Lab4_1　Pandas 数据分析

实验目的：学习 Pandas 数据分析基础，统计描述及数据可视化等
实验简介：Pandas 数据导入；数据变换处理；统计汇总描述；假设检验；可视化等。

1. 将数据导入 pandas

Pandas 基于两种数据类型：series 与 dataframe。一个 series 是一个一维的数据类型，其中每一个元素都有一个标签。series 类似于 Numpy 中元素带标签的数组。其中，标签可以是数字或者字符串。一个 dataframe 是一个二维的表结构。Pandas 的 dataframe 可以存储许多种不同的数据类型，并且每一个坐标轴都有自己的标签。首先需要将分析的数据导入数据 pandas。从 csv 文件中读取到数据，并将他们存入 dataframe 中，只需要调用 read_csv 函数并将 csv 文件的路径作为函数参数即可：

```
# 导入 pandas 库并用别名 pd 代替 pandas
import pandas as pd
# 读取 WEB 数据
data_url =
"https://raw.githubusercontent.com/alstat/Analysis-with-Programming/master/
2014/Python/Numerical-Descriptions-of-the-Data/data.csv"
df = pd.read_csv(data_url)
```

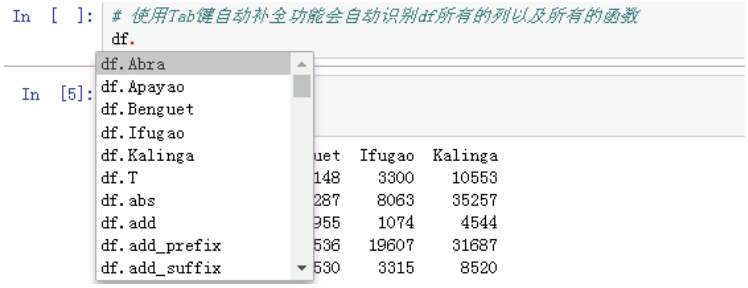
2019-09-18

pandas 的 read_csv 函数能够读取本地或 Web 数据，导入的 CSV 文件存放在名称为 df 的 dataframe 中，是五个地方的 Abra、Apayao、Benguet、Ifugao、Kalinga 的稻谷产量数据。

2. 查看浏览数据

在开始深入探究这些数据之前，先大致浏览一下，并从中获得一些有用信息，帮助确立探究方向。

（1）显示 df，以及 df 的行数：df，len(df)
　　思考如何查看 df 的列数
（2）查看不同列的数据类型：df.dtypes
（3）使用 Tab 键自动补全功能会自动识别 df 所有的列，以及所有的函数：



（4）查看 df 中头部和尾部的行：

```
# 打印数据的头部
df.head()
#示例输出
#    Abra  Apayao  Benguet  Ifugao  Kalinga
#0   1243    2934      148    3300    10553
#1   4158    9235     4287    8063    35257
#2   1787    1922     1955    1074     4544
#3  17152   14501     3536   19607    31687
#4   1266    2385     2530    3315     8520
# 打印数据的尾部
df.tail()
#示例输出
#     Abra  Apayao  Benguet  Ifugao  Kalinga
#74   2505   20878     3519   19737    16513
#75  60303   40065     7062   19422    61808
#76   6311    6756     3561   15910    23349
#77  13345   38902     2583   11096    68663
#78   2623   18264     3745   16787    16900
```

2019-09-18

head()函数和 tail()函数默认输出前 5 行数据和后 5 行数据，当然，也可以显示指定行数 n，使用 df.head(10)可以打印前 10 行数据，打印尾部的数据也是同样的道理，使用 df.head(10)。

（5）显示 columns、index、values：

```
# 提取列名
df.columns
# 示例输出
# Index(['Abra','Apayao', 'Benguet', 'Ifugao', 'Kalinga'], dtype='object')
```

```
# 提取行名或者索引
df.index
# 示例输出
# RangeIndex(start=0, stop=79, step=1)
# 提取值
df.values
```

