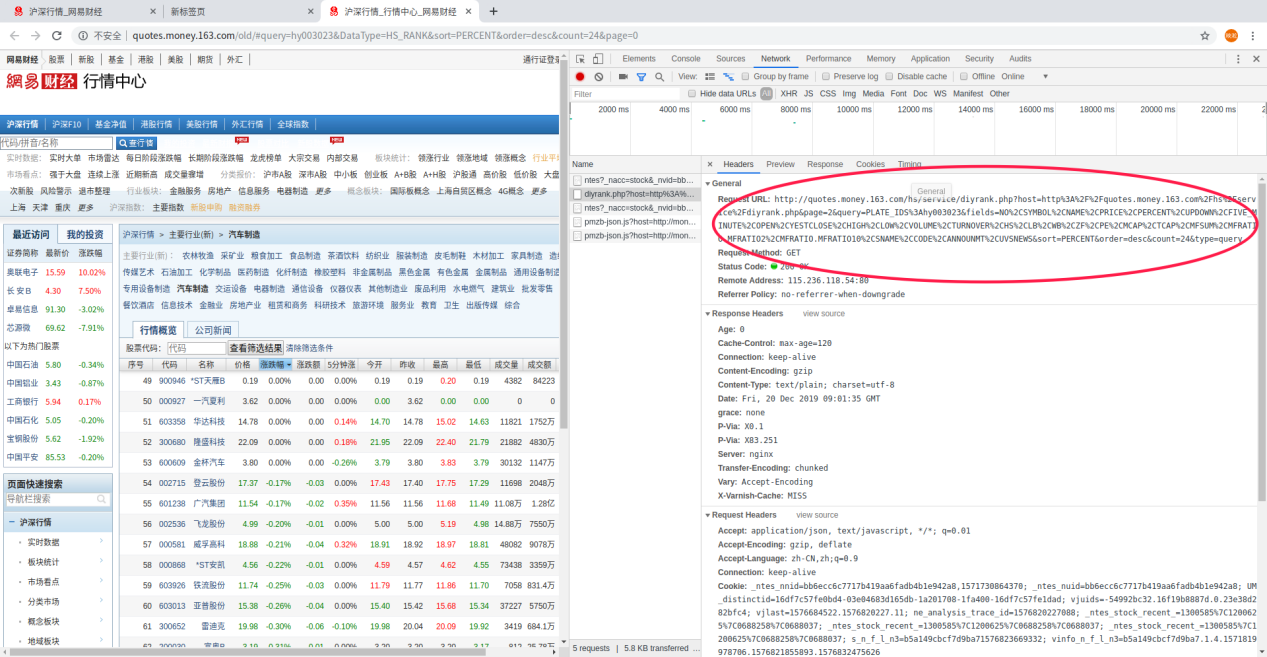
R语言大作业第二题

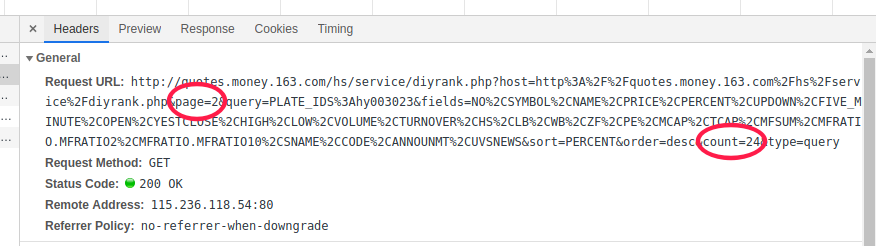
# 获取数据

由于数据在网站上，并且数量过多，手动下载工作量大，所以使用python语言写了脚本用来获取数据。

Step1：通过F12获得网站ajax截取的数据包地址



然后发现规律

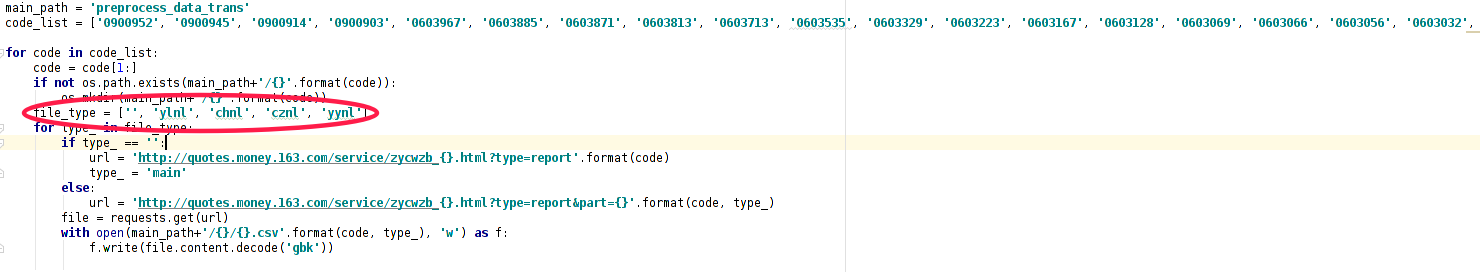


Page表示该板块的第几页，count表示每页的个股数量。

然后使用python中的request模块去迭代发送get请求，获得数据包。

然后使用json去读取板块下个股的CODE（股票编码）。

Step2： 通过观察五个数据的下载地址，我们又发现了其中的规律



下载网址是由CODE和type构成的，只要修改其中的CODE和type，我们就可以下载每一个个股下所有的数据。

# 数据处理

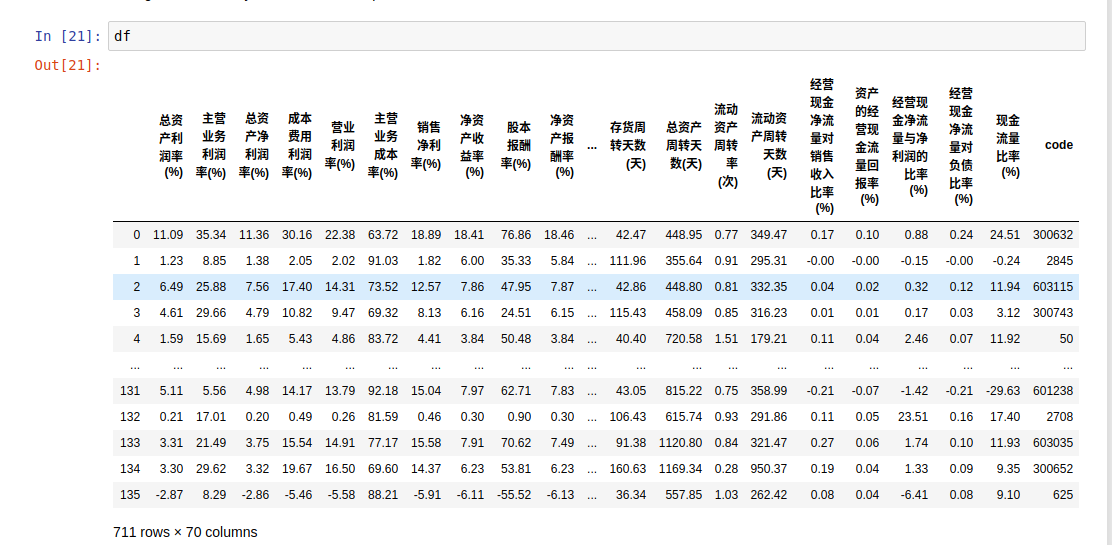
Step1：我们先把每一个CODE下的四个csv表组合成一张表。

Step2：在组合的过程中出现空缺值，我们的处理办法是取最近日期的值填补，因为表历史悠久，拿近十年的平均值不能较好的反应公司的运营情况，取最近的值更有可信度。如果近十年都没有数据，则取的是整个板块的平均值。

