海南大学
HAINAN UNIVERSITY

数据结构课程设计论文

基于最小生成树的股票分层工具
——以沪深300为例

2015.12.11

卢俊星
海南大学
StoryInStone@outlook.com
https://github.com/StoryInStone

通信工程
20120802310083

目录

1 概念

1.1 研究背景与意义

1.1.1 研究背景

对于风格投资的研究，国外的已有文献较多，Farrell[1]和Sharpe[1]都发现了股票收益间较高的相关性，与风格相关。一些实证研究也表明，基础价值协同性并不是收益协同性的唯一根源，Fama[1]和French的研究表明某些股票的收益协同性与基础价值协同性无关；另外，大量文献证明了初基础价值外，投资者偏好、交易地点、现金流、指数都可以导致收益协同性形成风格。Barberis等总结了已有的研究成果并提出了风格投资协同理论——风格投资产生收益协同性。对于同种风格的股票之间存在较强的相关性。

1.1.2 研究作用与意义

指数分层结构图横坐标为股票名称，纵坐标为股间距离。通过绘制指数分层结构图，可以清晰的看到股票的聚集状态，推导出风格的分布和组合样本。继而为研究风格形成因素、收益及风险水平等性质开辟了道路，为证券投资组合配置提供依据和参考。从上述可知，指数分层结构图对风格投资分析的重要性，然而在绘制指数分层结构图时用到的数据相当庞大，因此研究指数分层结构图的绘制算法具有实际的意义与价值。

2 目标

2.1 股票分层工具

制作一个简单的股票分层工具，能对沪深300中的股票进行聚类，输入股票代码能得到其同类风格的股票代码，给投资者提供分析基础。

2.2 巩固数据结构基础

数据结构是构建计算机世界的砖瓦，在本项目中，通过相关训练，巩固课堂上所学习的数据结构基础。

2.3 训练相关算法

算法的知识较为复杂，通过本文的训练，目标是能够入门写一些简单的算法程序，如最小生成树MST程序。

2.4 项目与工具

好的项目管理思想以及相关工具的使用，能够使工作事半功倍，在本项目的编码过程中，熟悉相关思想和工具使用方法。

3 知识与应用

3.1 数据结构基础应用

3.1.1 向量

本文中有沪深300股票代码数据，股票每日收盘数据，股票在某一时段内价格变化趋势序列三类主要数据，由于数据在程序的相关处理过程中并不需要动态操作，因此本文采用向量结构对数据进行存储和传递。

3.1.2 图

本文中研究的A股市场中代表性股票之间关系，任两支股票间都存在两两股间距离，故形成无向连通图，由于两两股票间的距离一定存在，故图为稠密图，本文采用邻接矩阵来表示股间距离图。

3.1.3 最小生成树

任何连通图都必存在一最小生成树，使图不成环。树结构可以转化为相应的层次结构，因此该最小生成树与股票分层结构图是一一对应的。

3.2 自动化项目管理知识应用

3.2.1 Make工具

4 方法

4.1 编程与写作环境

本文采用C/C++编译器为GCC 5.1.1，系统环境为Fedora 23，代码编辑器为VIM 7.4，项目编译工具为Make 4.0，文档编辑工具为Latex。

4.2 贪心算法求最小生成树

贪心算法一般不能解决实际问题，但是只要能使用贪心算法解决的问题，那么对于该问题而言，此算法即为最优的。

Algorithm 1 CH election algorithm

```
1: procedure CH–Election
2:     for each node i ∈ N do
3:         Broadcast HELLO message to its neighbor
4:         let k ∈ N1 (i) U i be s.t
5:         QOS(k) = max QOS(j) |j ∈ N1(i) U i
6:         MPRSet(i) = k
7:     end for
8: end procedure
```