（6）统计汇总

Pandas 获取数据的一些基本统计信息非常简单，通过 describe()函数实现：

```
# 描述数据的统计特性
df.describe()
# 示例输出
#              Abra        Apayao      Benguet       Ifugao       Kalinga
#count     79.000000     79.000000    79.000000     79.000000     79.000000
#mean   12874.379747  16860.645570  3237.392405  12414.620253  30446.417722
#std    16746.466945  15448.153794  1588.536429   5034.282019  22245.707692
#min      927.000000    401.000000   148.000000   1074.000000   2346.000000
#25%     1524.000000   3435.500000  2328.000000   8205.000000   8601.500000
#50%     5790.000000  10588.000000  3202.000000  13044.000000  24494.000000
#75%    13330.500000  33289.000000  3918.500000  16099.500000  52510.500000
#max    60303.000000  54625.000000  8813.000000  21031.000000  68663.000000
```

这将返回一个包含多种统计信息的表格，例如，计数，均值，标准差，最小值，最大值等。

（6）对数据转置

使用 T 方法对 df 进行转置：

```
# 数据转置
print(df.T)
# 示例输出
#              0     1     2     3     4     5     6     7     8     9   \
#Abra      1243  4158  1787  17152  1266  5576   927  21540  1039  5424
#Apayao    2934  9235  1922  14501  2385  7452  1099  17038  1382  10588
#Benguet    148  4287  1955   3536  2530   771  2796   2463  2592  1064
#Ifugao    3300  8063  1074  19607  3315  13134  5134  14226  6842  13828
#Kalinga  10553  35257  4544  31687  8520  28252  3106  36238  4973  40140
#以下部分省略
```


（7）按轴进行排序：df.sort_index(axis=1, ascending=False)

（8）按值进行排序：df.sort_value(columns = 'Abra')

3. 选择数据

当查看数据集时，可能希望获得一个特殊的样本数据。Pandas 提供了多种方法来选择数据。

pandas 数据访问方式：.at, .iat, .loc, .iloc。

- loc: only work on index
- iloc: work on position
- at: get scalar values. It's a very fast loc

- iat: Get scalar values. It's a very fast iloc

（1）选择一个单独的列，将会返回一个 Series：df['Abra']等同于 df. Abra
（2）通过[]进行选择，将会对行进行切片：df[0:3]
（3）通过 loc 选择，即通过标签选择，完成如下操作，并解释。
使用标签获取一个交叉区域：df.loc[0,['Abra']]
使用标签切片在多个轴上进行选择：df.loc[10:21, ['Abra', 'Apayao', 'Benguet']]
对于返回的对象进行维度缩减：df.loc[:,['Abra', 'Apayao', 'Benguet']]
获取某个位置的值：df.at[0, 'Apayao']
（4）通过 iloc 选择，即通过索引选择，通过 iloc 选择操作完成（3）的数据选择。
假设需要数据第一列的前 5 行：

```
# 选取数据第一列并打印前 5 行
print(df.iloc[:, 0].head())
# 示例输出
#0     1243
#1     4158
#2     1787
#3    17152
#4     1266
#Name: Abra, dtype: int64
```

2019-09-18

顺便提一下，Python 的索引是从 0 开始的，为取出从 10 到 20 行的前 3 列数据：

```
# 提取从 10 到 20 行的前 3 列的数据
print(df.iloc[10:21, 0:3])
# 示例输出
#    Abra  Apayao  Benguet
#10   981    1311     2560
#11 27366   15093     3039
#12  1100    1701     2382
#13  7212   11001     1088
#14  1048    1427     2847
#15 25679   15661     2942
#16  1055    2191     2119
#17  5437    6461      734
#18  1029    1183     2302
#19 23710   12222     2598
#20  1091    2343     2654
```

此，还有条件选择，有兴趣自己去探索。

4. 缺失值处理

在 pandas 中，使用 np.nan 来代替缺失值，这些值将默认不会包含在计算中。

- reindex()可改变/增加/删除指定轴上的索引，并将返回原始数据的一个拷贝
- dropna 去掉包含缺失值的行
- fillna 对缺失值进行填充
- isnull 对数据进行布尔填充