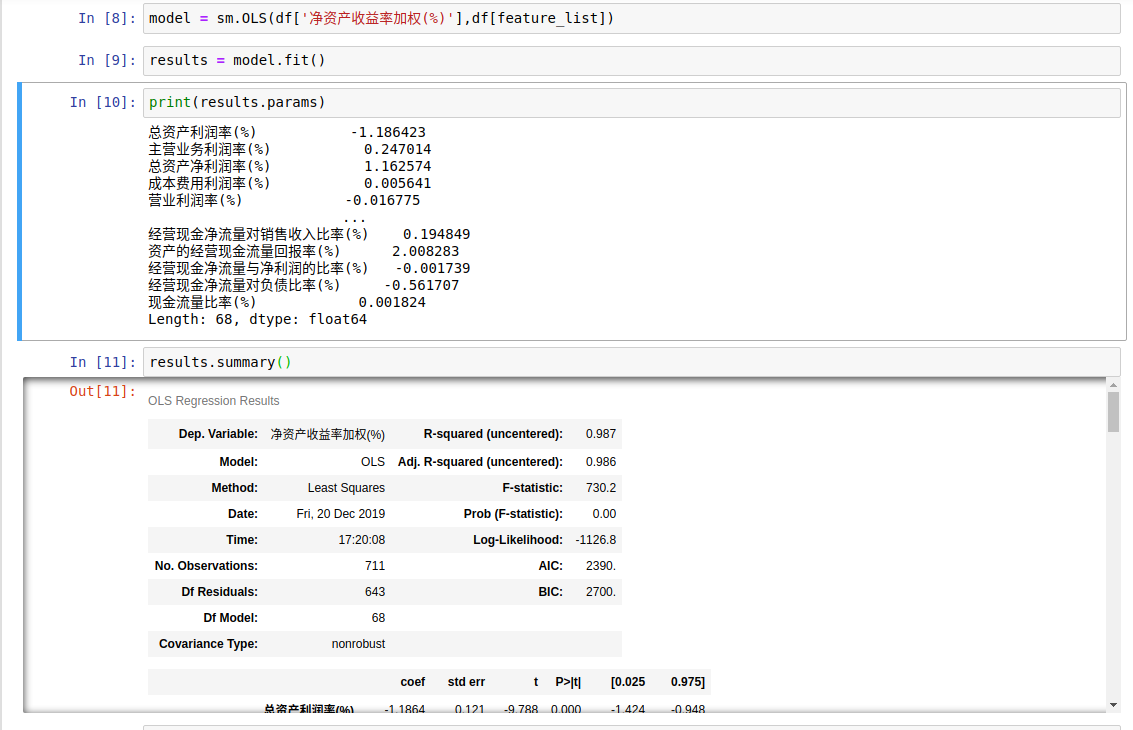
# **分析任务一：**

## 第一题：应用线性回归，预测净资产收益率加权 (%)

Step1：先用pandas模块整合五个板块。

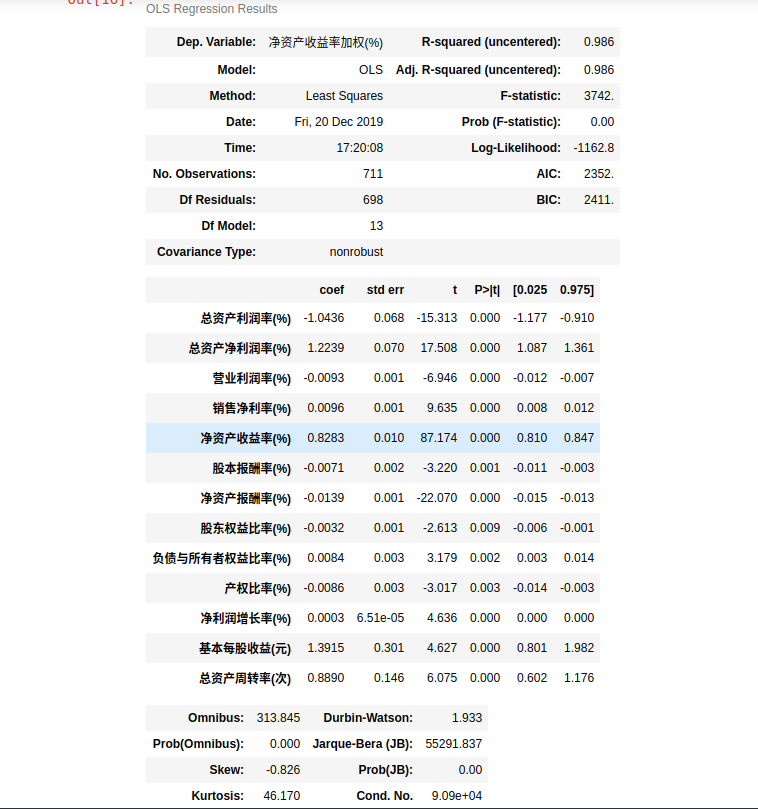


Step2：模型建立



Step3：模型优化

我们把模型中的因变量pvalue>0.05的特征去除，循环两轮。

最终得到

两轮后，特征中不存在pvalue>0.05的值

Step4：模型分析

决定系数和矫正决定系数都为0.986，证明模型拟合度较好。

原假设：该模型不显著存在线性关系

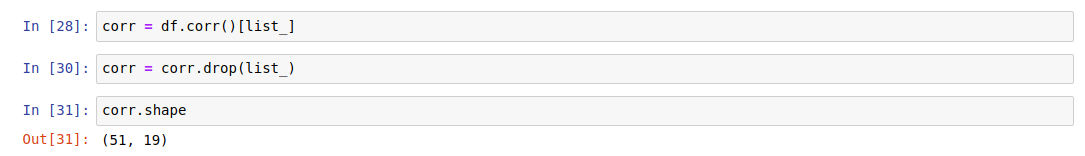
替代假设：该模型显著存在线性关系

方程：

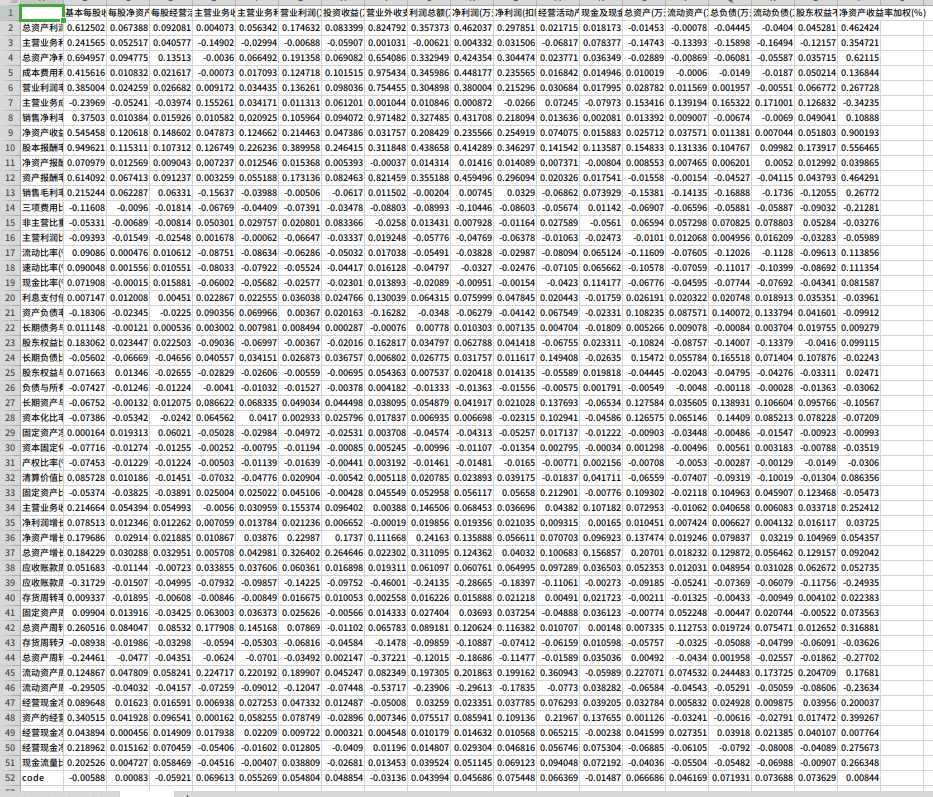


如图所示，可得线性回归方程。

## **第二题 应用典型相关分析，分析主要财务指标族与其他自变量之间的关系**



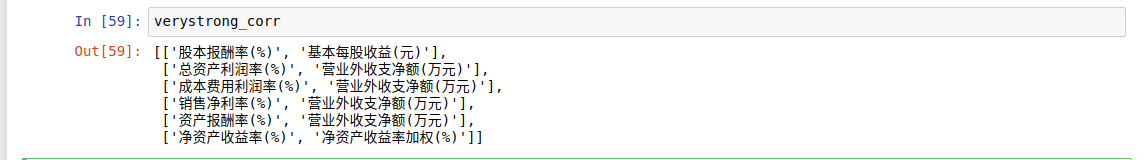
先取出主要财务指标中的特征，然后把列特征替换为主要财务指标，再把行特征中的主要财务指标剔除掉，得到结果。保存在csv文件中如下。



