数据分析技术 实验 1

16337090 黄家熙

一、

导入文件内容并选取感兴趣的数据，然后就是简单地实现公式。因为 numpy 中计算方差、标准差时是除以 n 而不是(n-1)，自己实现了两者的代码。

```
file_name = 'data_selected/000006.csv'

price = []
with open(file_name) as infile:
    reader = csv.reader(infile)
    header_row = next(reader)  # 去除表头
    for row in reader:  # 在读取时,输入数据的每一行都被解析并转换为字符串列表
        price.append((float(row[3])+float(row[4]))/2)
price = np.array(price, dtype=float)
```

均值 = 14.270976945902762
最小值 = 5.145
最大值 = 47.995
中值 = 12.825
下四分位点 = 9.985
上四分位点 = 16.785
极差 = 42.849999999999994
四分位极差 = 6.800000000000001
方差 = 43.483922721157775
标准差 = 6.594234051135717
变异系数 = 46.207306452337455%
偏度 = 1.8013931510673609
峰度 = 4.419173408952927

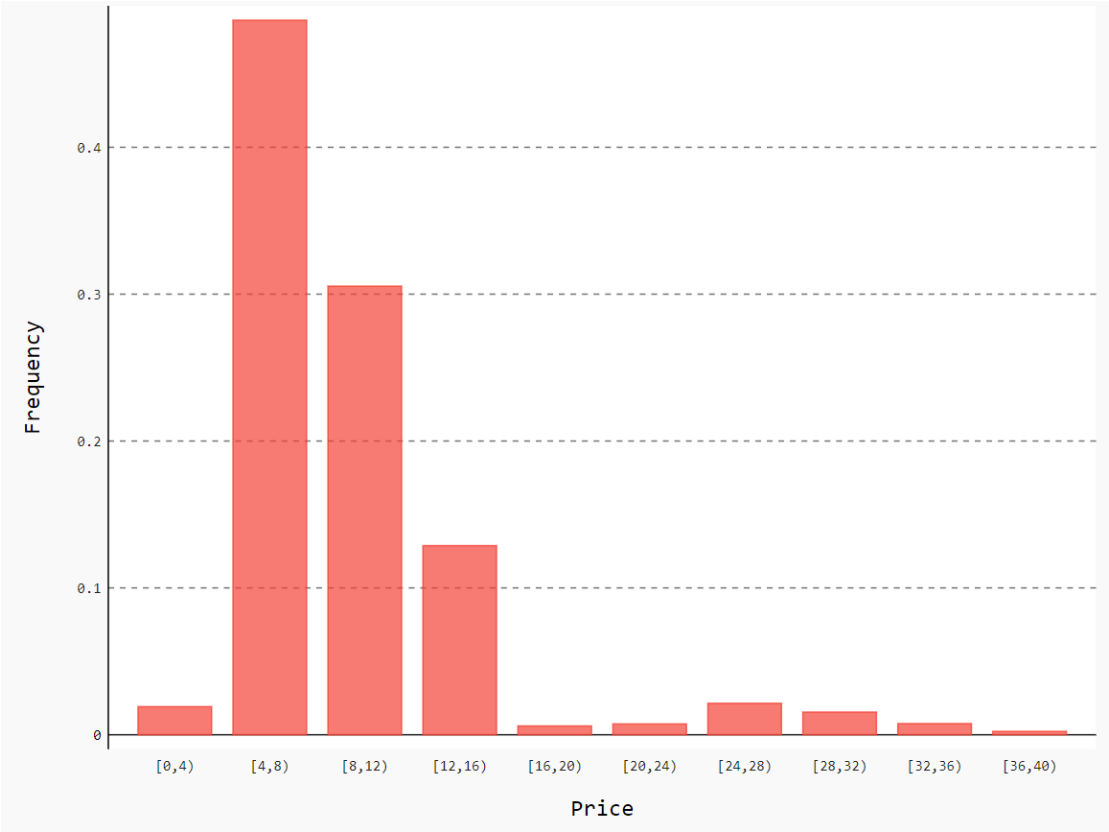
二、

本题难点在于画 QQ 图，如何计算标准正态分布函数的反函数 ϕ^{-1}（$\frac{i-0.375}{n+0.25}$）

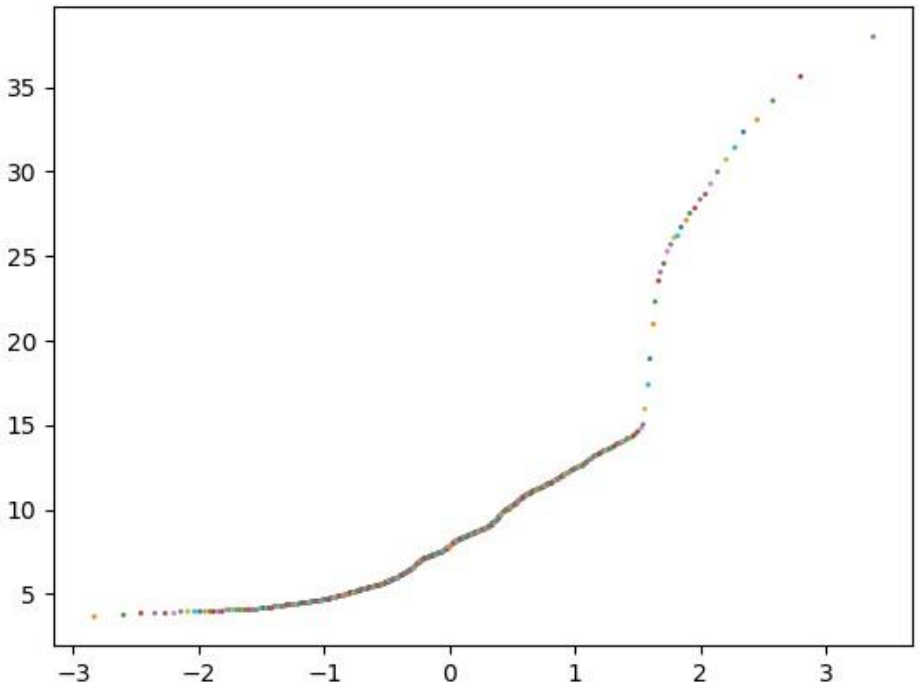
最终使用了 scipy.stat.norm，调用 norm.ppf，输入分位点，即可返回对应的样本值 x

```
norm.ppf((i-0.375)/(n+0.25)
```

