

# 数据分析作业--股票数据分析

## 一、作业过程及简单代码说明

1.读取数据，在列上仅保留：代码、简称，日期，开盘价(元)，收盘价(元)，成交金额(元)。在行上，删除包含空值的行。

```
# 读取数据
import pandas as pd

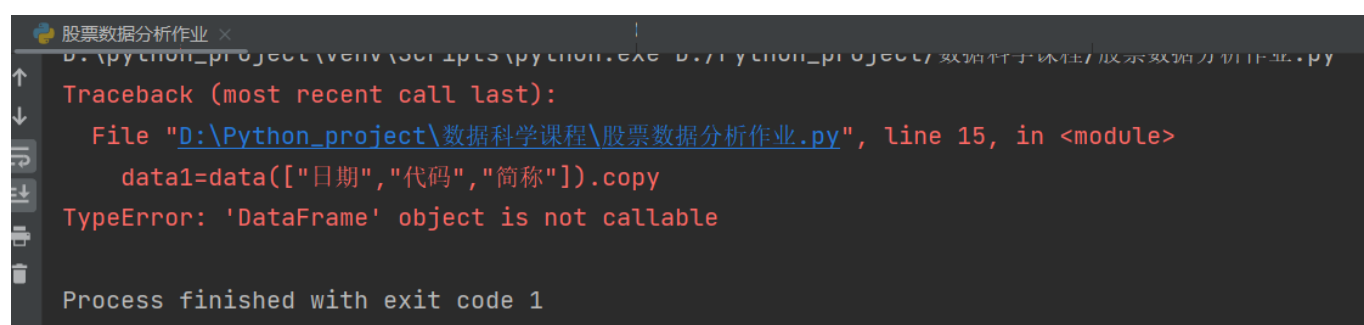
data = pd.read_csv("600009.SH.CSV", encoding="gbk", usecols=
["代码", "简称", "日期", "开盘价(元)", "收盘价(元)", "成交金额(元)"]) # 读取数据，并获取
指定的列
data = data.dropna(axis=0, how='any') # 删除空行
```

## 2.对数据进行汇总

根据不同的数据统计方法，将数据以日期为索引拆分成三个部分

```
data1 = data[["日期", "代码", "简称"]].copy() # 不用计算的部分
data2 = data[["日期", "开盘价(元)", "收盘价(元)"]].copy() # 计算平均值的部分
data3 = data[["日期", "成交金额(元)"]].copy() # 计算求和的部分
```

想要提取数据的一部分，却报错



```
股票数据分析作业 x
D:\python_project\venv\scripts\python.exe D:/python_project/数据科学课程/股票数据分析作业.py
Traceback (most recent call last):
  File "D:\Python_project\数据科学课程\股票数据分析作业.py", line 15, in <module>
    data1=data(['日期','代码','简称']).copy
TypeError: 'DataFrame' object is not callable

Process finished with exit code 1
```

后来发现data也是一个列表，将data1 = data(['日期', '代码', '简称']).copy()换成data1 = data[['日期', '代码', '简称']].copy()就成功了

```
股票数据分析作业 x
D:\python_project\venv\Scripts\python.exe D:/Python_project/数据科学课程/股票数据分析作业.py

      日期      代码      简称
0    1998-02-18  600009.SH  上海机场
1    1998-02-19  600009.SH  上海机场
2    1998-02-20  600009.SH  上海机场
3    1998-02-23  600009.SH  上海机场
4    1998-02-24  600009.SH  上海机场
...      ...      ...      ...
4435  2016-06-02  600009.SH  上海机场
4436  2016-06-03  600009.SH  上海机场
4437  2016-06-06  600009.SH  上海机场
4438  2016-06-07  600009.SH  上海机场
4439  2016-06-08  600009.SH  上海机场

[4393 rows x 3 columns]
```

去重失败，想要每个月只显示一次的效果，但是去重后只留下了一行

```
data1=data1.to_period('M')
data1=data1.drop_duplicates(keep='first')
print(data1)
```

```
D:\python_project\venv\Scripts\python.exe D:/Python_project/数据科学课程/股票数据分析作业

      代码      简称
日期
1998-02  600009.SH  上海机场

进程已结束,退出代码0
```