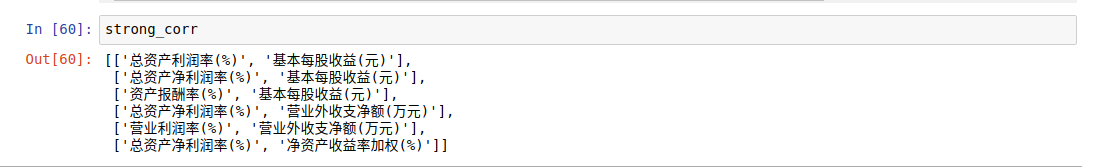
结论：利用百度中的等级解释对得到其中的相关系数进行等级划分。



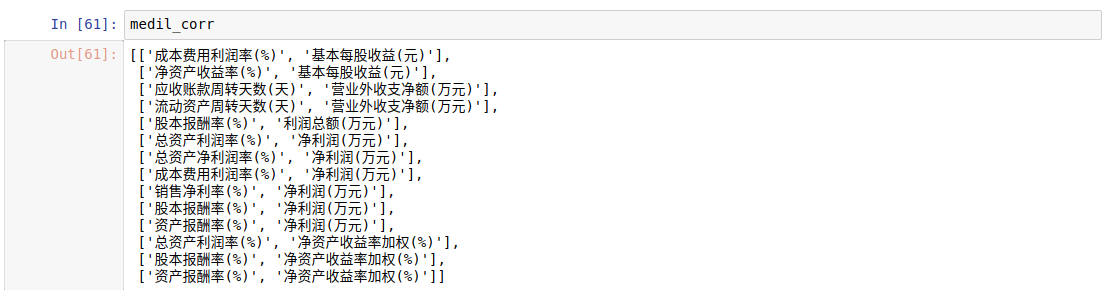
级强相关



强相关



中等相关：



弱相关：

[['主营业务利润率(%)', '基本每股收益(元)'],

['营业利润率(%)', '基本每股收益(元)'],

['主营业务成本率(%)', '基本每股收益(元)'],

['销售净利率(%)', '基本每股收益(元)'],

['销售毛利率(%)', '基本每股收益(元)'],

['主营业务收入增长率(%)', '基本每股收益(元)'],

['应收账款周转天数(天)', '基本每股收益(元)'],

['总资产周转率(次)', '基本每股收益(元)'],

['总资产周转天数(天)', '基本每股收益(元)'],

['流动资产周转天数(天)', '基本每股收益(元)'],

['资产的经营现金流量回报率(%)', '基本每股收益(元)'],

['经营现金净流量对负债比率(%)', '基本每股收益(元)'],

['现金流量比率(%)', '基本每股收益(元)'],

['流动资产周转率(次)', '主营业务收入(万元)'],

['股本报酬率(%)', '主营业务利润(万元)'],

['流动资产周转率(次)', '主营业务利润(万元)'],

['净资产收益率(%)', '营业利润(万元)'],

['股本报酬率(%)', '营业利润(万元)'],

['净资产增长率(%)', '营业利润(万元)'],

['总资产增长率(%)', '营业利润(万元)'],

['股本报酬率(%)', '投资收益(万元)'],

['总资产增长率(%)', '投资收益(万元)'],

['股本报酬率(%)', '营业外收支净额(万元)'],

['总资产周转天数(天)', '营业外收支净额(万元)'],

['总资产利润率(%)', '利润总额(万元)'],

['总资产净利润率(%)', '利润总额(万元)'],

['成本费用利润率(%)', '利润总额(万元)'],

['营业利润率(%)', '利润总额(万元)'],

['销售净利率(%)', '利润总额(万元)'],

['净资产收益率(%)', '利润总额(万元)'],

['资产报酬率(%)', '利润总额(万元)'],

['净资产增长率(%)', '利润总额(万元)'],

['总资产增长率(%)', '利润总额(万元)'],

['应收账款周转天数(天)', '利润总额(万元)'],

['流动资产周转天数(天)', '利润总额(万元)'],

['营业利润率(%)', '净利润(万元)'],

['净资产收益率(%)', '净利润(万元)'],

['应收账款周转天数(天)', '净利润(万元)'],

['流动资产周转率(次)', '净利润(万元)'],

['流动资产周转天数(天)', '净利润(万元)'],

['总资产利润率(%)', '净利润(扣除非经常性损益后)(万元)'],

['总资产净利润率(%)', '净利润(扣除非经常性损益后)(万元)'],

['成本费用利润率(%)', '净利润(扣除非经常性损益后)(万元)'],

['营业利润率(%)', '净利润(扣除非经常性损益后)(万元)'],

['销售净利率(%)', '净利润(扣除非经常性损益后)(万元)'],

['净资产收益率(%)', '净利润(扣除非经常性损益后)(万元)'],

['股本报酬率(%)', '净利润(扣除非经常性损益后)(万元)'],

['资产报酬率(%)', '净利润(扣除非经常性损益后)(万元)'],

['固定资产比重(%)', '经营活动产生的现金流量净额(万元)'],

['流动资产周转率(次)', '经营活动产生的现金流量净额(万元)'],

['资产的经营现金流量回报率(%)', '经营活动产生的现金流量净额(万元)'],

['总资产增长率(%)', '总资产(万元)'],

['流动资产周转率(次)', '总资产(万元)'],

['流动资产周转率(次)', '总负债(万元)'],

['流动资产周转率(次)', '股东权益不含少数股东权益(万元)'],

['主营业务利润率(%)', '净资产收益率加权(%)'],

['营业利润率(%)', '净资产收益率加权(%)'],

['主营业务成本率(%)', '净资产收益率加权(%)'],

['销售毛利率(%)', '净资产收益率加权(%)'],

['三项费用比重(%)', '净资产收益率加权(%)'],

['主营业务收入增长率(%)', '净资产收益率加权(%)'],

['应收账款周转天数(天)', '净资产收益率加权(%)'],

['总资产周转率(次)', '净资产收益率加权(%)'],

['总资产周转天数(天)', '净资产收益率加权(%)'],

['流动资产周转天数(天)', '净资产收益率加权(%)'],

['经营现金净流量对销售收入比率(%)', '净资产收益率加权(%)'],

['资产的经营现金流量回报率(%)', '净资产收益率加权(%)'],

['经营现金净流量对负债比率(%)', '净资产收益率加权(%)'],

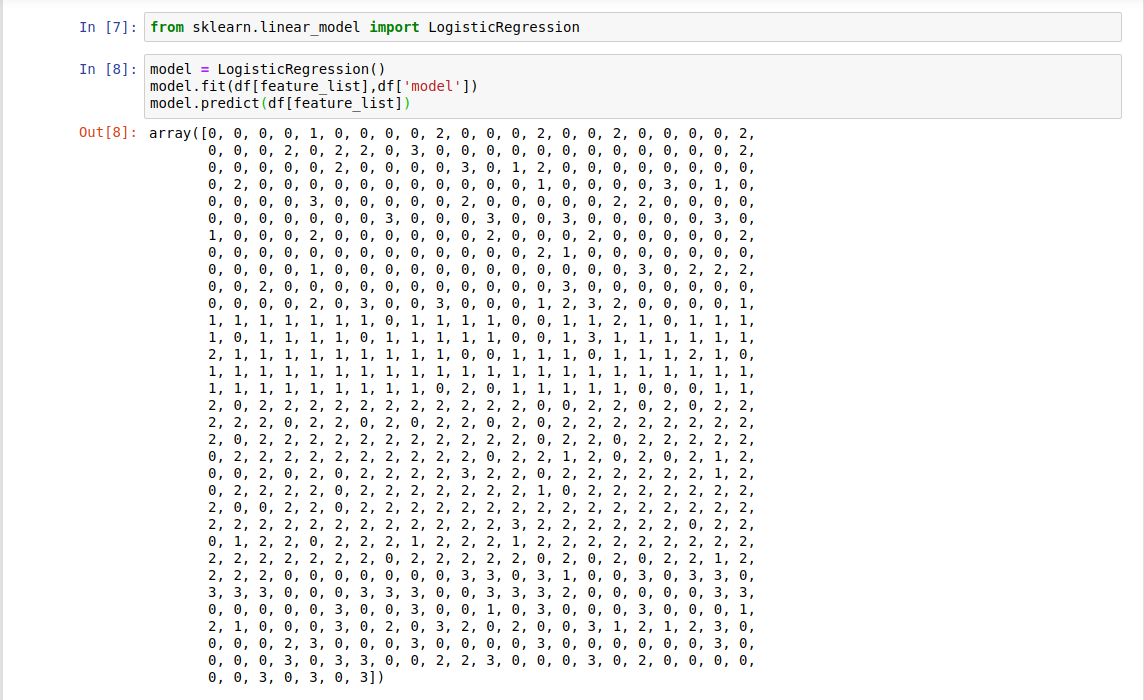
['现金流量比率(%)', '净资产收益率加权(%)']]

其余都是及弱相关或不相关。

## **第三题 应用 Logistic 回归，预测个股所在**

对板块进行标注划分出四个不同的板块，本想建立四个板块，但是在使用statesmodel的时候遇到其中两个模块的模型fit后会报出LinAlgError("Singular matrix")的error，所以我们只好换一种方法，使用sklearn中的逻辑回归函数，里面自动帮我们进行了模块的区分。

