	name	position_1	position_2	position_3	room	area	unitPrice	totalPrice
1	天恒水岸壹号	房山	良乡	良乡大学城西站地铁南侧400米，刺猬河旁	3室	185	58000	1073.0000
2	观唐云鼎	密云	密云其它	溪翁庄镇密溪路39号院（云佛山度假村对面）	3室	171	30000	513.0000
3	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	46000	460.0000
4	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	139	48500	674.1500
5	华远裘马四季	门头沟	大峪	增产路16号院	3室	150	60000	900.0000
6	合景映月台	海淀	清河	安宁庄西路与小营西路交汇处西南侧, 安宁庄西路与小营西路36号院项目	4室	182	120879.1209	2200
7	V7九间堂	通州	潞苑	通燕高速耿庄桥北出口中化石油对面	4室	220	68000	1496.0000
8	御汤山熙园	昌平	昌平其它	紧邻安泗路，距离北六环61号出口约2000米	4室	300	40000	1200.0000
9	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	54000	1593.0000
10	天资华府	房山	长阳	房山区CSD政务大厅5号门	2室	94	45000	423.0000
11	东方蓝海中心	昌平	北七家	未来科技城南区内科技城路与南区一路交界处（17号线未来科学城南站100米）	null	null	60000	500
12	赛球CLUB	通州	通州其它	西集镇京哈高速郎府出口南侧300米	2室	89	35000	311.5000
13	建邦·顺颐府	顺义	后沙峪	空港B区裕民大街30号	3室	270	55583	1500.7410
14	和悦春风	大兴	大兴新机场	庞各庄镇瓜乡路与永兴河交汇处	2室	76	35000	266.0000
15	阳光城溪山悦	密云	密云其它	密溪路33号	2室	80	22000	176.0000
16	融创公园壹号	大兴	南中轴机场商务区	孙村公园西侧	2室	73	40000	292.0000
17	招商·臻珑府	亦庄开发区	通州其它	亦庄经济技术开发区经海七路与科创十街交汇处	2室	57	47000	267.9000
18	融创亦庄壹号	亦庄开发区	通州其它	通惠干渠路与科创九街交汇处西南口（地铁亦庄线次渠南站北行900米）	2室	74	52695	389.9430
19	长安九里	石景山	石景山其它	古城南街东侧	2室	86	70000	602.0000
20	富力首开·金禧璞瑅	顺义	顺义其它	高泗路四村段23号	4室	140	47000	658.0000
21	中建国望府	丰台	丰台其它	射击场路中建国望府	5室	314	63630.5732	1998
22	阳光上东	朝阳	酒仙桥	东四环北路6号	3室	189	83000	1568.7000
23	北科建·翡翠华府	怀柔	怀柔	中高路与乐园大街交汇处（雁栖河生态廊道旁）	3室	95	40567	385.3865
24	保利绿城·和锦诚园	大兴	瀛海	8号线南段地铁站东1.6公里	3室	111	62000	688.2000
25	金悦郡	通州	通州其它	亦庄新城环景西一路与景盛南二街交叉口珠江逸景家园南区西侧	2室	72	36000	259.2000
26	北京恒大上河院	密云	密云其它	科技路东侧	2室	79	25500	201.4500
27	路劲·御合院	大兴	大兴新机场洋房别墅区	采育镇彩凤路与育进街交叉口	3室	89	29000	258.1000
28	熙红印	大兴	西红门	宏福路3号	2室	65	64400	418.6000
29	长安和玺	石景山	古城	古城南里长安和玺	4室	125	74000	925.0000
30	观承·望溪	顺义	后沙峪	京承高速8号出口东800米	4室	257	55800	1434.0600
31	富力首开·金禧璞瑅	顺义	顺义其它	高泗路四村段23号	5室	360	36000	1296.0000
32	碧桂园·京源著	延庆	延庆其它	百泉街与延康路交叉口北200米	3室	89	28000	249.2000
33	熙悦天寰	丰台	丰台其它	鑫博西路与大灰厂东路交口西行20米	3室	89	58000	516.2000
34	金融街武夷·融御	通州	武夷花园	通胡大街与东六环西侧路交汇处东南角	2室	50	62000	310.0000
35	中骏云景台	房山	房山其它	官道北21号	2室	75	28000	210.0000
36	北京金茂府二期	丰台	宋家庄	宋家庄地铁站F口旁	2室	75	103000	772.5000
37	禧瑞金海	平谷	平谷其它	平蓟路与环镇东路交汇	2室	87	22000	191.4000
38	禧瑞金海	平谷	平谷其它	平蓟路与环镇东路交汇	2室	125	22000	275.0000
39	中国铁建·山语澜廷	房山	房山其它	阎吕路中国铁建·山语澜廷	4室	223	30000	669.0000
40	京澜誉府	通州	万达	玉带河大街乙72号	3室	99	60000	594.0000
41	电建·洺悦湾	大兴	旧宫	旧宫北路旧宫地铁站东侧300米	2室	75	54715	410.3625
42	城市之光·东望	通州	通州其它	东五环化工桥向东京津高速第一出口通马路向南约500米	2室	70	51585	361.0950
43	合景泰富天汇	顺义	马坡	昌金路与通顺路交汇处	3室	89	38000	338.2000
44	金地旭辉·江山风华	大兴	黄村中	地铁4号线清源路站西侧800米	3室	89	55800	496.6200
45	水岸雁栖	怀柔	怀柔	雁栖镇京加路雁栖桥西北500米处	0室	56	34000	190.4000
47	懋源·璟玺	朝阳	中央别墅区	孙河京密路与京平辅路交叉口西行1000米	4室	262	86000	2253.2000
48	金隅学府	大兴	大兴其它	景园北街2号51-1号贵派大厦一楼	2室	82	52695	432.0990
49	台湖金茂悦	通州	通州其它	台湖镇亦庄新城惠民路与生态公园路交汇处	2室	80	50000	400.0000
50	中国铁建国际公馆	大兴	大兴其它	博兴十路中国铁建国际公馆	2室	80	52695	421.5600
51	迈宇平墅	顺义	顺义城	顺平路迈宇平墅	4室	217	38000	824.6000
52	观承·望溪	顺义	后沙峪	京承高速8号出口东800米	4室	130	55800	725.4000
53	山屿西山著	海淀	海淀北部新区	海淀区温泉镇太舟坞	3室	89	53000	471.7000