后来查网上的去重方法，发现一种方法很好，如下:loc函数提取要索取的数据，index按月份索引去重，前面的表示取反，效果是取data1.index.duplicated(keep='first')中的真值部分提取出来，打印出来的结果符合预期

```
data1 = data1.loc[~data1.index.duplicated(keep='first')] # 按月份索引去重
```

```
D:\python_project\venv\Scripts\python.exe D:/Python_project/数据科学课程/股票数据分析作业.py
```

```
代码    简称
```

```
日期
```

```
1998-02  600009.SH  上海机场
1998-03  600009.SH  上海机场
1998-04  600009.SH  上海机场
1998-05  600009.SH  上海机场
1998-06  600009.SH  上海机场
...
2016-02  600009.SH  上海机场
2016-03  600009.SH  上海机场
2016-04  600009.SH  上海机场
2016-05  600009.SH  上海机场
2016-06  600009.SH  上海机场
```

```
[221 rows x 2 columns]
```

```
进程已结束,退出代码0
```

对数据按月份进行处理，然后修改列名，合并数据

```
data1 = data1.to_period('M')
data1 = data1.loc[~data1.index.duplicated(keep='first')] # 按月份索引去重
data2 = data2.resample('M').mean().to_period('M') # 按月求平均值--平均开盘价（元），平均收盘价（元）
data3 = data3.resample('M').sum().to_period('M') # 按月求和--总成交金额(元)
# 修改列名
data2.rename(columns={"开盘价(元)": "平均开盘价（元）",
                     "收盘价(元)": "平均收盘价（元）"}, inplace=True)
data3.rename(columns={"成交金额(元)": "总成交金额（元）"}, inplace=True)
# 合并数据
data = pd.concat([data1, data2, data3], axis=1)
print(data)
```

合并数据后的效果，按月份分别有平均开盘价（元），平均收盘价（元），总成交金额（元），符合要求

```
股票数据分析作业 x
D:\python_project\venv\Scripts\python.exe D:/Python_project/数据科学课程/股票数据分析作业.py

      代码      简称      平均开盘价（元）      平均收盘价（元）      总成交金额（元）
日期
1998-02  600009.SH  上海机场  3.126062  3.081350  2.115369e+09
1998-03  600009.SH  上海机场  3.319895  3.331632  2.450726e+09
1998-04  600009.SH  上海机场  3.429950  3.426977  1.626719e+09
1998-05  600009.SH  上海机场  3.337050  3.330525  5.944312e+08
1998-06  600009.SH  上海机场  3.287232  3.278009  7.870084e+08
...      ...      ...      ...      ...      ...
2016-02  600009.SH  上海机场  27.020625  27.105625  3.110200e+09
2016-03  600009.SH  上海机场  28.779130  28.909130  4.371978e+09
2016-04  600009.SH  上海机场  28.826500  28.732000  3.004106e+09
2016-05  600009.SH  上海机场  27.623333  27.681429  3.112818e+09
2016-06  600009.SH  上海机场  27.658333  27.638333  6.540495e+08

[221 rows x 5 columns]

进程已结束,退出代码0
```