预测个股所在板块：

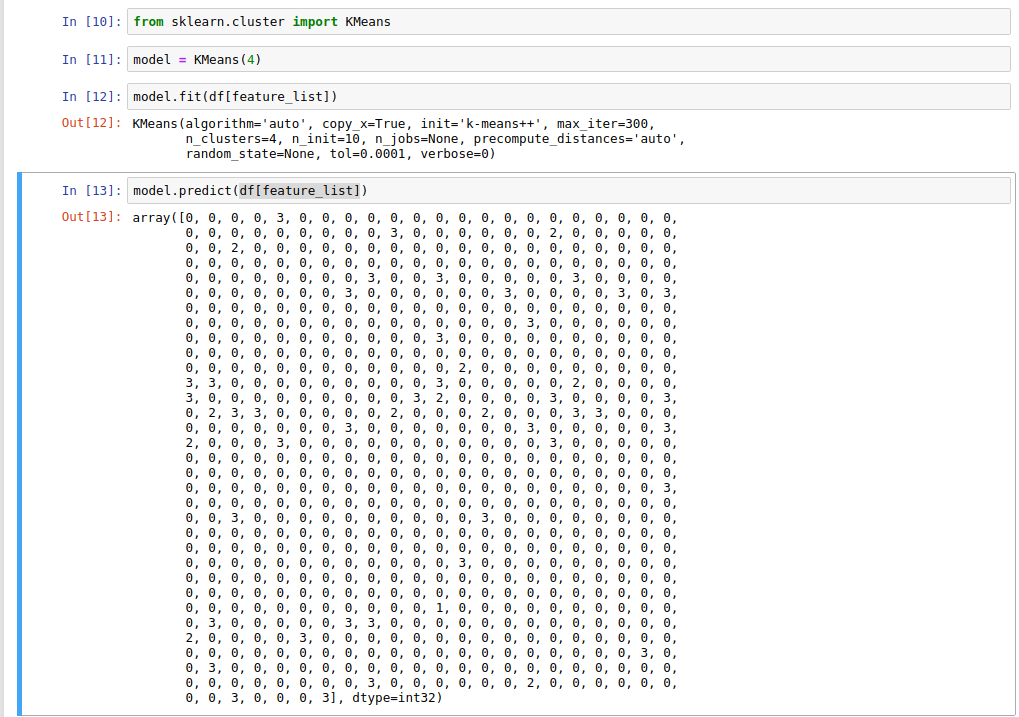


模型评估：



## **第四题：聚类分析以及评估**

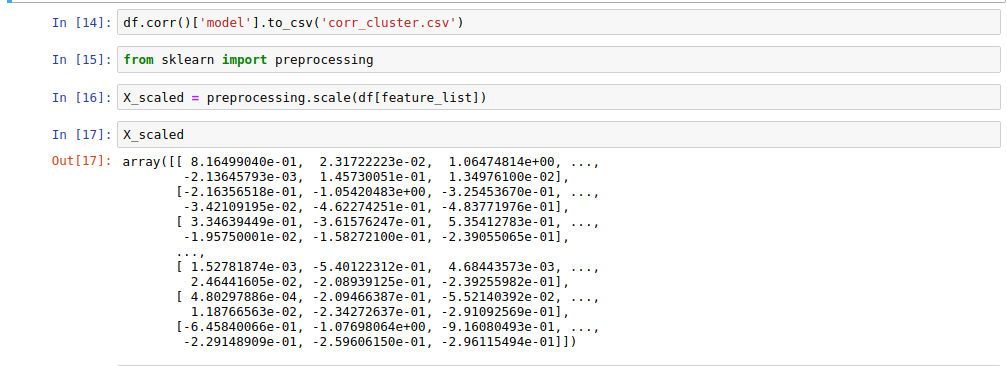
**4.1 应用聚类分析，对个股所在板块进行聚类，聚为 4 类**



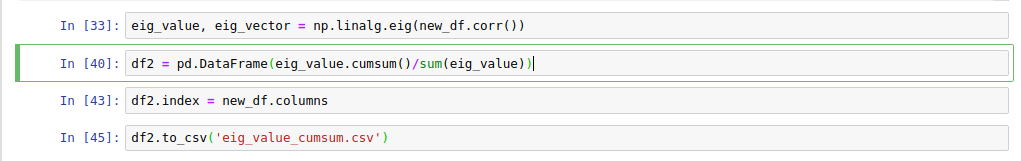
使用sklearn中的KMeans函数进行聚类，可以看出效果不是很好，第一类太多。