3.3 数据处理

```
import numpy as np
import pandas as pd
import time

# 1.打开CSV文件
fileNameStr = 'lianjia.csv'
df = pd.read_csv(fileNameStr, encoding='utf-8', dtype=str)
df.applymap(lambda x: x.strip() if type(x) == str else x)  # 去除空格
df[['area', 'unitPrice', 'totalPrice']] = df[['area', 'unitPrice', '
    totalPrice']].astype('float')

```

```
# 2.查看数据集的基本情况
print("2:head==================================")
print(df.head())
print("2:describe==============================")
print(df.describe())
print("2:info==================================")
print(df.info())

print("该数据集中，均价最高为：  %.4f" % df['unitPrice'].max())
print("均价最低为：  %.4f" % df['unitPrice'].min())
print("均价的中位数为：  %.4f" % df['unitPrice'].median())
print("该数据集中，总价最高为：  %.4f万元" % df['totalPrice'].max())
print("总价最低为：  %.4f万元" % df['totalPrice'].min())
print("总价的中位数为：  %.4f万元" % df['totalPrice'].median())

print(df.sort_values(by='totalPrice', ascending=False))
print(df.sort_values(by='unitPrice', ascending=False))
```

3.4 数据处理结果

```
该数据集中，均价最高为：  130000.0000
均价最低为：  21000.0000
均价的中位数为：  52695.0000
该数据集中，总价最高为：  5777.0000万元
总价最低为：  135.0000万元
总价的中位数为：  518.9200万元
```

总价最贵的房子是：

```
75     北京壹号总部   大兴          亦庄  ...    NaN   28000.0  5777.000
```

总价最便宜的房子是：

```
120    西长安壹号  门头沟       门头沟其它  ...    NaN   33000.0   135.000
```

均价最贵的房子是：

```
98     尊悦光华  朝阳   CBD  ...  133.0  130000.0000  1729.00
```

均价最便宜的房子是：

```
139  奥园北京源墅  密云  密云其它  ...  110.0  21000.0000  231.00
```

4 作业 2

4.1 要求 1

要求 1 的代码如下

```
import numpy as np
import pandas as pd
import time

# 1.打开CSV文件
fileNameStr = 'bj_data.csv'
df = pd.read_csv(fileNameStr, encoding='utf-8')

# 2.查看数据集的基本情况
print("2:head============================")
print(df.head())
print("2:describe========================")
print(df.describe())
print("2:info============================")
print(df.info())

# 3.查看是否有缺失值
print("3=================================")
print(df.isnull().sum().sort_values(ascending=False))

# 4.去掉无关的列
df.drop(df.columns[[0, 2, 3, 4, 5, 10, 11, 12, 13, 14, 15, 16, 17]],
    axis=1, inplace=True)
df.dropna(axis=0, how='all', subset=['PM_Dongsi', 'PM_Dongsihuan', '
    PM_Nongzhanguan', 'PM_US Post'], inplace=True)

# 5.计算平均值
df['sum'] = df[['PM_Dongsi', 'PM_Dongsihuan', 'PM_Nongzhanguan', '
    PM_US Post']].sum(axis=1)
df['count'] = df[['PM_Dongsi', 'PM_Dongsihuan', 'PM_Nongzhanguan', '
```

```
    PM_US Post']].count(axis=1)
df['ave'] = round(df['sum'] / df['count'], 2)

# 6.按照年做汇总，查看年的平均值
print(df.groupby("year")["ave"].mean())
```