2.1 股价分析



股价直方图

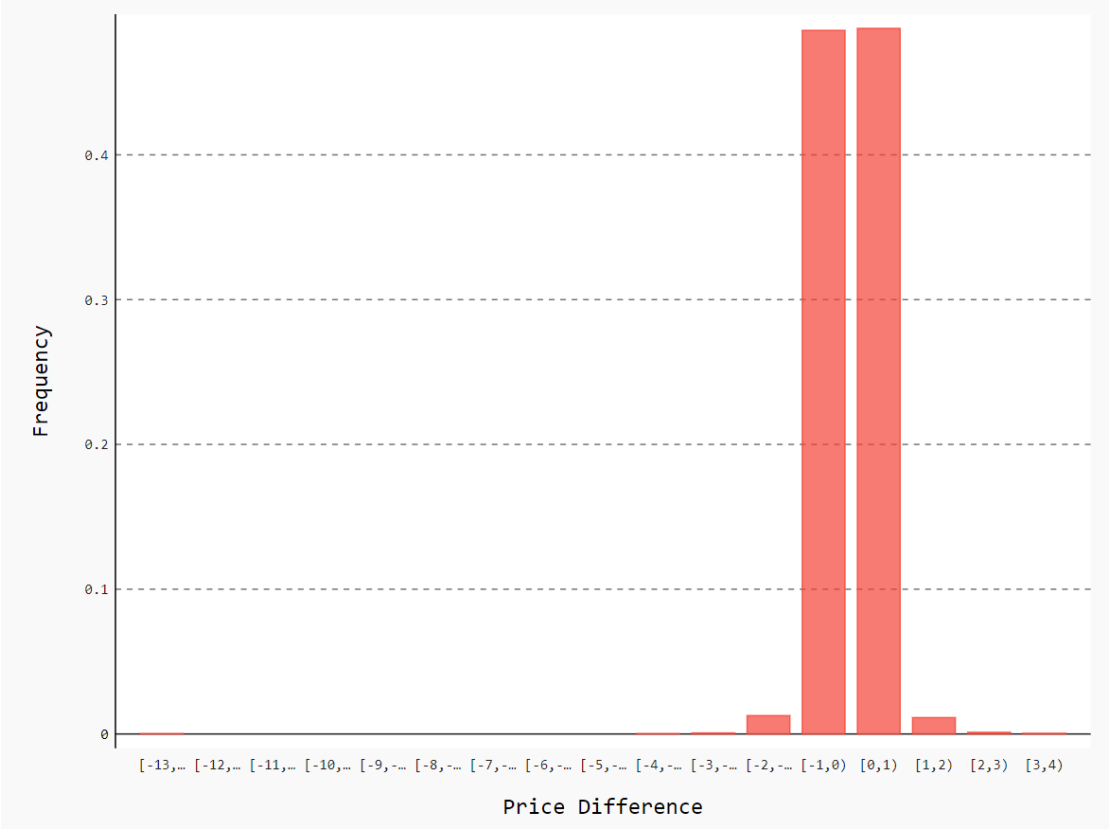


股价 QQ 图

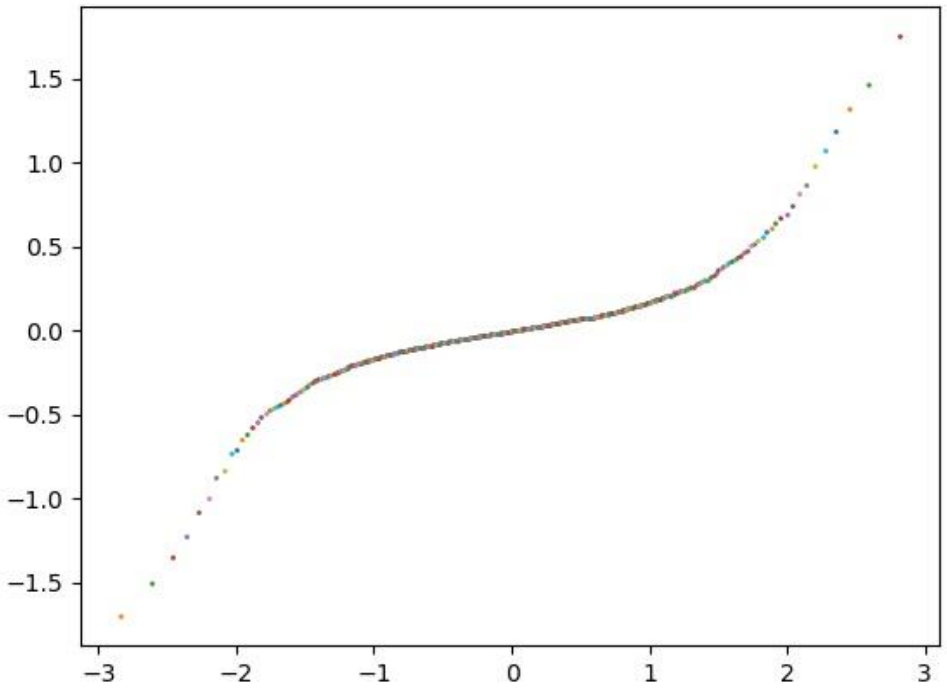
分析：不是正态分布，看 QQ 图，不是直线，尤其股价大于 15 时散点偏离直线很多，再看直方图，分布的尾部在右侧，是正偏的，偏度应该大于 0，计算的偏度也符合分析。

偏度 = 2.352125989978969

2.2 差值分析



差值直方图



差值 QQ 图

分析：不是正态分布，看 QQ 图，(-1,1)部分还可以用直线拟合，但是两端就偏离直线太远了，再看直方图，计算偏度小于 0，应该是因为最左侧[-13,-12)内出现了一个偏差值，导致偏度较小。

偏度 = -8.343887454884397

三、

计算 000012 股票的股价和成交量间的相关性

```
if __name__ == "__main__":
    file_name = 'data_selected/000012.csv'
    sample = load_data(file_name)

    cov_mat, pearson_mat = compute_pearson(sample)
    print("covariance = \n", cov_mat)
    print("Pearson = \n", pearson_mat)

    spearman_mat = compute_spearman(sample)
    print("Spearman = \n", spearman_mat)
```

计算结果如下，

协方差矩阵 = [[2.77955953e+01 3.25620655e+04]
[3.25620655e+04 4.29093191e+10]]

Pearson 相关系数 = [[1. 0.02981592]
[0.02981592 1.]]