### 3.绘制图形

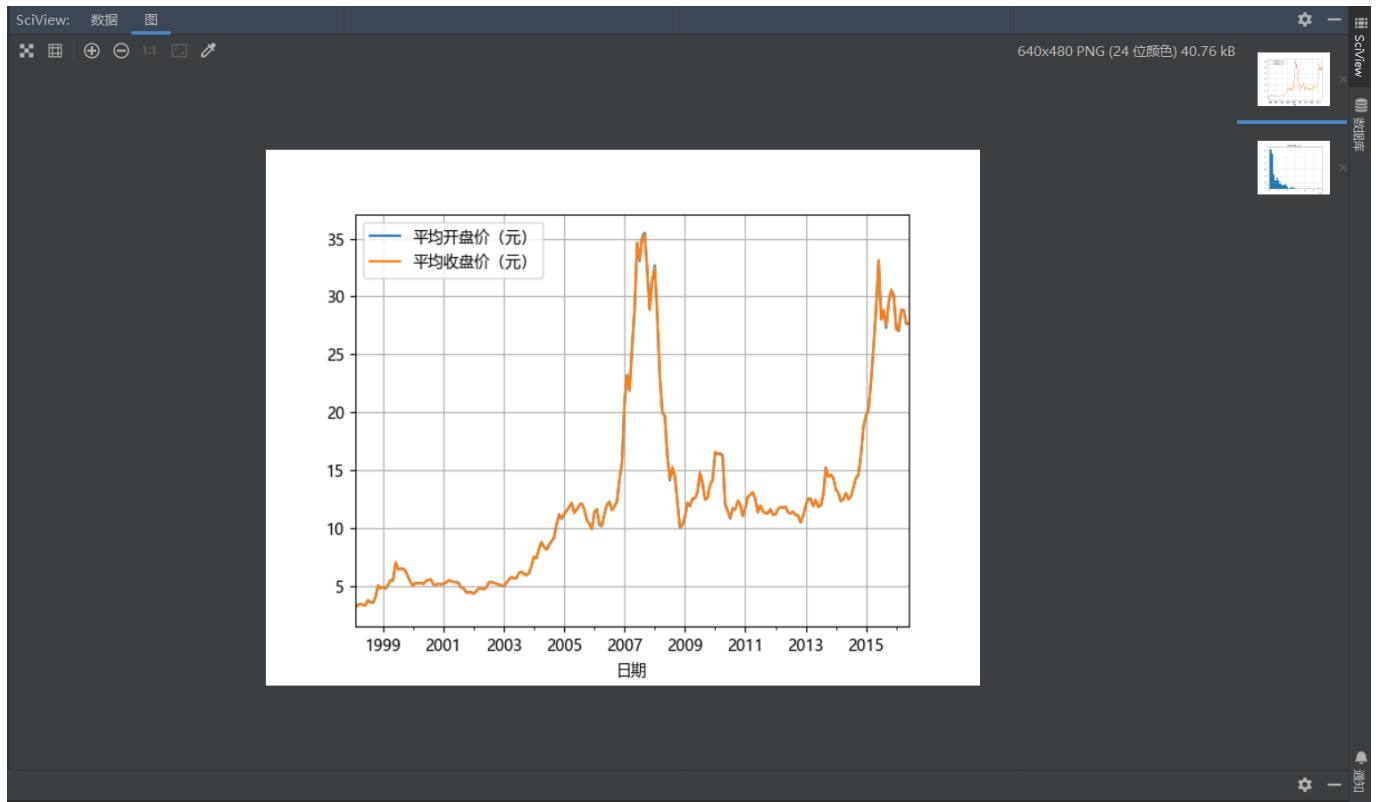
横轴为月份，纵轴为平均开盘价和平均收盘价，绘制图形

```
# 3_绘制图形
pic1 = data[["平均开盘价（元）", "平均收盘价（元）"]].copy()
# 横坐标是月份，纵坐标是股价，绘制平均开盘价（元）、平均收盘价（元）随月份的变化（两条曲线）。
pic1.plot(y=["平均开盘价（元）", "平均收盘价（元）"], grid=True)
plt.show()
```

以下两行代码为解决标题和图例不能正常显示的问题

```
mpl.rcParams['font.sans-serif'] = ['Microsoft YaHei']
mpl.rcParams['axes.unicode_minus'] = False
```