如果要舍弃数据中的列，比如舍弃列 1(Apayao)和列 2(Benguet)，使用 drop 方法：

```
# 舍弃列 1 和列 2
print(df.drop(df.columns[[1, 2]], axis = 1).head())
# OUTPUT
#    Abra  Ifugao  Kalinga
#0   1243    3300    10553
#1   4158    8063    35257
#2   1787    1074     4544
#3  17152   19607    31687
#4   1266    3315     8520
```

axis 参数告诉函数到底舍弃列还是行。如果 axis 等于 0，则舍弃行，否则舍弃列。

更多操作练习，请参考：

http://pandas.pydata.org/pandas-docs/stable/missing_data.html#missing-data

2019-09-18

5. 假设检验

Python 有一统计推断包，即 scipy 中的 stats。ttest_1samp 实现了单样本 t 检验，如果想检验数据 Abra 列的稻谷产量均值，假定总体稻谷产量均值为 15000，通过原假设：

```
#导入 scipy 中的 stats
import scipy.stats as ss
# 单个总体均值 μ 的 t 检验
Print(ss.ttest_1samp(a = df.loc[:, 'Abra'], popmean = 15000))
# 示例输出
# Ttest_1sampResult(statistic=-1.1281738488299586,
pvalue=0.26270472069109496)
```

ttest_1samp 方法返回由两个数值组成的元组：

第一个数是 t 统计量，即 statistic：浮点或数组类型的 t 统计量。

第二个数则是相应的 p 值，即 pvalue：浮点或数组类型的双侧概率值。

从输出结果，可以看到 p 值约为 0.267，远大于 0.05，因此，没有充分的证据说明平均稻谷产量不是 15000。将这个检验应用到所有的变量，同样假设均值为 15000：

```
# 对所有变量进行 t 检验
print(ss.ttest_1samp(a = df, popmean = 15000))
# 示例输出
#Ttest_1sampResult(statistic=array([ -1.12817385,   1.07053437,
#-65.81425599,  -4.564575  ,   6.17156198]),
#pvalue=array([  2.62704721e-01,   2.87680340e-01,   4.15643528e-70,
#  1.83764399e-05,   2.82461897e-08]))
```

第一个数组是 t 统计量，第二个数组则是对应的 p 值。

6. 可视化

Python 中有许多可视化模块，最流行的当属 matpalotlib 库。使用 matplotlib 库中的箱线图模块对数据进行绘图：

```
# Import the module for plotting
import matplotlib.pyplot as plt
plt.show(df.plot(kind = 'box'))
```

得到的箱线图如图 1 所示。

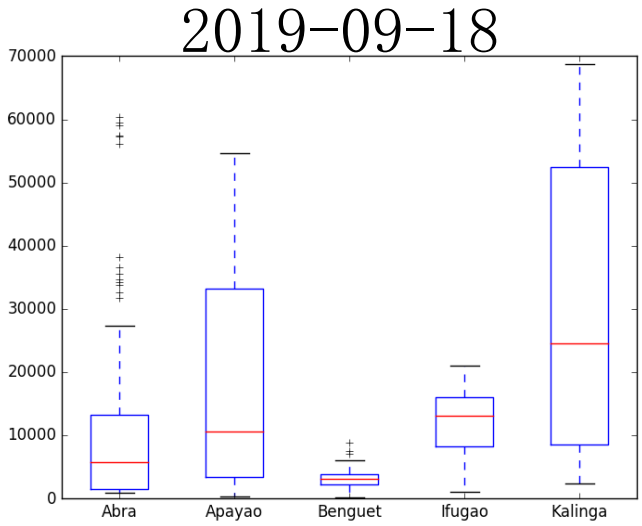


图 1 箱线图

可以用 pandas 模块中集成 R 的 ggplot 主题来美化图表，需要在上述代码中多加一行：

```
plt.style.use('ggplot') # 使用 ggplot 风格
plt.show(df.plot(kind = 'box'))
```