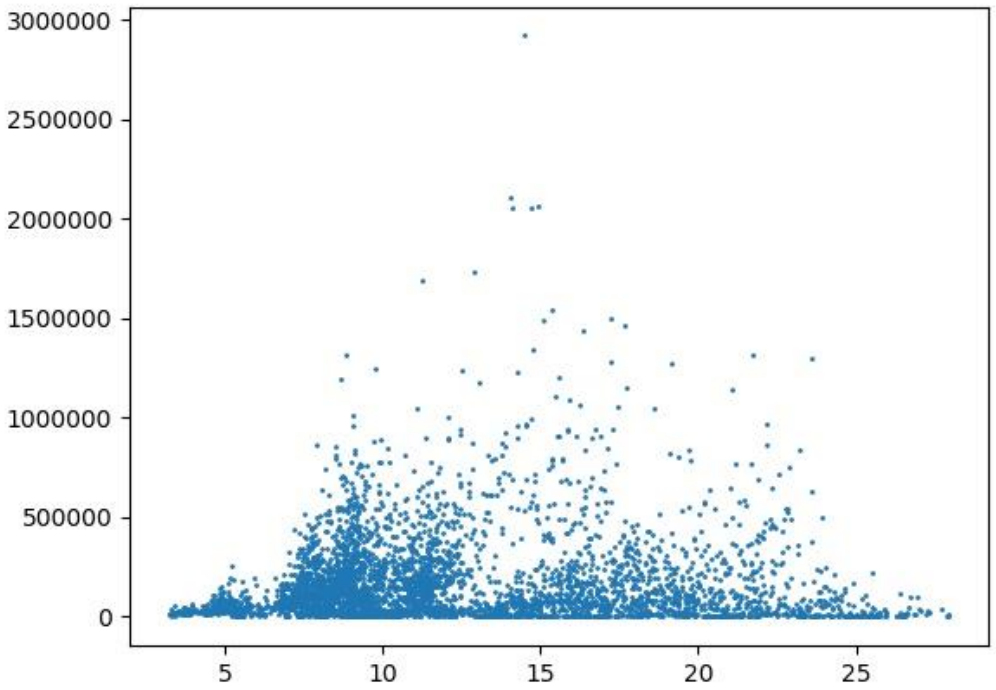
Spearson 相关系数 = [[1. -0.01869784]
[-0.01869784 1.]]