**4.2应用主成分分析，重新进行聚类分析**

Step1：使用scale函数进行归一化处理



Step2：计算特征值和特征向量

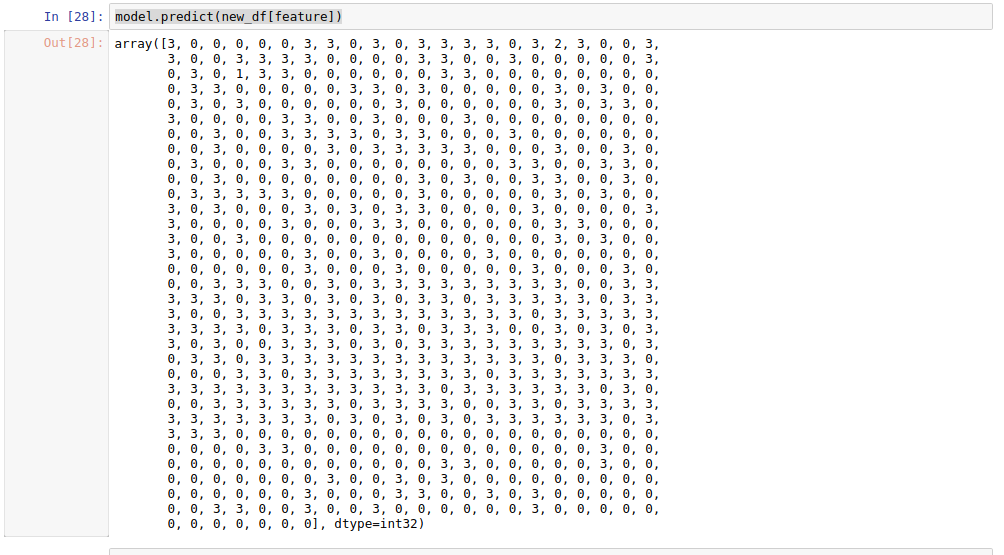


Step3：进行主成分分析，我们决定把影响结果90%的特征保留下来。

即从第2行到第27行的特征。

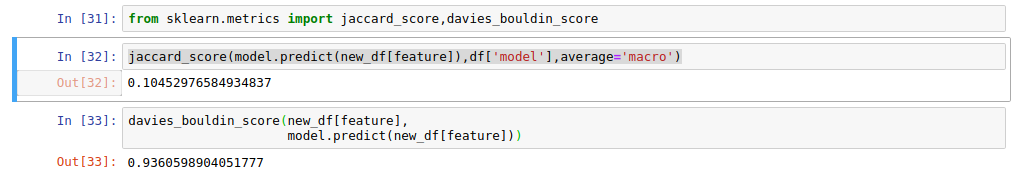


Step4:重新进行聚类查看结果



发现聚类效果还是比较差

**4.3 Jaccard系数和DBI系数如下：**



模型的聚类效果很不好。

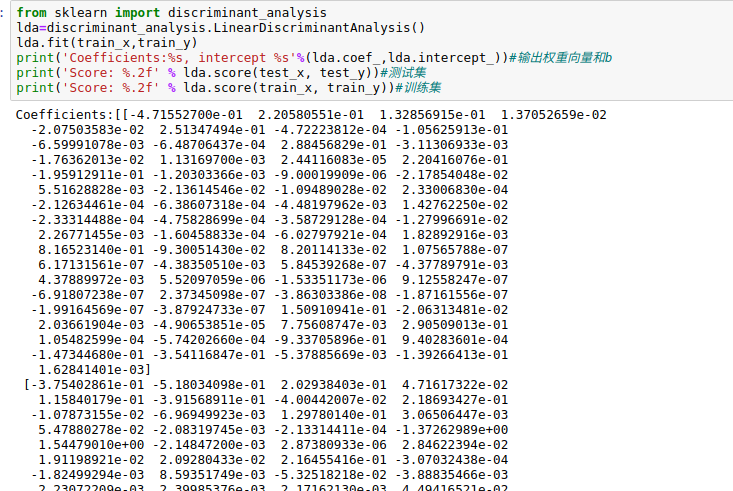
**第五题：分类分析以及评估：**

Step1：区分测试集和训练集，使用sklearn中的函数



Step2：应用判别分析预测个股所在板块。

采取线性判别分析



分别得到测试集和训练集的准确率



Step3：进行主成分分析之后 ，再进行判别分析

截取到占比99%的特征。

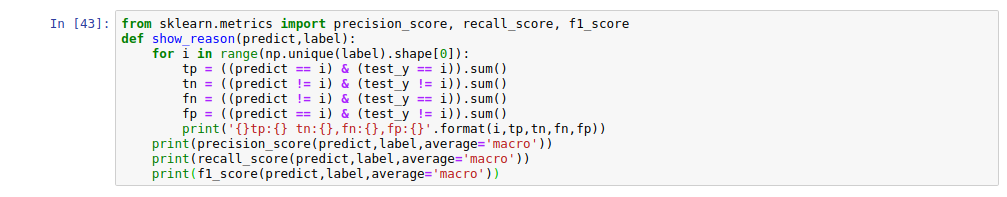
测试集准确率上升，模型泛化能力增强。



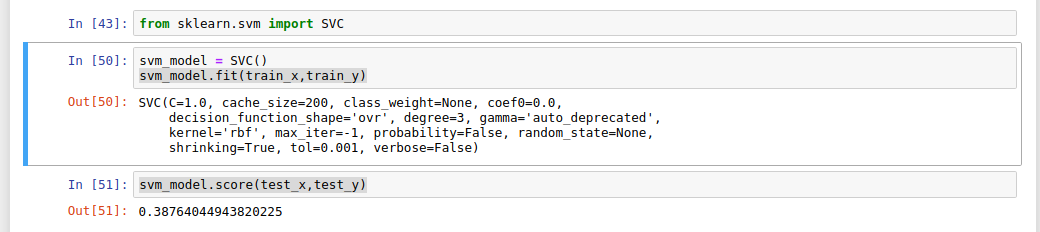
Step：4

运用四种算法进行分类：

其中，我们自己定义计算TP，FP，TN，FN，Precosion，Recall 以及 F1 值的函数如下：

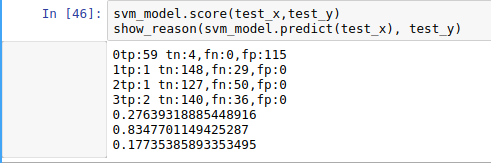


SVM：

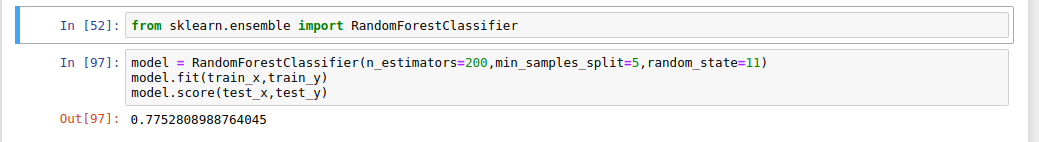


结论：模型效果较差

其中的TP，FP，TN，FN，Precosion，Recall 以及 F1 值如下：

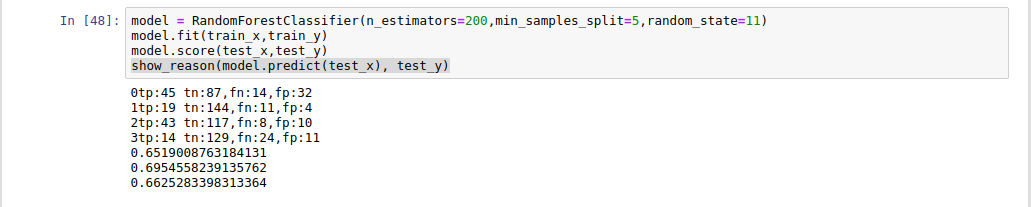


随机森林：

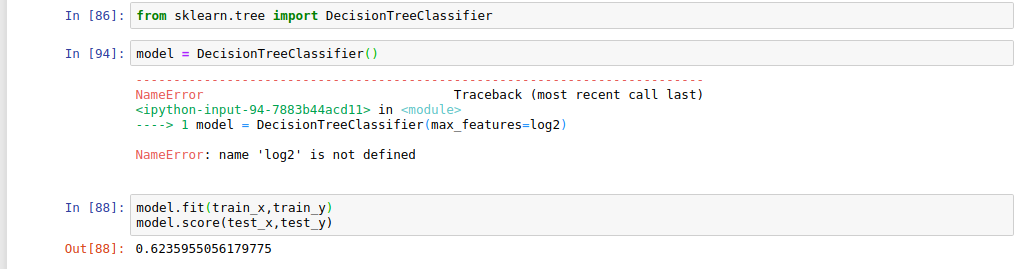


结论：模型效果相当好

其中的TP，FP，TN，FN，Precosion，Recall 以及 F1 值如下：

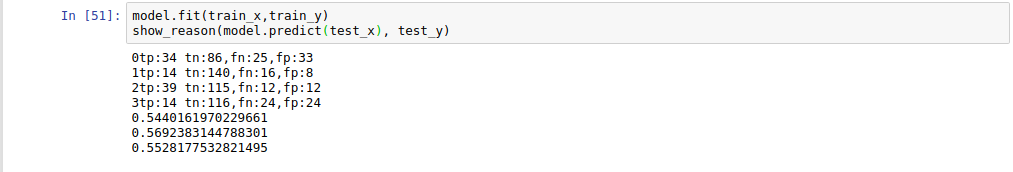


决策树：

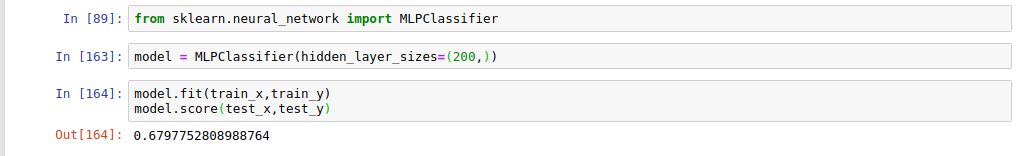


结论：模型效果尚可

其中的TP，FP，TN，FN，Precosion，Recall 以及 F1 值如下：

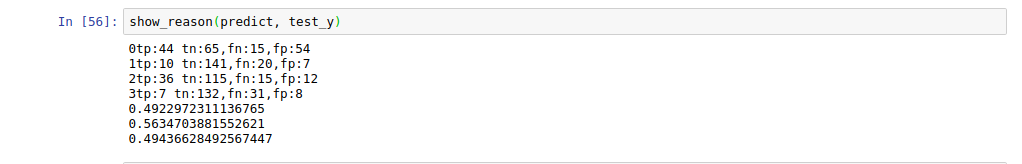


神经网络：



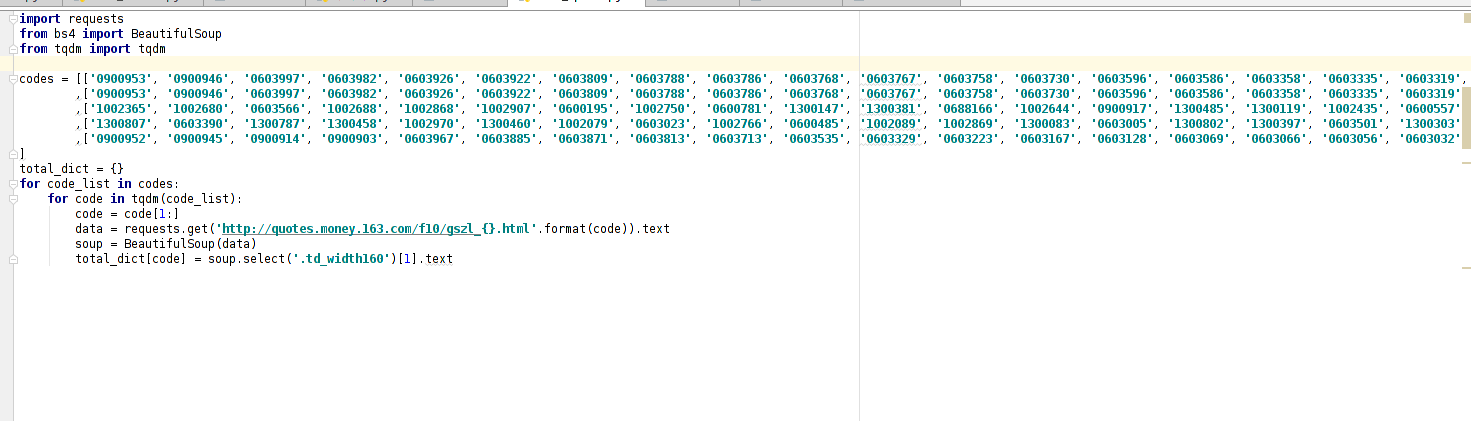
结论：模型效果尚可

其中的TP，FP，TN，FN，Precosion，Recall 以及 F1 值如下：



# 分析任务二：

## 数据获取



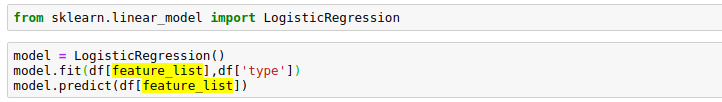
使用get请求网页，然后使用beautifulsoup解析地域和股票代码，再进行拼接即可。