运行结果为

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

从数据可以看出，北京市的 pm2.5 指数在整体上有下降的趋势，但是个别年份还是出现了回升，例如 2013 年 pm2.5 指数相对于 2012 年回升了 8。

4.2 要求 2

要求 2 的代码如下

```
import numpy as np
import pandas as pd
import time

fileNameStr = 'bj_data.csv'
df_1 = pd.read_csv(fileNameStr, encoding='utf-8')
df_2 = pd.read_csv(fileNameStr, encoding='utf-8')

df_1.dropna(axis=0, how='all', subset=['PM_Dongsi', 'PM_Dongsihuan',
    'PM_Nongzhanguan', 'PM_US Post'], inplace=True)
```

```
df_1['sum_pm'] = df_1[['PM_Dongsi', 'PM_Dongsihuan', 'PM_Nongzhanguan', 'PM_US Post']].sum(axis=1)
df_1['count_pm'] = df_1[['PM_Dongsi', 'PM_Dongsihuan', 'PM_Nongzhanguan', 'PM_US Post']].count(axis=1)
df_1['ave_pm'] = round(df_1['sum_pm'] / df_1['count_pm'], 2)

df_pm = df_1.groupby(["year", "month"])["ave_pm"].mean()
df_temp = df_2.groupby(["year", "month"])["TEMP"].mean()
pd.concat([df_pm, df_temp], axis=1, join='inner').to_csv("ave_by_month.csv")
```