4.3 开源支持

4.3.1 股票数据调取项目——Tushare

本文中所有股票数据全部来自于Tushare财经接口工具包项目，该项目采用Python编码，从各大官方财经站点调取股票数据。

5 Source Code

Listing 1: StdDeviation.h

```
1  //===================================================================================
2  //        Filename:  StdDeviation.h
3  //
4  //     Description:
5  //
6  //         Dersion:  1.0
7  //         Created:  12/08/2015 02:15:06 PM
8  //        Revision:  none
9  //        Compiler:  g++
10 //
11 //          Author:  Junxing Lu (SIS), StoryInStone@outlook.com
12 //         Company:
13 //===================================================================================
14
15 #include <math.h>
16
17 #define D 100
18
19 class StdDeviation
20 {
21     private:
22         int max;
23         double value[D];
24         double mean;
25     public:
26         double CalculateMean()
27         {
28             double sum = 0;
29             for(int i = 0; i < max; i++)
30                 sum += value[i];
31             return (sum / max);
32         }
33         double CalculateVariane()
34         {
35             mean = CalculateMean();
36             double temp = 0;
37             for(int i = 0; i < max; i++)
38                 temp += (value[i] - mean) * (value[i] - mean) ;
39             return temp / max;
40         }
41         double CalculateSampleVariane()
42         {
43             mean = CalculateMean();
44             double temp = 0;
45             for(int i = 0; i < max; i++)
46                 temp += (value[i] - mean) * (value[i] - mean) ;
47             return temp / (max - 1);
48         }
49         int SetValues(double *p, int count)
50         {
51             if(count > D)
52                 return -1;
```

```
53              max = count;
54              for(int i = 0; i < count; i++)
55                  value[i] = p[i];
56              return 0;
57          }
58          double Calculate_StandardDeviation()
59          {
60              return sqrt(CalculateVariane());
61          }
62          double Calculate_SampleStandardDeviation()
63          {
64              return sqrt(CalculateSampleVariane());
65          }
66  };
67  class Calculator
68  {
69      private:
70          double XSeries[D];
71          double YSeries[D];
72          int max;
73          StdDeviation x;
74          StdDeviation y;
75      public:
76          void SetValues(double *xvalues, double *yvalues, int count)
77          {
78              for(int i = 0; i < count; i++)
79              {
80                  XSeries[i] = xvalues[i];
81                  YSeries[i] = yvalues[i];
82              }
83              x.SetValues(xvalues, count);
84              y.SetValues(yvalues, count);
85              max = count;
86          }
87          double Calculate_Covariance()
88          {
89              double xmean = x.CalculateMean();
90              double ymean = y.CalculateMean();
91              double total = 0;
92              for(int i = 0; i < max; i++)
93              {
94                  total += (XSeries[i] - xmean) * (YSeries[i] - ymean);
95              }
96              return total / max;
97          }
98          double Calculate_Correlation()
99          {
100             double cov = Calculate_Covariance();
101             double correlation = cov / (x.Calculate_StandardDeviation() * y.
    Calculate_StandardDeviation());
102             return correlation;
103         }
104 };
```

Listing 2: Mst.h

```
//===============================================================================
//         Filename:  Mst.h
//
//      Description:
//
//          Version:  1.0
//          Created:  12/08/2015 04:22:14 PM
//         Revision:  none
//         Compiler:  g++
//
//           Author:  Junxing Lu (SIS), StoryInStone@outlook.com
//          Company:
//===============================================================================

#include <stdio.h>
#include <cstdlib>
#include <math.h>
#include <limits.h>
#include <iostream>
#include <string>

using namespace std;

#define V 300
class Mst
{
    private:
        int parent[V];
        double key[V];
        bool mstSet[V];
        double graph[V][V];
    public:
    void createGraph(double gra[V][V]) {
        for (int i=0;i<V;i++) {
            for (int j=0;j<V;j++) {
                graph[i][j] = gra[i][j];
            }
        }
    }
    int minKey()
    {
        int min = 65535, min_index;
        for (int v = 0; v < V; v++)
        {
            if (mstSet[v] == false && key[v] < min)
                min = key[v], min_index = v;
        }
        return min_index;
    }
    void primMST()
    {
        for (int i = 0; i < V; i++)
        {
            key[i] = 65535, mstSet[i] = false;
```

```
55          }
56          key[o] = o;
57          parent[o] = -1;
58          for (int count = o; count < V-1; count++)
59          {
60              int u = minKey();
61              mstSet[u] = true;
62              for (int v = o; v < V; v++)
63              {
64                  if (graph[u][v] && mstSet[v] == false && graph[u][v] < key[v
    ])
65                  {
66                      parent[v] = u;
67                      key[v] = graph[u][v];
68                  }
69              }
70          }
71          printMST();
72      }
73      void printMST()
74      {
75          for (int i = o; i < V; i++)
76              cout << parent[i] << "--" << i << "␣␣␣" << graph[i][parent[i]]
    << endl;
77      }
78  };
```