## 数据处理

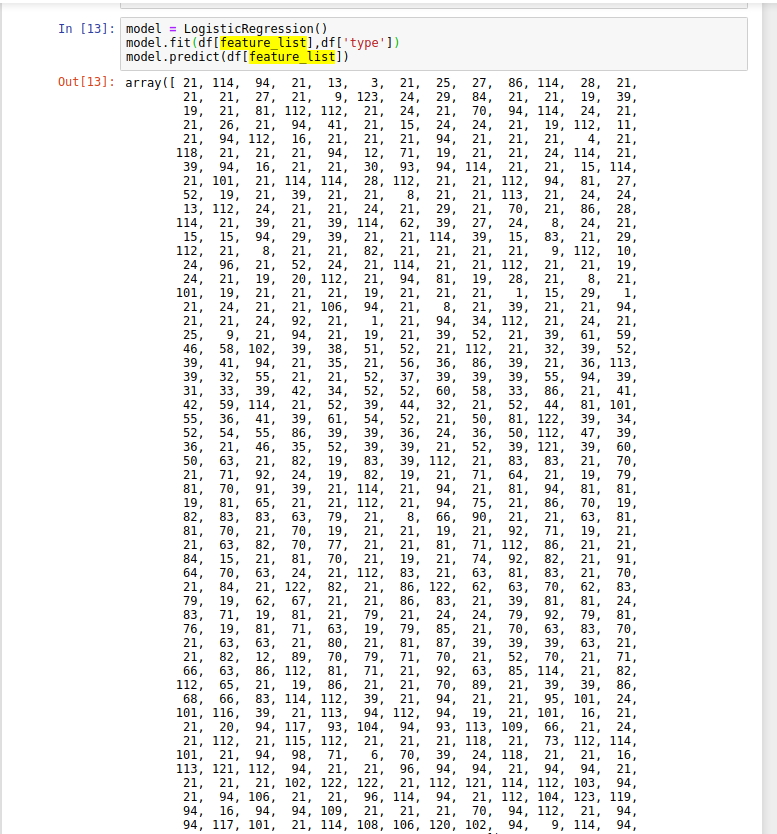
因为板块类别里有0，直接与31个地域相乘会有31类为0，于是我们采取用

（板块号+31）\*地域号，这样就不会有零值。

## 第一题：应用 Logistic 回归，预测个股所在板块 + 地域



预测结果：

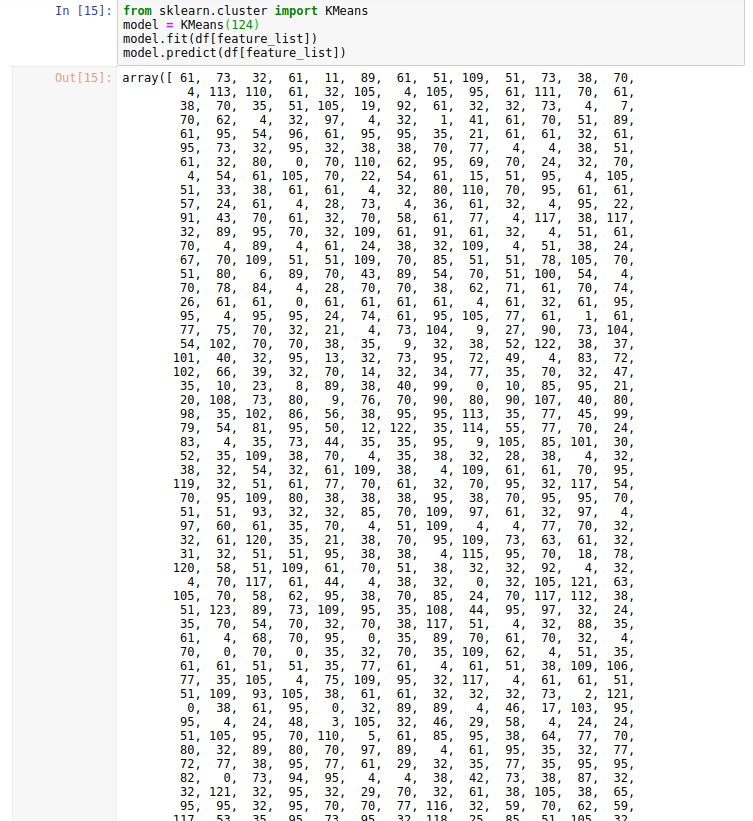


准确率为

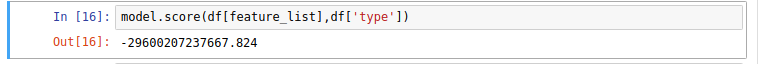


## **第二题：聚类分析以及评估**

**2.1 应用聚类分析，对个股所在板块+地域进行聚类，聚为124类**



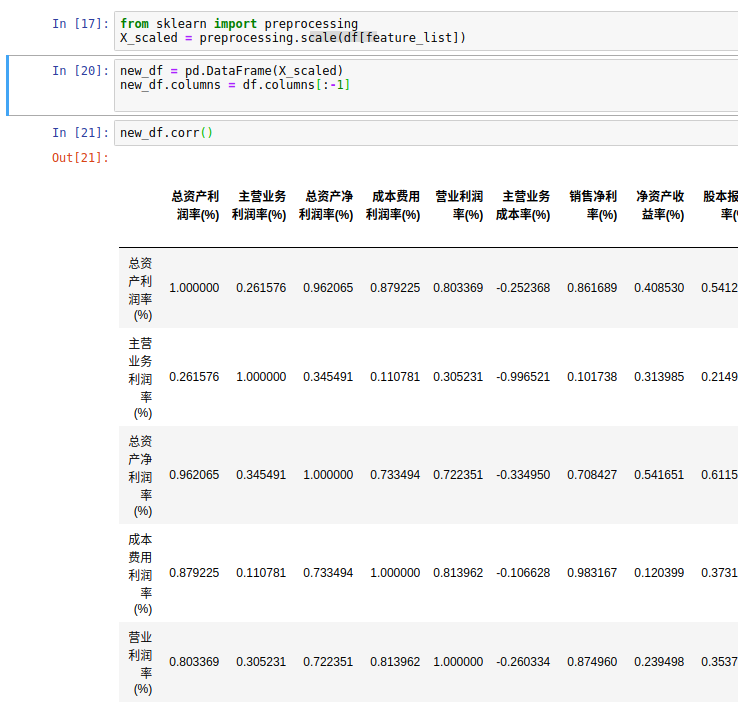
模型评分：



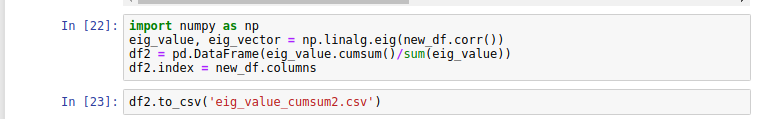
使用sklearn中的KMeans函数进行聚类，可以看出效果很不好。

**2.2应用主成分分析，重新进行聚类分析**

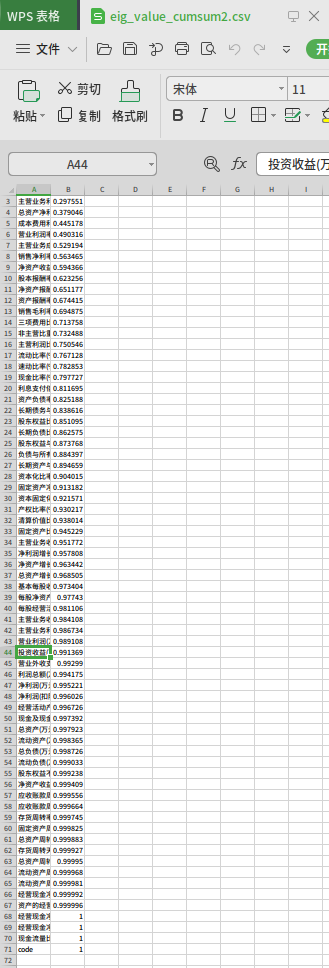
Step1：使用scale函数进行归一化处理



Step2：计算特征值和特征向量

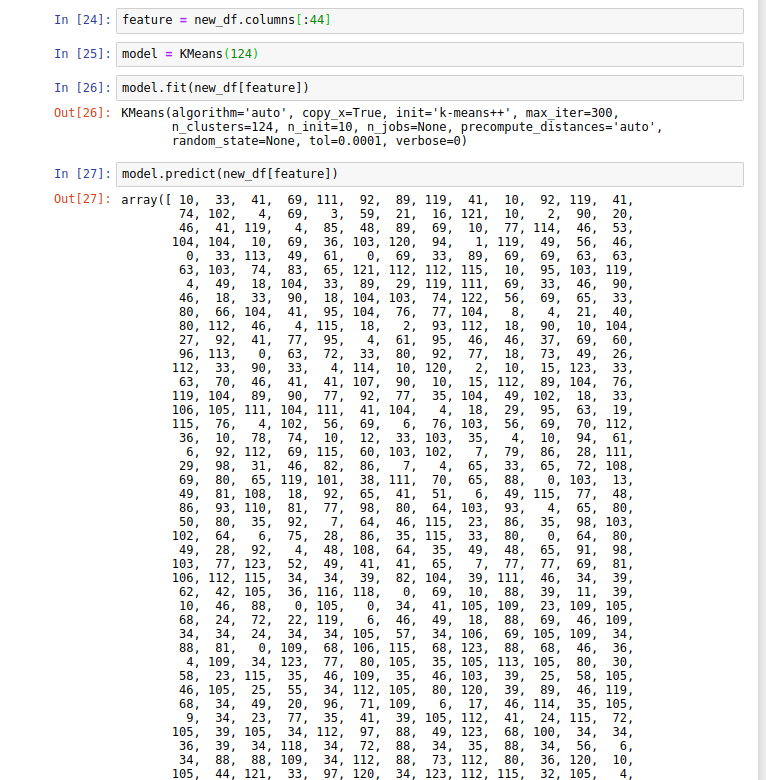


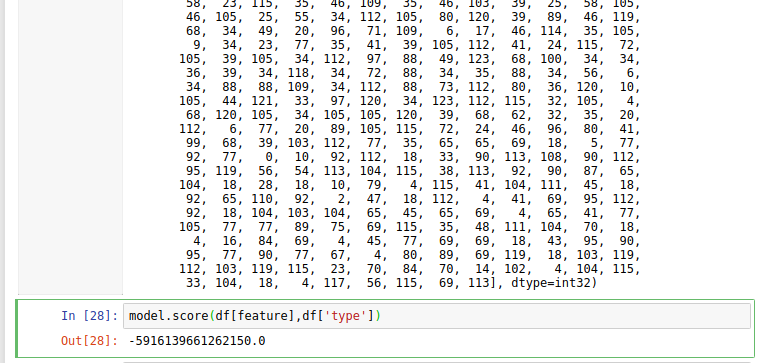
Step3：进行主成分分析，我们决定把影响结果99%的特征保留下来。



取到44行

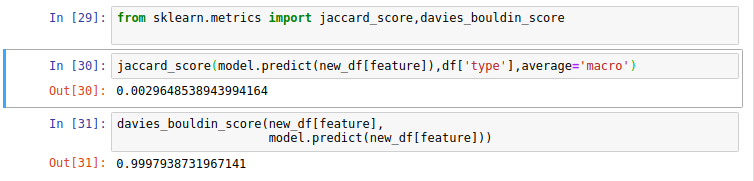
Step4:重新进行聚类查看结果





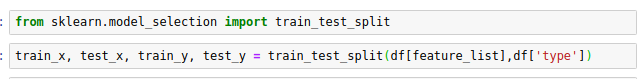
结论：效果还是不好。

**2.3 Jaccard系数和DBI系数如下：**



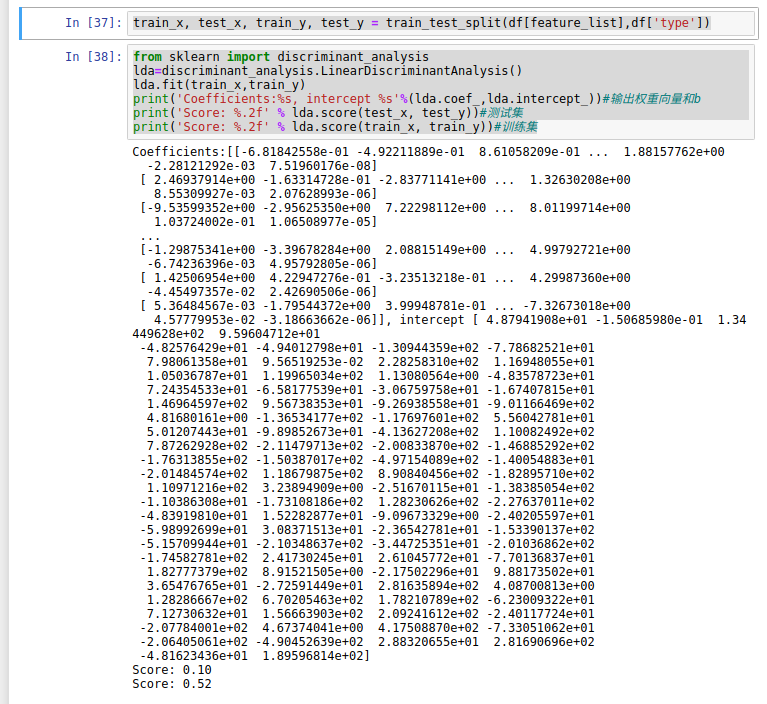
**第三题：分类分析以及评估：**

3.1：测试集和训练集的划分



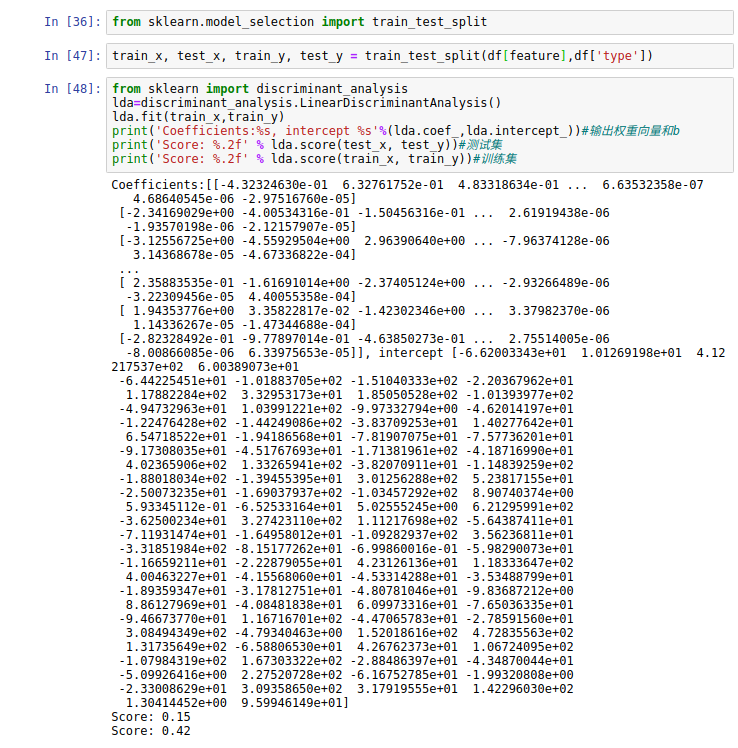
3.2：应用判别分析，预测个股所在板块 + 地域；应用主成分分析，重新进行判别分析