得到美化后的箱线图如图 2 所示。

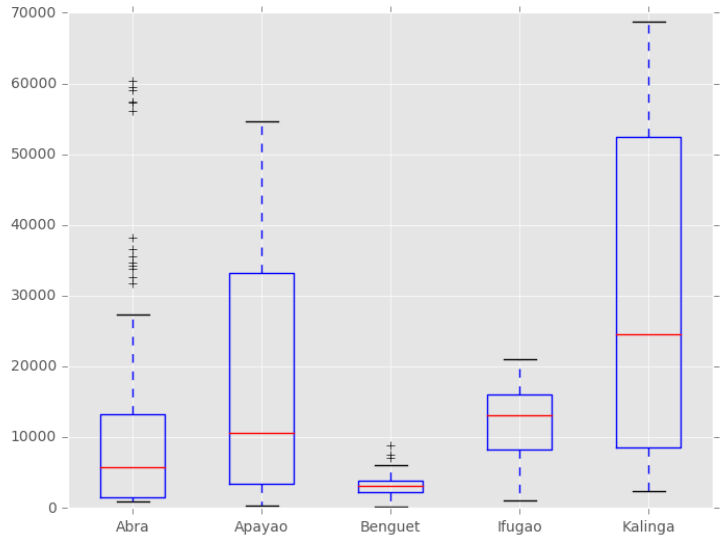


图 2 使用 gglpot 风格的箱线图

除了箱线图，还可以绘制其他统计图：

```
# 绘制均值的折线图，如图 3 所示
plt.show(df.mean().plot(kind = 'line'))
# 绘制均值的直方图，如图 4 所示
plt.show(df.mean().plot(kind = 'bar'))
```

2019-09-18

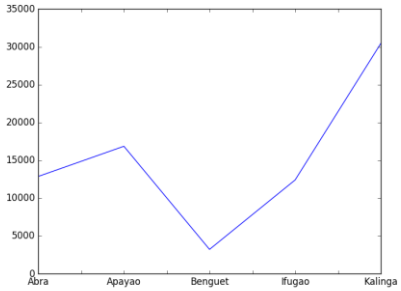


图 3 均值折线图

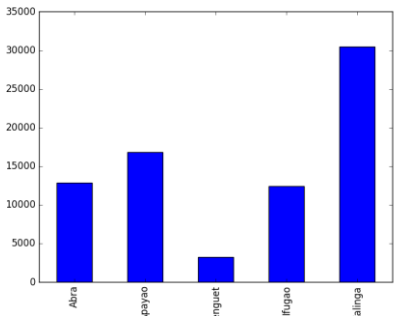


图 4 均值直方图

其中 ppf 方法返回累计分布函数反函数的值，mean(1)返回 y 轴上的均值。

7. 作业练习

（1）操作说明 pandas 数据缺失值处理：reindex()，dropna()，fillna()，isnull()。
（2）探索 pandas 中的可视化工具（折线图、饼状图等）对该数据集进行深入分析。
（3）将 df.DataFrame 转换成 Numpy 数据类型。
（4）探索 pandas 中分组及统计操作。利用 pandas 或 numpy 统计函数，计算 Anscombe 数据的统计值。

Anscombe's quartet (安斯库姆四重奏)

	dataset	x	y
0	I	10.0	8.04
1	I	8.0	6.95
2	I	13.0	7.58
3	I	9.0	8.81
4	I	11.0	8.33
5	I	14.0	9.96
6	I	6.0	7.24
7	I	4.0	4.26
8	I	12.0	10.84
9	I	7.0	4.82
10	I	5.0	5.68
11	II	10.0	9.14
12	II	8.0	8.14
13	II	13.0	8.74
14	II	9.0	8.77
15	II	11.0	9.26
16	II	14.0	8.10
17	II	6.0	6.13
18	II	4.0	3.10
19	II	12.0	9.13
20	II	7.0	7.26
21	II	5.0	4.74
22	III	10.0	7.46
23	III	8.0	6.77
24	III	13.0	12.74
25	III	9.0	7.11
26	III	11.0	7.81
27	III	14.0	8.84
28	III	6.0	6.08
29	III	4.0	5.39
30	III	12.0	8.15
31	III	7.0	6.42
32	III	5.0	5.73
33	IV	8.0	6.58
34	IV	8.0	5.76
35	IV	8.0	7.71
36	IV	8.0	8.84
37	IV	8.0	8.47
38	IV	8.0	7.04
39	IV	8.0	5.25
40	IV	19.0	12.50
41	IV	8.0	5.56
42	IV	8.0	7.91
43	IV	8.0	6.89

2019-09-18

参考链接：
https://pandas.pydata.org/pandas-docs/stable/getting_started/tutorials.html
https://pandas.pydata.org/pandas-docs/stable/user_guide/visualization.html