最终可视化效果如图



## 4.判断是否符合正太分布

取所有月份的总成交金额构成的一组数值，判定这组数值是否符合正态分布，

原假设H0：所有月份的总成交金额符合正态分布；

备择假设H1：所有月份的总成交金额不符合正态分布；

引入正态分布判断函数进行判断

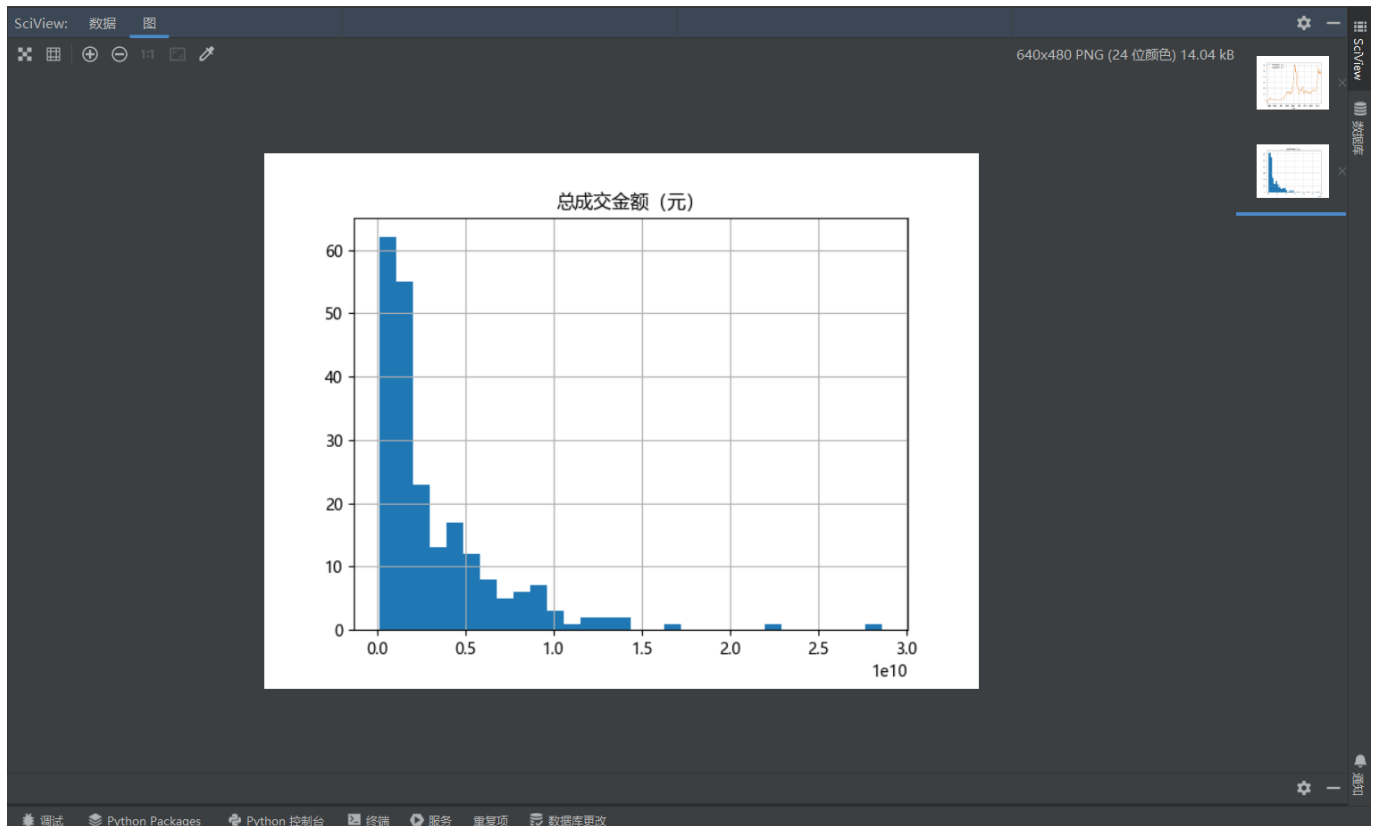
```
from scipy import stats
# 4_判定每个月总成交额是否符合正态分布
pic2 = data[["总成交金额 (元)"]].copy()
pic2.hist(bins=30) # 绘制直方图直观观察分布
plt.show()
print(stats.normaltest(pic2)) # 判断是否符合正态分布
```