进行判别分析：



测试集准确率较低。

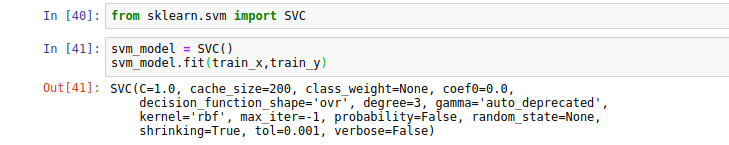
主成分分析后再进行判别分析：



测试集准确率提升，训练集准确率降低，模型泛化能力提升。

3.3应用分类算法，预测个股所在板块 + 地域，采用以下四种算法，SVM，神经网络，随机森林以及决策树

SVM：建模



准确率为：0.11797752808988764

**TTP，FP，TN，FN，Precosion，Recall 以及 F1 值（最后三行为recosion，Recall 以及 F1 值）**

0tp:0 tn:178,fn:0,fp:0

1tp:0 tn:168,fn:10,fp:0

2tp:0 tn:178,fn:0,fp:0

3tp:0 tn:178,fn:0,fp:0

4tp:0 tn:178,fn:0,fp:0

5tp:0 tn:178,fn:0,fp:0

6tp:0 tn:178,fn:0,fp:0

7tp:0 tn:178,fn:0,fp:0

8tp:0 tn:177,fn:1,fp:0

9tp:0 tn:177,fn:1,fp:0

10tp:0 tn:177,fn:1,fp:0

11tp:0 tn:178,fn:0,fp:0

12tp:0 tn:177,fn:1,fp:0

13tp:0 tn:177,fn:1,fp:0

14tp:0 tn:178,fn:0,fp:0

15tp:0 tn:176,fn:2,fp:0

16tp:0 tn:173,fn:5,fp:0

17tp:0 tn:178,fn:0,fp:0

18tp:0 tn:178,fn:0,fp:0

19tp:0 tn:176,fn:2,fp:0

20tp:0 tn:178,fn:0,fp:0

21tp:21 tn:0,fn:0,fp:157

22tp:0 tn:178,fn:0,fp:0

23tp:0 tn:178,fn:0,fp:0

24tp:0 tn:171,fn:7,fp:0

25tp:0 tn:177,fn:1,fp:0

26tp:0 tn:178,fn:0,fp:0

27tp:0 tn:175,fn:3,fp:0

28tp:0 tn:177,fn:1,fp:0

29tp:0 tn:175,fn:3,fp:0

30tp:0 tn:178,fn:0,fp:0

31tp:0 tn:176,fn:2,fp:0

32tp:0 tn:178,fn:0,fp:0

33tp:0 tn:176,fn:2,fp:0

34tp:0 tn:177,fn:1,fp:0

35tp:0 tn:178,fn:0,fp:0

36tp:0 tn:176,fn:2,fp:0

37tp:0 tn:177,fn:1,fp:0

38tp:0 tn:177,fn:1,fp:0

39tp:0 tn:175,fn:3,fp:0

40tp:0 tn:176,fn:2,fp:0

41tp:0 tn:177,fn:1,fp:0

42tp:0 tn:177,fn:1,fp:0

43tp:0 tn:178,fn:0,fp:0

44tp:0 tn:177,fn:1,fp:0

45tp:0 tn:178,fn:0,fp:0

46tp:0 tn:177,fn:1,fp:0

47tp:0 tn:177,fn:1,fp:0

48tp:0 tn:178,fn:0,fp:0

49tp:0 tn:178,fn:0,fp:0

50tp:0 tn:178,fn:0,fp:0

51tp:0 tn:178,fn:0,fp:0

52tp:0 tn:170,fn:8,fp:0

53tp:0 tn:178,fn:0,fp:0

54tp:0 tn:177,fn:1,fp:0

55tp:0 tn:178,fn:0,fp:0

56tp:0 tn:178,fn:0,fp:0

57tp:0 tn:178,fn:0,fp:0

58tp:0 tn:177,fn:1,fp:0

59tp:0 tn:178,fn:0,fp:0

60tp:0 tn:178,fn:0,fp:0

61tp:0 tn:178,fn:0,fp:0

62tp:0 tn:176,fn:2,fp:0

63tp:0 tn:173,fn:5,fp:0

64tp:0 tn:176,fn:2,fp:0

65tp:0 tn:176,fn:2,fp:0

66tp:0 tn:176,fn:2,fp:0

67tp:0 tn:178,fn:0,fp:0

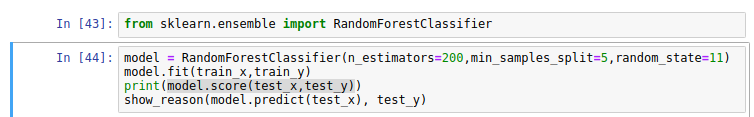
68tp:0 tn:177,fn:1,fp:0

0.014492753623188406

0.001709819247679531

0.00305877212147695

随机森林：建模



准确率为：0.19101123595505617

**TTP，FP，TN，FN，Precosion，Recall 以及 F1 值（最后三行为recosion，Recall 以及 F1 值）**

0tp:0 tn:178,fn:0,fp:0

1tp:0 tn:167,fn:10,fp:1

2tp:0 tn:178,fn:0,fp:0

3tp:0 tn:178,fn:0,fp:0

4tp:0 tn:178,fn:0,fp:0

5tp:0 tn:178,fn:0,fp:0

6tp:0 tn:178,fn:0,fp:0

7tp:0 tn:178,fn:0,fp:0

8tp:0 tn:177,fn:1,fp:0

9tp:0 tn:177,fn:1,fp:0

10tp:0 tn:177,fn:1,fp:0

11tp:0 tn:178,fn:0,fp:0

12tp:0 tn:177,fn:1,fp:0

13tp:0 tn:177,fn:1,fp:0

14tp:0 tn:178,fn:0,fp:0

15tp:0 tn:176,fn:2,fp:0

16tp:0 tn:172,fn:5,fp:1

17tp:0 tn:178,fn:0,fp:0

18tp:0 tn:178,fn:0,fp:0

19tp:0 tn:173,fn:2,fp:3

20tp:0 tn:178,fn:0,fp:0

21tp:16 tn:108,fn:5,fp:49

22tp:0 tn:178,fn:0,fp:0

23tp:0 tn:178,fn:0,fp:0

24tp:2 tn:164,fn:5,fp:7

25tp:0 tn:177,fn:1,fp:0

26tp:0 tn:178,fn:0,fp:0

27tp:0 tn:173,fn:3,fp:2

28tp:1 tn:177,fn:0,fp:0

29tp:0 tn:175,fn:3,fp:0

30tp:0 tn:178,fn:0,fp:0

31tp:0 tn:176,fn:2,fp:0

32tp:0 tn:176,fn:0,fp:2

33tp:0 tn:176,fn:2,fp:0

34tp:0 tn:177,fn:1,fp:0

35tp:0 tn:178,fn:0,fp:0

36tp:1 tn:175,fn:1,fp:1

37tp:0 tn:177,fn:1,fp:0

38tp:0 tn:177,fn:1,fp:0

39tp:3 tn:165,fn:0,fp:10

40tp:0 tn:176,fn:2,fp:0

41tp:0 tn:177,fn:1,fp:0

42tp:0 tn:177,fn:1,fp:0

43tp:0 tn:178,fn:0,fp:0

44tp:0 tn:177,fn:1,fp:0

45tp:0 tn:178,fn:0,fp:0

46tp:0 tn:177,fn:1,fp:0

47tp:0 tn:177,fn:1,fp:0

48tp:0 tn:178,fn:0,fp:0

49tp:0 tn:178,fn:0,fp:0

50tp:0 tn:178,fn:0,fp:0

51tp:0 tn:178,fn:0,fp:0

52tp:4 tn:163,fn:4,fp:7

53tp:0 tn:178,fn:0,fp:0

54tp:0 tn:177,fn:1,fp:0

55tp:0 tn:178,fn:0,fp:0

56tp:0 tn:178,fn:0,fp:0

57tp:0 tn:178,fn:0,fp:0

58tp:0 tn:177,fn:1,fp:0

59tp:0 tn:178,fn:0,fp:0

60tp:0 tn:178,fn:0,fp:0

61tp:0 tn:178,fn:0,fp:0

62tp:0 tn:175,fn:2,fp:1

63tp:0 tn:166,fn:5,fp:7

64tp:0 tn:176,fn:2,fp:0

65tp:0 tn:176,fn:2,fp:0

66tp:0 tn:176,fn:2,fp:0

67tp:0 tn:178,fn:0,fp:0

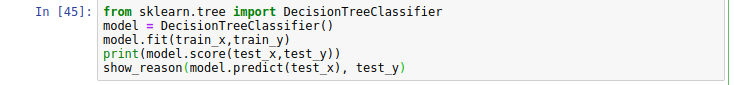