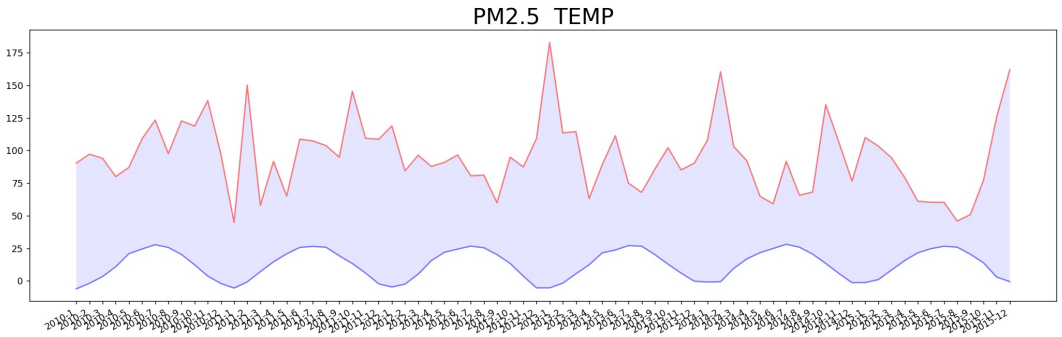
输出的 csv 文件如下

```
year,month,ave_pm,TEMP
2010,1,90.40366972477064,-6.162634408602151
2010,2,97.23994038748137,-1.9226190476190477
2010,3,94.04654442877292,3.293010752688172
2010,4,80.0724233983287,10.806944444444444
2010,5,87.0719131614654,20.831989247311828
2010,6,109.03893805309734,24.434722222222224
2010,7,123.4260752688172,27.72983870967742
2010,8,97.68343195266272,25.611559139784948
2010,9,122.79273504273505,20.21388888888889
2010,10,118.78436657681941,12.299731182795698
2010,11,138.38403614457832,3.609722222222222
2010,12,97.1157469717362,-2.064516129032258
2011,1,44.87369985141159,-5.553763440860215
2011,2,150.29017857142858,-0.8541666666666666
2011,3,57.99198717948718,7.068548387096774
2011,4,91.72067039106145,14.605555555555556
2011,5,65.10814606741573,20.713709677419356
2011,6,108.79465541490858,25.648611111111112
2011,7,107.38648648648649,26.469086021505376
2011,8,103.7338003502627,25.758064516129032
2011,9,94.96940194714882,19.231944444444444
2011,10,145.5568181818182,13.209677419354838
2011,11,109.43496503496503,5.980555555555555
2011,12,108.72139973082099,-2.3024193548387095
2012,1,118.92238805970149,-4.758064516129032
2012,2,84.44202898550725,-2.5114942528735633
```

```
2012,3,96.47432432432433,5.07258064516129
2012,4,87.83588317107093,15.473611111111111
2012,5,90.96671490593343,21.896505376344088
2012,6,96.63418079096046,24.3375
2012,7,80.64970930232558,26.657258064516128
2012,8,81.1653290529695,25.373655913978496
2012,9,59.95224719101124,20.08888888888889
2012,10,94.95135135135135,13.317204301075268
2012,11,87.43696275071633,3.6416666666666666
2012,12,109.18729641693811,-5.408602150537634
2013,1,183.19533783783783,-5.377688172043011
2013,2,113.56647321428571,-1.8214285714285714
2013,3,114.57275537634408,5.405913978494624
2013,4,63.047805555555556,12.248611111111112
2013,5,89.14853494623655,21.455645161290324
2013,6,111.35484722222225,23.677777777777777
2013,7,74.93284366576819,27.086021505376344
2013,8,67.92362903225806,26.571236559139784
2013,9,85.71787499999998,20.125
2013,10,102.20880376344086,12.821236559139784
2013,11,85.14629166666667,5.913888888888889
2013,12,90.3177688172043,-0.29301075268817206
2014,1,107.91174731182795,-0.9139784946236559
2014,2,160.5138988095238,-0.7023809523809523
2014,3,103.18325268817203,9.564516129032258
2014,4,92.16069444444442,16.844444444444445
2014,5,64.95850806451614,21.612903225806452
2014,6,59.15463888888888,24.833333333333332
2014,7,91.79994623655915,28.044354838709676
2014,8,65.66822341857336,25.801075268817204
2014,9,68.23266666666665,20.504166666666666
2014,10,135.26973118279568,13.341397849462366
2014,11,106.33752777777777,5.676388888893055
2014,12,76.62251344086023,-1.4193548387096775
2015,1,110.02276881720428,-1.3266129032258065
2015,2,103.4455505952381,0.9419642857142857
2015,3,94.48344086021504,8.26514131897712
2015,4,79.39705555555553,15.53888888888889
2015,5,61.16751344086024,21.493279569892472
```

```
2015,6,60.3323888888889,24.674547983310152
2015,7,60.22952956989249,26.567204301075268
2015,8,45.89607526881719,25.82907133243607
2015,9,50.92475000000001,20.408333333333335
2015,10,77.2577419354839,13.827956989247312
2015,11,125.80318055555553,2.8970792767732965
2015,12,162.17891129032256,-0.6177658142664872
```

画图如下



从图中可以看出温度在每年的 2 月份是最低的，在每年的 8、9 月份是最高的。PM2.5 指数也随着温度有着周期性的变化，一般来说，每年的 1、2 月份即温度最低时，北京的 pm2.5 指数是最高的，到达一个顶峰；每年的 8、9 月份即温度最高时，北京的 pm2.5 指数是最低的。

画图代码如下

```python
import csv
from matplotlib import pyplot as plt
from datetime import datetime

# 读取CSV文件数据
filename = 'ave_by_month.csv'
with open(filename) as f:  # 打开这个文件，并将结果文件对象存储在f中
    reader = csv.reader(f)  # 创建一个阅读器reader
    header_row = next(reader)  # 返回文件中的下一行
    dates, pm, temp = [], [], []  # 声明存储日期，最值的列表
    for row in reader:
        current_date = row[0] + "-" + row[1]  # 将日期数据转换为
            datetime对象
```

```
        dates.append(current_date)  # 存储日期
        high = float(row[2])  # 将字符串转换为数字
        pm.append(high)
        low = float(row[3])
        temp.append(low)

# 根据数据绘制图形
fig = plt.figure(dpi=128, figsize=(20, 6))
plt.plot(dates, pm, c='red', alpha=0.5)  # 实参alpha指定颜色的透明
    度，0表示完全透明，1（默认值）完全不透明
plt.plot(dates, temp, c='blue', alpha=0.5)
plt.fill_between(dates, pm, temp, facecolor='blue', alpha=0.1)  # 给
    图表区域填充颜色
plt.title('PM2.5  TEMP', fontsize=24)
plt.xlabel('', fontsize=16)
plt.tick_params(axis='x', which='major', labelsize=10)
fig.autofmt_xdate()  # 绘制斜的日期标签
plt.savefig("C:\\Users\\Lenovo\\Desktop\\10.jpg")
plt.show()
```