结果如下

```
NormaltestResult(statistic=array([153.32302529]), pvalue=array([5.08543472e-34]))
```

pvalue的值远远小于显著性水平0.05，所以推翻原假设H0，所以不是正态分布

直方图如下，能更直观地观察分布情况



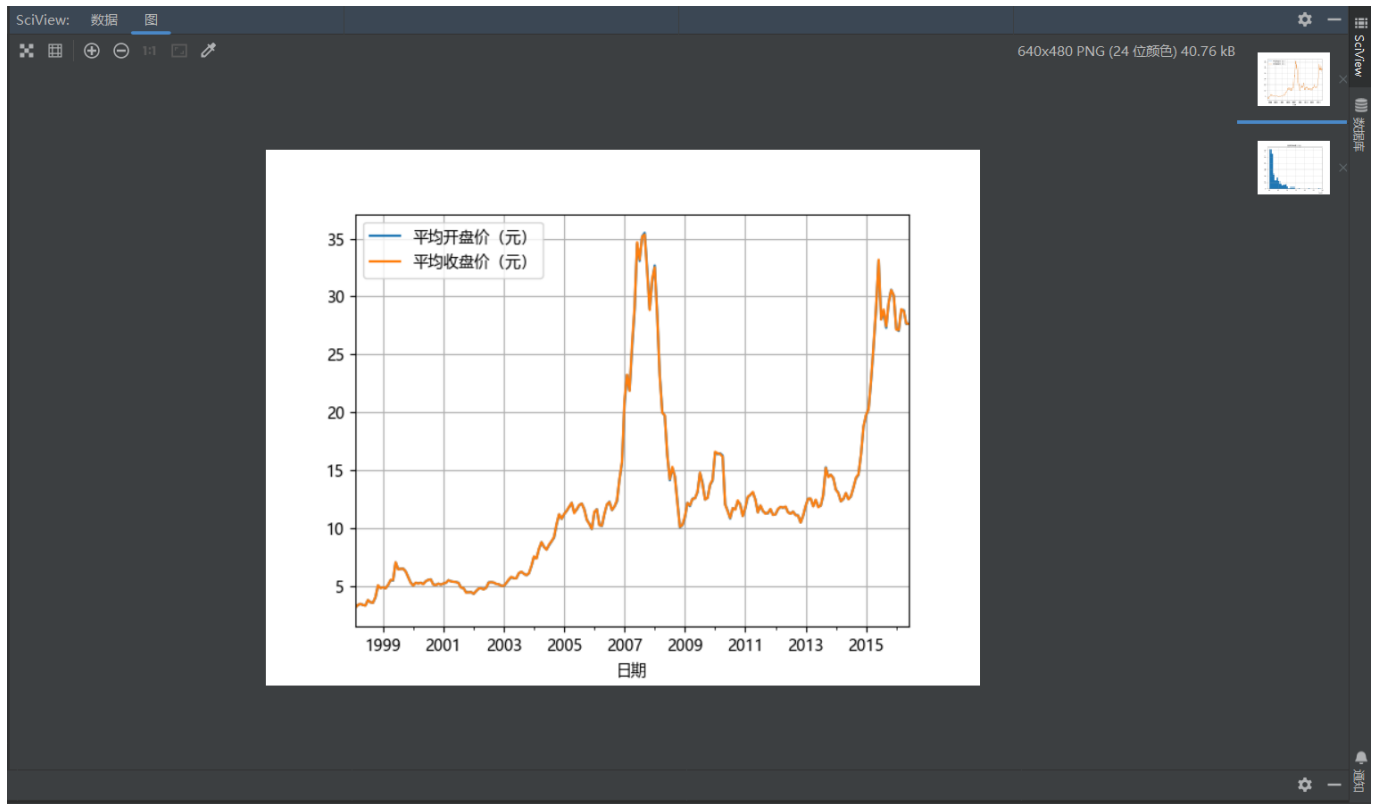
## 二、公式推导

正态分布检验的原理：

假设检验：基本思想是“小概率事件”原理，其统计推导方法是带有某种概率性质的反证法。小概率事件在一次试验中基本上不会发生。用反证法思想先提出检验假设，再用适当的统计方法，利用小概率原理，确定假设是否成立。即为了检验一个假设 $H_0$ 是否正确，然后根据样本对假设 $H_0$ 做出接受或拒绝的决策。如果样本观察值导致了“小概率事件”发生，就应拒绝假设 $H_0$ ，否则应接受假设 $H_0$ 。本次案例中，pvalue的值远远小于显著性水平0.05，所以推翻原假设 $H_0$ ，所以不是正态分布。

## 三、可视化结果

1.平均开盘价（元）、平均收盘价（元）随月份的变化（两条曲线）



## 2. 总成交金额 (元) 分布直方图