68tp:0 tn:177,fn:1,fp:0

0.0841876102292769

0.07054789346456013

0.06638146696413731

决策树：建模



准确率为：0.08426966292134831

**TTP，FP，TN，FN，Precosion，Recall 以及 F1 值（最后三行为recosion，Recall 以及 F1 值）**

0.08426966292134831

0tp:0 tn:178,fn:0,fp:0

1tp:0 tn:160,fn:10,fp:8

2tp:0 tn:178,fn:0,fp:0

3tp:0 tn:178,fn:0,fp:0

4tp:0 tn:178,fn:0,fp:0

5tp:0 tn:178,fn:0,fp:0

6tp:0 tn:178,fn:0,fp:0

7tp:0 tn:178,fn:0,fp:0

8tp:0 tn:177,fn:1,fp:0

9tp:0 tn:177,fn:1,fp:0

10tp:0 tn:176,fn:1,fp:1

11tp:0 tn:178,fn:0,fp:0

12tp:0 tn:176,fn:1,fp:1

13tp:0 tn:175,fn:1,fp:2

14tp:0 tn:178,fn:0,fp:0

15tp:0 tn:176,fn:2,fp:0

16tp:0 tn:172,fn:5,fp:1

17tp:0 tn:178,fn:0,fp:0

18tp:0 tn:178,fn:0,fp:0

19tp:0 tn:170,fn:2,fp:6

20tp:0 tn:178,fn:0,fp:0

21tp:9 tn:146,fn:12,fp:11

22tp:0 tn:178,fn:0,fp:0

23tp:0 tn:178,fn:0,fp:0

24tp:0 tn:167,fn:7,fp:4

25tp:0 tn:177,fn:1,fp:0

26tp:0 tn:178,fn:0,fp:0

27tp:0 tn:170,fn:3,fp:5

28tp:0 tn:176,fn:1,fp:1

29tp:0 tn:175,fn:3,fp:0

30tp:0 tn:177,fn:0,fp:1

31tp:0 tn:176,fn:2,fp:0

32tp:0 tn:176,fn:0,fp:2

33tp:0 tn:176,fn:2,fp:0

34tp:0 tn:177,fn:1,fp:0

35tp:0 tn:177,fn:0,fp:1

36tp:1 tn:174,fn:1,fp:2

37tp:0 tn:177,fn:1,fp:0

38tp:0 tn:177,fn:1,fp:0

39tp:1 tn:170,fn:2,fp:5

40tp:0 tn:176,fn:2,fp:0

41tp:0 tn:175,fn:1,fp:2

42tp:0 tn:177,fn:1,fp:0

43tp:0 tn:178,fn:0,fp:0

44tp:0 tn:174,fn:1,fp:3

45tp:0 tn:178,fn:0,fp:0

46tp:0 tn:177,fn:1,fp:0

47tp:0 tn:177,fn:1,fp:0

48tp:0 tn:178,fn:0,fp:0

49tp:0 tn:178,fn:0,fp:0

50tp:0 tn:177,fn:0,fp:1

51tp:0 tn:178,fn:0,fp:0

52tp:2 tn:167,fn:6,fp:3

53tp:0 tn:178,fn:0,fp:0

54tp:0 tn:177,fn:1,fp:0

55tp:0 tn:178,fn:0,fp:0

56tp:0 tn:178,fn:0,fp:0

57tp:0 tn:178,fn:0,fp:0

58tp:0 tn:177,fn:1,fp:0

59tp:0 tn:178,fn:0,fp:0

60tp:0 tn:177,fn:0,fp:1

61tp:0 tn:177,fn:0,fp:1

62tp:0 tn:174,fn:2,fp:2

63tp:0 tn:173,fn:5,fp:0

64tp:0 tn:175,fn:2,fp:1

65tp:0 tn:174,fn:2,fp:2

66tp:0 tn:175,fn:2,fp:1

67tp:0 tn:178,fn:0,fp:0

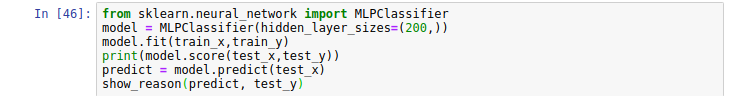
68tp:0 tn:177,fn:1,fp:0

0.023235800344234076

0.019199876428792093

0.02017110937413457

神经网络：建模



准确率为：0.056179775280898875

**TTP，FP，TN，FN，Precosion，Recall 以及 F1 值（最后三行为recosion，Recall 以及 F1 值）**

1tp:0 tn:161,fn:10,fp:7

2tp:0 tn:178,fn:0,fp:0

3tp:0 tn:178,fn:0,fp:0

4tp:0 tn:178,fn:0,fp:0

5tp:0 tn:178,fn:0,fp:0

6tp:0 tn:177,fn:0,fp:1

7tp:0 tn:178,fn:0,fp:0

8tp:0 tn:173,fn:1,fp:4

9tp:0 tn:176,fn:1,fp:1

10tp:0 tn:177,fn:1,fp:0

11tp:0 tn:178,fn:0,fp:0

12tp:0 tn:177,fn:1,fp:0

13tp:0 tn:176,fn:1,fp:1

14tp:0 tn:178,fn:0,fp:0

15tp:1 tn:175,fn:1,fp:1

16tp:0 tn:173,fn:5,fp:0

17tp:0 tn:178,fn:0,fp:0

18tp:0 tn:178,fn:0,fp:0

19tp:0 tn:172,fn:2,fp:4

20tp:0 tn:178,fn:0,fp:0

21tp:3 tn:151,fn:18,fp:6

22tp:0 tn:178,fn:0,fp:0

23tp:0 tn:178,fn:0,fp:0

24tp:0 tn:166,fn:7,fp:5

25tp:0 tn:174,fn:1,fp:3

26tp:0 tn:178,fn:0,fp:0

27tp:0 tn:170,fn:3,fp:5

28tp:1 tn:177,fn:0,fp:0

29tp:0 tn:175,fn:3,fp:0

30tp:0 tn:178,fn:0,fp:0

31tp:0 tn:176,fn:2,fp:0

32tp:0 tn:177,fn:0,fp:1

33tp:0 tn:175,fn:2,fp:1

34tp:0 tn:176,fn:1,fp:1

35tp:0 tn:178,fn:0,fp:0

36tp:0 tn:175,fn:2,fp:1

37tp:0 tn:177,fn:1,fp:0

38tp:0 tn:177,fn:1,fp:0

39tp:0 tn:167,fn:3,fp:8

40tp:0 tn:169,fn:2,fp:7

41tp:0 tn:177,fn:1,fp:0

42tp:0 tn:176,fn:1,fp:1

43tp:0 tn:178,fn:0,fp:0

44tp:0 tn:177,fn:1,fp:0

45tp:0 tn:178,fn:0,fp:0

46tp:0 tn:176,fn:1,fp:1

47tp:0 tn:176,fn:1,fp:1

48tp:0 tn:178,fn:0,fp:0

49tp:0 tn:178,fn:0,fp:0

50tp:0 tn:176,fn:0,fp:2

51tp:0 tn:178,fn:0,fp:0

52tp:2 tn:164,fn:6,fp:6

53tp:0 tn:178,fn:0,fp:0

54tp:0 tn:176,fn:1,fp:1

55tp:0 tn:176,fn:0,fp:2

56tp:0 tn:178,fn:0,fp:0

57tp:0 tn:178,fn:0,fp:0

58tp:0 tn:177,fn:1,fp:0

59tp:0 tn:177,fn:0,fp:1

60tp:0 tn:178,fn:0,fp:0

61tp:0 tn:178,fn:0,fp:0

62tp:0 tn:170,fn:2,fp:6

63tp:0 tn:163,fn:5,fp:10

64tp:0 tn:176,fn:2,fp:0

65tp:0 tn:176,fn:2,fp:0

66tp:0 tn:176,fn:2,fp:0

67tp:0 tn:178,fn:0,fp:0

68tp:0 tn:177,fn:1,fp:0

0.042244323889893506

0.03551336146272855

0.03616971401781528

随机森林效果最好，神经网络效果最差。