Listing 3: FileDeal.cc

```
//=====================================================================================
//        Filename:  FileDeal.cc
//
//     Description:
//
//         Version:  1.0
//         Created:  12/08/2015 03:00:42 PM
//        Revision:  none
//        Compiler:  g++
//
//          Author:  Junxing Lu (SIS), StoryInStone@outlook.com
//         Company:
//=====================================================================================

#include <fstream>
#include <vector>
#include <string>
#include <sstream>
#include <iostream>
#include <stdlib.h>

#define I 13
#define C 300
using namespace std;

vector<double> GetShareData(string code)
{
    string line;
    string url = "./lnShare/"+code+".txt";
    const char *ch = url.c_str();
    ifstream file (ch);
    if (file.is_open())
    {
        getline(file, line);
    }
    file.close();
    for (unsigned int j=0; j<line.length(); j++)
    {
        if (line[j] == ',')
            line[j] = ' ';
    }
    vector<double> arr;
    string tmp;
    int i = 0;
    stringstream ssin(line);
    while (ssin.good() && i < I)
    {
        ssin >> tmp;
        arr.push_back(atof(tmp.c_str()));
        ++i;
    }
    return arr;
}
vector<string> GetShareCode()
```

```
{
    string line;
    ifstream file ("./hs300_code.txt");
    if (file.is_open())
    {
        getline(file, line);
    }
    file.close();
    for (unsigned int j=0; j<line.length(); j++)
    {
        if (line[j] == ',')
            line[j] = ' ';
    }
    vector<string> arr;
    string tmp;
    int i = 0;
    stringstream ssin(line);
    while (ssin.good() && i < C)
    {
        ssin >> tmp;
        arr.push_back(tmp);
        ++i;
    }
    return arr;
}
```

Listing 4: main.cc

```
//===============================================================================
//
//       Filename:  main.cc
//
//    Description:
//
//        Version:  1.0
//        Created:  12/08/2015 03:48:31 PM
//       Revision:  none
//       Compiler:  g++
//
//         Author:  Junxing Lu (SIS), StoryInStone@outlook.com
//        Company:
//
//===============================================================================

#include "./CorrelationCalculator.h"
#include "./FileDeal.cc"
#include "./Mst.h"
#include <math.h>


using namespace std;
int main() {
    Calculator calc;
    Mst m;
    double cor[V][V];
    vector<string> hs300;
    hs300 = GetShareCode();
    string share[V];
    for (vector<double>::size_type ix = 1; ix != hs300.size(); ++ix)
    {
        share[ix-1] = hs300[ix];
    }
    int count = hs300.size() - 1;
    for (int i = 0; i < count; i++)
    {
        for (int j = 0; j < count; j++)
        {
            double xvalues[I];
            double yvalues[I];
            vector<double> xv;
            vector<double> yv;
            xv = GetShareData(share[i]);
            yv = GetShareData(share[j]);
            for (vector<double>::size_type ix = 1; ix != xv.size(); ++ix)
            {
                xvalues[ix] = xv[ix];
            }
            for (vector<double>::size_type ix = 1; ix != yv.size(); ++ix)
            {
                yvalues[ix] = yv[ix];
            }
            calc.SetValues(xvalues, yvalues, I);
```

```
55              cor[i][j] = fabs(calc.Calculate_Correlation());
56          }
57      }
58      m.createGraph(cor);
59      m.primMST();
60      return o;
61  }
```

参考文献

[1] 卢俊星, “这是一个测试,” *J. of Math*, vol. 58, no. 345-363, p. 5, 1936.