所以

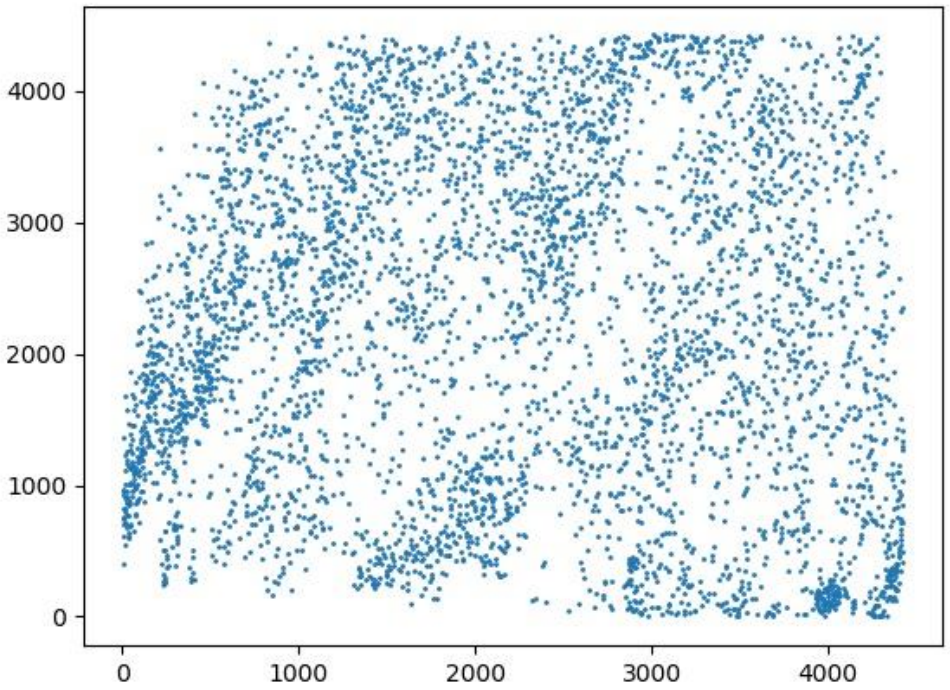
Pearson 相关系数 = 0.02981592

Spearman 相关系数 = -0.01869784

相关系数十分接近 0，说明两者没有什么线性关系，画出散点图直观感受两者的关系，



Pearson 相关性：横轴—股价，纵轴—成交量



Spearman 相关性：横轴—股价，纵轴—成交量，二者为次序统计量

四、

4.2

000001 和 000006 号股票的相关分析:

```
if __name__ == "__main__":
    fname1 = 'data_selected/000001.csv'
    fname2 = 'data_selected/000006.csv'
    price_pair = load_data(fname1, fname2)

    pearson = compute_pearson(price_pair)
    print("Pearson 相关系数 = \n", pearson)
    spearman = compute_spearman(price_pair)
    print("Spearman 相关系数 = \n", spearman)
```

函数的作用就如同名字，计算结果如下，

Pearson 相关系数 = 0.7771936543357362

Spearman 相关系数 = 0.5099077677958287

4.2

本题第二部分需要计算 100 支股票间的相关系数，而且因为两支股票的股价只取具有相同日期的，所以样本数量可能会因选择的两只股票不同而不同，所以只能分别两两计算，考虑到效率会因为数据量太大而不足，所以不使用自己写的代码计算 pearson 和 spearman 相关系数，而是调用 scipy.stat 的 pearsonr 和 spearmanr 函数。

```
if __name__ == "__main__":
    file_name_set = glob.glob('data_selected/*.csv')  # 读入所有csv文件
    n = len(file_name_set)
    Pearson_mat = np.ones((n, n), dtype=float)
    pp_mat = np.ones((n, n), dtype=float)
    Spearman_mat = np.ones((n, n), dtype=float)
    sp_mat = np.ones((n, n), dtype=float)
    for i in range(n):
        for j in range(i+1, n):
            print("i=", i, ", j=", j)
            # 选取相同日期的股价,计算相关系数并返回
            pearson, pp, spearman, sp = analysis(
                file_name_set[i], file_name_set[j])
            Pearson_mat[i, j] = pearson
            Pearson_mat[j, i] = pearson
            pp_mat[i, j] = pp
            pp_mat[j, i] = pp
            Spearman_mat[i, j] = spearman
            Spearman_mat[j, i] = spearman
            sp_mat[i, j] = sp
            sp_mat[j, i] = sp

    # 计算相关性最强和最弱的5对股票
    min_pair1, max_pair1 = max_min(Pearson_mat, n)
    min_pair2, max_pair2 = max_min(Spearman_mat, n)
```

100 支股票的相关性分析结果：

表头：

股票 1　股票 2　相关系数　p 值

零假设 H_0：股票 1 和股票 2 之间没有相关性

Pearson 相关性最弱

000632　000525　-0.0002854860231784521　0.9849317585223375

000090　000049　0.0006476817190810115　0.9662156046568375

000661　000521　-0.0008275803432526166　0.9564784342147806

000661　000601　0.0010438976200018892　0.9452637333434777

000425　000036　-0.001680314485977578　0.9126746287567745

Pearson 相关性最强

000069　000046　0.9494846389143637　0.0

000046 000006 0.9464481049611131 0.0
000567 000025 0.927051782130761 0.0
000069 000006 0.9150686814687826 0.0
000708 000059 0.906159711693867 0.0

Spearman 相关性最弱
000667 000567 -0.00036783203180637455 0.9805995239980376
000538 000078 -0.00039880469825196466 0.9792253172494055
000088 000001 -0.0006760491563398693 0.9648234758690551
000667 000418 0.0007418725124183771 0.9611931821137563
000544 000425 -0.0013241252559623492 0.9313191683717976

Spearman 相关性最强
000661 000028 0.9615927289440876 0.0
000567 000025 0.9431301023573374 0.0
000418 000025 0.9360251004541741 0.0
000661 000423 0.931103078545676 0.0
000702 000421 0.9308496459665726 0.0