

```

1  #include <iostream>           // cout, endl
2  #include <fstream>           // fstream
3  #include <vector>
4  #include <algorithm>         // copy
5  #include <iterator>          // ostream_operator
6  #include<sstream>
7  #include<cstdlib>
8  #include<ostream>
9  #include <string>
10 #include <iomanip>
11 #include <random>
12 #include<cmath>
13 #include<utility>
14 #include <boost/tokenizer.hpp>
15 #include <boost/accumulators/accumulators.hpp>
16 #include <boost/accumulators/statistics.hpp>
17 #include <boost/bind.hpp>
18 #include <boost/ref.hpp>
19
20 #define SAMPLESIZE 8
21 #define TOTALSIZE 60
22 #define NLEX 2
23 #define TOTALSAMPLE 4
24
25
26 typedef std::vector<double> DVector;
27 typedef std::vector<std::string> ColHead;
28 typedef std::vector<double> TempVec;
29 typedef std::vector<int> N1delns;
30 typedef std::vector<int> N2delns;
31 typedef std::vector<std::pair<std::string,double>> GeneMap;
32
33 using namespace boost::accumulators;
34 typedef accumulator_set<double, stats<tag::mean, tag::variance>> accSet;
35 double differentiation(std::vector<double> genedata,int group1,int group2);
36 double ttest(accSet acc1,accSet acc2,int n1,int n2);
37
38 struct sort_pred {
39     bool operator()(const std::pair<std::string,double> &left, const std::pair<std::string,double> &right)
40     {
41         return left.second < right.second;
42     }
43 };
44
45
46 int main()
47 {
48     using namespace std;
49     using namespace boost;
50
51     string data("NCI60.csv");
52
53     ifstream in(data.c_str());
54     if (!in.is_open()) return 1;
55     std::filebuf fb;
56     fb.open ("test.txt",std::ios::out);
57     std::ostream os(&fb);
58
59     typedef tokenizer< escaped_list_separator<char> > Tokenizer;
60
61     DVector vec;
62     TempVec tempv;
63     ColHead colhead;
64     N1delns n1deln;
65     N2delns n2deln;

```

```

66     GeneMap gene;
67
68     string line;
69     int n=0;
70     int a=0;
71
72     while (getline(in,line))
73     {
74
75         if(n==0){
76             Tokenizer firstline(line);
77             Tokenizer::const_iterator token_iterator = firstline.begin();
78             std::istringstream to_string;
79
80             for(token_iterator;token_iterator!=firstline.end();token_iterator++)
81             {
82                 string datum;
83
84                 to_string.clear();
85                 to_string.str(*token_iterator);
86                 to_string >> datum;
87                 colhead.push_back(datum);
88             }
89
90         }
91
92     else{
93
94         Tokenizer tok(line);
95         Tokenizer::const_iterator token_iterator = tok.begin();
96
97
98
99         std::istringstream to_double;
100
101         std::string id(*token_iterator);
102
103
104         for(++token_iterator; token_iterator != tok.end(); ++token_iterator)
105         {
106             int m=0;
107             double datum = 0.0;
108
109             to_double.clear();
110             to_double.str(*token_iterator);
111             to_double >> datum;
112             tempv.push_back(datum);
113
114
115
116             if(!to_double)
117                 continue;
118
119             vec.push_back(datum);
120
121         }
122         int n1del=0;
123         int n2del=0;
124         for(int i=0;i<SAMPLESIZE;i++)
125         {
126             if(tempv[i]==fabs(0))
127                 n1del++;
128
129         }
130
131         double dscore= differentiation(vec,SAMPLESIZE-n1del,vec.size()+n1del-SAMPLESIZE);

```

```

132         gene.push_back(make_pair(id,dscore));
133
134     os << gene[n-1].first<<" "<<gene[n-1].second<<endl;
135         //cout<<gene[n-1].first<<"\t"<<gene[n-1].second<<endl;
136         tempv.clear();
137
138         vec.clear();
139
140     }
141     //cout<<gene.size()<<endl;
142     n++;
143 }
144
145
146     std::sort(gene.begin(),gene.end(),
147         boost::bind(&std::pair<std::string, double>::second, _1) >
148         boost::bind(&std::pair<std::string, double>::second, _2));
149     //cout<<gene.size()<<endl;
150     for (GeneMap::iterator it = gene.begin(); it != gene.begin()+20; ++it)
151
152         cout << it->first <<" "<< it->second << endl;
153
154
155 }
156
157
158 double differentiation(std::vector<double> genedata,int group1,int group2)
159 {
160     accSet acc1;
161     accSet acc2;
162     double dscore;
163     std::for_each(genedata.begin(), genedata.begin()+group1 ,
164         boost::bind<void>(boost::ref(acc1), _1));
165
166
167     std::for_each(genedata.begin()+group1, genedata.end() ,
168         boost::bind<void>(boost::ref(acc2), _1));
169
170
171     double tSamp = ttest(acc1,acc2,group1,group2);
172
173     accSet randacc1;
174     accSet randacc2;
175     accSet randTest;
176     for(int i=0;i<100;i++)
177     {
178
179
180         std::random_shuffle(genedata.begin(), genedata.end());
181
182         std::for_each(genedata.begin(), genedata.begin()+group1 ,
183             boost::bind<void>(boost::ref(randacc1), _1));
184
185         std::for_each(genedata.begin()+group1, genedata.end() ,
186             boost::bind<void>(boost::ref(randacc2), _1));
187
188
189     double tscore= ttest(randacc1,randacc2,group1,group2);
190     randTest(tscore);
191
192     }
193     dscore = fabs((double)tSamp - (double)mean(randTest)) /
194         sqrt(variance(randTest));
195
196
197     return dscore;

```

```
198 }
199
200
201 double ttest(accSet acc1,accSet acc2,int n1,int n2)
202 {
203
204 double mean1 = mean(acc1);
205 double mean2 = mean(acc2);
206 double variancel = variance(acc1);
207 double variance2 = variance(acc2);
208 double tscore= ((mean2-mean1)/(sqrt(((variancel*variancel)/((double)n1)+((variance2*variance2)/((double)n2
209 )))));
210 return tscore;
211 }
212
```